

Program: Master of Computer Applications

Curriculum Scheme: CBCGS

Examination: MCA 1<sup>st</sup> Year SEMESTER II

Course Code: MCA23\_and Course Name: Information Security

Time: 2 HRS

Max. Marks: 80

Section I - MCQS (40 Marks) – 40 Minutes

Section II – Subjective (40 Marks) – 80 Minutes

**The timings**

**Examination Time is 11:00 am to 1:00 noon**

**Section I – 11:00 am – 11:40 am**

**Section II – 11.40 am – 1:00 pm**

=====

**SECTION II**

**Q2. Solve any two questions out of three.**

**20 marks**

A) Explain Data Encryption Standard in detail.

B) Explain the working of MD5 in detail.

C) Explain the SSL handshake Protocol.

**Q3. Solve any Four from five.**

**20 marks**

A) Use the Euclidean Algorithm to find the greatest common divisor of 44 and 17.

B) What is the principle of information security?

C) Explain MAC in detail.

D) Write short note on MIME and PGP.

E) Explain Bastion Host.

10/8/21

Program: Master of Computer Applications

Curriculum Scheme: MCA 2 year Course

Examination: MCA I YEAR SEMESTER II

Course Code: MCA23 and Course Name: Information Security

Time: 2 HRS

Max. Marks: 80

Section I - MCQS (40 Marks) – 40 Minutes

Section II – Subjective (40 Marks) – 80 Minutes

---

## SECTION II

**Q2. Solve any four questions (Each question carries 05 Marks)**

1. Distinguish between symmetric and asymmetric key encryption. State the advantage of both.
2. Explain Digital Certificates Process.
3. Discuss SSL as an internet security protocol and three major protocols used in SSL?
4. Using Euclidean algorithm, find the greatest common divisor of the following:
  - i. 24 and 320
  - ii. 401 and 700
5. Discuss Role-Based Access Control.
6. What are the principles of Security?

**Q3. Solve any two questions (Each question carries 10 Marks)**

1. Explain and differentiate between various architectures of firewall implementation.
2. What is mutual authentication? Explain various ways of implementing the mutual authentication.
3. Explain RSA with a suitable example.



1 | Page

# **INFORMATION SECURITY**

## **Introduction**

**– Mrs. Vaishali Gatty**

# CONTENTS

- Introduction to information security
- Principles
- Services and attacks
- Functional requirements of security
- Current trends in security

# HISTORY OF INFORMATION SECURITY

- 1960s: Password protection
- 1970s:
  - [Cybersecurity](#)'s history began with a research project during the 1970s, on what was then known as the ARPANET (The Advanced Research Projects Agency Network).
  - A researcher named Bob Thomas created a [computer program](#) which was able to move ARPANET's network, leaving a small trail wherever it went.
  - He named the program 'CREEPER'
  - Ray Tomlinson – the man who invented email – later designed a program which took CREEPER to the next level, making it self-replicating and the first ever computer worm.
  - Revealed a number of flaws in [ARPANET's network](#) security
  - he then wrote another program called Reaper which chased CREEPER and deleted it- [Antivirus](#)

# HISTORY OF INFORMATION SECURITY

- 1980s: The internet

- computers started to become more and more connected, computer viruses became more advanced
  - 1986-
    - Russians employed German computer hacker Marcus Hess to steal US military secrets. hacked into over 400 military computers.
  - 1988- Morris Worm-
    - Named after its inventor Robert Morris, the worm was designed to propagate across networks, infiltrate terminals using a known bug, and then copy itself.
    - Its aim was to identify lacking areas in a network intrusion prevention system.
    - But worm replicated so aggressively that it rendered targeted computers inoperable and slowed the internet down to a crawling pace and caused untold damage.
- ARPANET network -> Internet
- Made available to public as WWW in 1989

# HISTORY OF INFORMATION SECURITY

- 1990s: The rise of firewalls
  - Network security threats increased exponentially and, as such, firewalls and antivirus programs had to be produced on a mass basis to protect the public.
  - A nasa researcher created the very first firewall program design, following a computer virus attack at their california base.
- 2000s:
  - Sever punishments to clamp down on the criminality of hacking
- 2010s: The era of major breaches
  - **Snowden & The NSA, 2013**
  - **Yahoo, 2013 – 2014**
  - **WannaCry, 2017**-first ‘ransomworm’-demanded ransom payments in the Bitcoin cryptocurrency infected over 230,000 computers across 150 countries in a day.

# WHAT IS INFORMATION SECURITY ?

- The protection of information and its critical elements, including systems and hardware that use, store, and transmit that information
- Necessary tools: policy, awareness, training, education, technology

## Security Approaches

- No security
- Security through obscurity
- Host Security
- Network Security

# WEAKNESSES IN THE SYSTEM

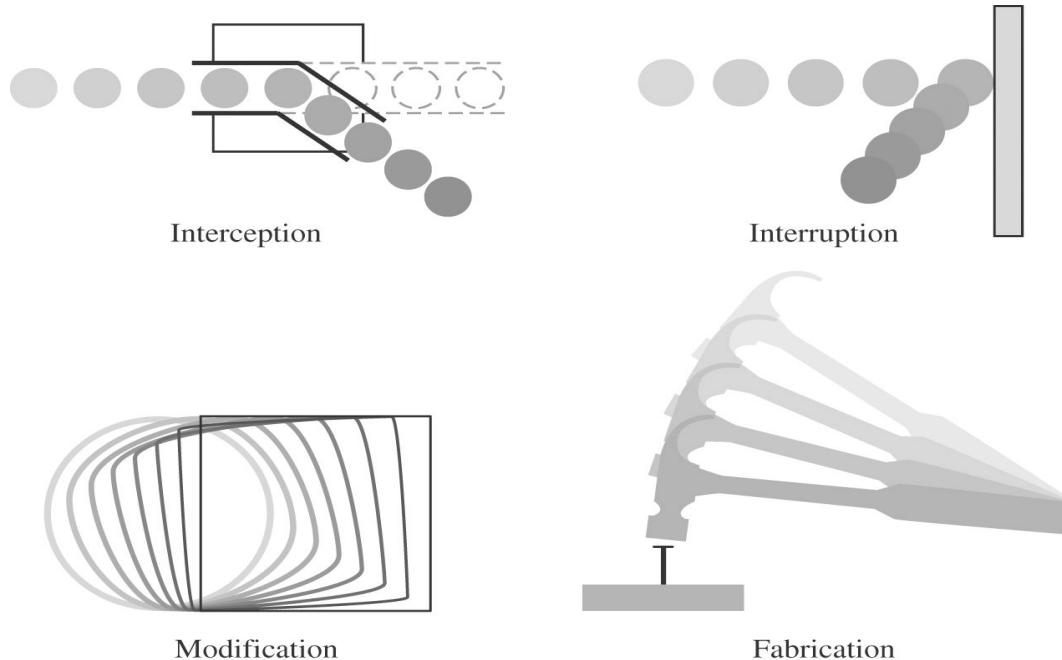
- **vulnerability** – is a weakness in the security system.
  - for example, in procedures, design, or implementation, that might be exploited to cause loss or harm.
  - For instance, a particular system may be vulnerable to unauthorized data manipulation because the system does not verify a user's identity before allowing data access.
- **threat** -is a set of circumstances that has the potential to cause loss or harm.
- **attack** – an assault on system security, a deliberate attempt to evade security services

A *threat* is blocked by *control of a vulnerability*.

# PRINCIPLES OF SECURITY

- **Confidentiality** (Secrecy or Privacy)- assets accessed only by authorized parties
  - Not only reading but viewing, printing or knowing about the asset
- **Integrity**- assets modified only by authorized parties
  - Includes writing, changing, changing the status, deleting or creating
- **Authentication**– helps to establish Proof of identities
- **Non-Repudiation** - prevents either sender or receiver from denying a transmitted message
- **Access Control** - prevention of the unauthorized use of a resource
- **Availability** –Resources should be available to authorized parties at all times

# TYPE OF THREATS



Pfleeger/Pfleeger Fig. 01-02

# TYPE OF THREATS

- **Interception:**

- some unauthorized party gains access to an asset.
- The outside party can be a person, a program, or a computing system.
- Examples: copying of program or data files, or wiretapping to obtain data in a network.

# TYPE OF THREATS

- **Interruption:**
  - an asset of the system becomes lost, unavailable, or unusable.
  - Example: malicious destruction of a hardware device, erasure of a program or data file, or malfunction of an operating system file manager so that it cannot find a particular disk file.

# TYPE OF THREATS

- **Modification:**

- an unauthorized party not only accesses but changes the state of an asset.
- someone might change the values in a database, alter a program so that it performs an additional computation, or modify data being transmitted electronically.

# TYPE OF THREATS

- **Fabrication:**

- an unauthorized party might create a **fabrication** of counterfeit objects on a computing system.
- The intruder may insert bogus transactions to a network communication system or add records to an existing database.

# SECURITY ATTACKS

Attacks can be classified as active or passive.

- **Passive attacks** attempts to learn or make use of information from the system but does not affect system resources.
- **Active attacks** involve some modification of the data stream or the creation of a false stream .

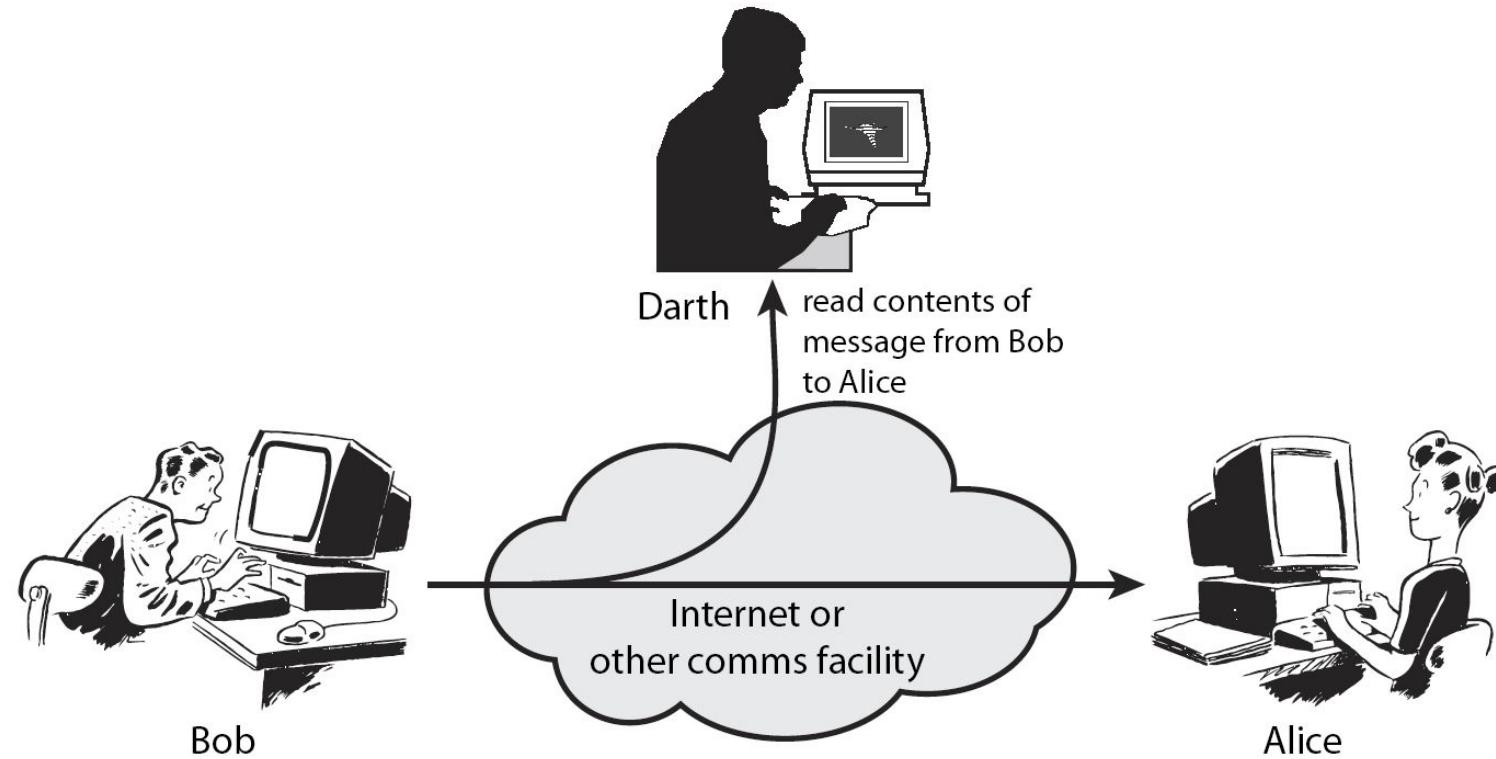
Attacks can also be categorized as inside attack or outside attack.

- An **inside attack** is initiated from within the physical boundary of network by an authorized person
- An **outside attack** is caused by an external entity, an intruder who does not have privilege to access the enterprise network

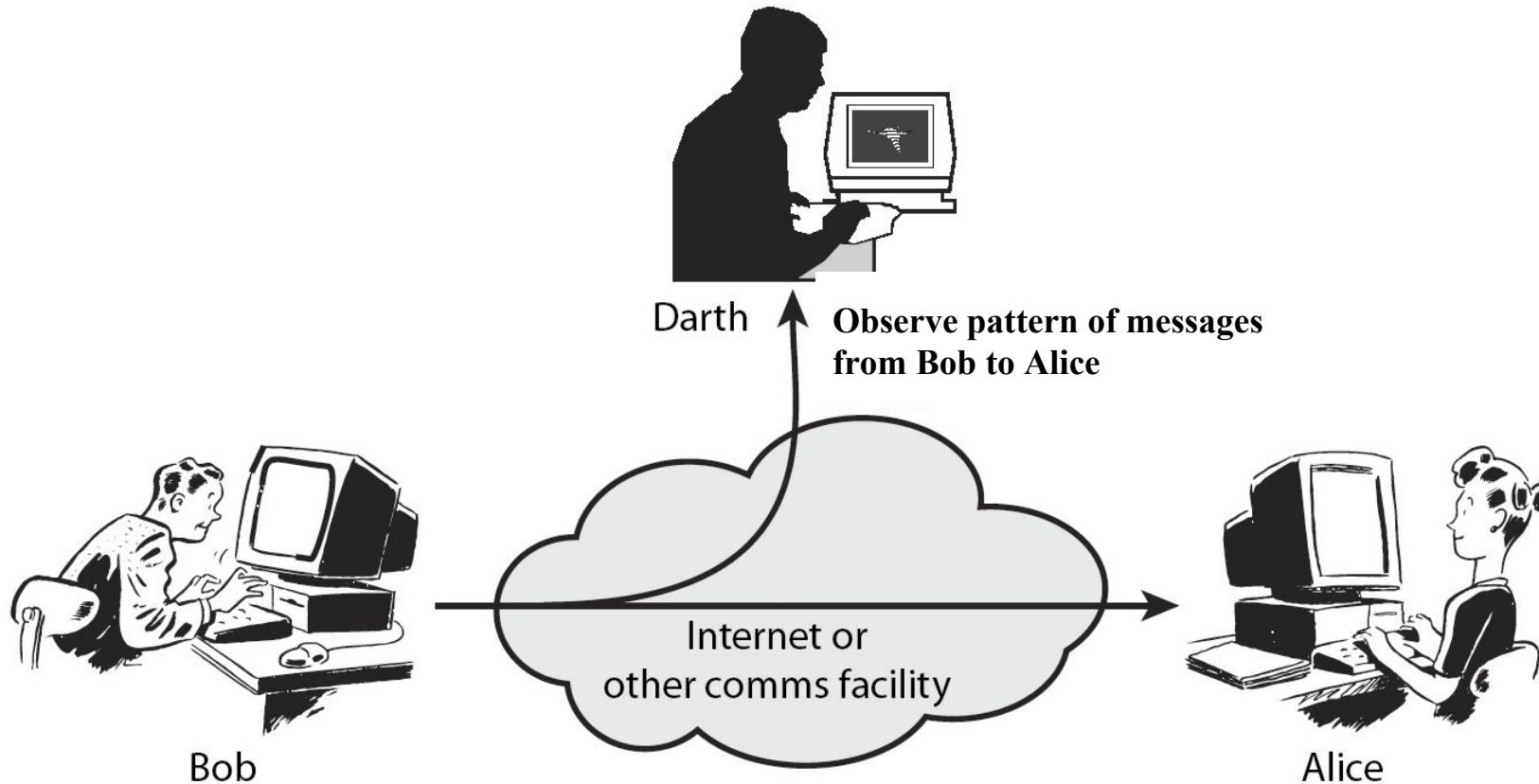
# PASSIVE ATTACKS

- The goal of passive attack is to obtain information that is being transmitted.
- Two types of passive attacks are:
  1. release of message contents
  2. traffic analysis - monitor traffic flow to determine location and identity of communicating hosts and could observe the frequency and length of messages being exchanged
- These attacks are difficult to detect because they do not involve any alteration of the data.

# PASSIVE ATTACK – RELEASE OF MESSAGE CONTENTS



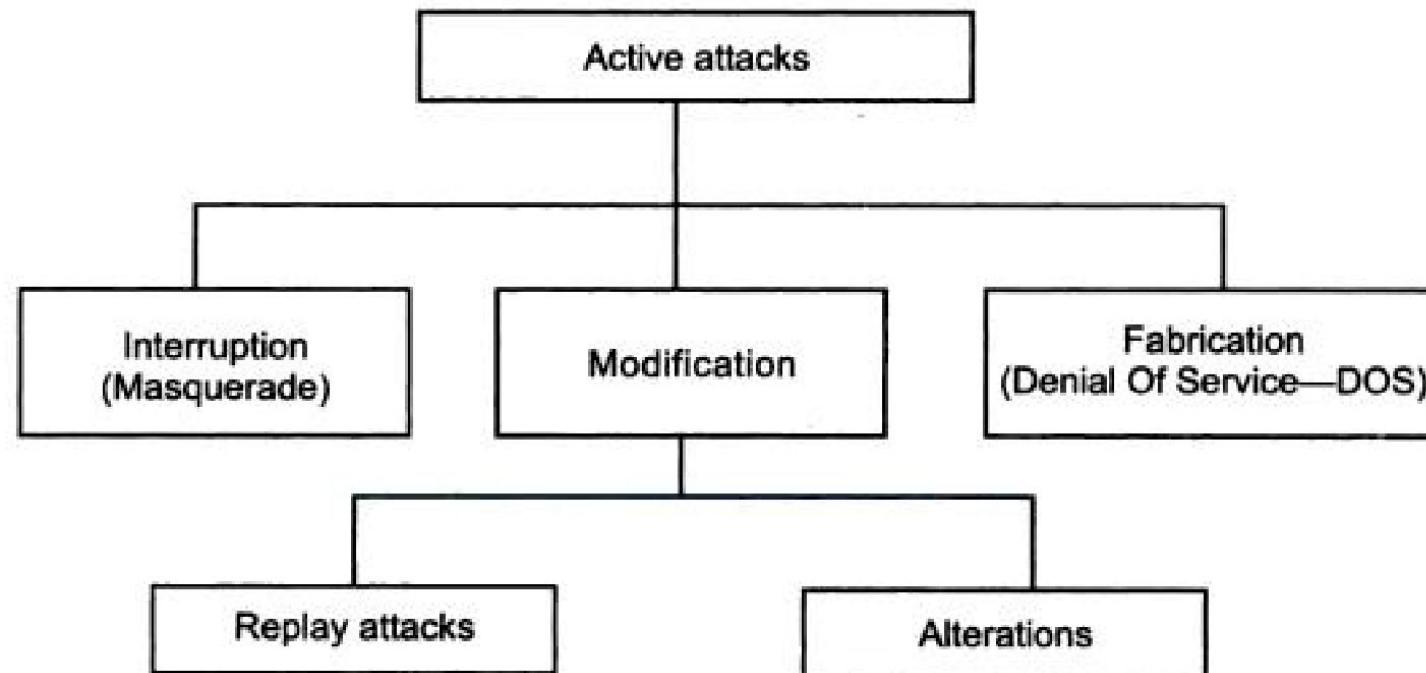
# PASSIVE ATTACK: TRAFFIC ANALYSIS



# ACTIVE ATTACKS

- Four categories of Active attacks :
  1. Masquerade
  2. Replay
  3. Modification of messages
  4. Denial of service

# ACTIVE ATTACKS

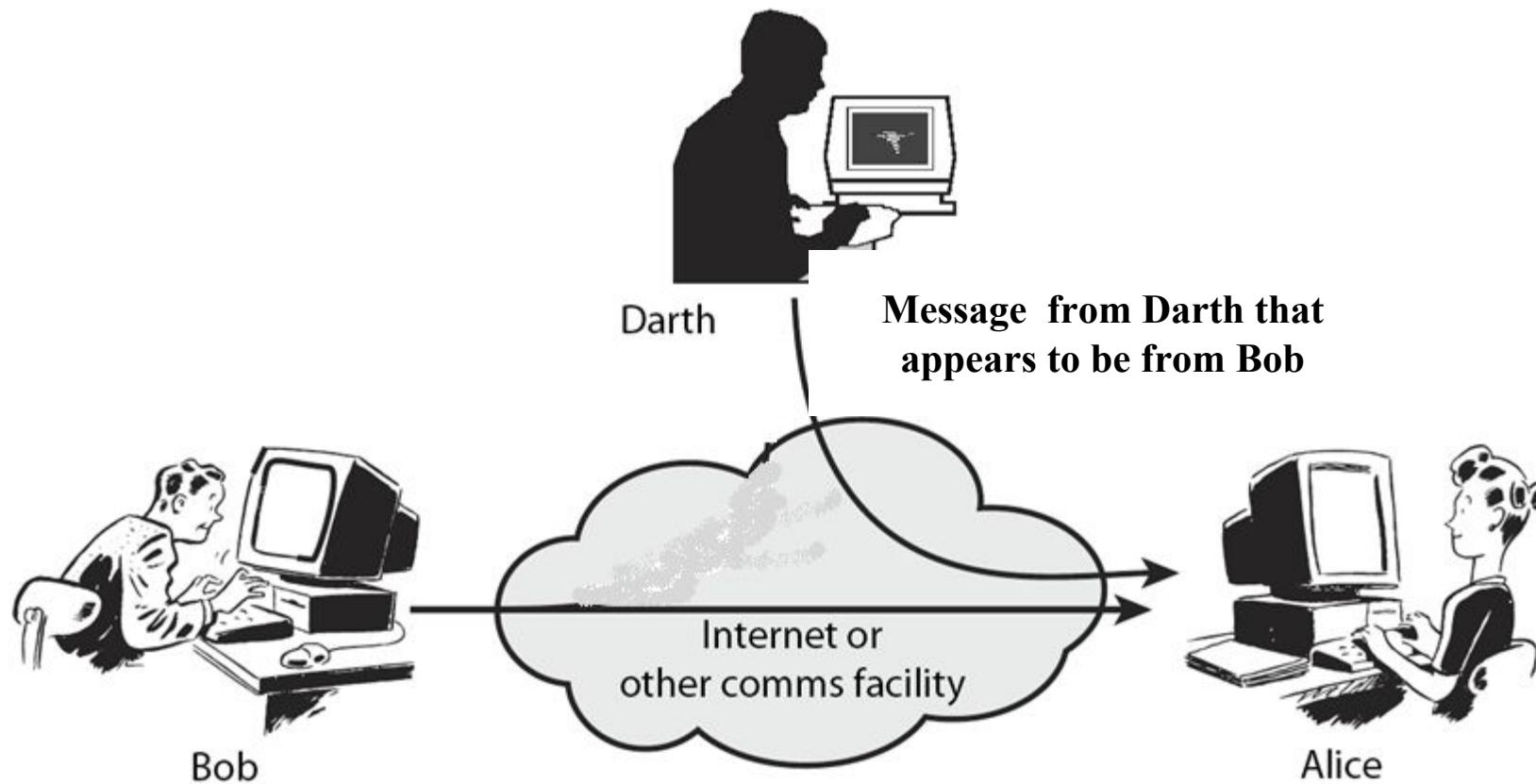


# ACTIVE ATTACKS

## Masquerade :

- one entity pretends to be a different entity.
- **Example:** Authentication sequences can be captured and replayed after a valid authentication sequences has taken place, thus enabling an authorized entity with few privileges to obtain extra privileges by impersonating an entity that has those privileges.

# ACTIVE ATTACK: MASQUERADE



# ACTIVE ATTACKS

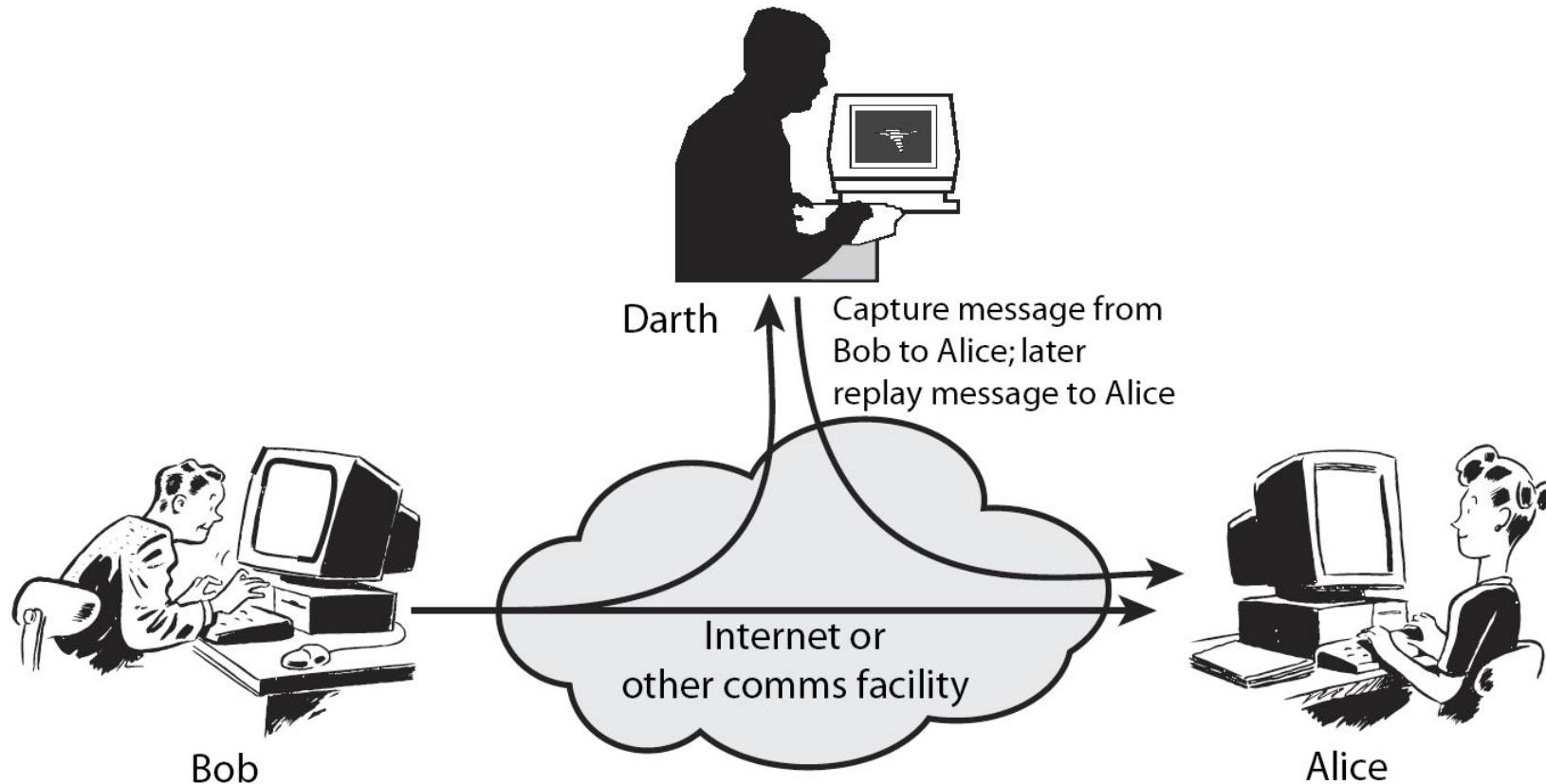
## **Replay :**

- Involves passive capture of data unit and its subsequent retransmission to produce an unauthorized effect.

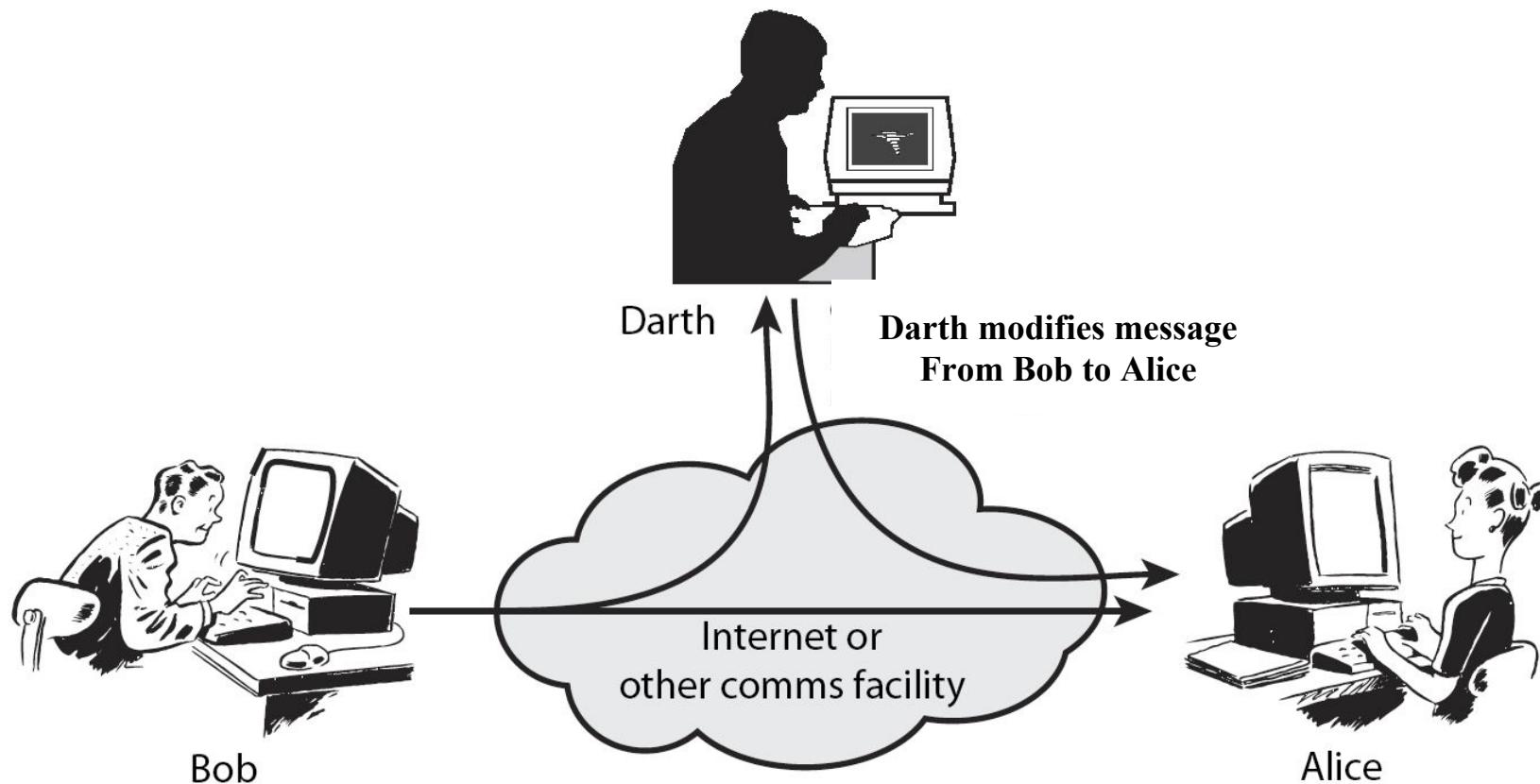
## **Modification of messages:**

- Some portion of legitimate message is altered, or that messages are delayed or recorded, to produce an unauthorized effect.

# ACTIVE ATTACK: REPLAY



# ACTIVE ATTACK: MODIFICATION OF MESSAGES

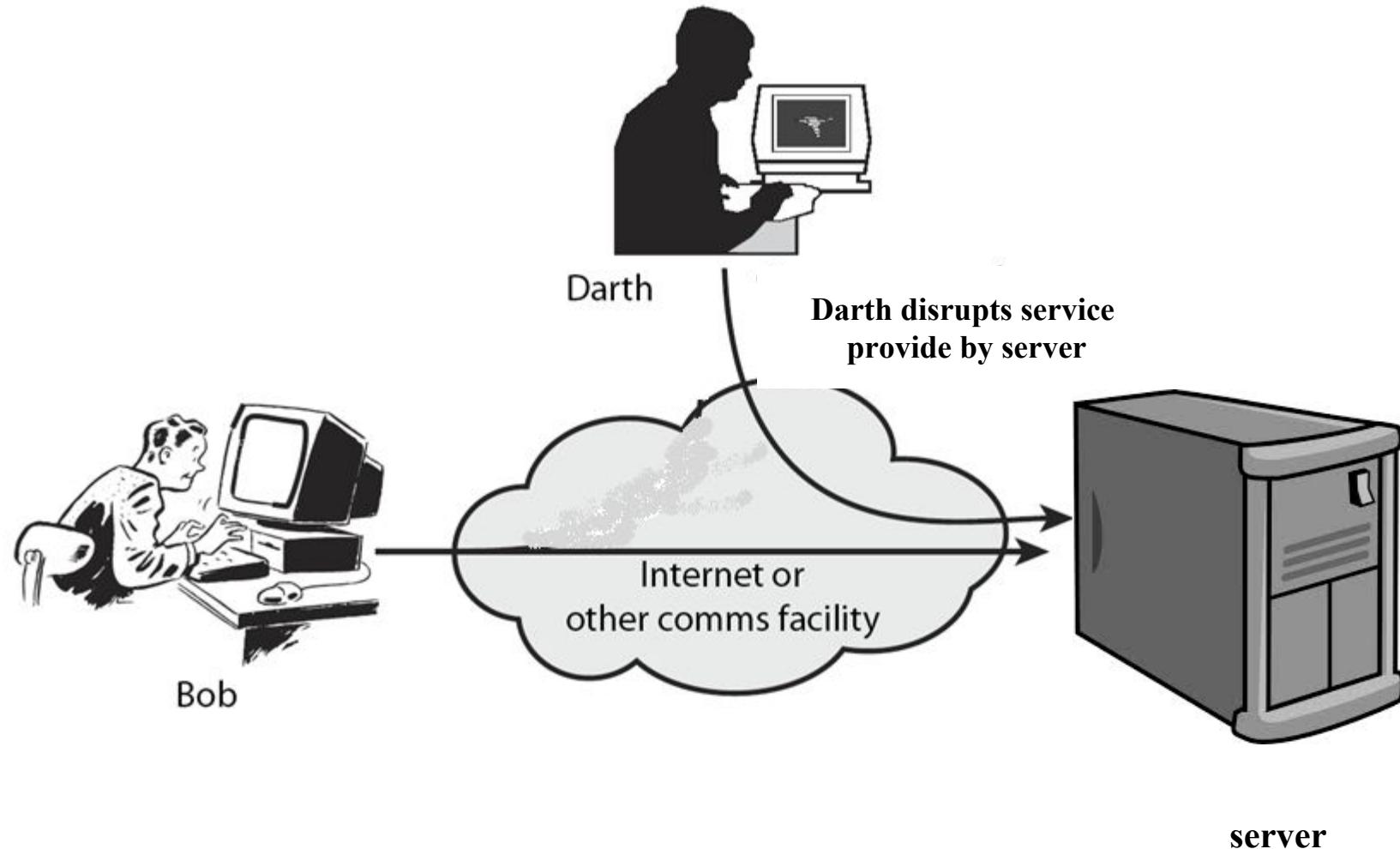


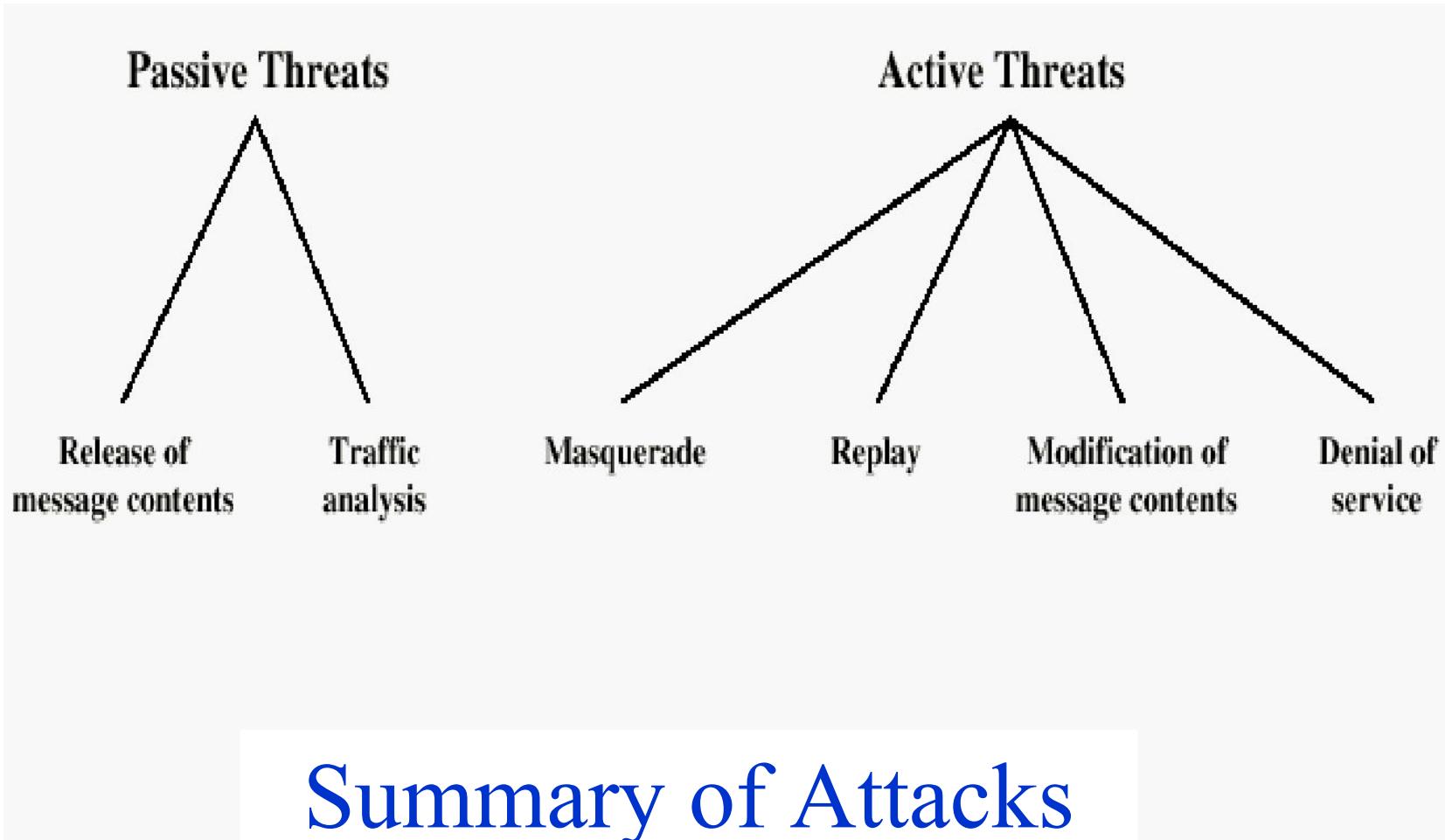
# ACTIVE ATTACKS

## **Denial of service :**

- Prevents the normal use of management of communications facilities. This attack may have specific target.
- Example: an entity may suppress all messages directed to a particular destination, disruption of an entire network either by disabling the network or by overloading it with messages so as to degrade performance.

# ACTIVE ATTACK: DENIAL OF SERVICE

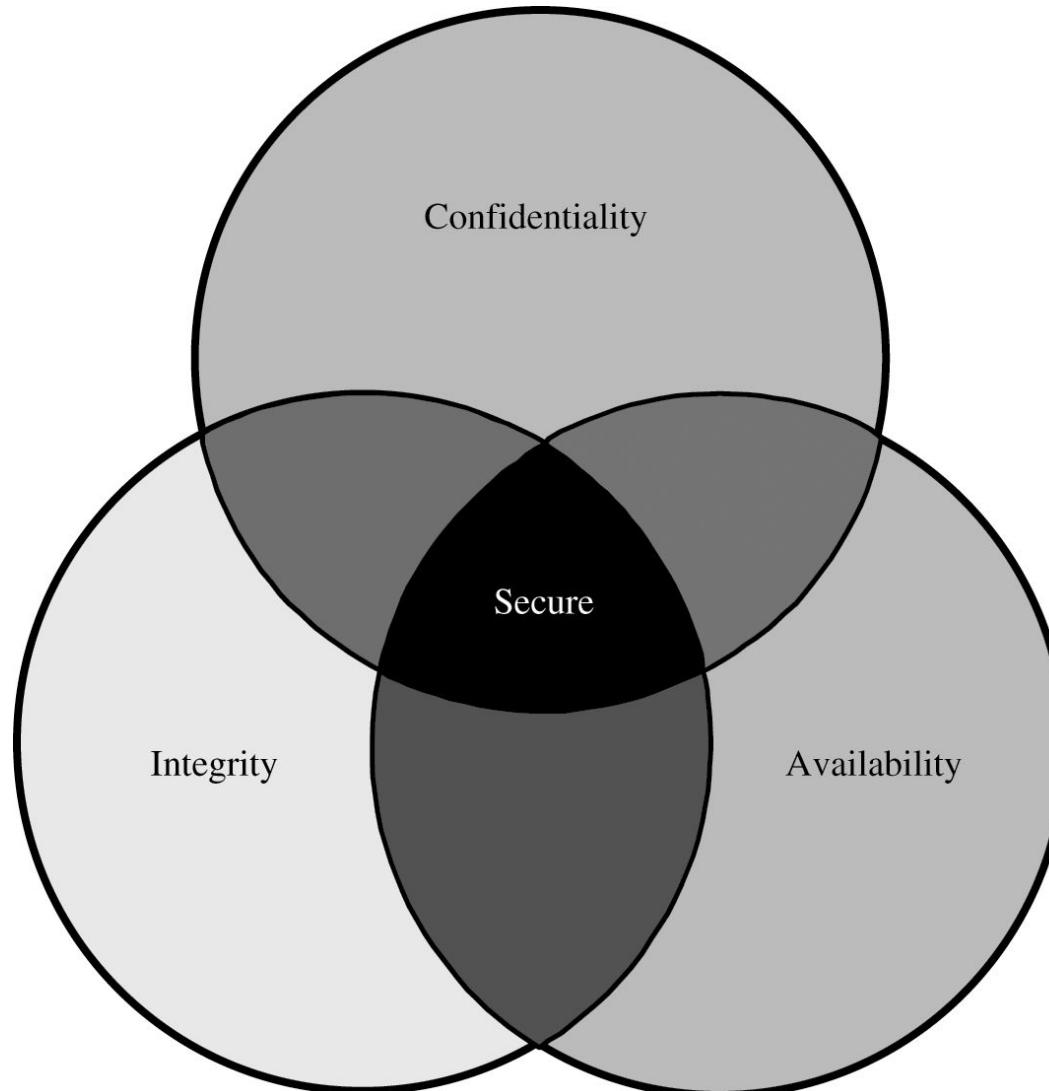




# HANDLING ATTACKS

- Passive attacks – focus on Prevention
  - Easy to stop
  - Hard to detect
- Active attacks – focus on Detection and Recovery
  - Hard to stop
  - Easy to detect

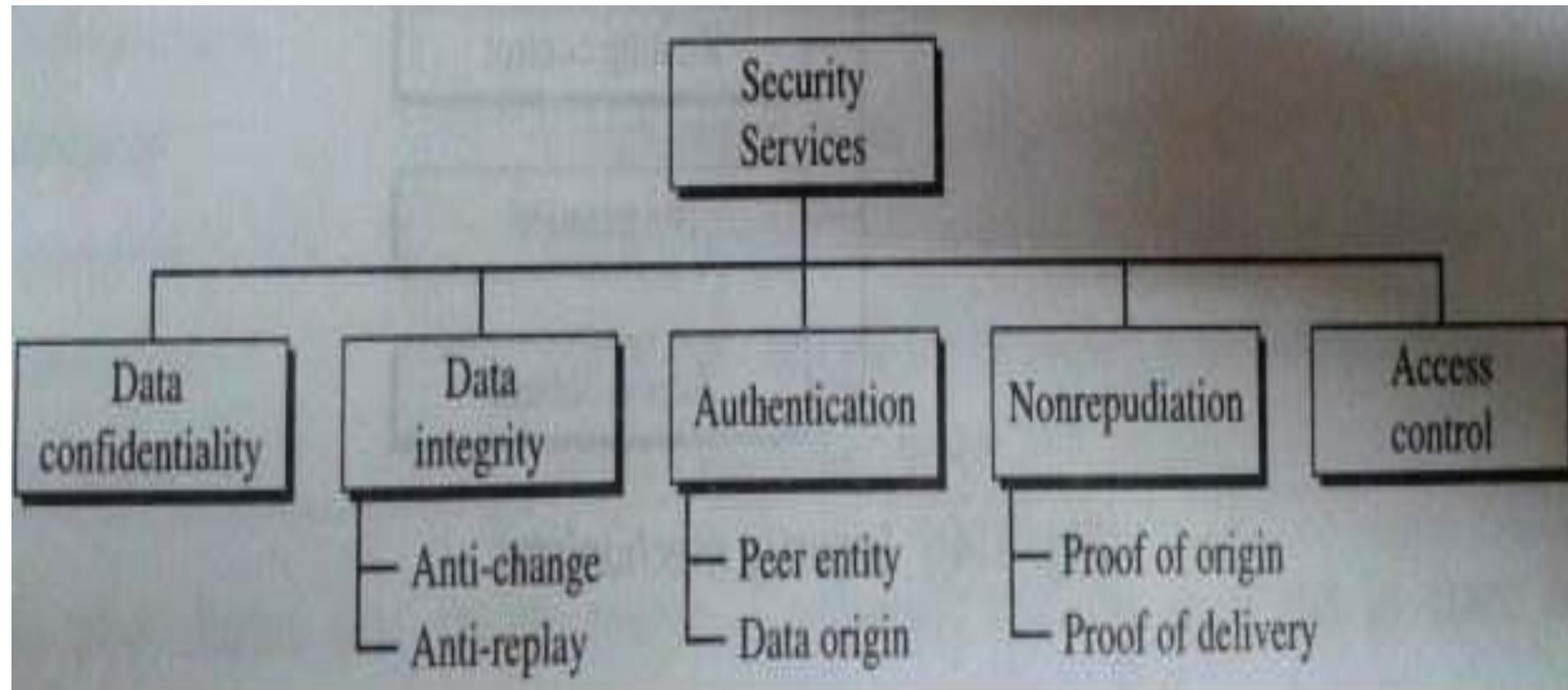
# SECURITY GOALS



# SECURITY GOALS

- Secure is:
  - **Confidentiality** (Secrecy or Privacy)- assets accessed only by authorized parties
    - Not only reading but viewing, printing or knowing about the asset
  - **Integrity** – assets modified only by authorized parties
    - Includes writing, changing, changing the status, deleting or creating
  - **Availability** – assets are accessible to authorized parties at appropriate times.
    - Denial of Service

# SECURITY SERVICES (X.800)



# SECURITY SERVICES (X.800)

- **Data Confidentiality**

- protection of data from unauthorized disclosure.
- It is designed to prevent snooping and traffic analysis attack.

- **Data Integrity**

- assurance that data received is as sent by an authorized entity.
- It is designed to protect and replaying by an adversary.
- It may protect the whole message or a part of message.

# SECURITY SERVICES (X.800)

## • Data Integrity (Five Types)

1. Connection integrity with recovery:
  - Provides integrity to data on a connection.
  - Detects modifications, replay of any data.
  - recovery attempted.
2. Connection integrity without recovery:
  - provides only detection without recovery.
3. Selective field connection integrity:
  - Provides integrity to selected fields in data
4. Connectionless integrity :
  - Provides integrity to connection less data block.
5. Selective field connectionless integrity:

# SECURITY SERVICES (X.800)

- **Authentication**

- assurance that the communicating entity is the one claimed
- **Peer entity authentication:** authentication of sender and receiver in connection oriented communication.
- **Data origin Authentication:** authentication of source of data in connectionless communication

- **Access Control** - prevention of the unauthorized use of a resource

# SECURITY SERVICES (X.800)

- **Non-Repudiation** – prevents either sender or receiver from denying a transmitted message.
  - **Non-repudiation of origin** : Protects against a sender of data **denying** that data was sent.
  - **Non-repudiation of delivery** : Protects against a receiver of data denying that data was received
  -

# SECURITY MECHANISMS

- Security mechanisms provide and support security services.
- Can be divided into two classes:
  - **Specific security mechanisms:**
    - used to provide specific security services.
    - They are incorporated into protocol layer to provide OSI security services.
  - **Pervasive security mechanisms:**
    - Mechanisms that are not specific to any particular OSI security services or protocol layer.

# SPECIFIC SECURITY MECHANISMS

- **Eight types:**

- Encipherment
- Digital signature
- Access control mechanisms
- Data integrity mechanisms
- Authentication exchanges
- Traffic padding
- Routing control
- Notarization

# SPECIFIC MECHANISMS

- **Encipherment mechanisms = encryption algorithms.**
  - Transformation and recovery of data using encryption keys.
- **Digital signature mechanisms**
  - signing procedure (using private key),
  - verification procedure (using public key).
  - Can provide, origin authentication and data integrity services.

# SPECIFIC MECHANISMS

- **Access Control mechanisms**

- Enforce access rights to the resources.
  - E.g. access control lists

- **Data integrity mechanisms**

- Protection against modification of data.

- **Authentication exchange mechanisms**

- Provide entity authentication service.
- provides peer entity authentication and data origin authentication.

# SPECIFIC MECHANISMS

- **Traffic padding mechanisms**
  - The insertion of bits into gaps in a data stream to frustrate traffic analysis attempts.
- **Routing control mechanisms**
  - means selecting and continuously changing different available routes between sender and receiver to prevent the opponent from eavesdropping on particular route.
- **Notarization mechanisms**
  - Integrity, origin and/or destination of data can be guaranteed by using a trusted third party to assure certain properties of data exchange.

# PERVASIVE MECHANISMS

- **Trusted functionality**
  - Any functionality providing or accessing security mechanisms should be trustworthy.
  - May involve combination of software and hardware.
- **Security labels**
  - Any resource (e.g. stored data, processing power, communications bandwidth) may have security label associated with it to indicate security sensitivity.
  - Similarly labels may be associated with users. Labels may need to be securely bound to transferred data.
- **Security audit trail**
  - Log of past security-related events.
  - Permits detection and investigation of past security breaches.

# PERVASIVE MECHANISMS

- **Event detection**

- Includes detection of
  - attempted security violations,
  - legitimate security-related activity.
  - Can be used to trigger event reporting (alarms), event logging, automated recovery.

- **Security recovery**

- Includes mechanisms to handle requests to recover from security failures.
- May include immediate abort of operations, temporary invalidation of an entity, addition of entity to a blacklist.

# RELATIONSHIP BETWEEN SECURITY SERVICES AND MECHANISMS

**Table 1.2** *Relation between security services and security 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 AND AUTHENTICATION

## UNIT 2

# CONTENTS

- Concept of Cryptography
- Symmetric and Asymmetric Cryptography.
- Mathematics of cryptography
- Stream Cipher and Block Cipher
- Concept of Confusion and Diffusion.
- Modes of Operation of Block Cipher
- **Authentication**

# WHAT IS CRYPTOGRAPHY

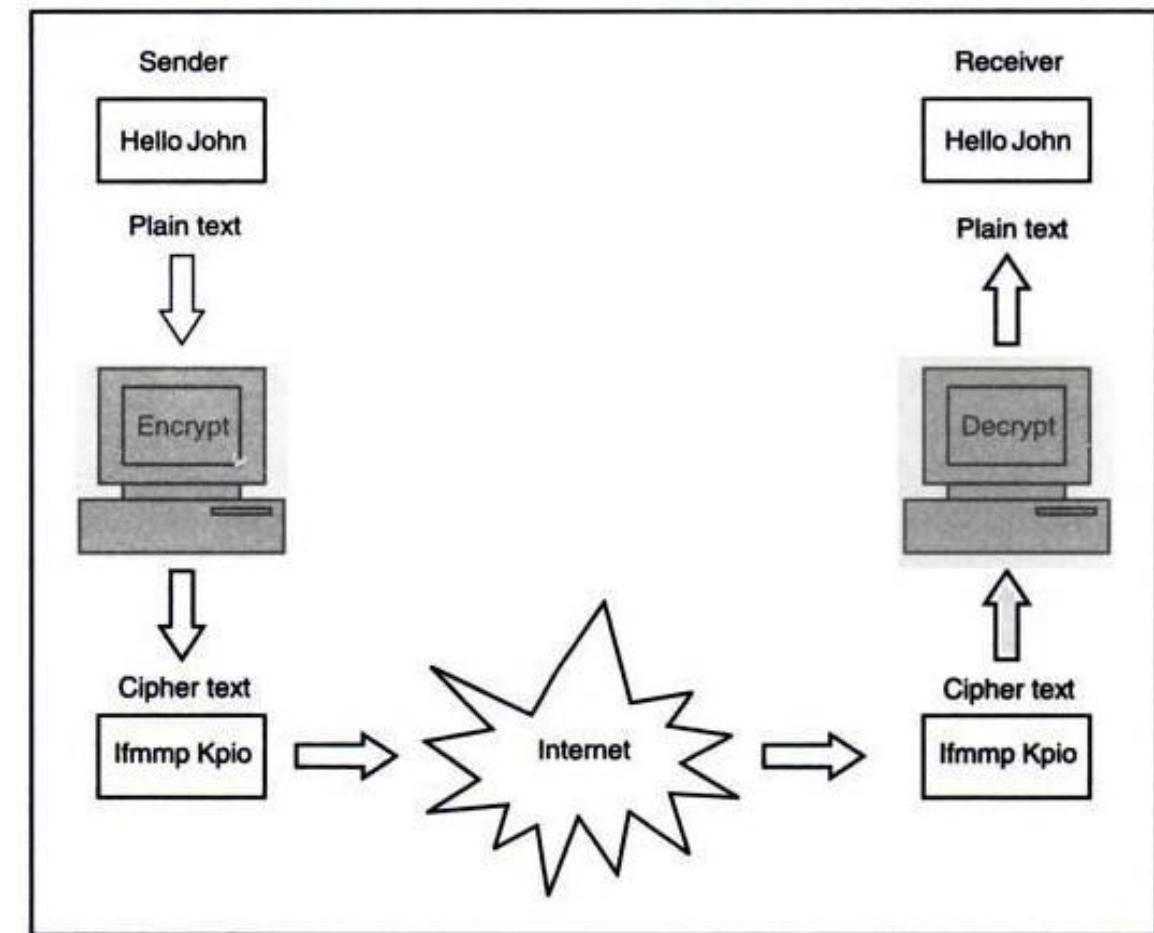
- Cryptography is the art of achieving security by encoding messages to make them non readable
  - process of writing using various methods (“ciphers”) to keep messages secret.
  - Cryptography not only protects data from theft or alteration, but can also be used for user authentication
- Cryptanalysis is the science of attacking ciphers, finding weaknesses, or even proving that a cipher is secure.
- Cryptology covers both; it’s the complete science of secure communication.

# BASIC TERMINOLOGY/NOTATION

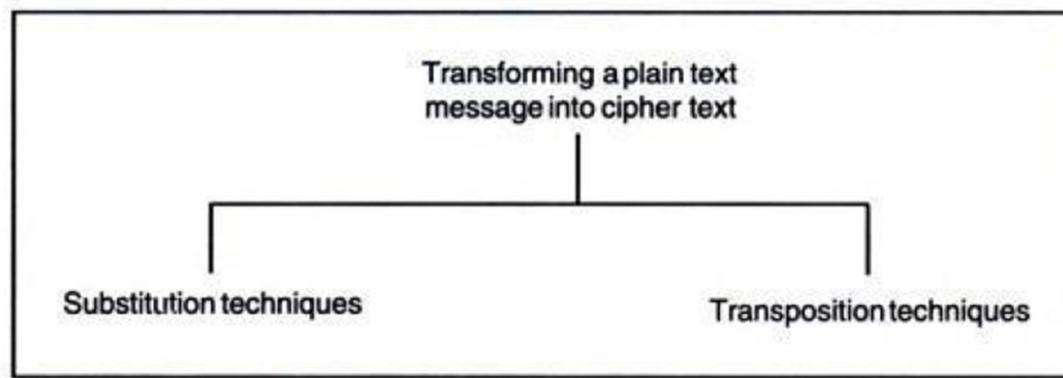
- $P$  is the plaintext. This is the original readable message (written in some standard language, like English, French, Cantonese, Hindi, Icelandic, ...).
- $C$  is the ciphertext. This is the output of some encryption scheme, and is not readable by humans.
- $E$  is the encryption function. We write, for example,  $E(P) = C$  to mean that applying the encryption process  $E$  to the plaintext  $P$  produces the ciphertext  $C$ .
- $D$  is the decryption function, i.e.  $D(C) = P$ .
- Note  $D(E(P)) = P$  and  $E(D(C)) = C$ .

# ENCRYPTION AND DECRYPTION

- Encryption transforms understandable text (plaintext) into an unintelligible piece of data (ciphertext).
- Decryption restores the understandable text from the unintelligible data.
- Both functions involve a mathematical formula (the algorithm) and secret data (the key).



# ENCRYPTION TECHNIQUES



- Substitution technique is a classical encryption technique where the characters present in the original message are replaced by the other characters or numbers or by symbols.
- In transposition Cipher Technique, plain text characters are rearranged with respect to the position.

# CAESAR CIPHER

- earliest known substitution cipher
- by Julius Caesar
- first attested use in military affairs
- replaces each letter by a letter **three** places down the alphabet
- example:

meet me after the toga party  
PHHW PH DIWHU WKH WRJD SDUWB
- Modified version of ceaser cipher-

- **The formula of encryption is:**

- $E_n(x) = (x + n) \text{ mod } 26$

- **The formula of decryption is:**

- $D_n(x) = (x - n) \text{ mod } 26$

E denotes the encryption

D denotes the decryption

x denotes the letters value

n denotes the key value (shift value)

Plaintext: J → 09

$E_n: (09 + 3) \text{ mod } 26$

Ciphertext: 12 → M

# CRYPTANALYSIS OF CAESAR CIPHER

- only have 26 possible keys
  - Could shift  $K = 0, 1, 2, \dots, 25$  slots
- could simply try each in turn
- a **brute force search**
- given ciphertext, just try all shifts of letters
- do need to recognize when have plaintext
- Test:break ciphertext

GCUA VQ DTGCM

## MONOALPHABETIC CIPHER

- rather than just shifting the alphabet
- could shuffle the letters arbitrarily
- each plaintext letter maps to a different random ciphertext letter
- hence key is 26 letters long

Plain: abcdefghijklmnopqrstuvwxyz

Cipher: DKVQFIBJWPESCXHTMYAUOLRGZN

Plaintext: ifewewishtoreplaceletters

Ciphertext: WIRFRWAJUHYFTSDVFSUUUFYA

# MONOALPHABETIC CIPHER SECURITY

- now have a total of  $26! = 4 \times 10^{26}$  keys
- with so many keys, might think is secure
  - The simplicity and strength of the monoalphabetic substitution cipher dominated for the first millenium AD.
- but would be !!!**WRONG!!!**
  - First broken by Arabic scientists in 9<sup>th</sup> century

# SUBSTITUTION CIPHER

- **Homophonic substitution cipher** -replacing each letter with a variety of substitutes, the number of potential substitutes being proportional to the frequency of the letter.
- For ex:A can be replaced with D,H,P,R ; B can be replaced with E,I,Q,S etc.
- **Polygram Substitution Cipher**- replaces one block of plain text with a block of cipher text -it does not work on a character by character basis.(**playfair cipher**)
- Ex: HELLO -> YUQQW , HELL -> TEUI
- **Poly alphabetic Substitution Cipher**- cipher alphabet for the plain alphabet may be different at different places during the encryption process (**Vigenere Cipher**)

# HOMOPHONIC SUBSTITUTION CIPHER

Plaintext: HE EATS THEN SLEEPS AND DREAMS      Ciphertext: ??

Key: The following table is the key.

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
11	21	89	58	QP	15	JK	55	BC	B	47	PB	AA	49	50	ZS	A	43	JK	J	90	76	CT	93	30	13
AZ	12	ZA	ND	69	RF	FR	OG	61	71	GC	VC	CG	BB	49	SC	SP	CR	87	77	QQ	VM	59	HR	XT	NE
@		47	??			81												XX	WO						
QA			99			SS												88							
XF			101																						
			DD																						
			YD																						

Homophonic Cipher Table

Ciphertext:

55QP69 11JK 77OG?? 4987PB 99101ZS XXAZBB 58ND43 DD@AA 88

# PLAYFAIR CIPHER

- In Playfair cipher, initially a key table is created. The key table is a  $5 \times 5$  grid of alphabets that acts as the key for encrypting the plaintext.
- The sender and the receiver decide on a particular key, say ‘tutorials’.
- Each of the 25 alphabets must be unique and one letter of the alphabet (usually J) is omitted from the table as we need only 25 alphabets instead of 26. If the plaintext contains J, then it is replaced by I.
- We want to encrypt the message “hide money”. It will be written as –  
**HI DE MO NEYZ**
- The rules of encryption are –
- If both the letters are in the same column, take the letter below each one (going back to the top if at the bottom)
- If both letters are in the same row, take the letter to the right of each one (going back to the left if at the farthest right)
- If neither of the preceding two rules are true, form a rectangle with the two letters and take the letters on the horizontal opposite corner of the rectangle.

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

# PLAYFAIR CIPHER

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

'H' and 'I' are in same column, hence take letter below them to replace.  
HI → QC

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

'M' and 'O' nor on same column or same row,  
hence form rectangle as shown, and replace letter  
by picking up opposite corner letter on same row  
MO → NU

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

'D' and 'E' are in same row, hence take letter to the right of them to  
replace. DE → EF

Cipher text generated: QC EF NU MF ZV

# VIGENERE CIPHER

- let's assume the key is 'point'. Each alphabet of the key is converted to its respective numeric value: In this case,
- $p \rightarrow 16, o \rightarrow 15, i \rightarrow 9, n \rightarrow 14$ , and  $t \rightarrow 20$ .
- Thus, the key is: 16 15 9 14 20.

a	t	t	a	c	k	f	r	o	m	s	o	u	t	h	e	a	s	t
16	15	9	14	20	16	15	9	14	20	16	15	9	14	20	16	15	9	14

- The sender wants to encrypt the message, say 'attack from south east'. He will arrange plaintext and numeric key as follows
- He now shifts each plaintext alphabet by the number written below it to create ciphertext as shown –

a	t	t	a	c	k	f	r	o	m	s	o	u	t	h	e	a	s	t
16	15	9	14	20	16	15	9	14	20	16	15	9	14	20	16	15	9	14
Q	I	C	O	W	A	U	A	C	G	I	D	D	H	B	U	P	B	H

Here, each plaintext character has been shifted by a different amount – and that amount is determined by the key. The key must be less than or equal to the size of the message.

- For decryption, the receiver uses the same key and shifts received ciphertext in reverse order to obtain the plaintext.

Q	I	C	O	W	A	U	A	C	G	I	D	D	H	B	U	P	B	H
16	15	9	14	20	16	15	9	14	20	16	15	9	14	20	16	15	9	14
a	t	t	a	c	k	f	r	o	m	s	o	u	t	h	e	a	s	t

# TRANSPOSITION TECHNIQUE

- Rail Fence technique – Involves writing plain text as sequence of diagonals and then reading it row by row to produce cipher text.
  - "defend the east wall", with a key of 3.
- |   |   |   |   |   |   |   |   |   |  |  |
|---|---|---|---|---|---|---|---|---|--|--|
| D |   | N |   | E |   | T |   | L |  |  |
| E | E | D | H | E | S | W | L | X |  |  |
| F |   | T |   | A |   | A |   | X |  |  |
- The Rail Fence Cipher with a key of 3. Notice the nulls added at the end of the message to make it the right length.
  - The ciphertext is read off row by row to get "DNETLEEDHESWLXFTAAX".

## SIMPLE COLUMNAR TRANSPOSITION-

- Simple columnar transposition- writing the plaintext out in rows, and then reading the ciphertext off in columns

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

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

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

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

## SIMPLE COLUMNAR TRANSPOSITION WITH MULTIPLE ROUNDS

encrypt the message "The tomato is a plant in the nightshade family" using the keyword tomato

T	O	M	A	T	O
5	3	2	1	6	4
T	H	E	T	O	M
A	T	O	I	S	A
P	L	A	N	T	I
N	T	H	E	N	I
G	H	T	S	H	A
D	E	F	A	M	I
L	Y	X	X	X	X

T	O	M	A	T	O
5	3	2	1	6	4
T	I	N	E	S	A
X	E	O	A	H	T
F	X	H	T	L	T
H	E	Y	M	A	I
I	A	I	X	T	A
P	N	G	D	L	O
S	T	N	H	M	X

We put the plaintext into the grid below the keyword tomato to get the ciphertext "TINES AXEOA HTFXH TLTHE YMAII AIXTA PNGDL OSTNH MX"

We now write the ciphertext retrieved from the grid to the left in the grid in rows as if it was the plaintext.

After this double transposition, we get the ciphertext "EATMX DHNOH YIGNI EXEAN TATTI AOXTX FHIPS SHLAT LM".

## VERNAM CIPHER

- The Vernam Cipher, also called as One time Pad is implemented using a random set of non-repeating characters as the input cipher text.
- One time pad is discarded after single use. So is suitable for short messages.

1. Treat each plain text alphabet as a number in an increasing sequence, i.e. A = 0, B = 1, ... Z = 25.
2. Do the same for each character of the input cipher text.
3. Add each number corresponding to the plain text alphabet to the corresponding input cipher text alphabet number.
4. If the sum thus produced is greater than 26, subtract 26 from it.
5. Translate each number of the sum back to the corresponding alphabet. This gives the output cipher text.

## VERNAME CIPHER

1. Plain text	H 7	O 14	W 22	A 0	R 17	E 4	Y 24	O 14	U 20
+									
2. One-time pad	13 N	2 C	1 B	19 T	25 Z	16 Q	0 A	17 R	23 X
3. Initial Total	20	16	23	19	42	20	24	30	33
4. Subtract 26, if > 25	20	16	23	19	16	20	24	5	17
5. Cipher text	U	Q	X	T	Q	U	Y	F	R

Fig. 2.17 Example of vernam cipher

## BOOK CIPHER/RUNNING KEY CIPHER

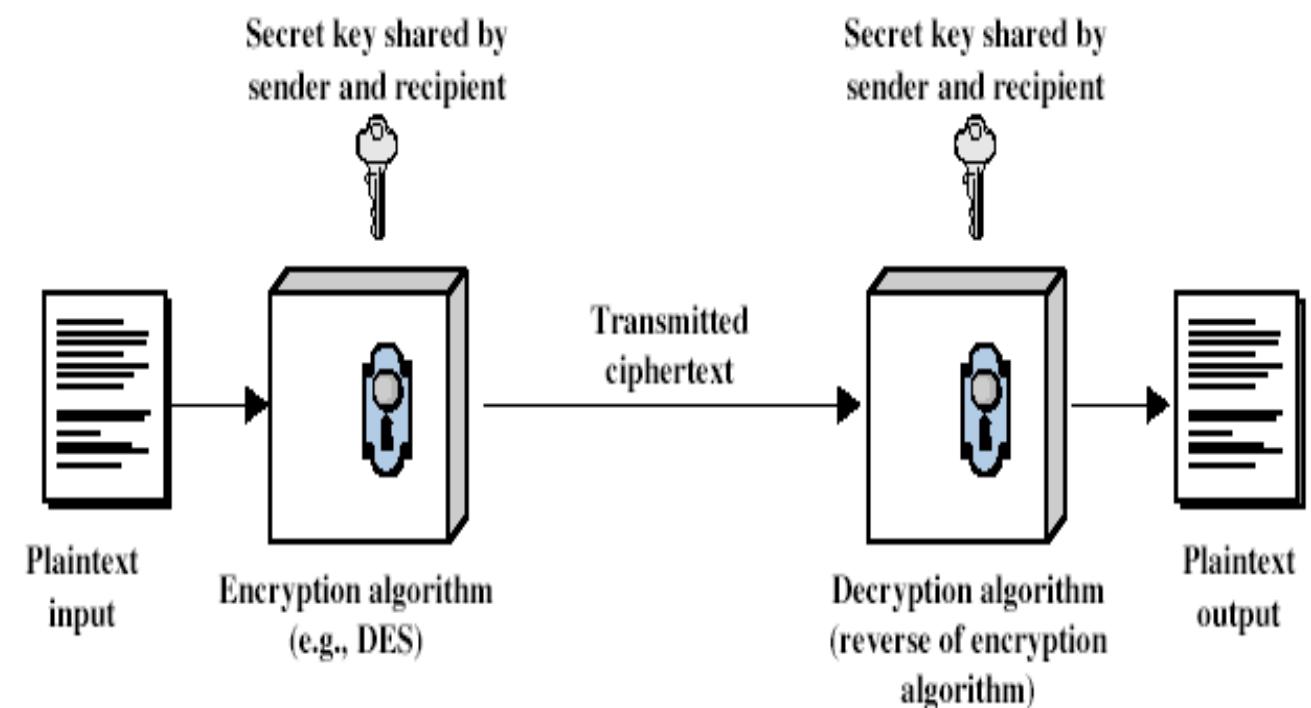
- Some portion of text from a book is used , which serves the purpose of a one time pad.
- Works similar to Vernam cipher.

# CRYPTOGRAPHIC TECHNIQUES

- *Three types of cryptographic techniques used in general.*
  1. Symmetric-key cryptography
  2. Hash functions.
  3. Asymmetric-key cryptography

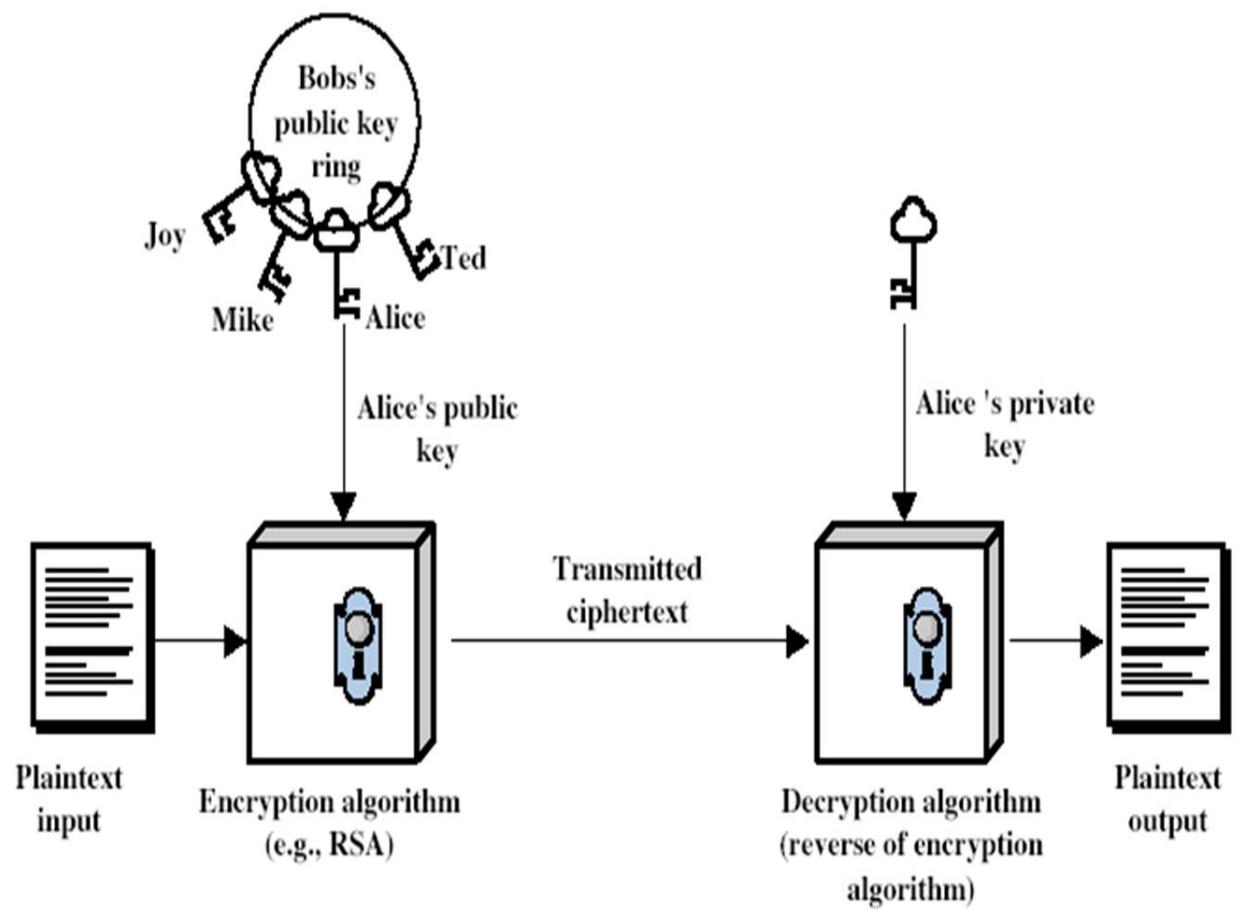
# PRIVATE-KEY/SYMMETRIC KEY CRYPTOGRAPHY

- Traditional **private/secret/single key** cryptography uses **one key**
- Both parties share the same key (which is kept secret).
- Before communications begin, both parties must exchange the shared secret key.
- Each pair of communicating entities requires a unique shared key.
- The key is not shared with other communication partners.
- If this key is disclosed communications are compromised
- Also is **symmetric**, parties are equal
- Hence does not protect sender from receiver forging a message & claiming is sent by sender



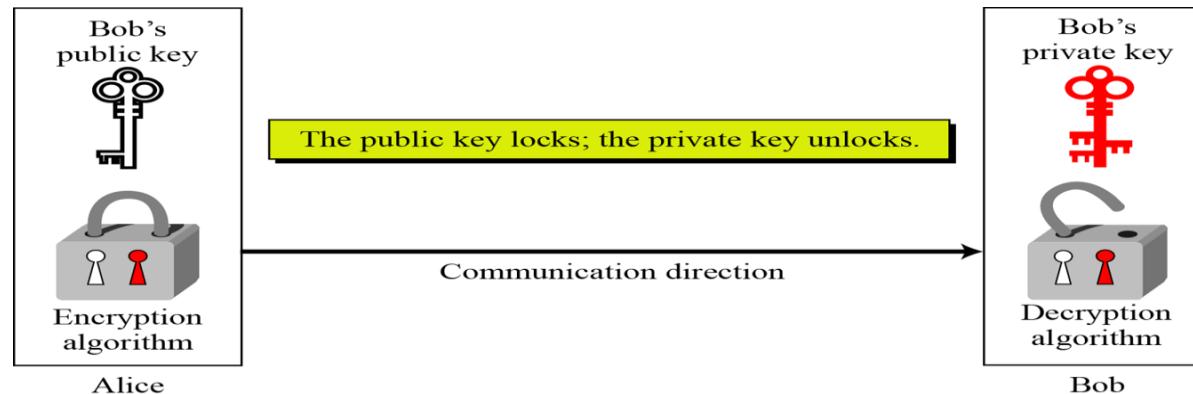
# PUBLIC-KEY/ASYMMETRIC KEY CRYPTOGRAPHY

- Probably most significant advance in the 3000 year history of cryptography
- Uses **two** keys – a public & a private key
- **Asymmetric** since parties are **not** equal
- Uses clever application of number theoretic concepts to function
- Complements **rather than** replaces private key crypto



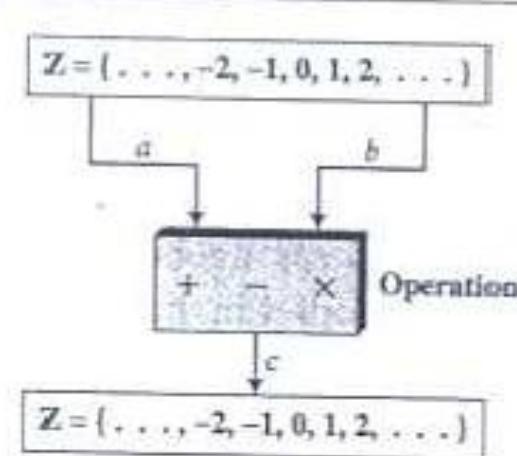
# PUBLIC-KEY CRYPTOGRAPHY

- **public-key/two-key/asymmetric** cryptography involves the use of **two keys**:
  - a **public-key**, which may be known by anybody, and can be used to **encrypt messages**, and **verify signatures**
  - a **private-key**, known only to the recipient, used to **decrypt messages**, and **sign (create) signatures**
- is **asymmetric** because
  - those who encrypt messages or verify signatures **cannot** decrypt messages or create signatures



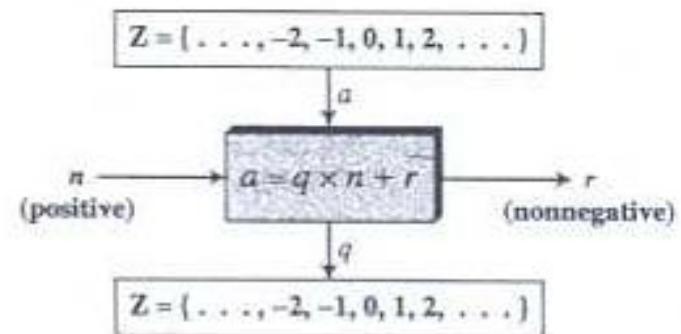
# MATHEMATICS OF CRYPTOGRAPHY

- Cryptography is based on some specific areas of mathematics, including number theory, linear algebra and algebraic structures.
- Set of Integers denoted by  $Z$  contains all integral numbers (with no fraction) from negative infinity to positive infinity
- $Z=\{\dots, -2, -1, 0, 1, 2, \dots\}$
- Binary operations on  $Z$
- Addition, subtraction and multiplication



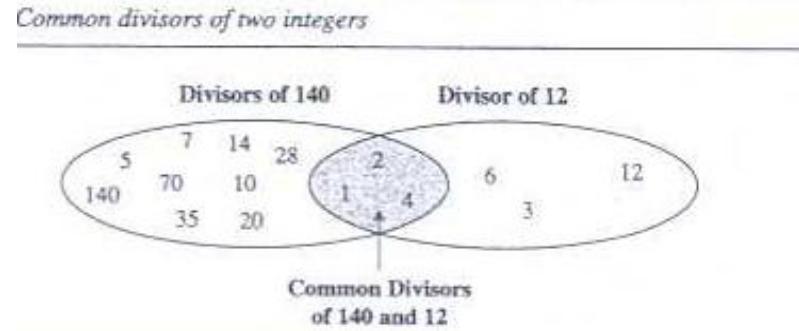
# INTEGER DIVISION

- If we divide  $a$  by  $n$  we get  $q$  and  $r : a = q \times n + r$
- Division relation is used in cryptography with 2 restrictions: divisor  $n$  be a positive integer ( $n > 0$ )
- Remainder be a non negative integer ( $r \geq 0$ )



When we use a computer or a calculator,  $r$  and  $q$  are negative when  $a$  is negative. How can we apply the restriction that  $r$  needs to be positive? The solution is simple, we decrement the value of  $q$  by 1 and we add the value of  $n$  to  $r$  to make it positive.

- **Divisibilty:** n divides a or a divisible by n is represented as  $n | a$ . Not divisible is represented as  $n \nmid a$
- **Greatest common divisor(GCD)-**

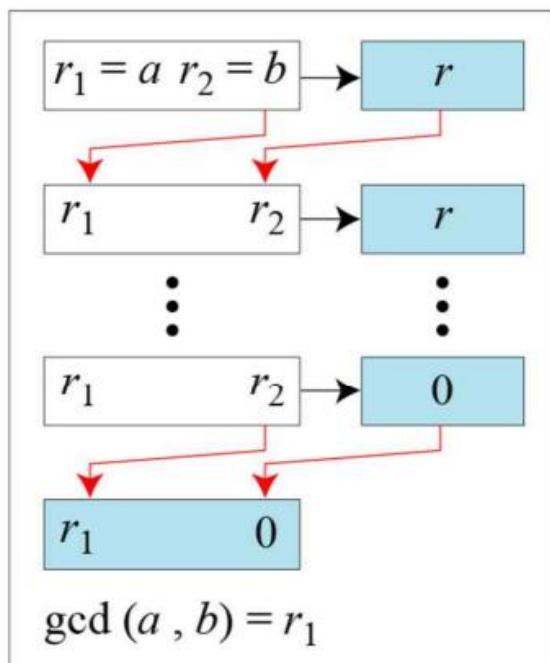


- Euclidian algorithm: Developed by Euclid for finding the GCD of large integers
- Fact 1 :  $\gcd(a,0)=a$
- Fact 2:  $\gcd(a,b)=\gcd(b,r)$ , where r is the remainder of dividing a by b

Eg:  $\gcd(36,10)=\gcd(10,6)=\gcd(6,4)=\gcd(4,2)=\gcd(2,0)=2$

## EUCLIDIAN ALGORITHM:

$\text{GCD}(a, b)$  = the greatest common divisor of integers  $a$  and  $b$ .



a. Process

```
r1 ← a;      r2 ← b;      (Initialization)  
while (r2 > 0)  
{  
    q ← r1 / r2;  
    r ← r1 - q × r2;  
    r1 ← r2;  r2 ← r;  
}  
gcd(a, b) ← r1
```

b. Algorithm

## EXAMPLE I

- Find the greatest common divisor of 2740 and 1760.

$q$	$r_1$	$r_2$	$r$
1	2740	1760	980
1	1760	980	780
1	980	780	200
3	780	200	180
1	200	180	20
9	180	20	0
	<b>20</b>	0	

We have  $\gcd(2740, 1760) = 20$ .

## EXAMPLE 2

- Find the greatest common divisor of 25 and 60.

$q$	$r_1$	$r_2$	$r$
0	25	60	25
2	60	25	10
2	25	10	5
2	10	5	0
	<b>5</b>	0	

We have  $\gcd(25, 60) = 5$ .

## EXTENDED EUCLID ALGORITHM

- Given two integers  $a$  and  $b$ , we often need to find other two integers,  $s$  and  $t$ , such that

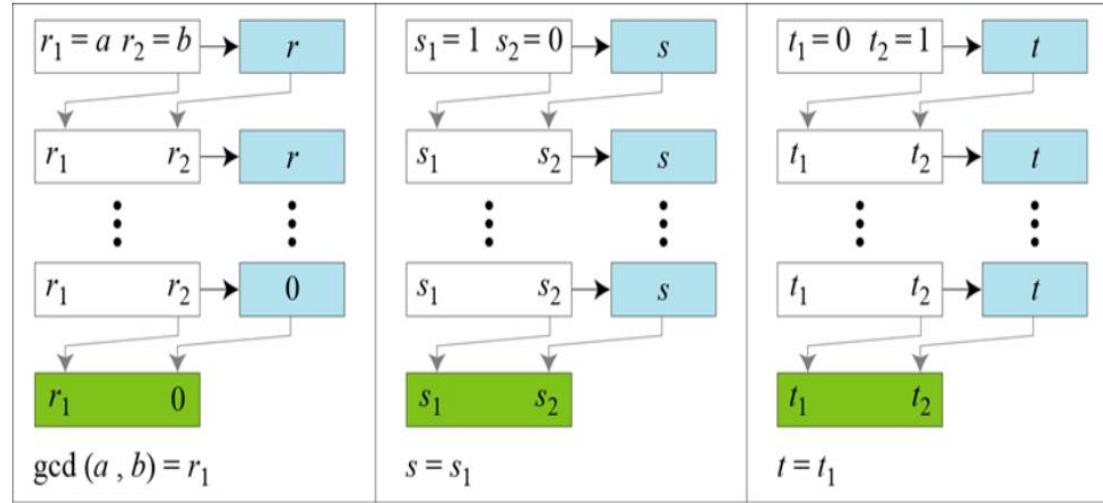
$$s \times a + t \times b = \gcd(a, b)$$

Example :  $\gcd(161, 28) = 7$ ,  $s = -1$  and  $t = 6$ .

$$(-1) \times 161 + 6 \times 28 = 7$$

- The extended Euclidean algorithm can calculate the  $\gcd(a, b)$  and at the same time calculate the value of  $s$  and  $t$ .

**where  $r = r_1 - q \times r_2$ ,  $s = s_1 - q \times s_2$ ,  $t = t_1 - q \times t_2$  in every step.**



a. Process

### Extended Euclidean algorithm

```

 $r_1 \leftarrow a; \quad r_2 \leftarrow b;$ 
 $s_1 \leftarrow 1; \quad s_2 \leftarrow 0;$ 
 $t_1 \leftarrow 0; \quad t_2 \leftarrow 1;$  (Initialization)

while ( $r_2 > 0$ )
{
     $q \leftarrow r_1 / r_2;$ 
     $r \leftarrow r_1 - q \times r_2;$ 
     $r_1 \leftarrow r_2; \quad r_2 \leftarrow r;$  (Updating  $r$ 's)

     $s \leftarrow s_1 - q \times s_2;$ 
     $s_1 \leftarrow s_2; \quad s_2 \leftarrow s;$  (Updating  $s$ 's)

     $t \leftarrow t_1 - q \times t_2;$ 
     $t_1 \leftarrow t_2; \quad t_2 \leftarrow t;$  (Updating  $t$ 's)
}

 $\text{gcd}(a, b) \leftarrow r_1; \quad s \leftarrow s_1; \quad t \leftarrow t_1$ 

```

b. Algorithm

## EXAMPLE 1

- Given  $a = 161$  and  $b = 28$ , find  $\gcd(a, b)$  and the values of  $s$  and  $t$ .

$q$	$r_1$	$r_2$	$r$	$s_1$	$s_2$	$s$	$t_1$	$t_2$	$t$
5	161	28	21	1	0	1	0	1	-5
1	28	21	7	0	1	-1	1	-5	6
3	21	7	0	1	-1	4	-5	6	-23
	7	0		-1	4		6	-23	

$i$	$r_i$	$q_i$	$s_i$	$t_i$
0	161	1	0	0
1	28	5	0	1
2	21	1	1	-5
3	7	3	-1	6
4	0			

We get  $\gcd(161, 28) = 7$ ,  $s = -1$  and  $t = 6$ .

$$s \times a + t \times b = \gcd(a, b)$$

## EXAMPLES

Given  $a=17$   $b=0$ , find the  $\text{gcd}(a,b)$  and the values of  $s$  and  $t$ .

q	r1	r2	r	s1	s2	s	t1	t2	t
	17	0		1	0		0	1	

- $\text{gcd}=17, s=1, t=0$

Given  $a=0$   $b=45$ , find the  $\text{gcd}(a,b)$  and the values of  $s$  and  $t$ .

q	r1	r2	r	s1	s2	s	t1	t2	t
0	0	45	0	1	0	1	0	1	0
	45	0		0	1		1	0	

$\text{gcd}=45, s=0, t=1$

$\text{Gcd}(291,42)$  and also find  $s$  and  $t$  - HW

# MODULAR ARITHMETIC

- In modular arithmetic we are only interested in only one of the outputs ie.  $r$ .  $a \equiv q \times n + r$
- The modulo operator is shown as mod.  
The second input ( $n$ ) is called the modulus.  
The output  $r$  is called the residue.
- Set of Residues**
- The modulo operation creates a set, which in modular arithmetic is referred to as the set of least residues modulo  $n$ , or  $Z_n$ .  $Z_n = \{0, 1, 2, 3, \dots, n-1\}$ ,  $Z_2 = \{0, 1\}$ ,  $Z_6 = \{0, 1, 2, 3, 4, 5\}$  etc.

Find the result of the following operations:

- a.  $27 \bmod 5$
- b.  $36 \bmod 12$
- c.  $-18 \bmod 14$
- d.  $-7 \bmod 10$

$$Z_n = \{0, 1, 2, 3, \dots, (n-1)\}$$

$$Z_2 = \{0, 1\} \quad Z_6 = \{0, 1, 2, 3, 4, 5\} \quad Z_{11} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$$

## Congruence

- Two integers are called congruent modulo  $n$  if they have the same residue for the modulus  $n$ .  
To show that two integers are congruent, we use the congruence operator ( $\equiv$ ). Eg:  $2 \bmod 10 \equiv 2$ ,  $12 \bmod 10 \equiv 2$ ,  $22 \bmod 10 \equiv 2$  etc.. So integers like 2, 12, 22 are called congruent mod 10 .

$$2 \equiv 12 \pmod{10}$$

$$3 \equiv 8 \pmod{5}$$

$$13 \equiv 23 \pmod{10}$$

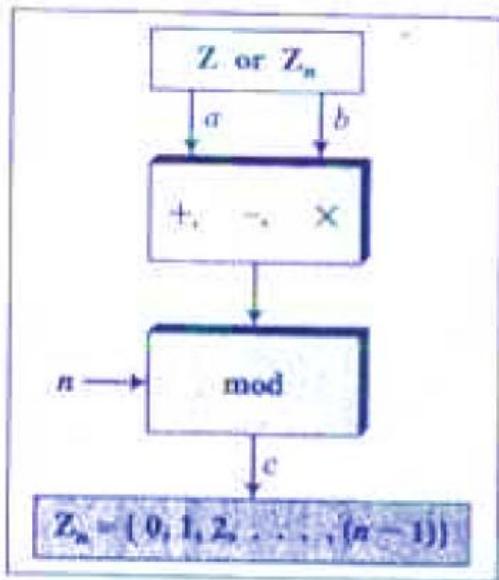
$$8 \equiv 13 \pmod{5}$$

$$34 \equiv 24 \pmod{10}$$

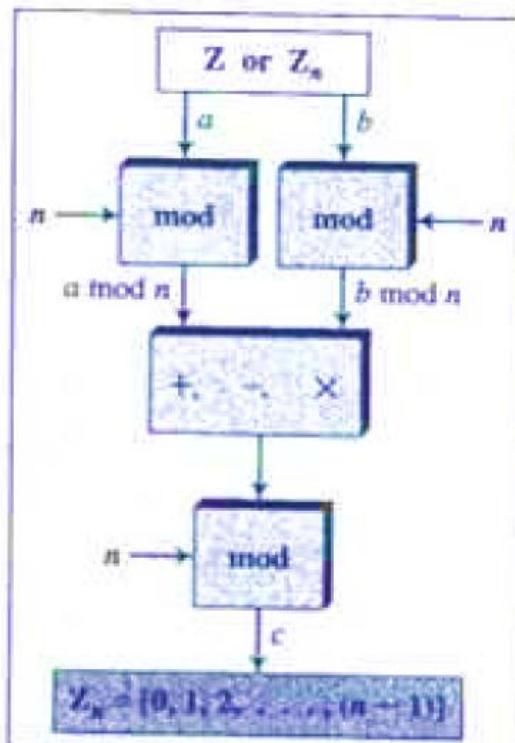
$$23 \equiv 33 \pmod{5}$$

$$-8 \equiv 12 \pmod{10}$$

$$-8 \equiv 2 \pmod{5}$$



a. Original process



b. Applying properties

### Example:

1.  $(1,723,345 + 2,124,945) \text{ mod } 11 = (8 + 9) \text{ mod } 11 = 6$
2.  $(1,723,345 - 2,124,945) \text{ mod } 11 = (8 - 9) \text{ mod } 11 = 10$
3.  $(1,723,345 \times 2,124,945) \text{ mod } 11 = (8 \times 9) \text{ mod } 11 = 6$

## OPERATION IN ZN

- In modular arithmetic we often need to find  $10 \bmod 3$ ,  $10^2 \bmod 3$ ,  $10^3 \bmod 3$  etc
- $10 \bmod 3 = 1 \rightarrow 10^n \bmod 3 = (10 \bmod 3)^n = 1^n$
- $10 \bmod 7 = 3 \rightarrow 10^n \bmod 7 = (10 \bmod 7)^n = 3^n$

$$10^n \bmod x = (10 \bmod x)^n \quad \text{Applying the third property } n \text{ times.}$$

# INVERSES

## Additive inverse

- In modular arithmetic, each integer has an additive inverse. The sum of an integer and its additive inverse is congruent to 0 modulo n.

$$a + b \equiv 0 \pmod{n}$$

## ADDITIVE INVERSE

In  $\mathbb{Z}_n$ , the additive inverse of  $a$  can be calculated as  $b = n - a$ . For example, the additive inverse of 4 in  $\mathbb{Z}_{10}$  is  $10 - 4 = 6$ .

---

In modular arithmetic, each integer has an additive inverse.

The sum of an integer and its additive inverse is congruent to 0 modulo  $n$ .

---

Note that in modular arithmetic, each number has an additive inverse and the inverse is unique; each number has one and only one additive inverse. However, the inverse of the number may be the number itself.

- Find all **additive inverse** pairs in  $Z_{10} = \{0-9\}$

- The six pairs of additive inverses are

$(0, 0), (1, 9), (2, 8), (3, 7), (4, 6)$ , and  $(5, 5)$ .

## MULTIPLICATIVE INVERSE

- In  $Z_n$ , two numbers  $a$  and  $b$  are the multiplicative inverse of each other if,

$$a \times b \equiv 1 \pmod{n}$$

- Ex: if modulus is 10 then the multiplicative inverse of 3 is 7.  $(3 \times 7) \pmod{10} = 1$
- In modular arithmetic, an integer may or may not have a multiplicative inverse. When it does, the product of the integer and its multiplicative inverse is congruent to 1 modulo n.
- The integer  $a$  in  $Z_n$  has a multiplicative inverse if and only if  $\gcd(n, a) = 1$ .

## EXAMPLES

Find the multiplicative inverse of 8 in  $Z_{10}$ .

### Solution

There is no multiplicative inverse because  $\gcd(10, 8) = 2 \neq 1$ . In other words, we cannot find any number between 0 and 9 such that when multiplied by 8, the result is congruent to 1.

- Find all multiplicative inverses in  $Z_{10}$ .
- There are only three pairs: (1, 1), (3, 7) and (9, 9). The numbers 0, 2, 4, 5, 6, and 8 do not have a multiplicative inverse.
  
- Find all multiplicative inverse pairs in  $Z_{11}$ .
- We have six pairs: (1, 1), (2, 6), (3, 4), (5, 9), (7, 8), and (10, 10).

## HOW TO FIND THE MULTIPLICATIVE INVERSE OF B IN $Z_N$ ?

- Using Extended Euclidean algorithm
- $(s \times n) + (t \times b) = \text{gcd}(n, b)$
- The inverse of b exists only if  $\text{gcd}(n, b) = 1$ . Now apply modulo operator to both sides .Therefore , $(s \times n) + (t \times b) = 1$

$$[(s \times n) + (t \times b)] \bmod n = 1 \bmod n$$

$$[(s \times n) \bmod n + (t \times b) \bmod n] = 1 \bmod n$$

$$0 + (t \times b) \bmod n = 1$$

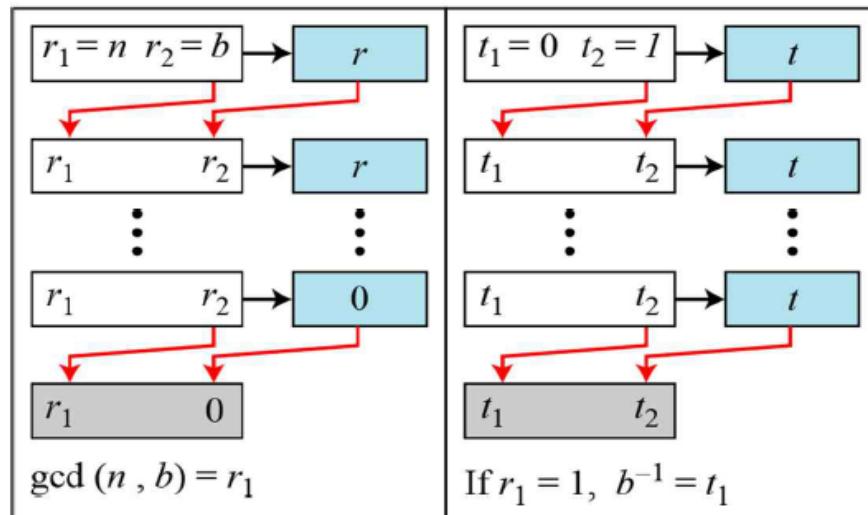
$$t \times b = 1 \pmod{n}$$

$t = b^{-1} \pmod{n}$ . T is the multiplicative inverse of b in  $Z_n$

Note:[ $(s \times n) \bmod n = 0$ , because  $s * n / n$  quotient =s, remainder=0]

# FINDING MULTIPLICATIVE INVERSES –USING EXTENDED EUCLIDEAN ALGORITHM

**Inverse exists only if  $\gcd(n, b) = 1$**



a. Process

```
r1 ← n;      r2 ← b;  
t1 ← 0;      t2 ← 1;
```

while ( $r_2 > 0$ )

```
{  
    q ← r1 / r2;
```

```
    r ← r1 - q × r2;
```

```
    r1 ← r2;      r2 ← r;
```

```
    t ← t1 - q × t2;
```

```
    t1 ← t2;      t2 ← t;
```

}

if ( $r_1 = 1$ ) then  $b^{-1} \leftarrow t_1$

b. Algorithm

The extended Euclidean algorithm finds the multiplicative inverses of  $b$  in  $Z_n$  when  $n$  and  $b$  are given and  $\gcd(n, b) = 1$ .

The multiplicative inverse of  $b$  is the value of  $t$  after being mapped to  $Z_n$ .

## EXAMPLE

- Find the multiplicative inverse of 11 in  $\mathbb{Z}_{26}$ .
- $n=26, a=11$

q	r1	r2	r	t1	t2	t
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	

- $\gcd=1$  and  $t=-7$     multiplicative inverse of 11 is  $-7 \bmod 26 = 19$
- EX: find the multiplicative inverse of 23 in  $\mathbb{Z}_{100}$  [Ans:87]

# CRYPTOGRAPHY AND AUTHENTICATION

---

# Contents Covered

- Stream Cipher and Block Cipher
- Concept of Confusion and Diffusion.
- Modes of Operation of Block Cipher: ECB, CBC, OFB, CFB
- DES, RSA
- Numerical on RSA

# Stream Cipher and Block Cipher

- Stream ciphers and Block ciphers are two encryption/decryption algorithms that belong to the family of symmetric key ciphers.
- Typically a cipher takes a plain-text as input and produces a cipher text as output.
- Block ciphers encrypts fixed-length block of bits using an unvarying transformation.
- Eg: play fair cipher , polyalphabetic cipher etc.
- Stream ciphers encrypt streams of bits with varying length and use varying transformation on each bit.
- Eg: Vigenere cipher ,Mono alphabetic cipher etc.

# Confusion and Diffusion

- In cryptography, confusion and diffusion are two properties of the operation of a secure cipher identified by Claude Shannon in his 1945 classified report A Mathematical Theory of Cryptography.
- These properties, when present, work to thwart the application of statistics and other methods of cryptanalysis.
- **Diffusion :** The idea of diffusion is to hide the relationship between the cipher text and the plain text.
- Diffusion implies that each symbol in the cipher text is dependent on some or all symbols in the plain text.
- This will frustrate the adversary who use cipher text statistics to find the plain text.

# Confusion and Diffusion

- **Confusion:** The idea of confusion is to hide the relationship between the cipher text and the key.
- If a single bit in the key is changed most or all bits in the cipher text will also be changed.
- the calculation of the values of most or all of the bits in the cipher text will be affected.
- Confusion increases the ambiguity of cipher text and it is used by both block and stream ciphers.
- This will frustrate the adversary who use cipher text statistics to find the key

# BLOCK CIPHER

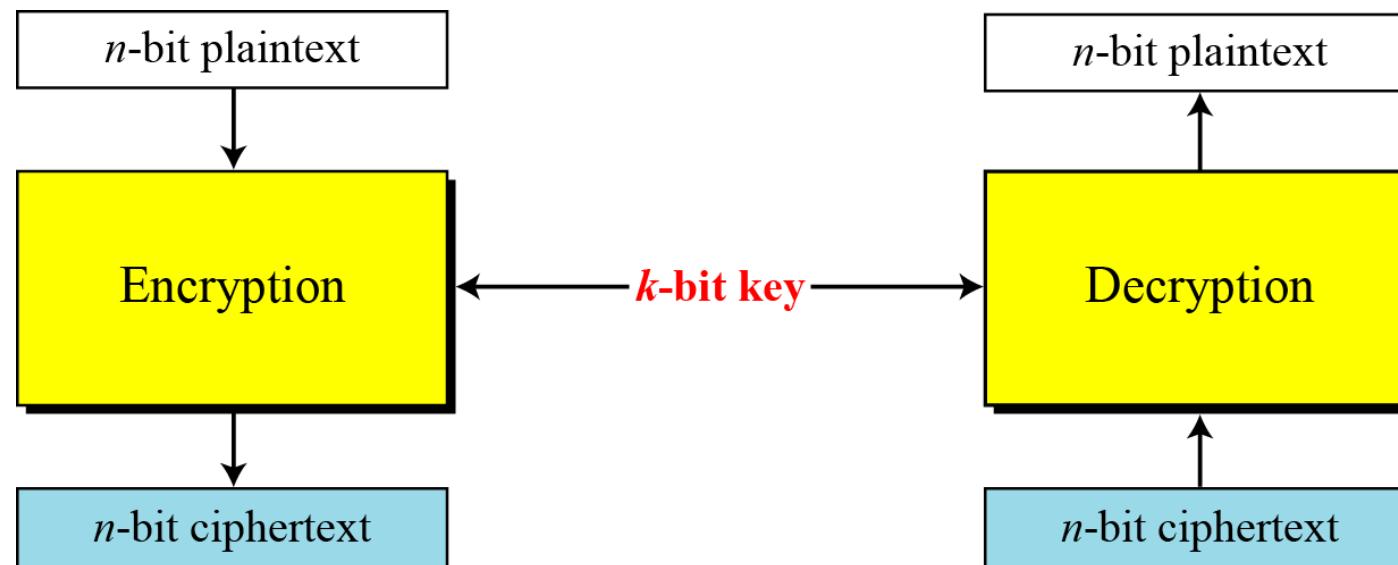
---

A block Cipher encrypt a group of plain text symbol as one block.

A stream Cipher convert one symbol of plain text into a symbol of cipher text.

# Block cipher

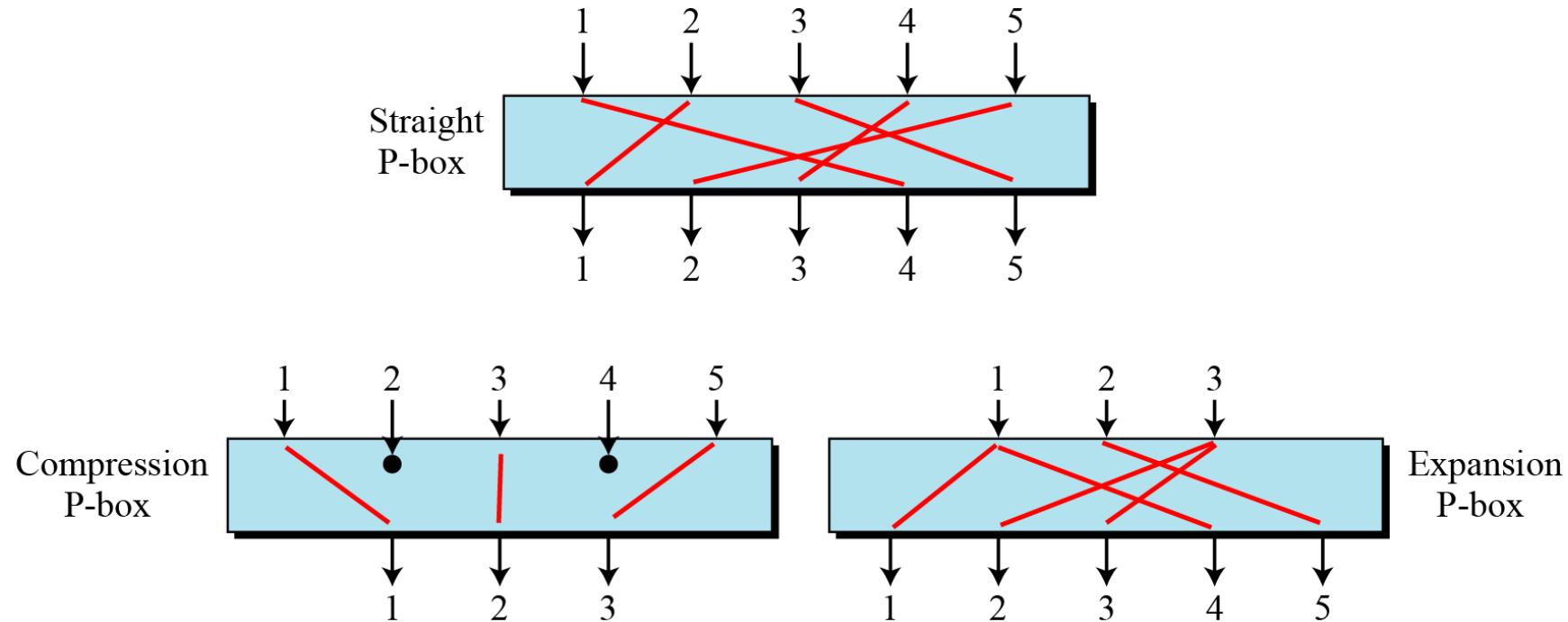
- A *symmetric-key modern block cipher encrypts an  $n$ -bit block of plaintext or decrypts an  $n$ -bit block of ciphertext. The encryption or decryption algorithm uses a  $k$ -bit key.*



# Components of modern block cipher

- A modern block cipher is made of a combination of transposition unit for diffusion-permutation box (P-box) and confusion - substitution unit called (S-box)
- P-box: A P-box parallels the traditional transposition cipher for characters . It transposes bits.
  - Straight P-box :with n input and n output is a permutation . There are  $n!$  possible mapping
  - Expansion P-box :with n bit input and m output with  $m > n$ , some input connected to more than one output. It is used when we need to transpose bit at same time increase the no of bit for next stage.
  - Compression P-box : with n input and m output with  $m < n$  , some input are blocked to reach the output .It is used when we need to permute bits and decrease the number of bits in next stage.

# *Three types of P-boxes*

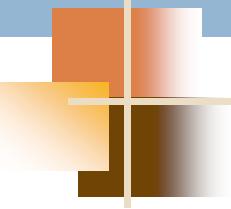


A straight P-box is invertible, but compression  
and expansion P-boxes are not.

## S-box

- An S-box(Substitution box)is a substitution cipher that can have different number of inputs and outputs.
- Input can be thought of n-bit word ,but output can be an m-bit word.

An S-box is an  $m \times n$  substitution unit, where  $m$  and  $n$  are not necessarily the same.



In an S-box with three inputs and two outputs, we have

$$y_1 = x_1 \oplus x_2 \oplus x_3 \quad y_2 = x_1$$

The S-box is linear because  $a_{1,1} = a_{1,2} = a_{1,3} = a_{2,1} = 1$  and  $a_{2,2} = a_{2,3} = 0$ . The relationship can be represented by matrices, as shown below:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

## Continued

In an S-box with three inputs and two outputs, we have

$$y_1 = (x_1)^3 + x_2 \quad y_2 = (x_1)^2 + x_1 x_2 + x_3$$

The S-box is nonlinear because there is no linear relationship between the inputs and the outputs.

Leftmost bit

Rightmost bits

Based on the table, an input of 010 yields the output 01. An input of 101 yields the output of 00.

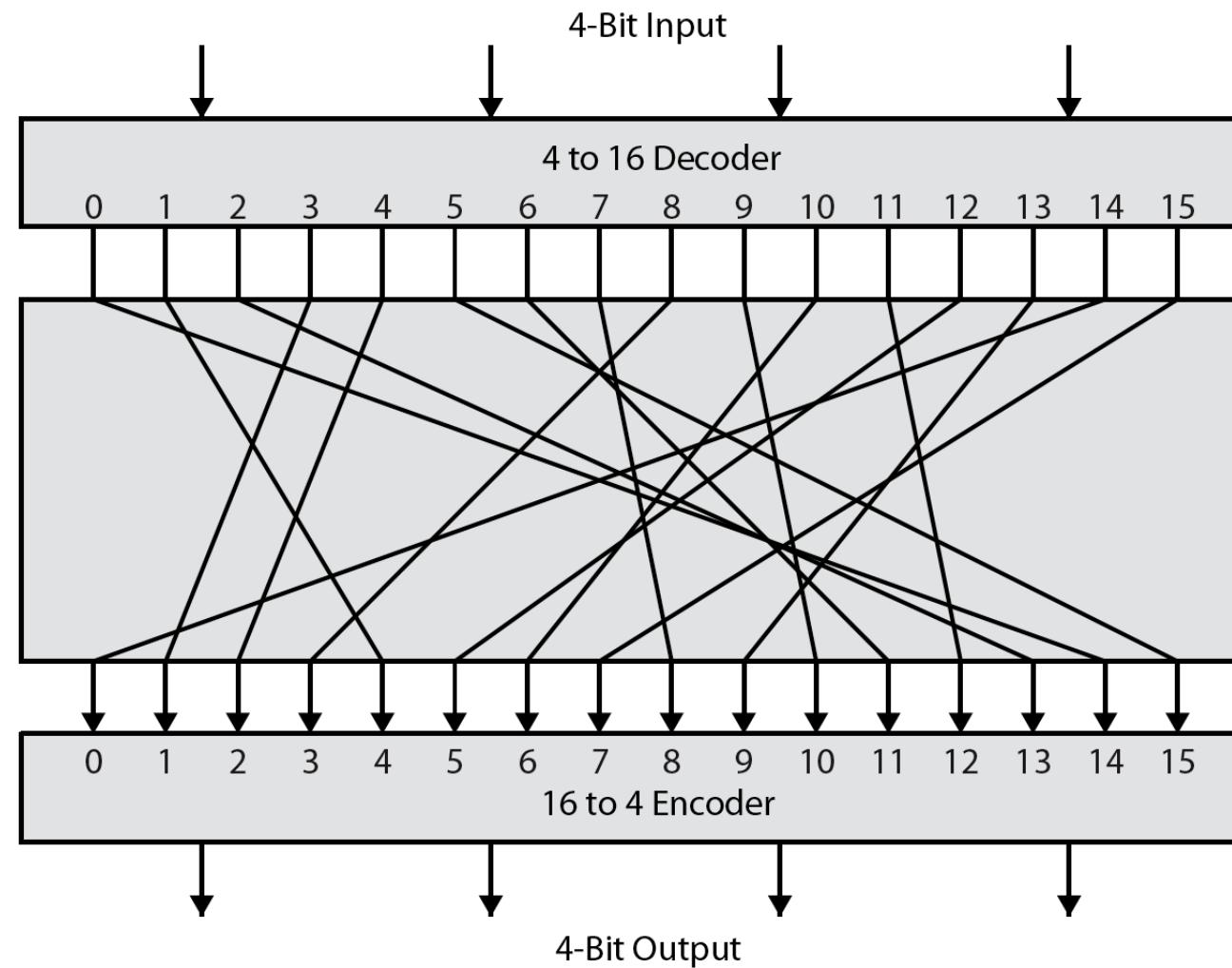
		00	01	10	11
0	0	00	10	01	11
	1	10	00	11	01

Output bits

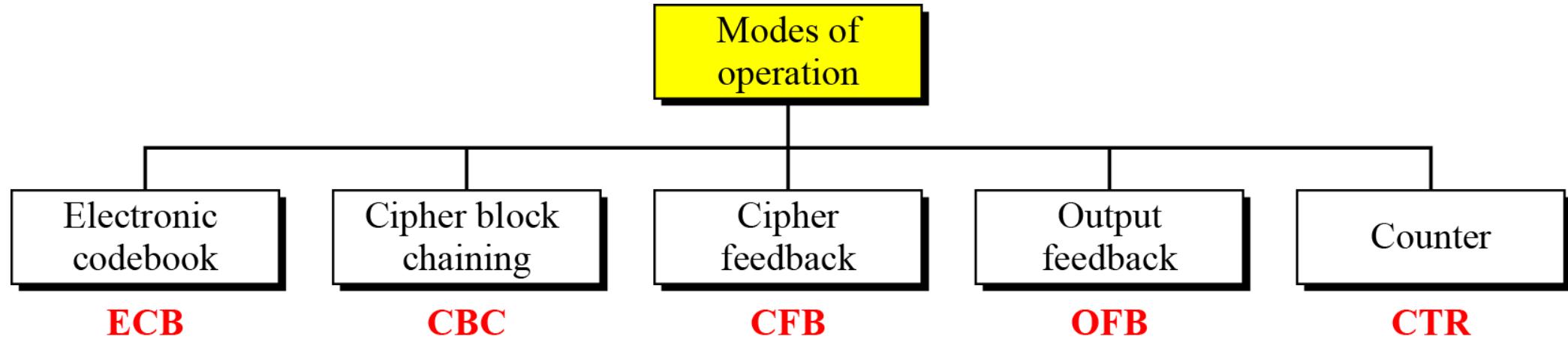
# Product cipher

- Claude Shannon introduced idea of substitution-permutation (S-P) networks in 1949 paper
- form basis of modern block ciphers
- S-P nets are based on the two primitive cryptographic operations seen before:
  - *substitution* (S-box)
  - *permutation* (P-box)
- provide *confusion & diffusion* of message & key

# Ideal Block Cipher



# Modes of Operation



- Modes of operation have been devised to encipher text of any size employing either DES or AES.
- A mode of operation describes how to repeatedly apply a cipher's single-block operation to securely transform amounts of data larger than a block.

# Electronic Codebook (ECB) Mode

The simplest mode of operation is called the electronic codebook (ECB) mode.

- Plain text divided into blocks of 64 bits each
- Encrypt each block independently with the same key
- Same process is applied during decryption.

$$\text{Encryption: } C_i = E_K(P_i)$$

$$\text{Decryption: } P_i = D_K(C_i)$$

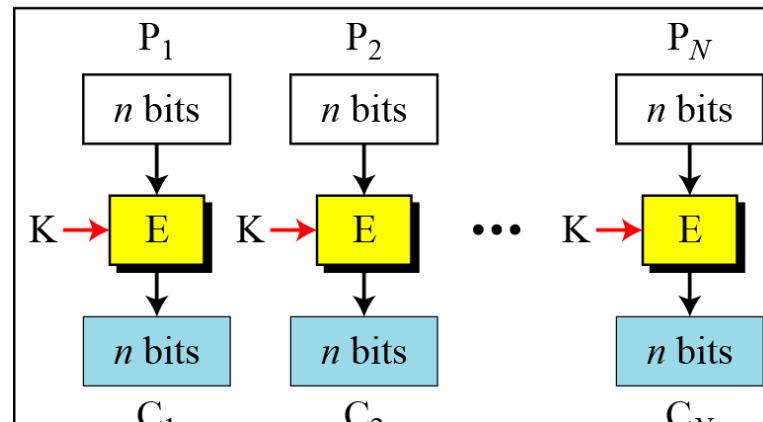
E: Encryption

D: Decryption

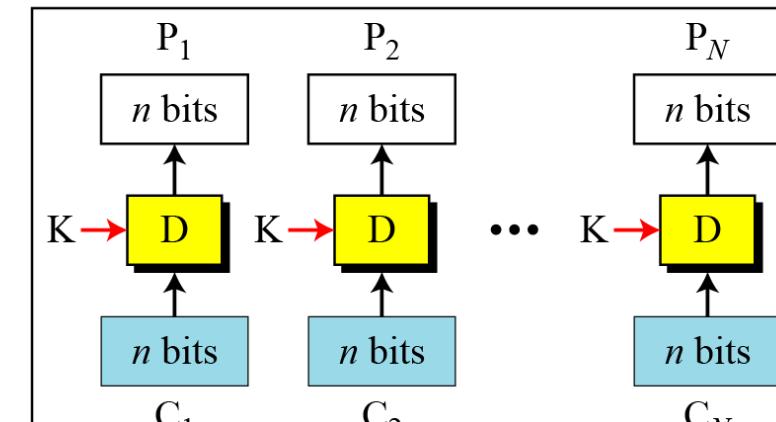
$P_i$ : Plaintext block  $i$

$C_i$ : Ciphertext block  $i$

K: Secret key



Encryption



Decryption

# ECB

- Security Issues:
  1. Pattern at the block level are preserved- single key usage
  2. The block independency creates opportunities for eve to exchange some ciphertext blocks without knowing the key.

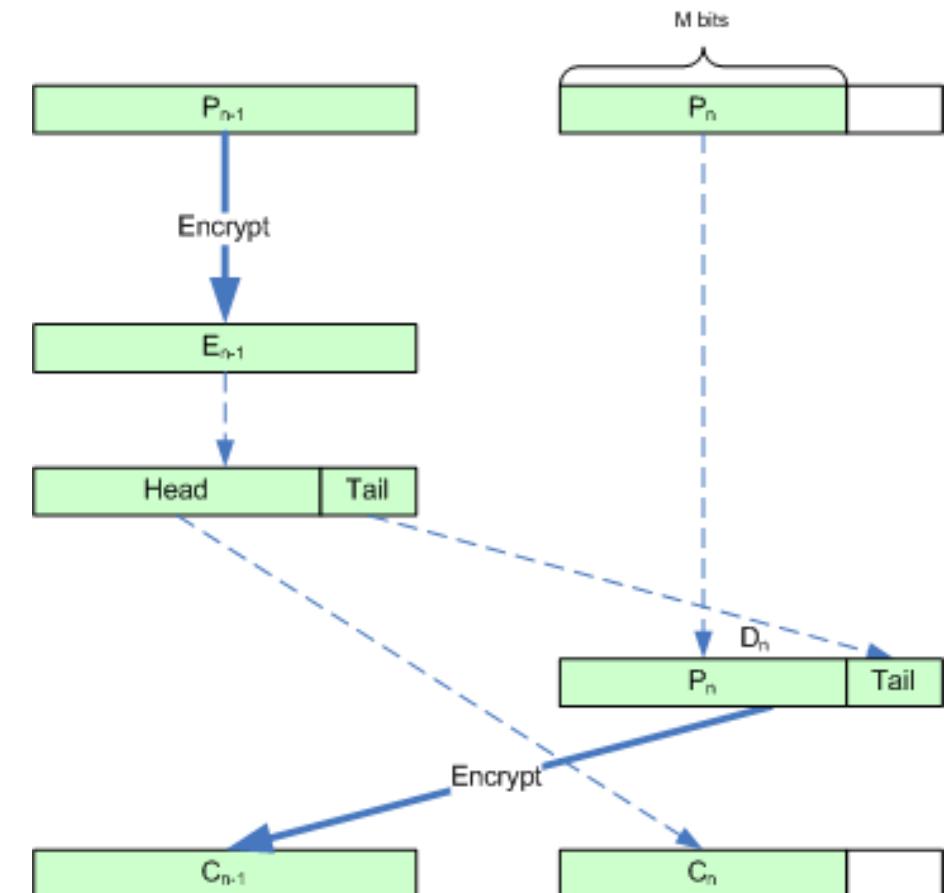
Error propagation: Error does not have any effect on the other blocks.

# Ciphertext Stealing

- A technique called ciphertext stealing (CTS) can make it possible to use ECB mode without padding.
- It allows for processing of messages that are not evenly divisible into blocks without resulting in any expansion of the cipher text, at the cost of slightly increased complexity.
- In this technique the last two plaintext blocks,  $P_{N-1}$  and  $P_N$ , are encrypted differently and out of order, as shown below, assuming that  $P_{N-1}$  has  $n$  bits and  $P_N$  has  $m$  bits, where  $m \leq n$ .

$$X = E_K(P_{N-1}) \rightarrow C_N = \text{head}_m(X)$$

$$Y = P_N \mid \text{tail}_{n-m}(X) \rightarrow C_{N-1} = E_K(Y)$$

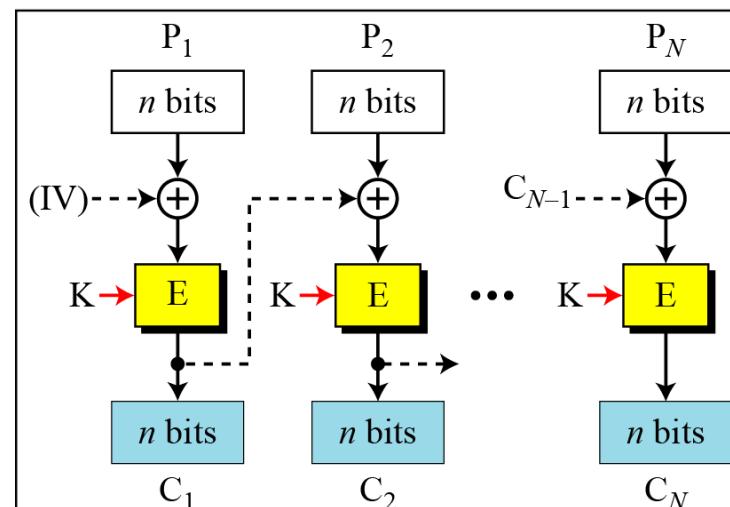


# Cipher Block Chaining (CBC) Mode

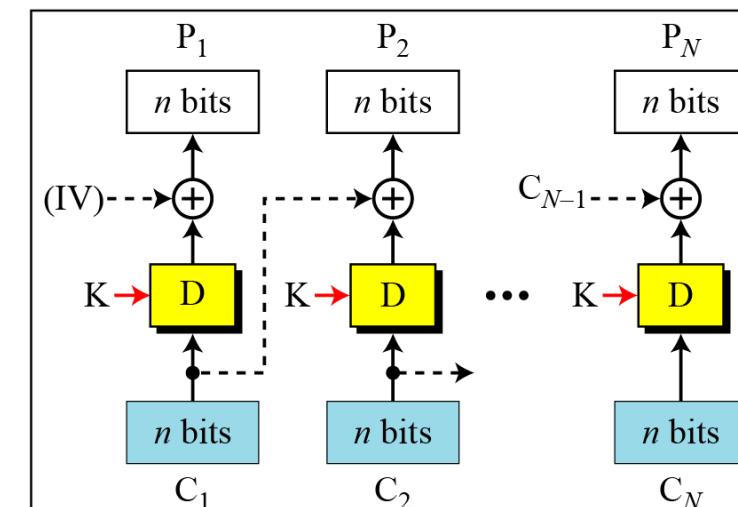
- In CBC mode, each plaintext block is Exclusive-ORed with the previous ciphertext block before being encrypted.
- The IV(Initialization Vector) is exclusive ORed with the first block of plaintext prior to the encryption step; the block of ciphertext just produced is exclusive-ORed with the next block of plaintext, and so on.
- You must use the same IV to decipher the data.

E : Encryption  
 $P_i$ : Plaintext block  $i$   
K: Secret key

D : Decryption  
 $C_i$ : Ciphertext block  $i$   
IV: Initial vector ( $C_0$ )



Encryption



Decryption

# Cipher Feedback (CFB) Mode

*In some situations, we need to use DES or AES as secure ciphers, but the plaintext or ciphertext block sizes are to be smaller.*

E : Encryption

$P_i$ : Plaintext block  $i$

K: Secret key

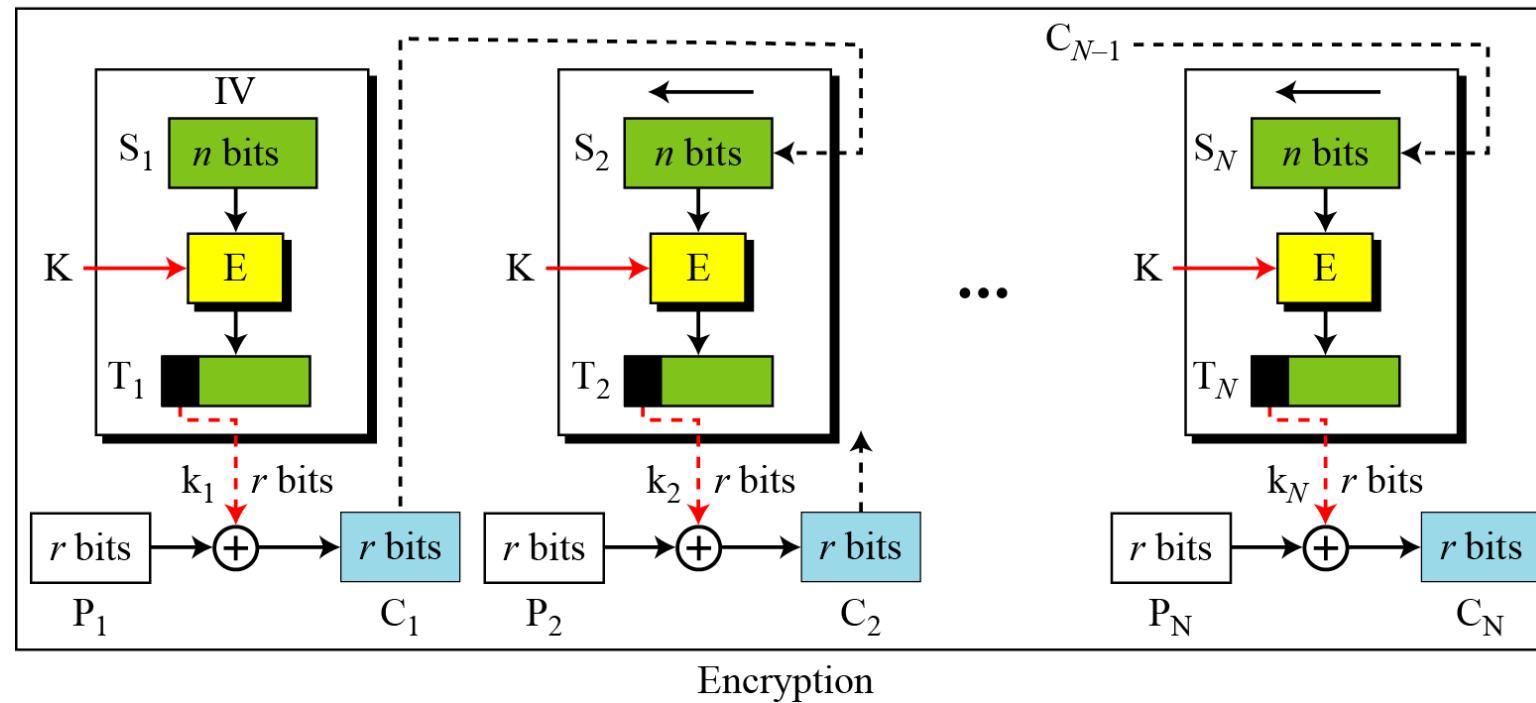
D : Decryption

$C_i$ : Ciphertext block  $i$

IV: Initial vector ( $S_1$ )

$S_i$ : Shift register

$T_i$ : Temporary register



# CFB

- The idea is to use DES or AES not to encrypt and decrypt the PL and CT but to encrypt or decrypt the content of a shift register S of size n.
- Encryption is done by exclusive oring an r-bit plain text with r bit of the shift register.
- Decryption is done by exclusive oring an r bit cipher text block with r bit of shift register.
- For each block the shift register  $S_i$  is made by shifting the shift register  $s_{i-1}$  r bits to left and filling the rightmost r bit with  $c_{i-1}$
- Only the rightmost r bit of  $T_i$  are exclusive-ored with the plain text block  $p_i$  to make  $c_i$ .

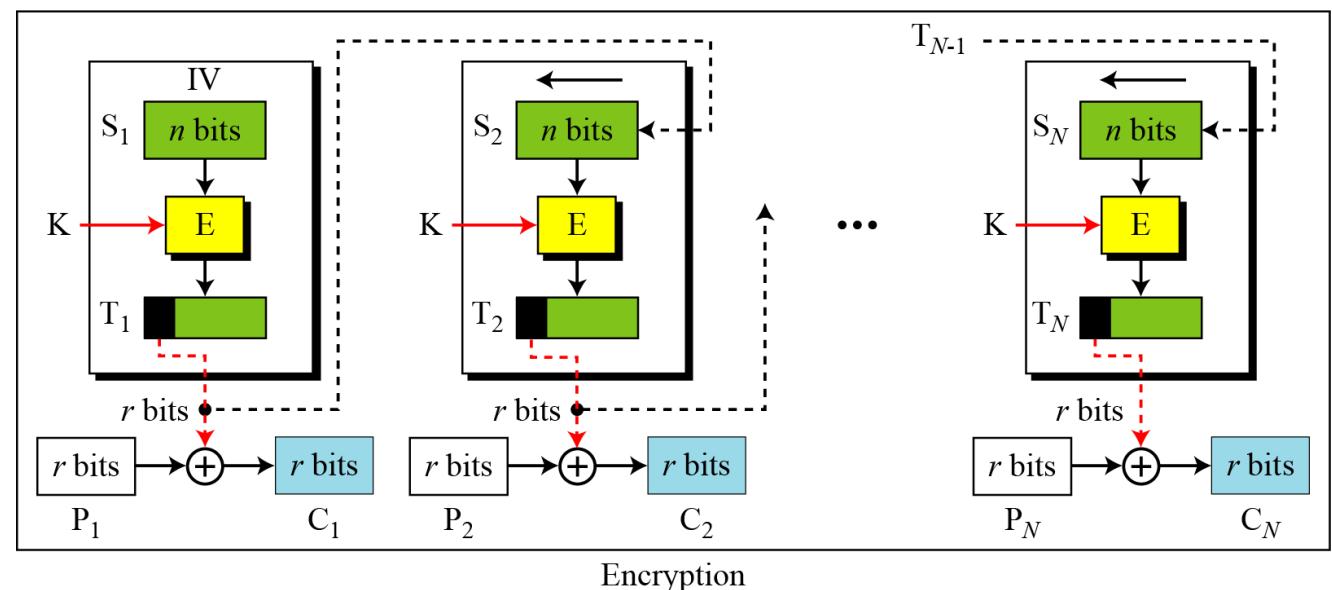
# Output Feedback (OFB) Mode

*In this mode each bit in the ciphertext is independent of the previous bit or bits. This avoids error propagation.*

*The key stream generated is XOR-ed with the plaintext blocks.*

*The OFB mode requires an IV as the initial random  $n$ -bit input block.*

E : Encryption  
P<sub>i</sub>: Plaintext block i  
K: Secret key  
D : Decryption  
C<sub>i</sub>: Ciphertext block i  
IV: Initial vector ( $S_1$ )  
 $S_i$ : Shift register  
 $T_i$ : Temporary register



*Encryption in output feedback (OFB) mode*

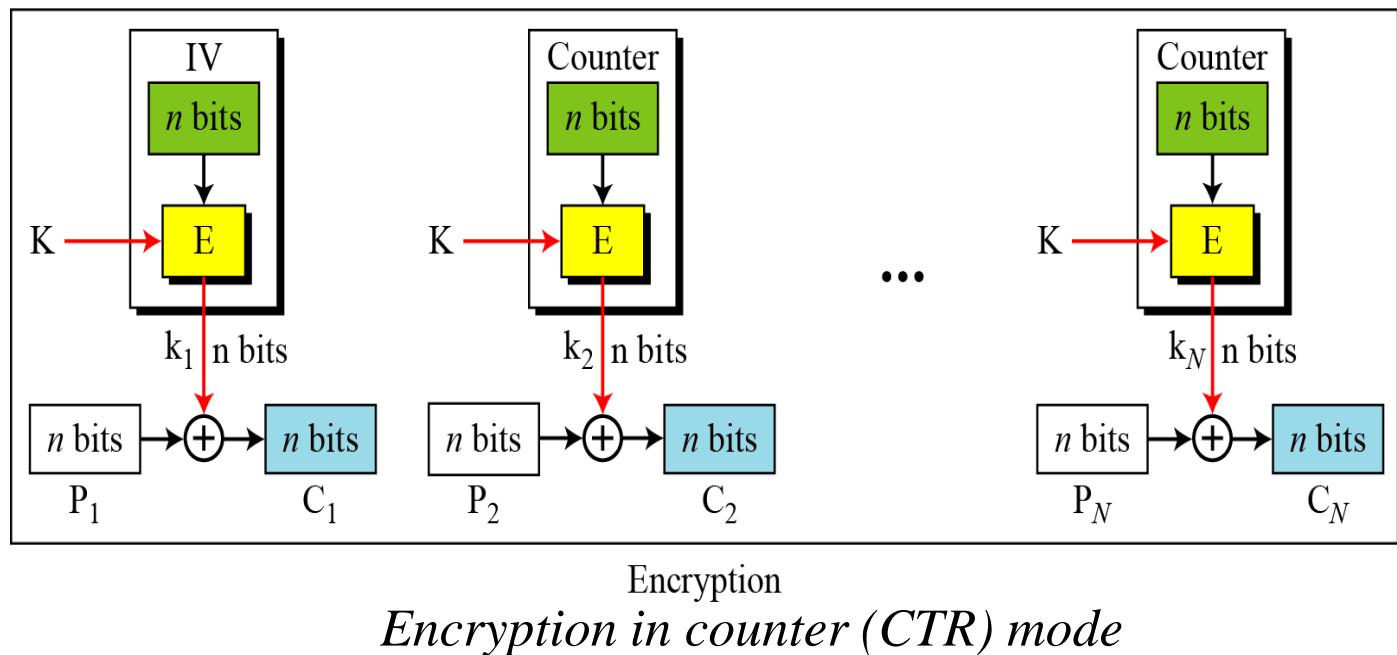
# Counter (CTR) Mode

- In the counter (CTR) mode, there is no feedback.
- The pseudo randomness in the key stream is achieved using a counter.
- In this mode, both the sender and receiver need to access to a reliable counter, which computes a new shared value each time a ciphertext block is exchanged.
- This shared counter is not necessarily a secret value, but challenge is that both sides must keep the counter synchronized.

E : Encryption  
P<sub>i</sub>: Plaintext block *i*  
K : Secret key

IV: Initialization vector  
C<sub>i</sub>: Ciphertext block *i*  
k<sub>i</sub> : Encryption key *i*

The counter is incremented for each block.



## Counter (CTR)Mode:

- An n-bit counter is initialized to a predetermined value (IV) and incremented based on a predefined rule ( $\text{mod } 2^n$ )
- CTR vs OFB: Like OFB, CTR creates a key stream that is independent from the previous cipher text blocks but CTR does not use feedback.
- CTR vs ECB:n bit cipher text blocks that are independent from each other,they depend on the value of the counter.

# Comparison of different Modes

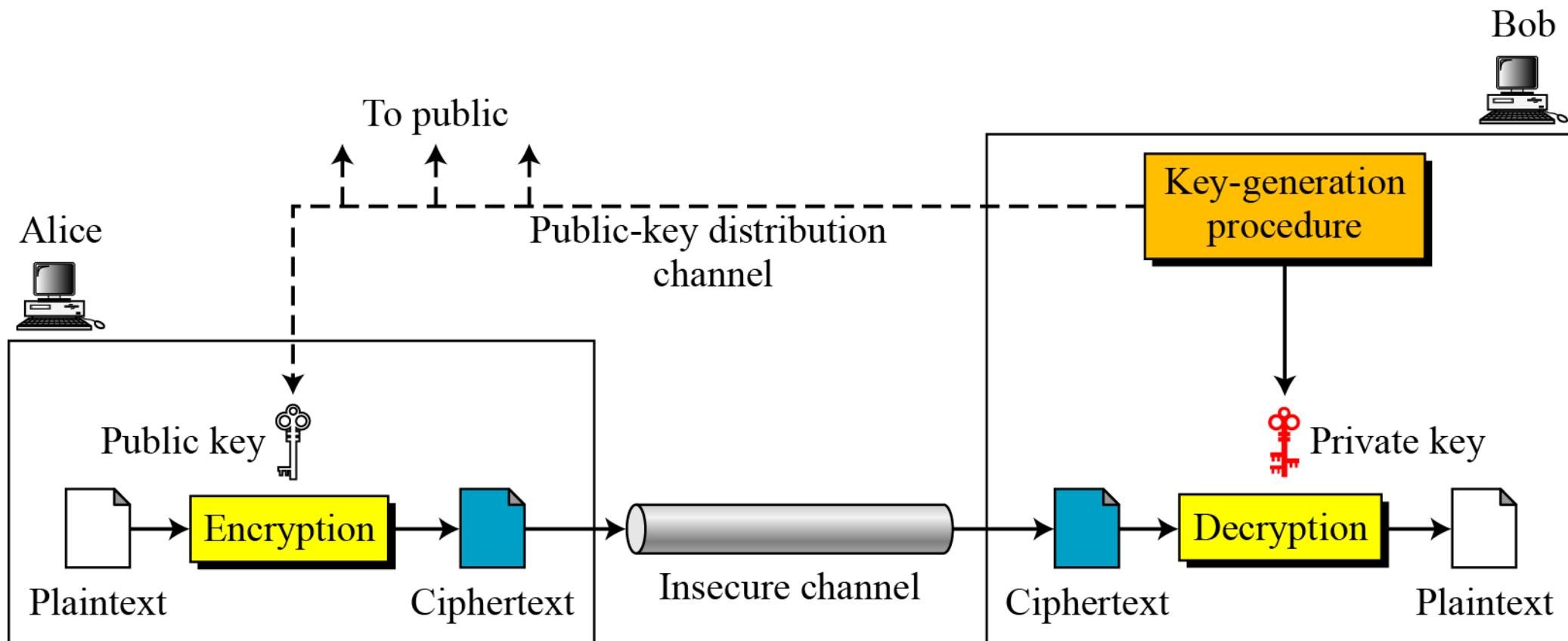
Table 8.1 *Summary of operation modes*

<i>Operation Mode</i>	<i>Description</i>	<i>Type of Result</i>	<i>Data Unit Size</i>
ECB	Each $n$ -bit block is encrypted independently with the same cipher key.	Block cipher	$n$
CBC	Same as ECB, but each block is first exclusive-ored with the previous ciphertext.	Block cipher	$n$
CFB	Each $r$ -bit block is exclusive-ored with an $r$ -bit key, which is part of previous cipher text	Stream cipher	$r \leq n$
OFB	Same as CFB, but the shift register is updated by the previous $r$ -bit key.	Stream cipher	$r \leq n$
CTR	Same as OFB, but a counter is used instead of a shift register.	Stream cipher	$n$

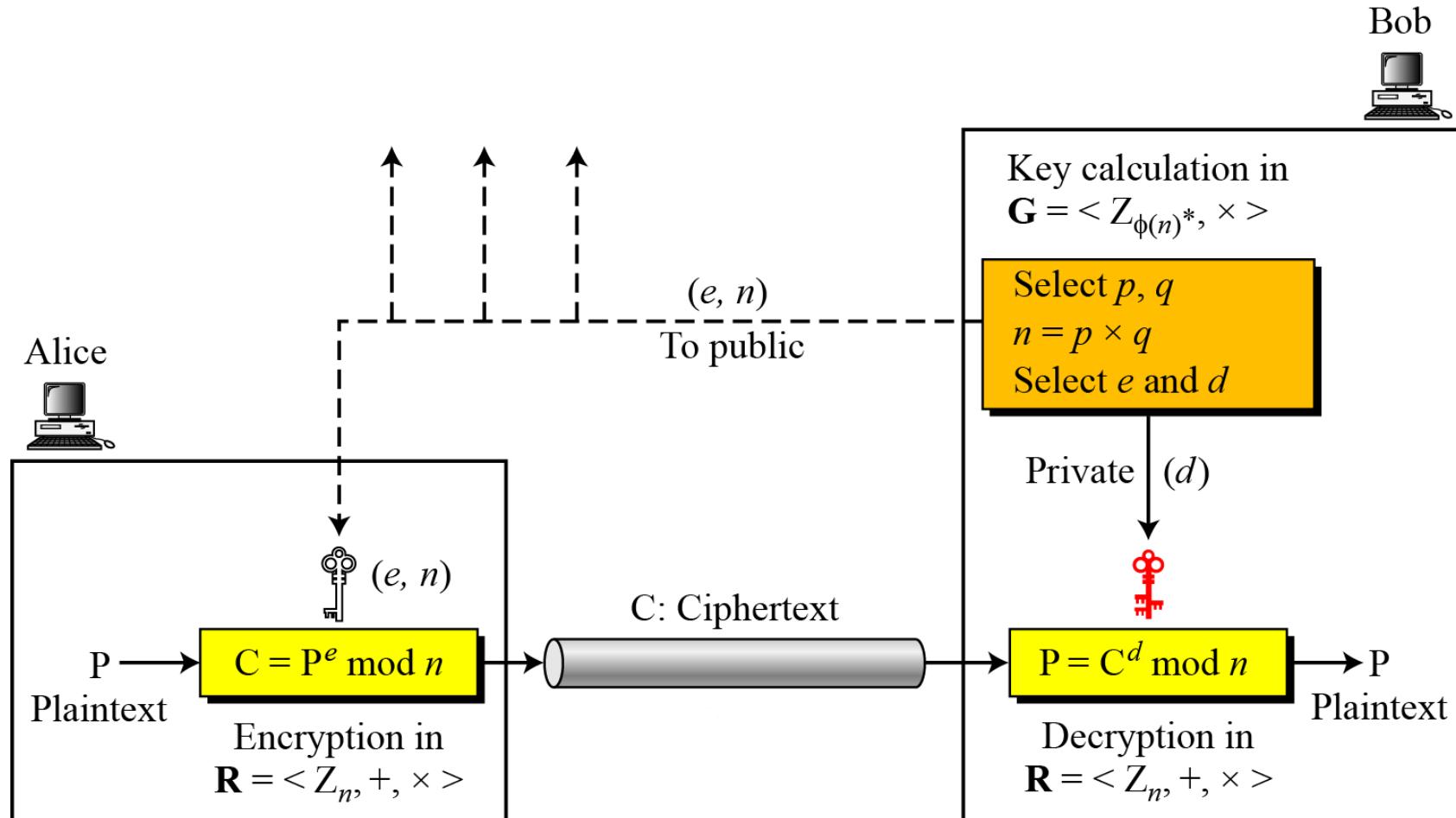
# RSA

RSA is a public-key cryptosystem that is widely used for secure data transmission. It is also one of the oldest.

The acronym RSA comes from the surnames of Ron Rivest, Adi Shamir and Leonard Adleman, who publicly described the algorithm in 1977.



# RSA Steps



# RSA STEPS

- Select 2 large prime numbers of about the same size,  $p$  and  $q$  such that  $p \neq q$
- Compute  $n = pq$ , and  $\Phi(n) = (p-1)(q-1)$
- Select the Public Key (encryption key) $e$  such that it has no factor common with  $(p-1)(q-1)$ , where  $1 < e < \Phi(n)$ , such that  $\gcd(e, \Phi(n)) = 1$  ( $e$  is coprime to  $\Phi(n)$ )
- Compute  $d$ ,  $1 < d < \Phi(n)$  such that  $ed \equiv 1 \pmod{\Phi(n)}$  (using the Extended Euclidean Algorithm)
- <https://youtu.be/LYmb8Adr6Wc> - Reference video link

# Using extended Euclidean to find d

q	r <sub>1</sub>	r <sub>2</sub>	r	s <sub>1</sub>	s <sub>2</sub>	s	t <sub>1</sub>	t <sub>2</sub>	t
9	120	13	3	1	0	1	0	1	-9
4	13	3	1	0	1	-4	1	-9	37
3	3	1	0	1	-4	13	-9	37	-120
	1	0		-4	13		37	-120	

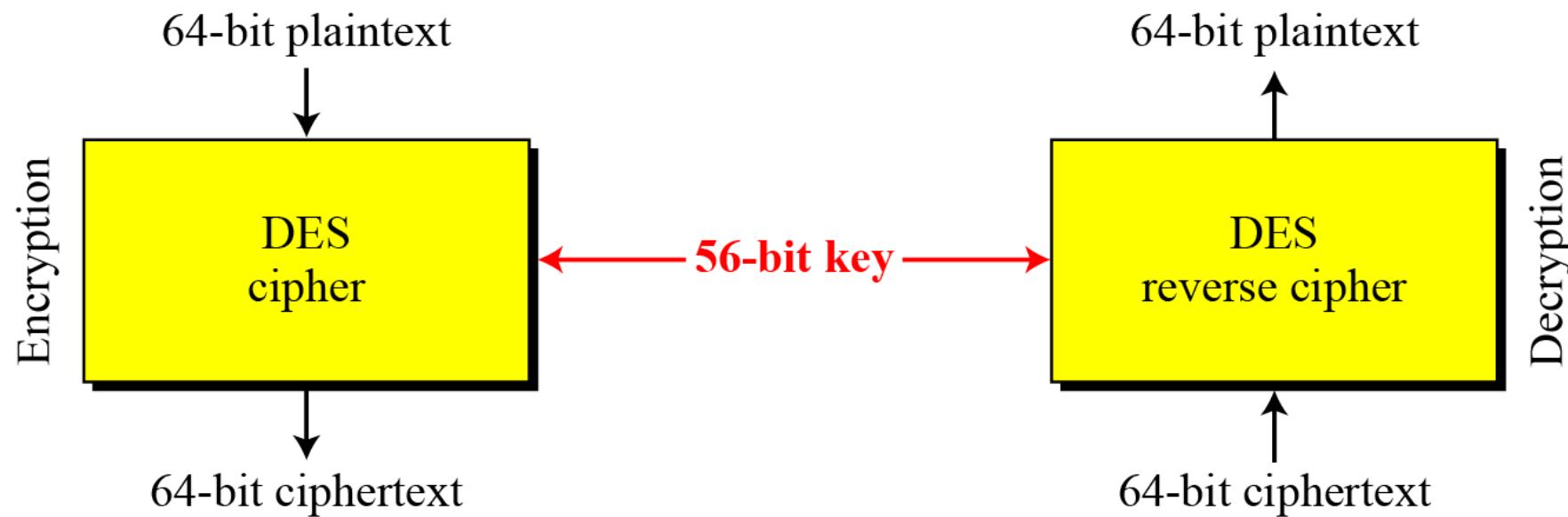
- P=13, q=11
- n=143
- $\Phi(n)=120$
- e=13
- $ed \equiv 1 \pmod{\Phi(n)}$
- $\text{Gcd}(\Phi(n), e) = 1$
- $\text{gcd}(120, 13) = 120s + 13t = 1$
- D=37
- Ex: p=3, q=11

# DATA ENCRYPTION STANDARD (DES)

---

# Overview of DES

DES is a Symmetric key block cipher. Published by National Institute of Standards and Technology(NIST) 1975



# History of DES Algorithm

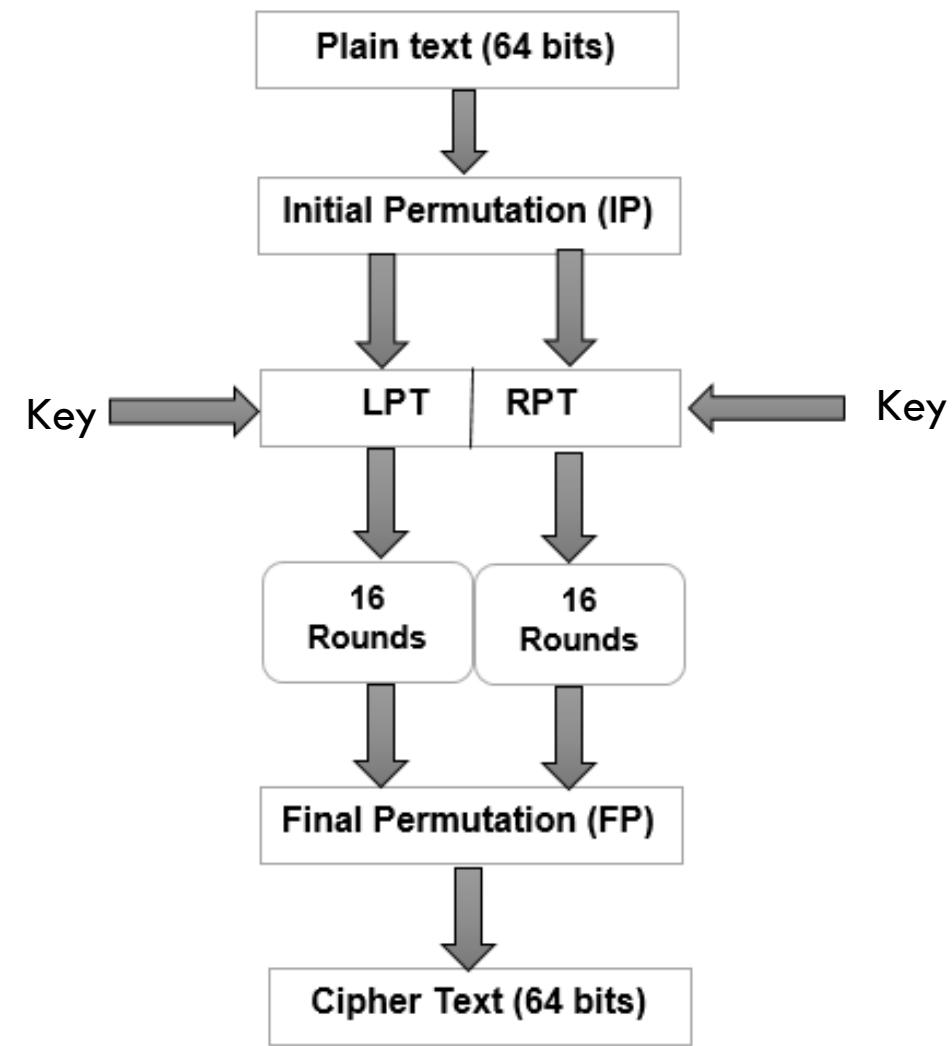
- DES is based on the Feistel block cipher, called LUCIFER, developed in 1971 by IBM cryptography researcher Horst Feistel.
- DES uses 16 rounds of the Feistel structure, using a different key for each round.
- DES became the approved federal encryption standard in November 1976 and was subsequently reaffirmed as the standard in 1983, 1988, and 1999.
- DES's dominance came to an end in 2002, when the Advanced Encryption Standard (AES) replaced the DES encryption algorithm as the accepted standard

# DES - Basics

- DES uses the two basic techniques of cryptography - confusion and diffusion.
- At the simplest level, diffusion is achieved through numerous permutations and confusions is achieved through the XOR operation and the S-Boxes.
- This is also called an S-P network.

# Steps in DES

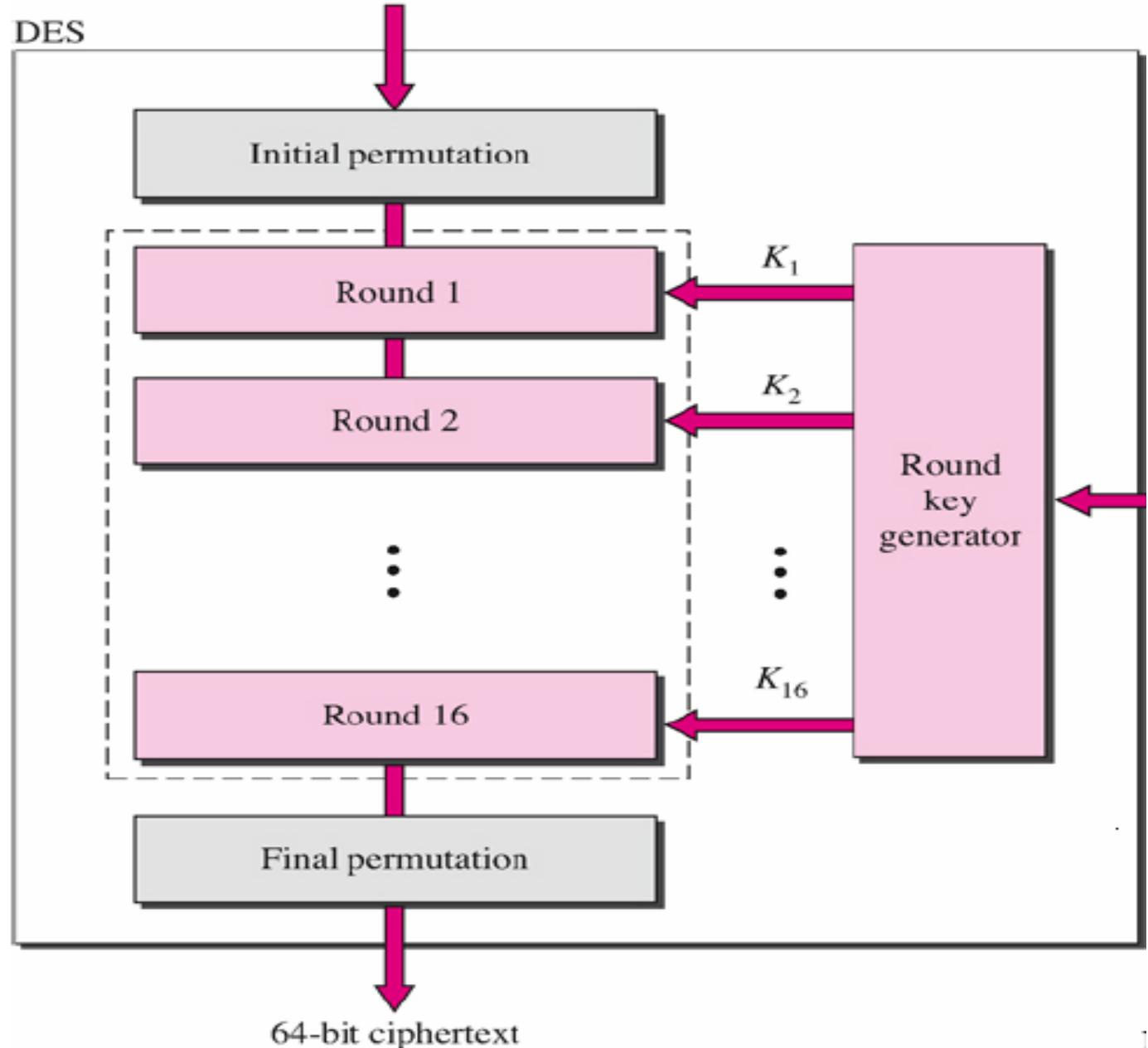
1. The process begins with the 64-bit plain text block getting handed over to an initial permutation (IP) function.
2. The initial permutation (IP) is then performed on the plain text.
3. Next, the initial permutation (IP) creates two halves of the permuted block, referred to as Left Plain Text (LPT) and Right Plain Text (RPT).
4. Each LPT and RPT goes through 16 rounds of the encryption process.
5. Finally, the LPT and RPT are rejoined, and a Final Permutation (FP) is performed on the newly combined block.
6. The result of this process produces the desired 64-bit cipher text.



[Figure: Broad level steps in DES]

# General Structure of DES

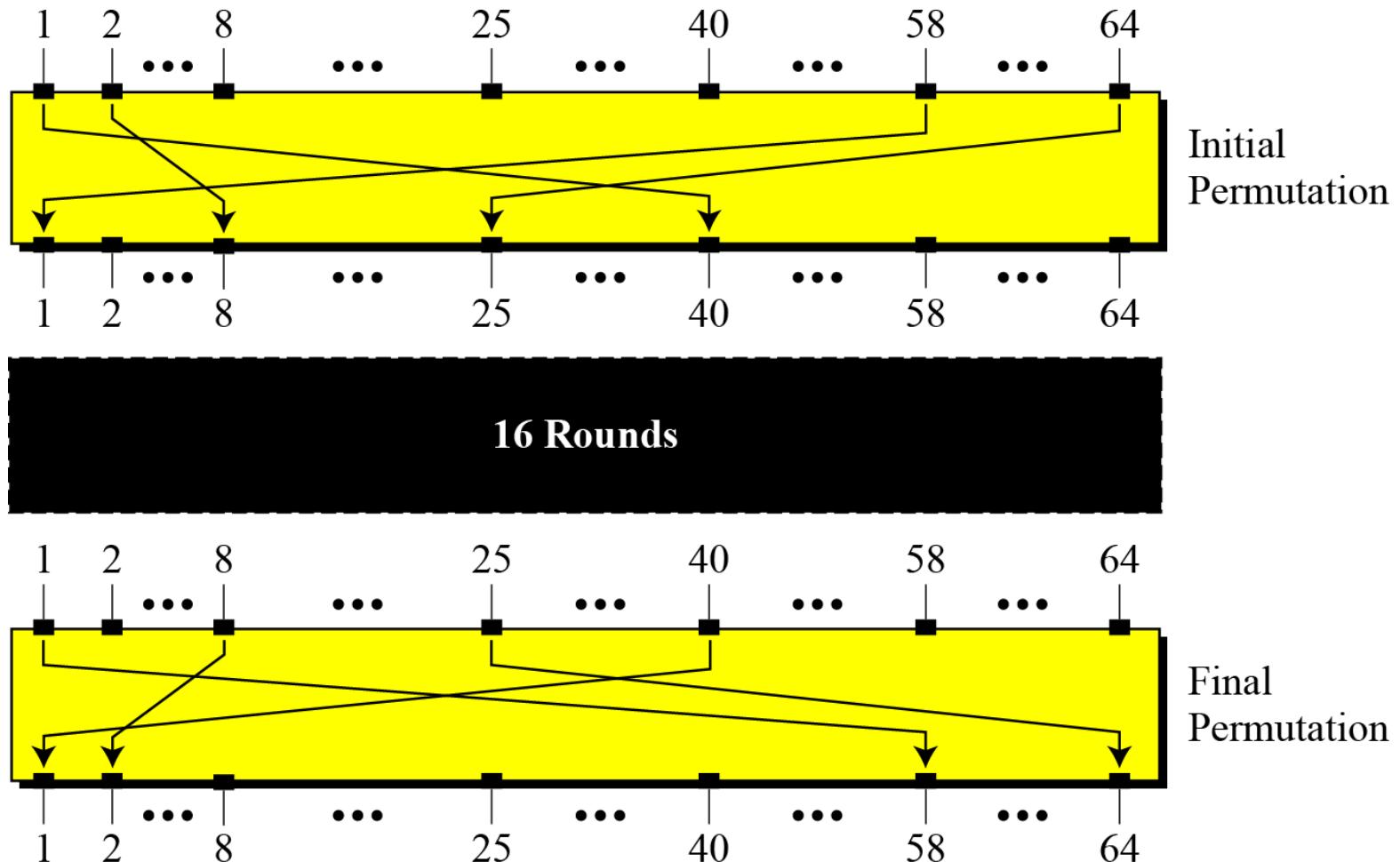
The encryption process is made of two permutations (P-boxes), which we call initial and final permutations, and sixteen Feistel rounds.



## Initial and final permutation

- DES has an initial permutation and a final permutation after 16 rounds.
- These permutations are inverses of each other and operate on 64 bits.
- They have no cryptographic significance.
- The designers did not disclose their purpose.

# *Initial and final permutation steps in DES*



# *Initial and final permutation tables*

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

# Encryption process

- The encryption process step (step 4) is further broken down into five stages:
- Key transformation
- Expansion permutation
- S-Box permutation
- P-Box permutation
- XOR and swap

# Key transformation

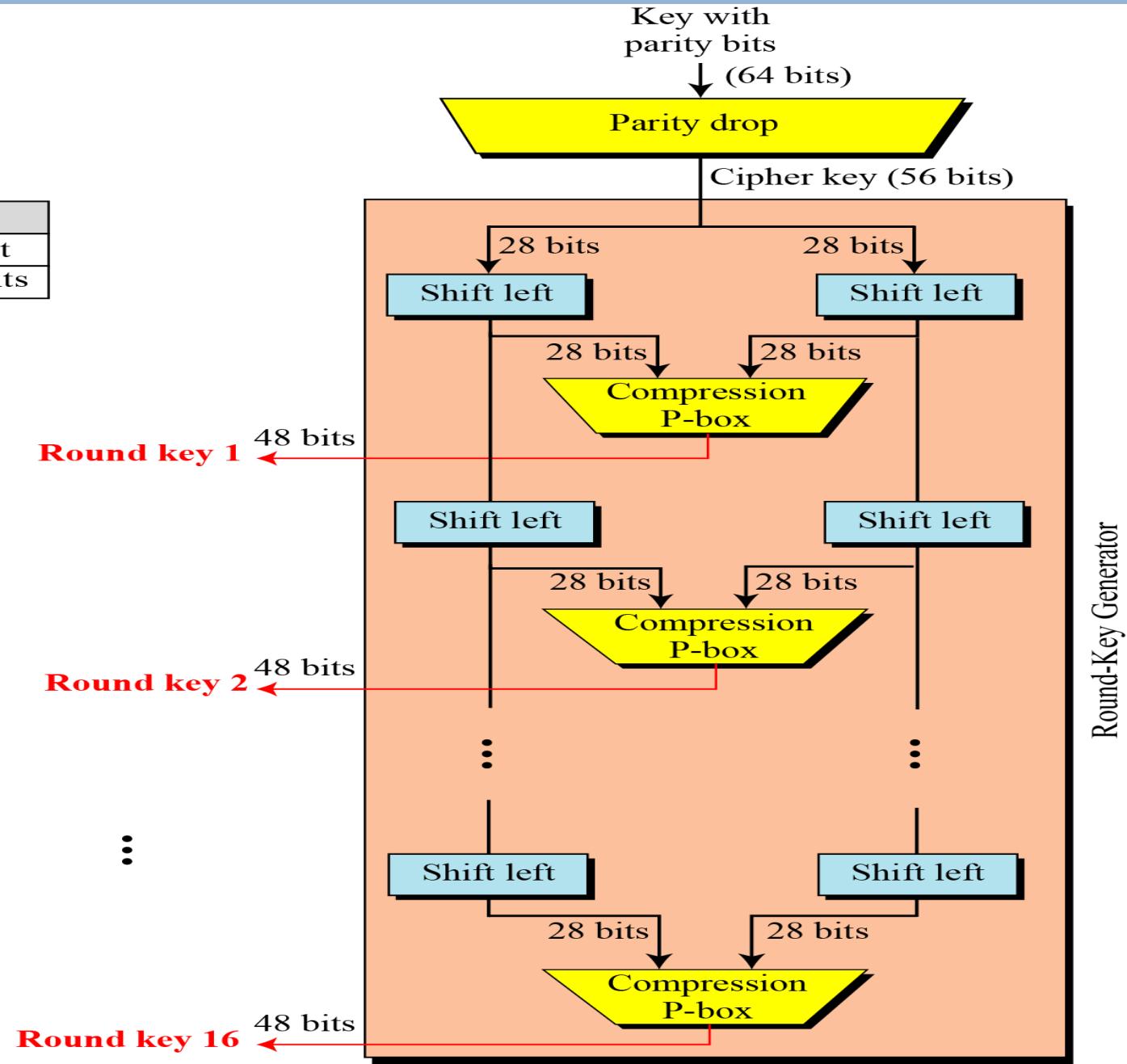
- The 64 bit key is first subjected to permutation or parity Drop.
- It drop Parity bit(8,16,24,32,40,48,56,64)
- The resulting 56 bit key is then treated as two 28 bit quantities
- At each round they are subjected to a circular left shift of 1 or 2 bits governed by the following
  - For rounds 1, 2, 9 and 16 the shift  $r_i$  is 1, and in all other rounds  $r_i$  is 2
  - These shifted values serve as input to the next round and permutation which produce 48 bit output that serve as input (key).

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

Figure - compression permutation

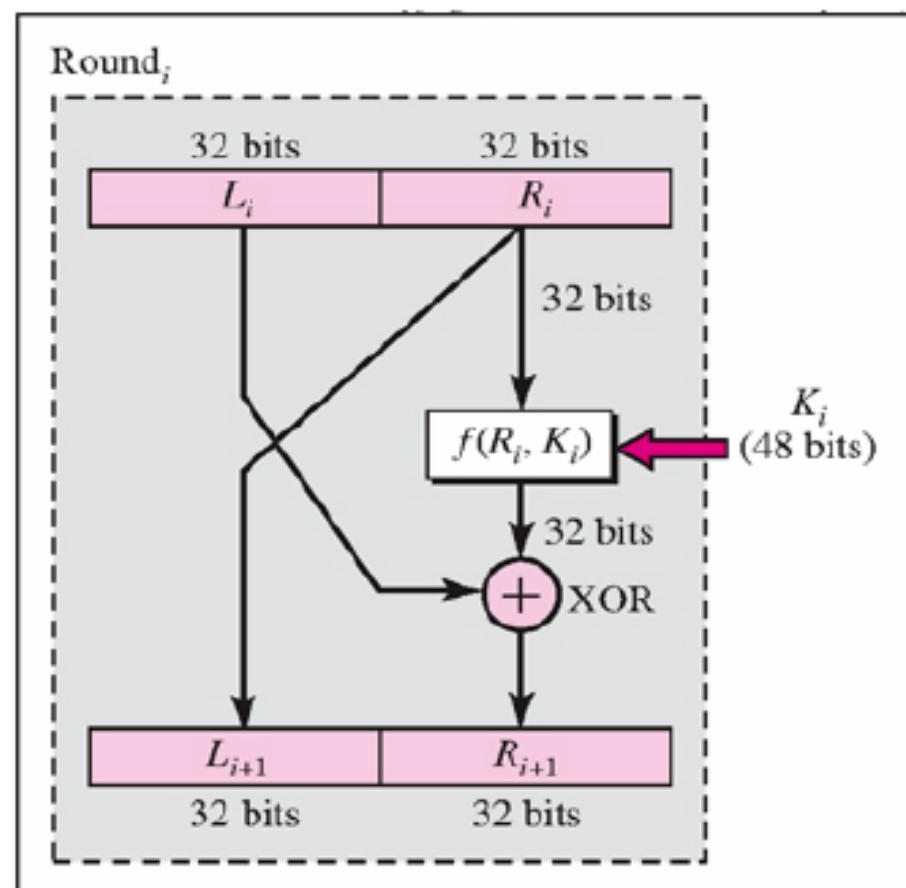
# Key generation

Shifting	
Rounds	Shift
1, 2, 9, 16	one bit
Others	two bits

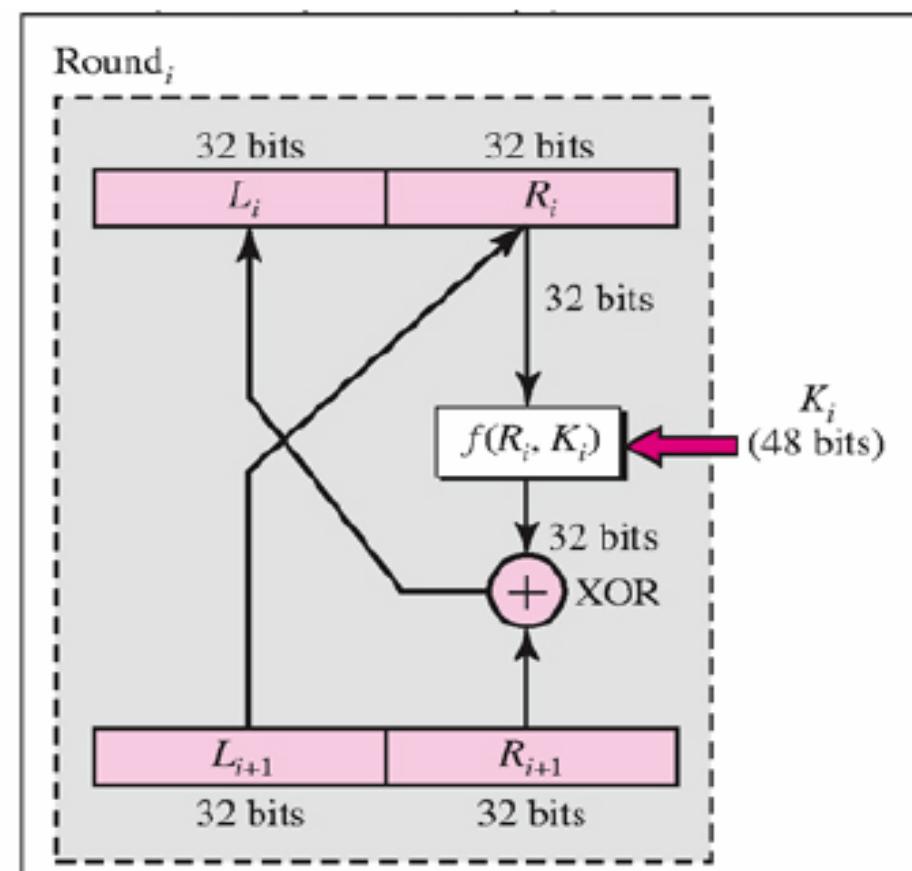


# DES - One Round

## Rounds



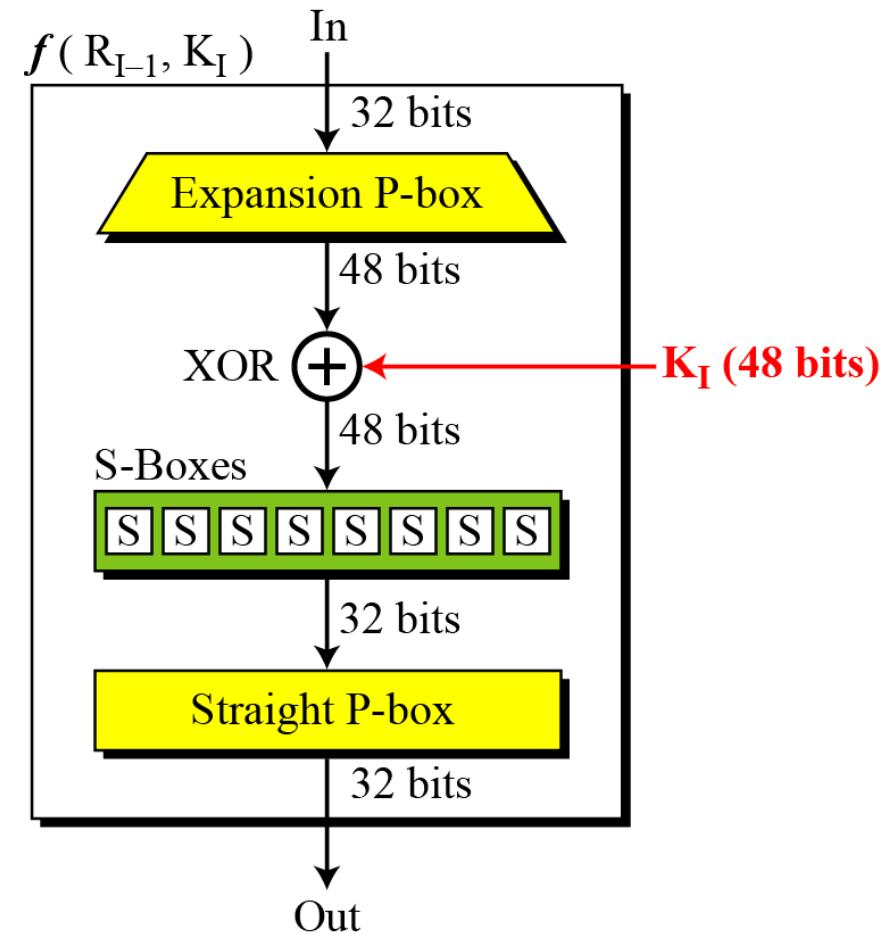
a. Encryption round



b. Decryption round

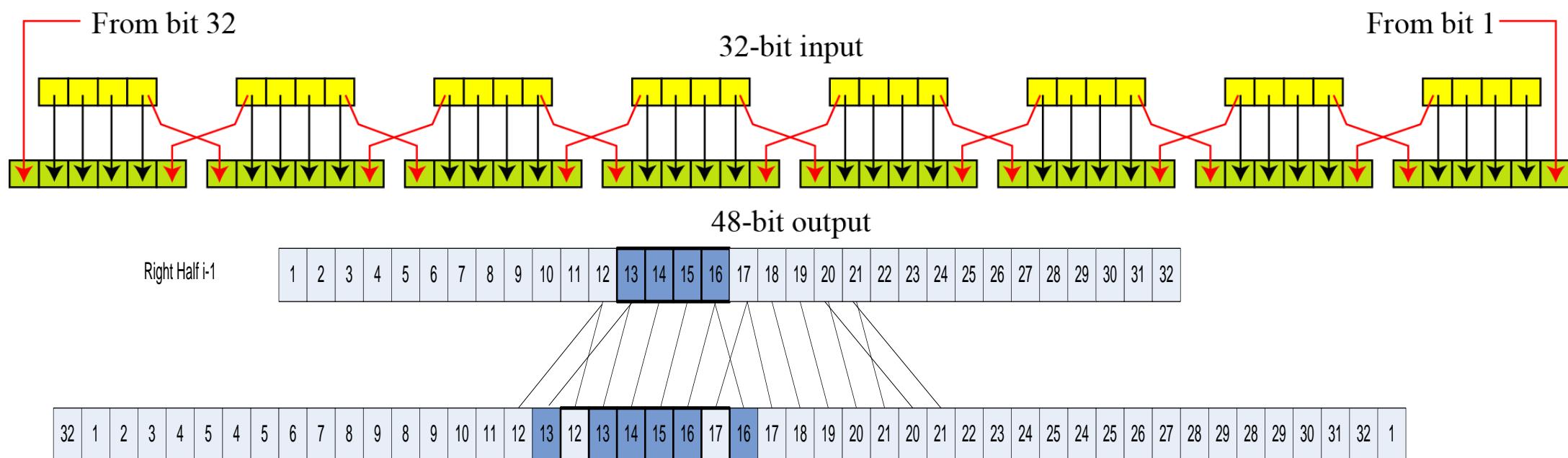
# Round structure

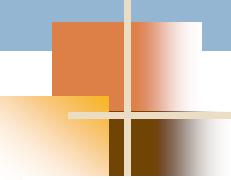
- Uses two 32-bit L & R halves
- As for any Feistel cipher can describe as:  
$$L_i = R_{i-1}$$
  
$$R_i = L_{i-1} \oplus f(R_{i-1}, k_i)$$
- F takes 32-bit R half and 48-bit subkey:
  - Expands R to 48-bits using permutation
  - Adds to subkey using XOR
  - Passes through 8 s-boxes to get 32-bit result
  - Finally permutes using 32-bit permutation



# DES function : 4 section

- Expansion P-box: $R_{i-1}$  is divided Eight 4-bit sections. Each 4-bit section is expanded into 6-bit each.
- Input bit 1,2,3,4 are copied to output bits 2,3,4,5
- Output bit 1 comes from previous bit 4
- Output bit 6 comes from bit 1 of next section





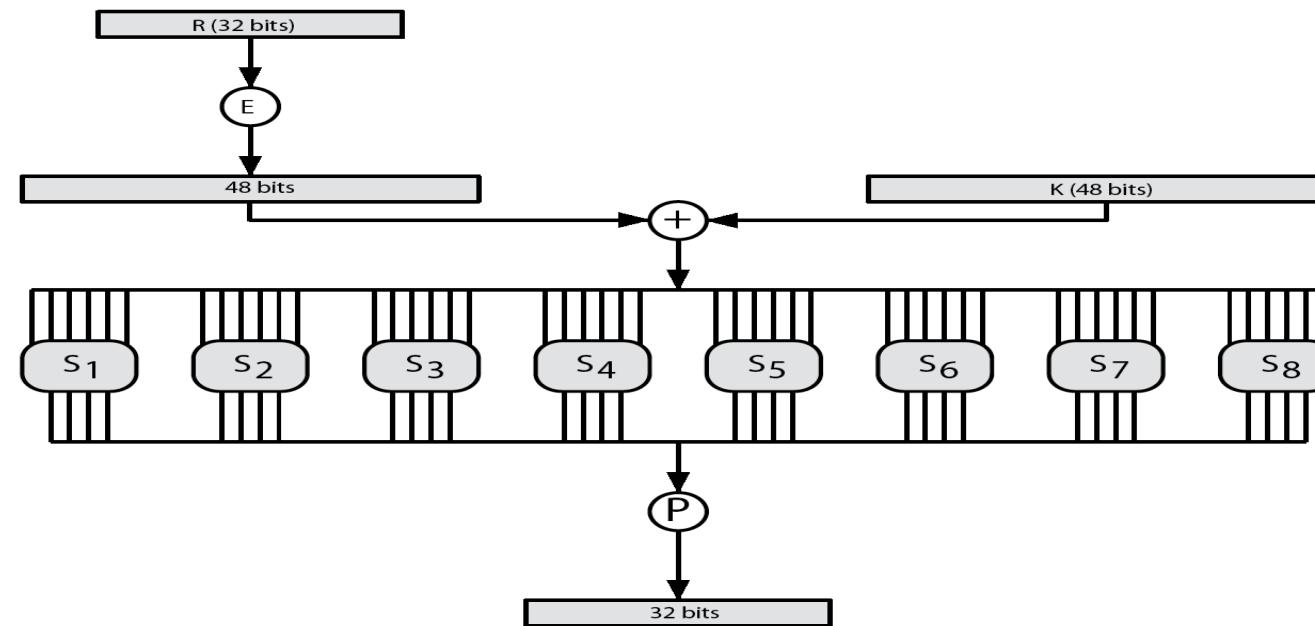
*Although the relationship between the input and output can be defined mathematically, DES uses Table to define this P-box.*

*Expansion P-box table*

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

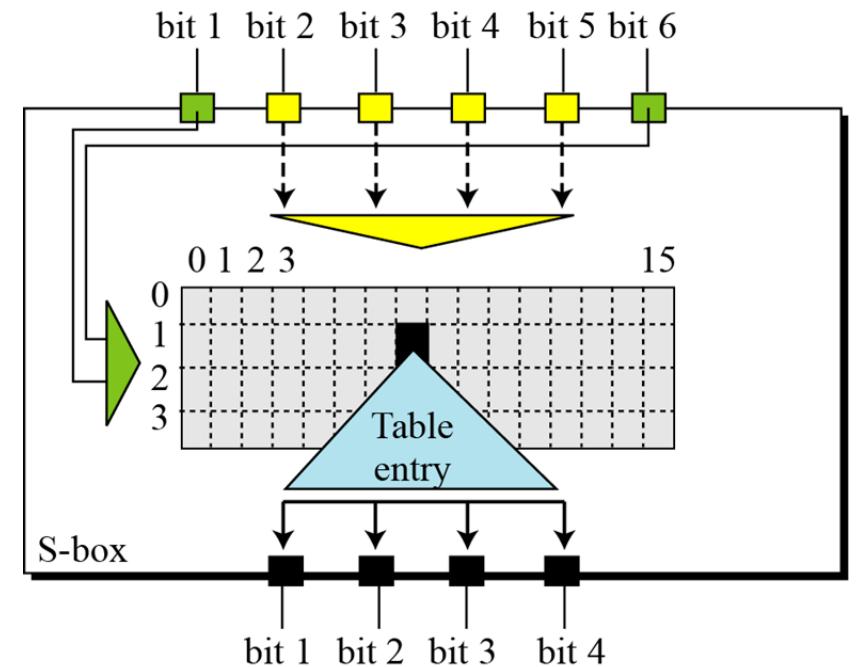
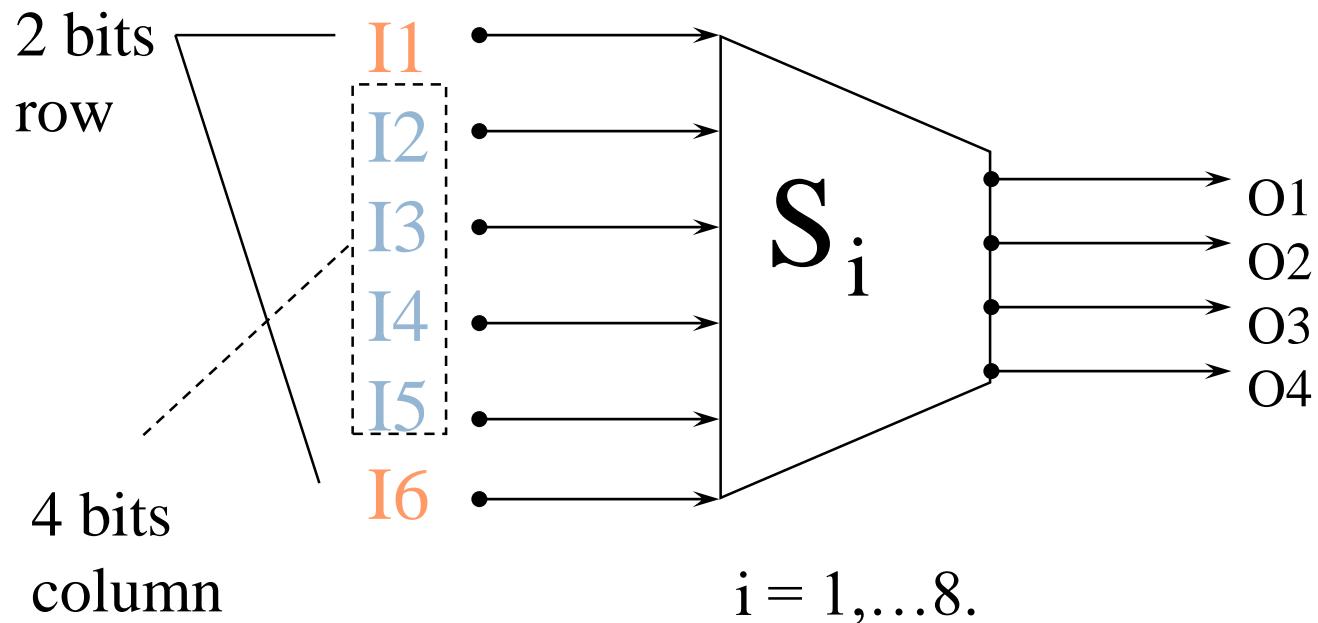
# Whitener (XOR)

- After the expansion permutation ,DES uses the XOR operation on the expanded right section and the round key.
- S-BOX: The S-box do the real mixing(confusion)

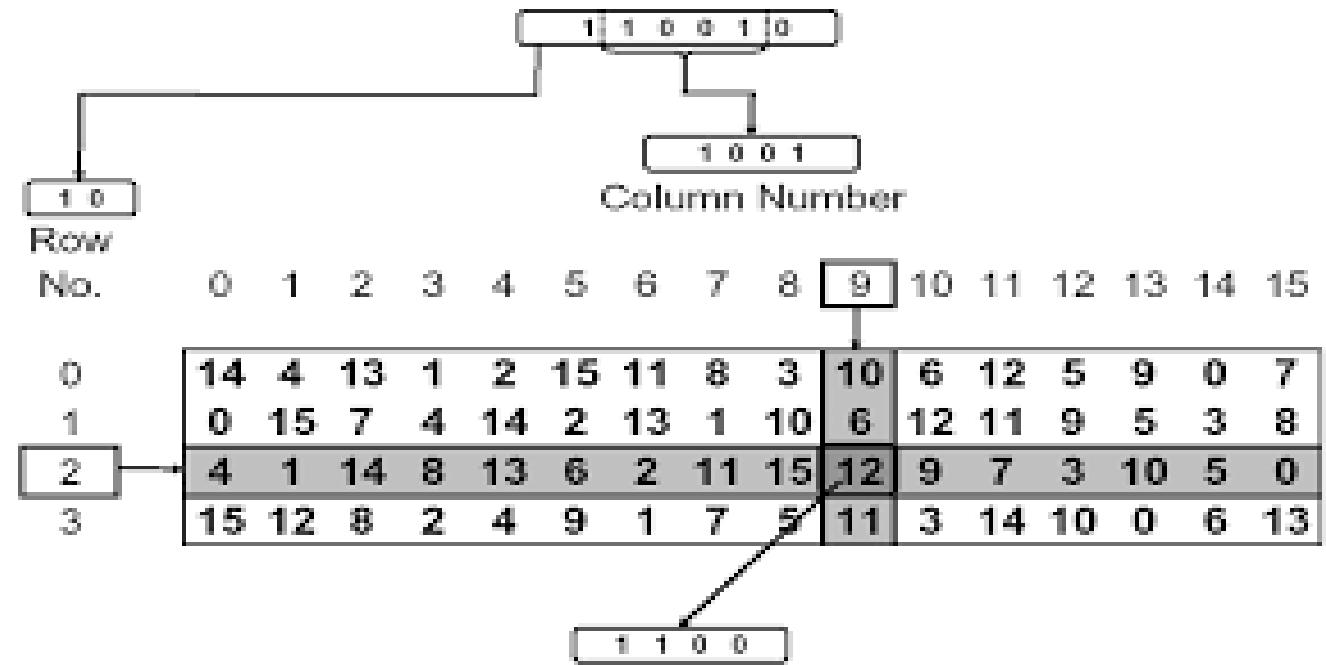
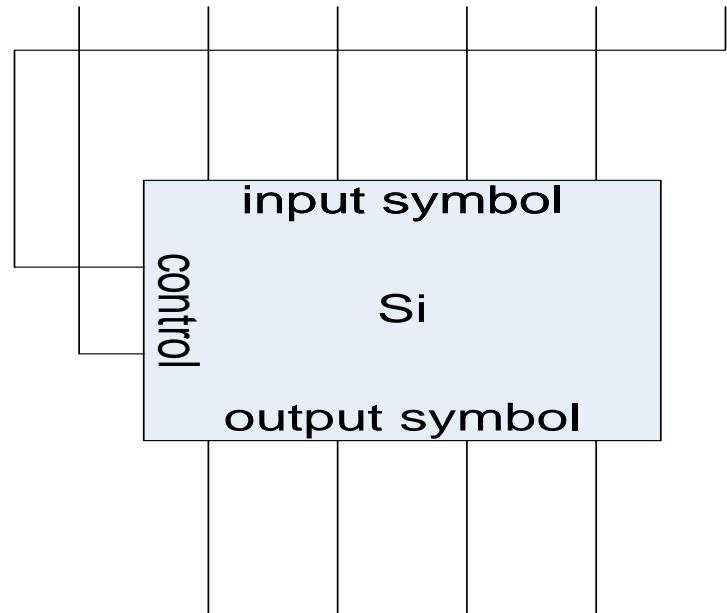


# S-Box (Substitute and Shrink)

- 48 bits ==> 32 bits. ( $8 \times 6 ==> 8 \times 4$ )
- 2 bits used to select amongst 4 substitutions for the rest of the 4-bit quantity



# Substitution Boxes S



- each of the eight s-boxes is different
- each s-box reduces 6 bits to 4 bits
- so the 8 s-boxes implement the 48-bit to 32-bit contraction substitution

**S-Box with Table entries in decimal**  
**Each row and column contain different numbers**

### S1: one of the S-boxes

Each row and column contain different numbers.

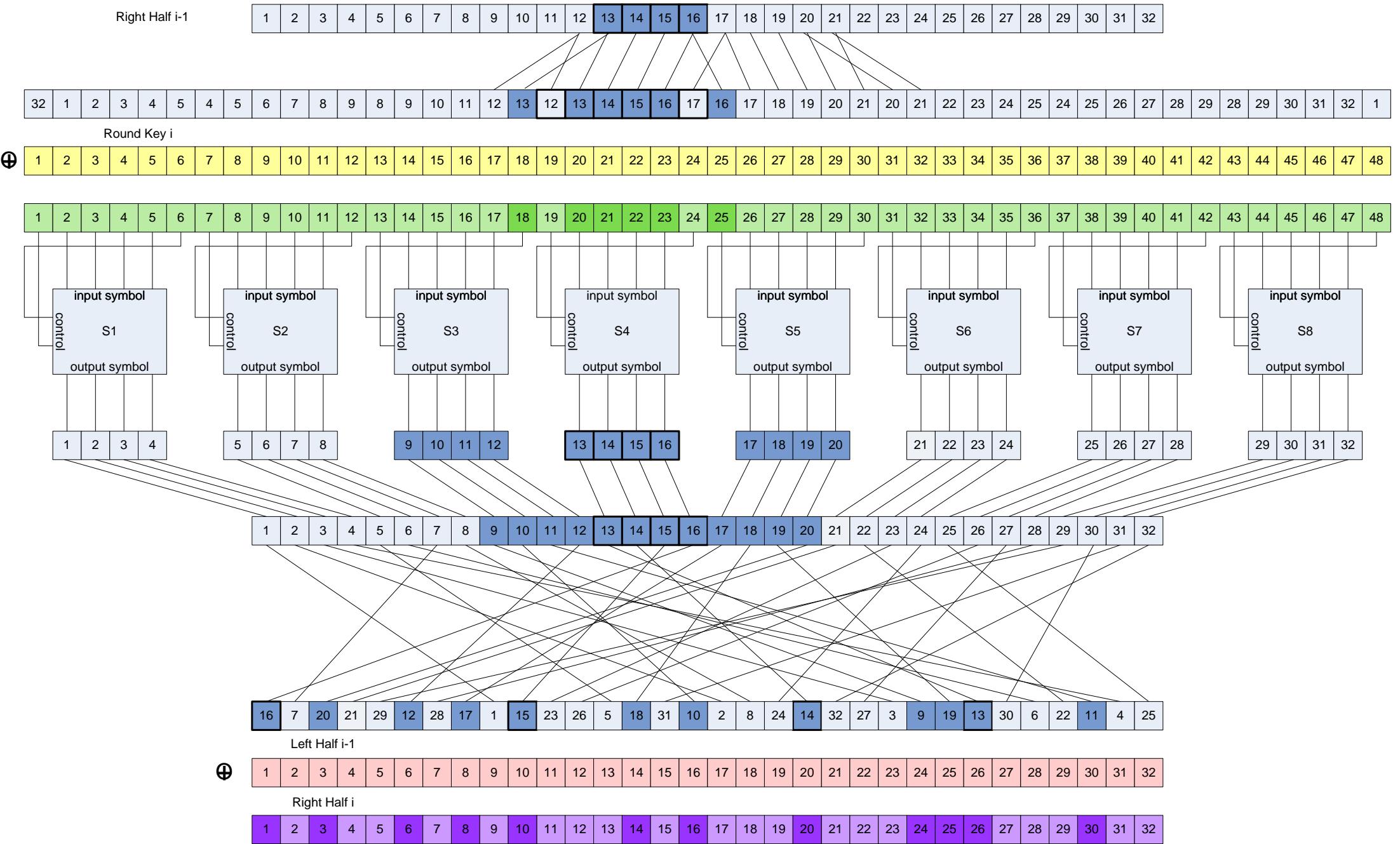
	0	1	2	3	4	5	6	7	8	9.... 15
0	14	4	13	1	2	15	11	8	3	
1	0	15	7	4	14	2	13	1	10	
2	4	1	14	8	13	6	2	11	15	
3	15	12	8	2	4	9	1	7	5	

Example: input: 100110 output: ???

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

# S-boxes (S1-S8)

<b>S<sub>1</sub></b>	<table border="1"> <tbody> <tr><td>14</td><td>4</td><td>13</td><td>1</td><td>2</td><td>15</td><td>11</td><td>8</td><td>3</td><td>10</td><td>6</td><td>12</td><td>5</td><td>9</td><td>0</td><td>7</td></tr> <tr><td>0</td><td>15</td><td>7</td><td>4</td><td>14</td><td>2</td><td>13</td><td>1</td><td>10</td><td>6</td><td>12</td><td>11</td><td>9</td><td>5</td><td>3</td><td>8</td></tr> <tr><td>4</td><td>1</td><td>14</td><td>8</td><td>13</td><td>6</td><td>2</td><td>11</td><td>15</td><td>12</td><td>9</td><td>7</td><td>3</td><td>10</td><td>5</td><td>0</td></tr> <tr><td>15</td><td>12</td><td>8</td><td>2</td><td>4</td><td>9</td><td>1</td><td>7</td><td>5</td><td>11</td><td>3</td><td>14</td><td>10</td><td>0</td><td>6</td><td>13</td></tr> </tbody> </table>	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7																																																		
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8																																																		
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0																																																		
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13																																																		
<b>S<sub>2</sub></b>	<table border="1"> <tbody> <tr><td>15</td><td>1</td><td>8</td><td>14</td><td>6</td><td>11</td><td>3</td><td>4</td><td>9</td><td>7</td><td>2</td><td>13</td><td>12</td><td>0</td><td>5</td><td>10</td></tr> <tr><td>3</td><td>13</td><td>4</td><td>7</td><td>15</td><td>2</td><td>8</td><td>14</td><td>12</td><td>0</td><td>1</td><td>10</td><td>6</td><td>9</td><td>11</td><td>5</td></tr> <tr><td>0</td><td>14</td><td>7</td><td>11</td><td>10</td><td>4</td><td>13</td><td>1</td><td>5</td><td>8</td><td>12</td><td>6</td><td>9</td><td>3</td><td>2</td><td>15</td></tr> <tr><td>13</td><td>8</td><td>10</td><td>1</td><td>3</td><td>15</td><td>4</td><td>2</td><td>11</td><td>6</td><td>7</td><td>12</td><td>0</td><td>5</td><td>14</td><td>9</td></tr> </tbody> </table>	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10																																																		
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5																																																		
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15																																																		
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9																																																		
<b>S<sub>3</sub></b>	<table border="1"> <tbody> <tr><td>10</td><td>0</td><td>9</td><td>14</td><td>6</td><td>3</td><td>15</td><td>5</td><td>1</td><td>13</td><td>12</td><td>7</td><td>11</td><td>4</td><td>2</td><td>8</td></tr> <tr><td>13</td><td>7</td><td>0</td><td>9</td><td>3</td><td>4</td><td>6</td><td>10</td><td>2</td><td>8</td><td>5</td><td>14</td><td>12</td><td>11</td><td>15</td><td>1</td></tr> <tr><td>13</td><td>6</td><td>4</td><td>9</td><td>8</td><td>15</td><td>3</td><td>0</td><td>11</td><td>1</td><td>2</td><td>12</td><td>5</td><td>10</td><td>14</td><td>7</td></tr> <tr><td>1</td><td>10</td><td>13</td><td>0</td><td>6</td><td>9</td><td>8</td><td>7</td><td>4</td><td>15</td><td>14</td><td>3</td><td>11</td><td>5</td><td>2</td><td>12</td></tr> </tbody> </table>	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8																																																		
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1																																																		
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7																																																		
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12																																																		
<b>S<sub>4</sub></b>	<table border="1"> <tbody> <tr><td>7</td><td>13</td><td>14</td><td>3</td><td>0</td><td>6</td><td>9</td><td>10</td><td>1</td><td>2</td><td>8</td><td>5</td><td>11</td><td>12</td><td>4</td><td>15</td></tr> <tr><td>13</td><td>8</td><td>11</td><td>5</td><td>6</td><td>15</td><td>0</td><td>3</td><td>4</td><td>7</td><td>2</td><td>12</td><td>1</td><td>10</td><td>14</td><td>9</td></tr> <tr><td>10</td><td>6</td><td>9</td><td>0</td><td>12</td><td>11</td><td>7</td><td>13</td><td>15</td><td>1</td><td>3</td><td>14</td><td>5</td><td>2</td><td>8</td><td>4</td></tr> <tr><td>3</td><td>15</td><td>0</td><td>6</td><td>10</td><td>1</td><td>13</td><td>8</td><td>9</td><td>4</td><td>5</td><td>11</td><td>12</td><td>7</td><td>2</td><td>14</td></tr> </tbody> </table>	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15																																																		
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9																																																		
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4																																																		
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14																																																		
<b>S<sub>5</sub></b>	<table border="1"> <tbody> <tr><td>2</td><td>12</td><td>4</td><td>1</td><td>7</td><td>10</td><td>11</td><td>6</td><td>8</td><td>5</td><td>3</td><td>15</td><td>13</td><td>0</td><td>14</td><td>9</td></tr> <tr><td>14</td><td>11</td><td>2</td><td>12</td><td>4</td><td>7</td><td>13</td><td>1</td><td>5</td><td>0</td><td>15</td><td>10</td><td>3</td><td>9</td><td>8</td><td>6</td></tr> <tr><td>4</td><td>2</td><td>1</td><td>11</td><td>10</td><td>13</td><td>7</td><td>8</td><td>15</td><td>9</td><td>12</td><td>5</td><td>6</td><td>3</td><td>0</td><td>14</td></tr> <tr><td>11</td><td>8</td><td>12</td><td>7</td><td>1</td><td>14</td><td>2</td><td>13</td><td>6</td><td>15</td><td>0</td><td>9</td><td>10</td><td>4</td><td>5</td><td>3</td></tr> </tbody> </table>	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9																																																		
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6																																																		
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14																																																		
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3																																																		
<b>S<sub>6</sub></b>	<table border="1"> <tbody> <tr><td>12</td><td>1</td><td>10</td><td>15</td><td>9</td><td>2</td><td>6</td><td>8</td><td>0</td><td>13</td><td>3</td><td>4</td><td>14</td><td>7</td><td>5</td><td>11</td></tr> <tr><td>10</td><td>15</td><td>4</td><td>2</td><td>7</td><td>12</td><td>9</td><td>5</td><td>6</td><td>1</td><td>13</td><td>14</td><td>0</td><td>11</td><td>3</td><td>8</td></tr> <tr><td>9</td><td>14</td><td>15</td><td>5</td><td>2</td><td>8</td><td>12</td><td>3</td><td>7</td><td>0</td><td>4</td><td>10</td><td>1</td><td>13</td><td>11</td><td>6</td></tr> <tr><td>4</td><td>3</td><td>2</td><td>12</td><td>9</td><td>5</td><td>15</td><td>10</td><td>11</td><td>14</td><td>1</td><td>7</td><td>6</td><td>0</td><td>8</td><td>13</td></tr> </tbody> </table>	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11																																																		
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8																																																		
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6																																																		
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13																																																		
<b>S<sub>7</sub></b>	<table border="1"> <tbody> <tr><td>4</td><td>11</td><td>2</td><td>14</td><td>15</td><td>0</td><td>8</td><td>13</td><td>3</td><td>12</td><td>9</td><td>7</td><td>5</td><td>10</td><td>6</td><td>1</td></tr> <tr><td>13</td><td>0</td><td>11</td><td>7</td><td>4</td><td>9</td><td>1</td><td>10</td><td>14</td><td>3</td><td>5</td><td>12</td><td>2</td><td>15</td><td>8</td><td>6</td></tr> <tr><td>1</td><td>4</td><td>11</td><td>13</td><td>12</td><td>3</td><td>7</td><td>14</td><td>10</td><td>15</td><td>6</td><td>8</td><td>0</td><td>5</td><td>9</td><td>2</td></tr> <tr><td>6</td><td>11</td><td>13</td><td>8</td><td>1</td><td>4</td><td>10</td><td>7</td><td>9</td><td>5</td><td>0</td><td>15</td><td>14</td><td>2</td><td>3</td><td>12</td></tr> </tbody> </table>	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1																																																		
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6																																																		
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2																																																		
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12																																																		
<b>S<sub>8</sub></b>	<table border="1"> <tbody> <tr><td>13</td><td>2</td><td>8</td><td>4</td><td>6</td><td>15</td><td>11</td><td>1</td><td>10</td><td>9</td><td>3</td><td>14</td><td>5</td><td>0</td><td>12</td><td>7</td></tr> <tr><td>1</td><td>15</td><td>13</td><td>8</td><td>10</td><td>3</td><td>7</td><td>4</td><td>12</td><td>5</td><td>6</td><td>11</td><td>0</td><td>14</td><td>9</td><td>2</td></tr> <tr><td>7</td><td>11</td><td>4</td><td>1</td><td>9</td><td>12</td><td>14</td><td>2</td><td>0</td><td>6</td><td>10</td><td>13</td><td>15</td><td>3</td><td>5</td><td>8</td></tr> <tr><td>2</td><td>1</td><td>14</td><td>7</td><td>4</td><td>10</td><td>8</td><td>13</td><td>15</td><td>12</td><td>9</td><td>0</td><td>3</td><td>5</td><td>6</td><td>11</td></tr> </tbody> </table>	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7																																																		
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2																																																		
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8																																																		
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11																																																		

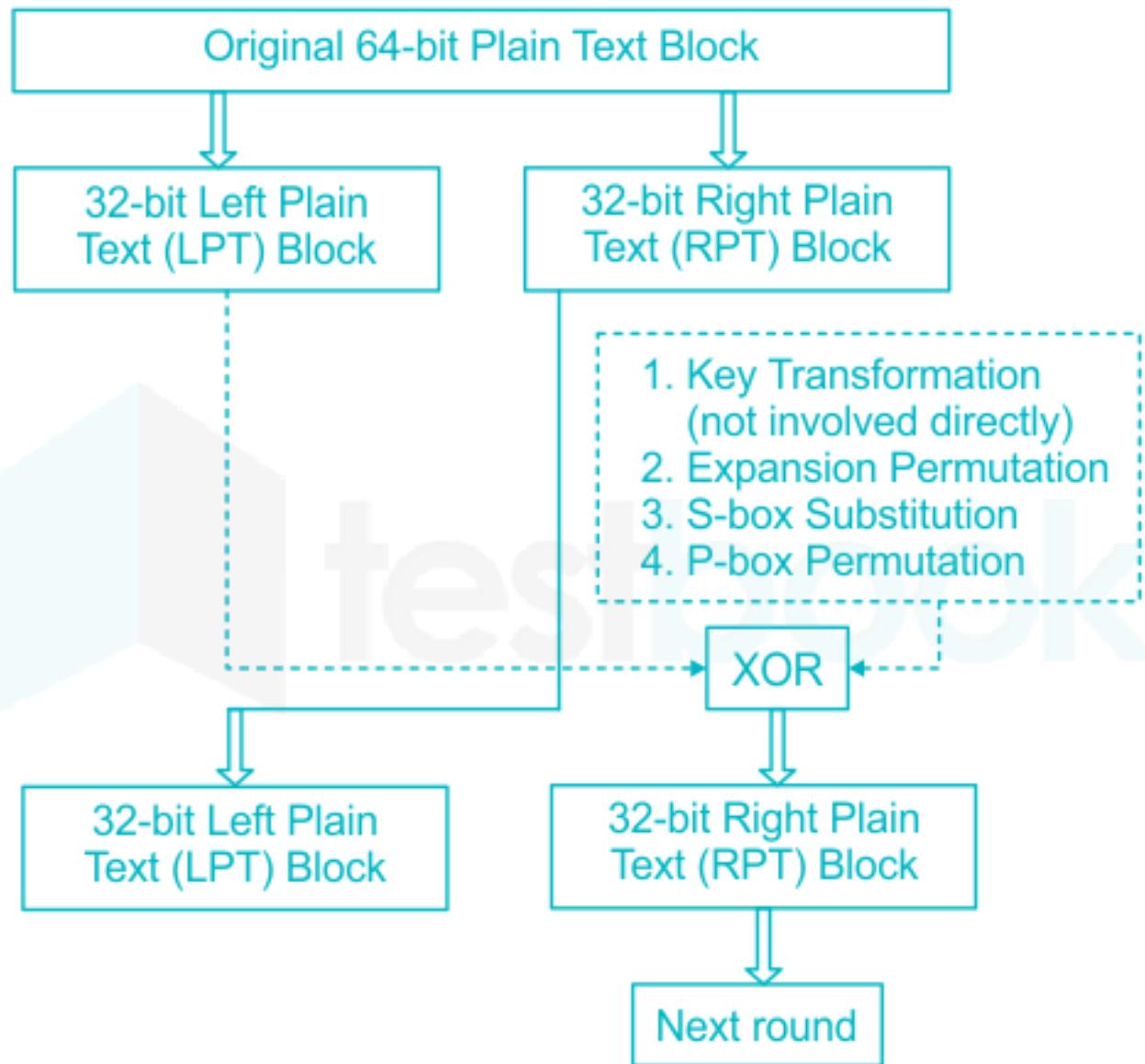


# *Straight Permutation*

**Straight permutation table**

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

# XOR and Swap



# DES ANALYSIS

- Two desired properties of a block cipher are the *avalanche effect* and the *completeness*.

Avalanche effect means a small change in the plaintext (or key) should create a significant change in the ciphertext. DES has been proved to be strong with regard to this property.

- 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

# *Design Criteria*

- *S-Boxes*
- *The design provides confusion and diffusion of bits from each round to the next.*
- *P-Boxes*
- *They provide diffusion of bits*
- *Number of Rounds*
- *DES uses sixteen rounds of Feistel ciphers. the cipher text is thoroughly a random function of plaintext and cipher text.*

# *DES Weaknesses*

- *Weaknesses in Cipher Design*
- 1. *Weaknesses in S-boxes*
- 2. *Weaknesses in P-boxes*
- 3. *Weaknesses in Key*

A weak key is the one which after parity drop operation, consists either of all 0's, all 1's or half 0's and half 1's.

- Four out of the 256 keys are weak keys.

# Example of weak key

- **Keys before parity drop**

- FEFE FEFE FEFE FEFE
- EOEO EOEO F1F1 F1F1
- 1F1F 1F1F OEOE OEOE
- 0101 0101 0101 0101

- **Keys after parity drop**

- FFFFFFF FFFFFFF
- FFFFFFF 0000000
- 0000000 FFFFFFF
- 0000000 0000000

# Consequence of weak keys

- The round keys created from any of these weak keys are the same.
  - For example, for the first weak key, all the round keys are 0.
  - The second key leads to half 0s, and half 1s.
- If we encrypt a block with a weak key and subsequently encrypt the result with the same weak key, we get the original block.

# Semi Weak Keys

- A semi weak key creates only two different round keys and each of them is repeated eight times.
- There are six key pairs that are called semi weak Keys.  
e.g. 01fe 01fe 01fe 01fe
- Possible weak key: A key that create only 4 distinct round key.

- *The major criticism of DES regards its key length.*
- *Fortunately DES is not a group.*
- *This means that we can use double or triple DES to increase the key size.*

# Authentication

# What is authentication?

Positive verification of identity (man or machine)

Verification of a person's claimed identity

Who are you? Prove it.

3 Categories:

What you know

What you have

Who you are

# Means of User Authentication

## Something the individual knows

- Examples include a password, a personal identification number (PIN), or answers to a prearranged set of questions

## Something the individual possesses

- Examples include cryptographic keys, electronic keycards, smart cards, and physical keys
  - This is referred to as a token

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

## Something the individual is (static biometrics)

- Examples include recognition by fingerprint, retina, and face

## Something the individual does (dynamic biometrics)

- Examples include recognition by voice pattern, handwriting characteristics, and typing rhythm

For network-based user authentication, the most important methods involve cryptographic keys and something the individual knows, such as a password

# Physical Biometrics

Fingerprint

- Smell
- Thermal Face

Iris

- Hand Vein

Hand Geometry

- Nail Bed

Finger Geometry

- DNA

Face Geometry

- Palm Print

Ear Shape

Retina

# Behavioral Biometrics

Signature

Voice

Keystroke

Gait

# Authentication Techniques

Four main types of authentication available are:



# Passwords

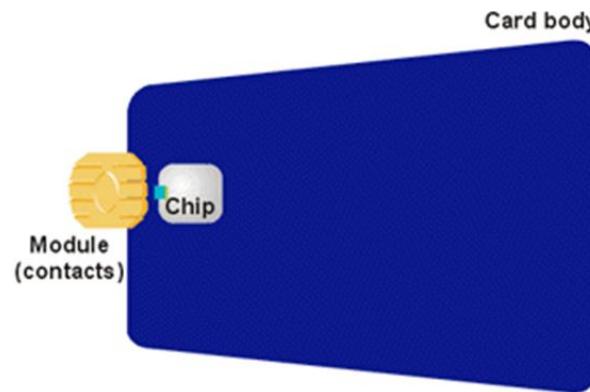
- Sequence of characters
  - Examples: 10 digits, a string of letters, etc.
  - Generated randomly, by user, by computer with user input
- Sequence of words
  - Examples: pass-phrases
- Algorithms
  - Examples: challenge-response, one-time passwords

# Storage

- Store as cleartext
  - If password file compromised, *all* passwords revealed
- Encipher file
  - Need to have decipherment, encipherment keys in memory
  - Reduces to previous problem
- Store one-way hash of password
  - If file read, attacker must still guess passwords or invert the hash

# Tokens

- Cards can also incorporate a variety of technologies, including pictures, magnetic strips containing secret or biometric information, or even a microprocessor and memory in a very small package

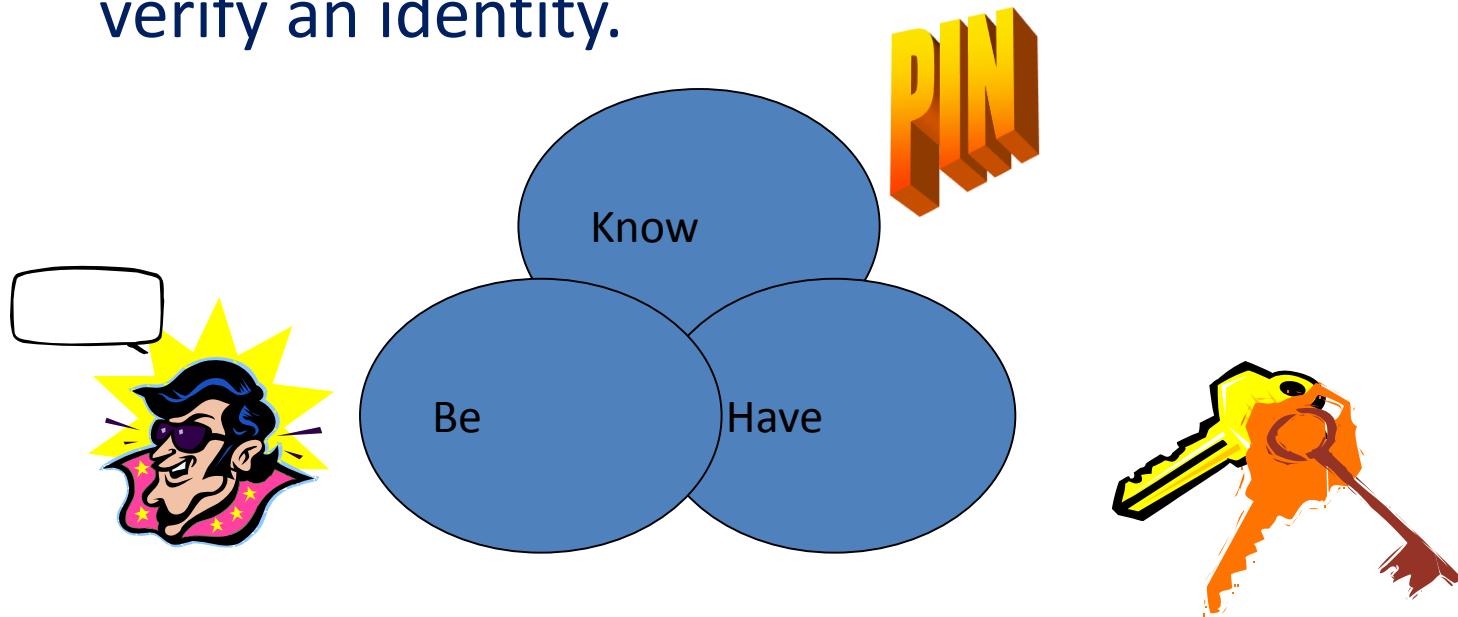


Source: GEMPLUS - All About Smart Cards



# What is Biometrics?

- The automated use of behavioral and physiological characteristics to determine or verify an identity.



# Frauds in industry happens in the following situations:

- Safety deposit boxes and vaults
- Bank transaction like ATM withdrawals
- Access to computers and emails
- Credit Card purchase
- Purchase of house, car, clothes or jewellery
- Getting official documents like birth certificates or passports
- Obtaining court papers
- Drivers licence
- Getting into confidential workplace
- writing Checks

# Why Biometric Application?

- To prevent stealing of possessions that mark the authorised person's identity e.g. security badges, licenses, or properties
- To prevent fraudulent acts like faking ID badges or licenses.
- To ensure safety and security, thus decrease crime rates

# Uses of Biometrics:

- Simple:
  - Verification – Is this who he claims to be?
  - Identification – who is this?
- Advanced:
  - Detecting multiple identities
  - Patrolling public spaces

# Biometrics Today

- Fingerprints
- Retina Prints
- Face Prints
- DNA Identification
- Voice Prints
- Palm Prints
- Handwriting Analysis
- Etc...

# Policy Issues That Effect Biometrics:

- Strong identification may not be necessary or appropriate in many circumstances
  - Voters may be scared off if forced to give a fingerprint
- Authorization can be granted to the *individual* or to the *template*.
  - It is frequently *not necessary* to identify an individual with a name.
- Biometrics and Privacy
  - Long association of biometrics with crime-fighting
  - Biometrics collected for one purpose can be used for another
- Other Issues:
  - Stability of Characteristic over Lifetime
  - Suitability for Logical and Physical Access
  - Difficulty of Usage

# Biometric: authentication process

## 1. Sensing

- User's characteristic must be presented to a sensor
- Output is a function of:
  - Biometric measure
  - The way it is presented
  - Technical characteristics of sensor

## 2. Signal Processing

- Feature extraction
- Extract the desired biometric pattern
  - remove noise and signal losses
  - discard qualities that are not distinctive/repeatable
  - Determine if feature is of “good quality”

# Biometric: authentication process

## 3. Pattern matching

- Sample compared to original signal in database
- Closely matched patterns have “small distances” between them
- Distances will hardly ever be 0 (perfect match)

## 4. Decisions

- Decide if the match is close enough
- Trade-off:
  - ↓ false non-matches leads to ↑false matches
- False Accept Ratio(FAR)
- False Reject Ratio(FRR)

# Biometrics in Detail

# Finger-scan

- A live acquisition of a person's fingerprint.
- Image Acquisition → Image Processing → Template Creation → Template Matching
- Acquisition Devices:
  - Glass plate
  - Electronic
  - Ultrasound



# Fingerprints

## Approaches

- Minutiae- Based
- Image Based

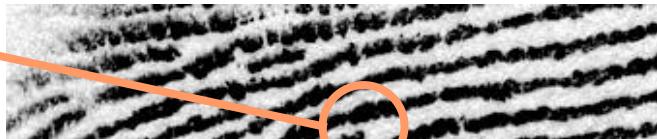
Enclosure



Ridge ending



Island



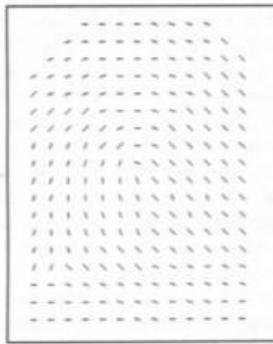
Bifurcation



# Fingerprint Extraction and Matching



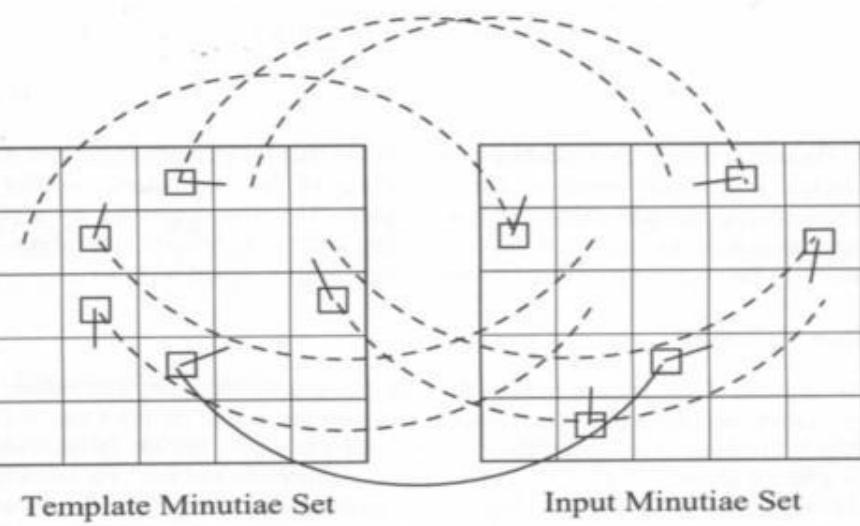
b)



d)



f)

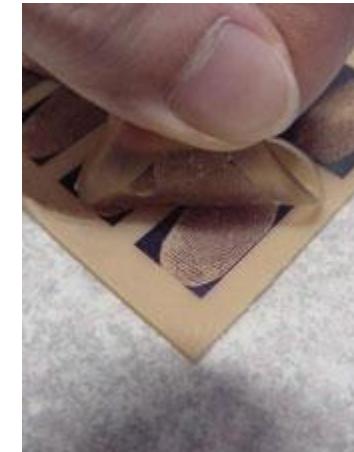


Template Minutiae Set

Input Minutiae Set

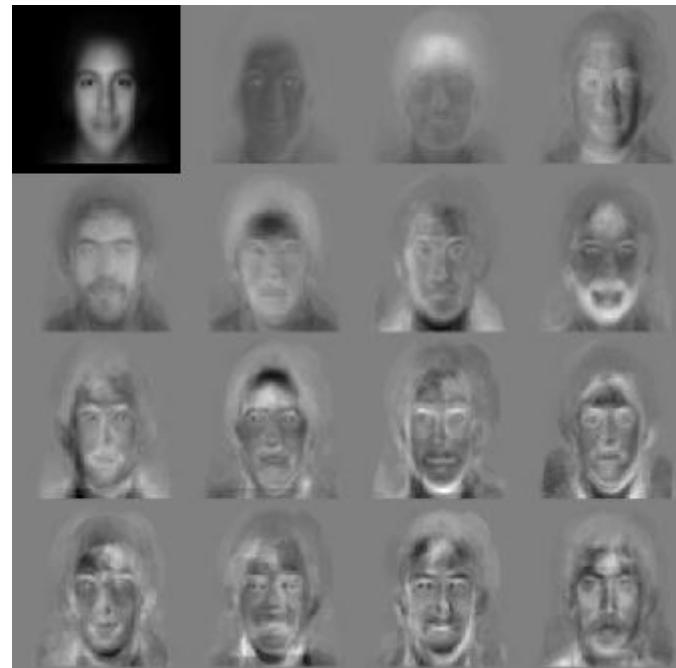
# Fingerprint SWAD

- Strengths:
  - Fingerprints don't change over time
  - Widely believed fingerprints are unique
- Weaknesses:
  - Scars, Age, Diseases
- Attacks:
  - Surgery to alter or remove prints
  - Finger Decapitation
  - “Gummy fingers”
  - Corruption of the database
- Defenses:
  - Measure physical properties of a live finger (pulse)



# Facial Scan

- Based on video Images
- Templates can be based on previously-recorded images
- Technologies:
  - Eigenface Approach
  - Feature Analysis (Visionics)
  - Neural Network



# Facial Scan

- Strengths:
  - Database can be built from driver's license records, visas, etc.
  - Can be applied covertly (surveillance photos).
  - Few people object to having their photo taken
- Weaknesses:
  - No real scientific validation
- Attacks:
  - Surgery
  - Facial Hair
  - Hats
  - Turning away from the camera
- Defenses:
  - Scanning stations with mandated poses

# Iris Scan

- Image Acquisition → Image Processing → Template Creation → Template Matching
- Uses to date:
  - Physical access control
  - Computer authentication



# Iris Scan

- Strengths:
  - 300+ characteristics; 200 required for match
- Weaknesses:
  - Fear
  - Discomfort
  - Proprietary acquisition device
  - Algorithms may not work on all individuals
  - No large databases
- Attacks:
  - Surgery

# Voice Identification

- Speaker Verification
- Verbal Information Verification
- Strengths:
  - Most systems have audio hardware
  - Works over the telephone
  - Can be done covertly
  - Lack of negative perception
- Weaknesses:
  - Background noise (airplanes)
  - No large database of voice samples
- Attacks:
  - Tape recordings
  - Identical twins / sound-alikes

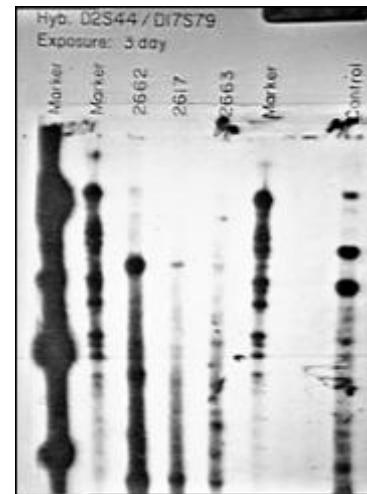
# Hand Scan

- Typical systems measure 90 different features:
  - Overall hand and finger width
  - Distance between joints
  - Bone structure
- Primarily for access control:
  - Machine rooms
  - Olympics
- Strengths:
  - Reasonably robust systems
- Weaknesses:
  - Accuracy is limited; can only be used for 1-to-1 verification
  - Bulky scanner



# DNA Identification

- RFLP - Restriction Fragment Length Polymorphism
- Widely accepted for crime scenes
- Twin problem



# Behavior Biometrics:

- Handwriting (static & dynamic)
  - Add information of timing (dynamics) of movements
    - Signing on an electronic tablet
- Keystroke dynamics
  - Each person has a distinct typing timing/style
    - Hand/finger movements

# Performance Metrics

False acceptance rate

$$FAR = \frac{\text{Total False Acceptances}}{\text{Total Accepted}}$$

False rejection rate.

$$FRR = \frac{\text{Total False Rejections}}{\text{Total Rejected}}$$

# Biometrics-enabled Authentication Applications

1. Cell phones, Laptops, Work Stations, PDA & Handheld device set.



2. Door, Car, Garage Access



3. ATM Access, Smart card



# Biometrics-enabled Identification Applications

1. Forensic : Criminal Tracking  
e.g. Fingerprints, DNA Matching
2. Car park Surveillance
2. Frequent Customers Tracking

# Location Based Authentication

- **Location-based authentication** is a special procedure to prove an individual's identity on appearance simply by detecting its presence at a distinct location.
- If you know where user is, validate identity by seeing if person is where the user is
- Requires special-purpose hardware to locate user
- **GPS (global positioning system)** device gives location signature of entity
- **Host uses LSS (location signature sensor)** to get signature for entity

## THIRD PARTY AUTHENTICATION

KDC and Kerberos



# AUTHENTICATION PROTOCOL

Kerberos

# INTRODUCTION

## WHY WE NEED KERBEROS?

- Kerberos is an authentication protocol, and at the same time a KDC
- Kerberos was developed for Project Athena at MIT
- The name Kerberos signifies a multi headed dog (Greek mythology) as the hound of Hades, is a multi-headed dog that guards the gates of the Underworld to prevent the dead from leaving.
- Scenario: user at workstation need to access service on server distributed throughout the network
  - Threat exist
    - A user gain access to a particular workstation and pretend to be another user operating for that workstation
    - A user alter network address of a workstation
    - A user may eavesdrop and use a replay attack

# INTRODUCTION

- Kerberos provide a centralized authentication server whose function is to authenticate
  - user to server and
  - server to users
- Relies on symmetric encryption
- Two version : 4 and 5
- Kerberos has separated user verification from the process of issuing tickets that allow the user to access different servers.
- Supports client-server applications, such as FTP

# REQUIREMENTS FOR KERBEROS

- A network eavesdropper should not be able to obtain the necessary information to impersonate a user

Secure

Reliable

Scalable

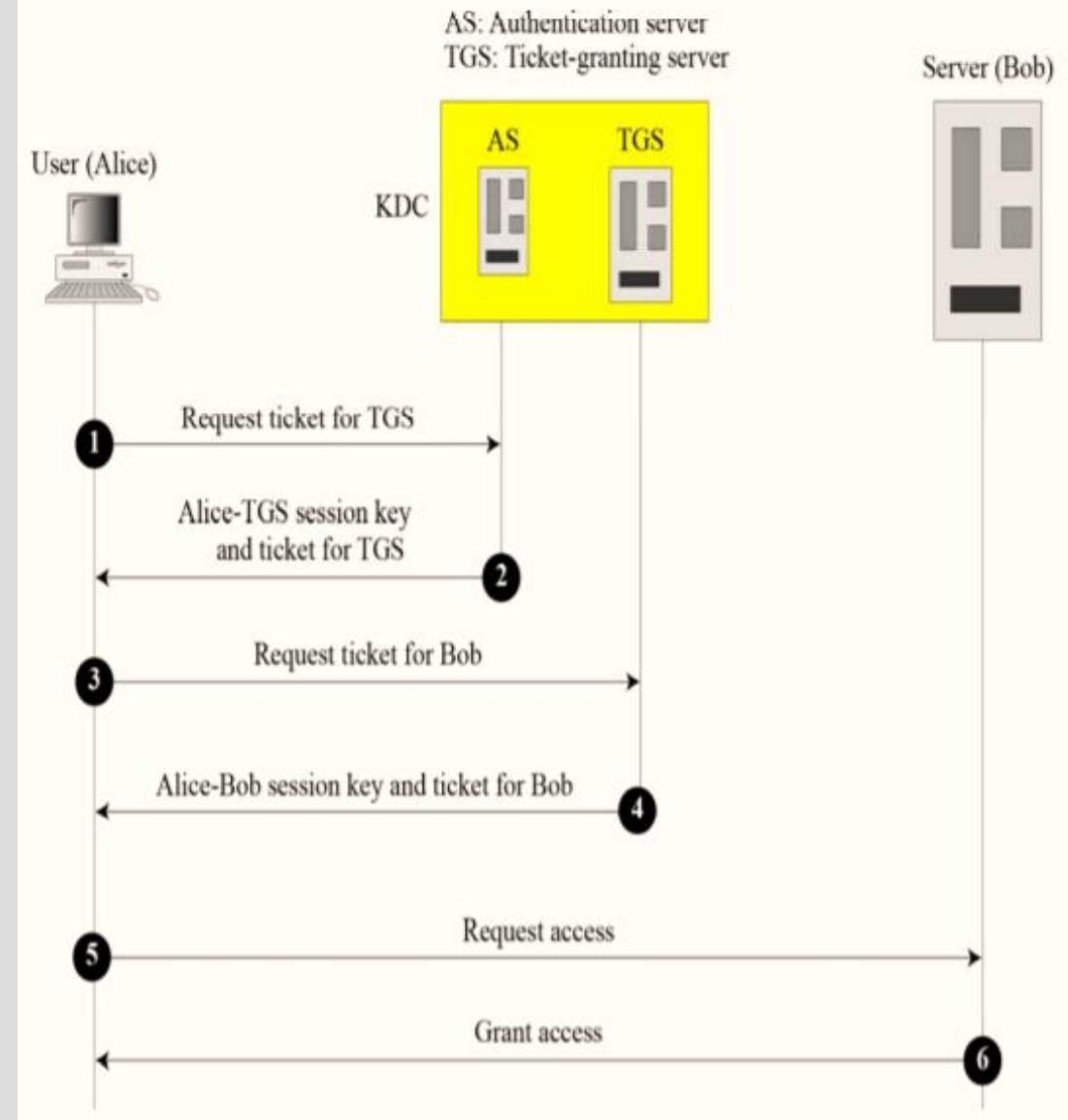
Transparent

- The system should be capable of supporting large numbers of clients and servers

- Ideally, the user should not be aware that authentication is taking place beyond the requirement to enter a password

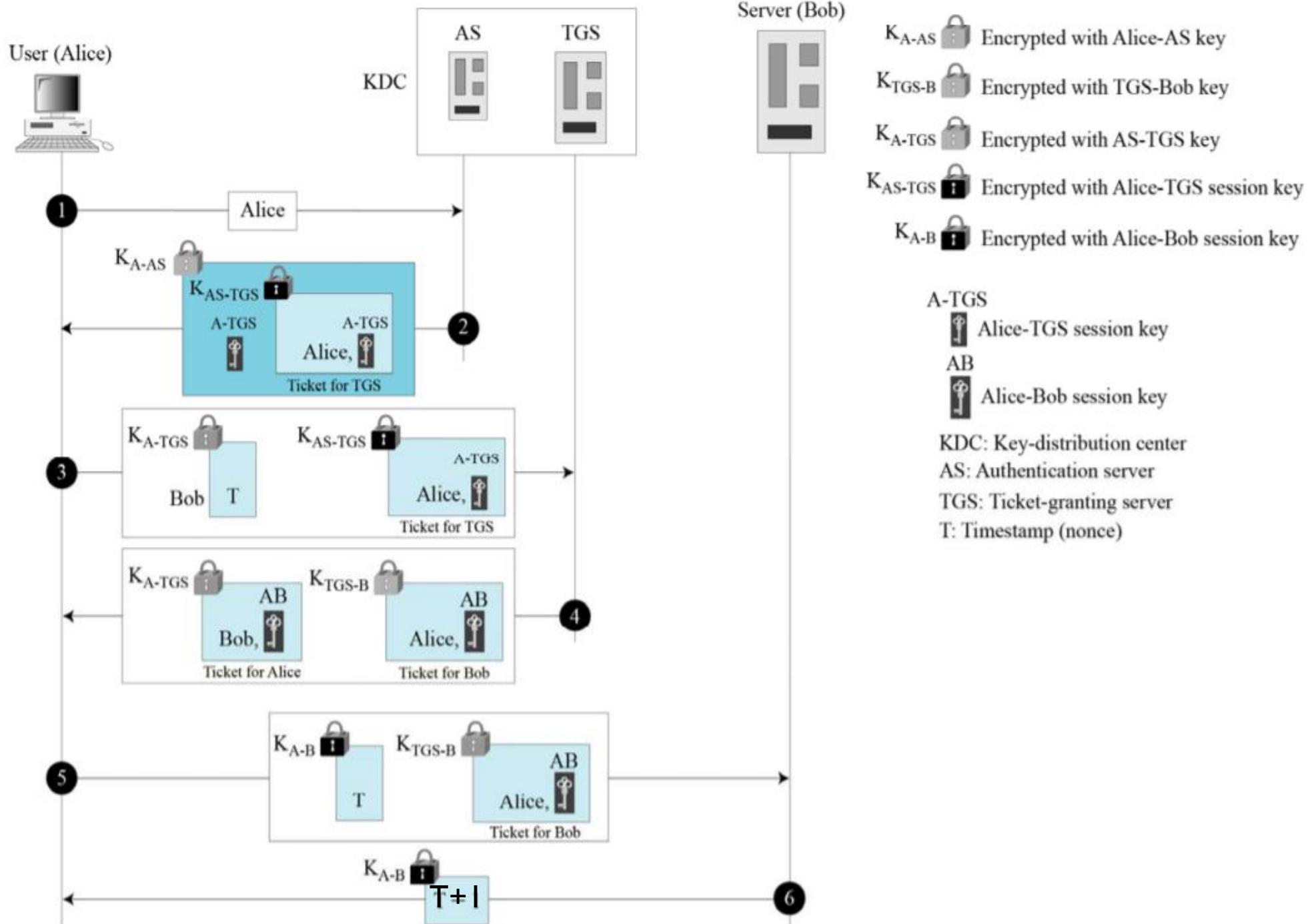
# KERBEROS SERVERS

- **Authentication Server (AS):** is the KDC in the Kerberos protocol. Each user registers with AS and is granted a user ID and password.
- **Ticket-Granting Server (TGS)** issues a ticket for the real server (Bob). It also provides the session key ( $K_{AB}$ ) between the user and the real server.
- **Real Server (Bob)** provides services for the user (Alice).

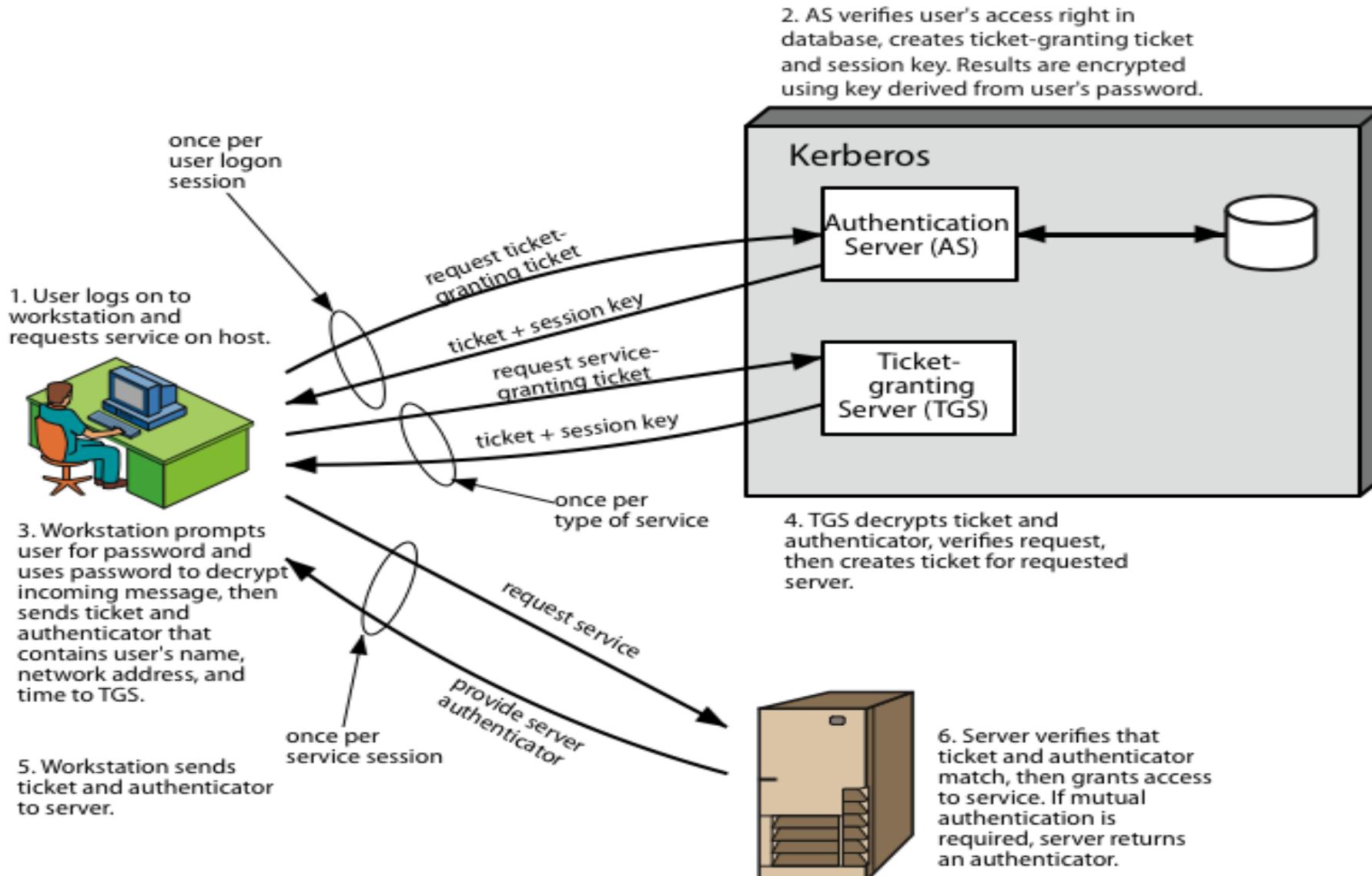


# KERBEROS OPERATION

- The client process (Alice) can access the real server process (Bob) in six steps
1. Alice sends her request to AS in plaintext.
  2. The AS sends a message encrypted with Alice's key  $K_{A-AS}$ .
    - The message contains a session key  $K_{A-TGS}$  that will be used by Alice to contact TGS and a ticket for TGS encrypted using TGS's key  $K_{AS-TGS}$ .(TGT)
    - When the message arrives, Alice types her password which is used by the client process to create  $K_{A-AS}$ , then decrypt the message to extract the session key and the ticket.
  3. Alice sends three items to TGS: The ticket from AS(TGT), the name of the real server (Bob), and a timestamp encrypted with  $K_{A-TGS}$ .
  4. TGS sends to Alice two tickets both containing the session key  $K_{A-B}$  between Alice and Bob .  
Alice's ticket is encrypted with the session key  $K_{A-TGS}$  and Bob's ticket is encrypted with Bob's password/key  $K_{TGS-B}$ .
  5. Alice sends Bob's ticket with the timestamp encrypted with  $K_{A-B}$ .
  6. Bob responds by adding I from the timestamp and encrypts the response with  $K_{A-B}$ .



# KERBEROS 4 OVERVIEW



# KERBEROS V4 DIALOGUE

(1)  $C \rightarrow AS \quad ID_c \parallel ID_{tgs} \parallel TS_1$

(2)  $AS \rightarrow C \quad E(K_c, [K_{c,tgs} \parallel ID_{tgs} \parallel TS_2 \parallel Lifetime_2 \parallel Ticket_{tgs}])$

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

(a) Authentication Service Exchange to obtain ticket-granting ticket

(3)  $C \rightarrow TGS \quad ID_v \parallel Ticket_{tgs} \parallel Authenticator_c$

(4)  $TGS \rightarrow C \quad E(K_{c,tgs}, [K_{c,v} \parallel ID_v \parallel TS_4 \parallel Ticket_v])$

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

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

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

(b) Ticket-Granting Service Exchange to obtain service-granting ticket

(5)  $C \rightarrow V \quad Ticket_v \parallel Authenticator_c$

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

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

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

(c) Client/Server Authentication Exchange to obtain service

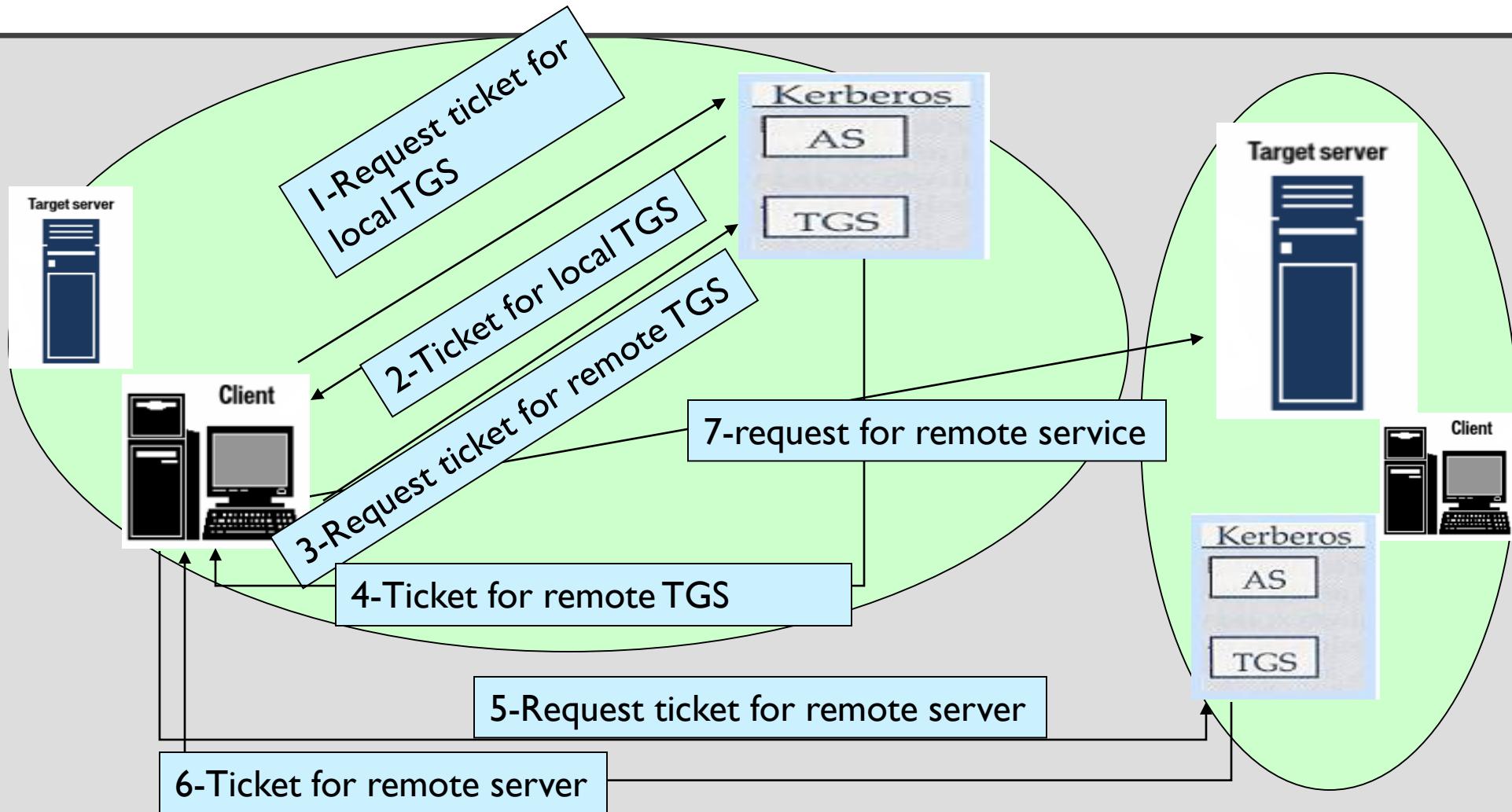
# TERMS IN AUTHENTICATION DIALOGUE

- C = client
- AS = authentication server
- $ID_C$  = identifier of user on C
- $ID_V$  = identifier of server V
- TGS = Ticket-granting server
- $ID_{TGS}$  = Identifier of the TGS
- $Ticket_{TGS}$  = *Ticket-granting ticket or TGT*
- $TS_1$  = timestamp
- $Lifetime_1$  = lifetime for the TGT
- $K_{(C)}$  = key derived from user's password

# KERBEROS REALMS

- a Kerberos environment consists of:
  - a Kerberos server
  - a number of clients, all registered with server
  - application servers, sharing keys with server
- this is termed a *realm*
  - typically a single administrative domain
  - if have multiple realms, their Kerberos servers must share keys and trust

# REQUEST FOR SERVICE IN ANOTHER REALM:



# New Elements in Kerberos Version 5

- **Realm**
  - Indicates realm of the user
- **Options** :Used to request that certain flags be set in the returned ticket.
- **Times**
  - From: the desired start time for the ticket
  - Till: the requested expiration time
  - Rtime: requested renew-till time
- **Nonce**
  - A random number to be repeated in the message back to the client to assure that the response is fresh and has not been replayed by an attacker.

**Table 14.3 Summary of Kerberos Version 5 Message Exchanges**

(1) **C → AS** Options ||  $ID_c$  ||  $Realm_c$  ||  $ID_{tgs}$  || Times ||  $Nonce_1$

(2) **AS → C**  $Realm_c$  ||  $ID_C$  ||  $Ticket_{tgs}$  ||  $E(K_c, [K_{c,tgs} \parallel Times \parallel Nonce_1 \parallel Realm_{tgs} \parallel ID_{tgs}])$

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

**(a) Authentication Service Exchange to obtain ticket-granting ticket**

(3) **C → TGS** Options ||  $ID_v$  || Times || ||  $Nonce_2$  ||  $Ticket_{tgs}$  ||  $Authenticator_c$

(4) **TGS → C**  $Realm_c$  ||  $ID_C$  ||  $Ticket_v$  ||  $E(K_{c,tgs}, [K_{c,v} \parallel Times \parallel Nonce_2 \parallel Realm_v \parallel ID_v])$

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

$$Ticket_v = E(K_v, [Flags \parallel K_{c,v} \parallel Realm_c \parallel ID_C \parallel AD_C \parallel Times])$$

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

**(b) Ticket-Granting Service Exchange to obtain service-granting ticket**

(5) **C → V** Options ||  $Ticket_v$  ||  $Authenticator_c$

(6) **V → C**  $E_{K_{c,v}} [ TS_2 \parallel Subkey \parallel Seq\# ]$

$$Ticket_v = E(K_v, [Flags \parallel K_{c,v} \parallel Realm_c \parallel ID_C \parallel AD_C \parallel Times])$$

$$Authenticator_c = E(K_{c,v}, [ID_C \parallel Realm_c \parallel TS_2 \parallel Subkey \parallel Seq\#])$$

**(c) Client/Server Authentication Exchange to obtain service**

The **subkey field** is a client's choice for an encryption key to be used to protect this specific application session.

If omitted, session key from the ticket  $K_{c,v}$  is used.

The **Sq# field** is an optional field that specifies the starting sequence number to be used by server for messages sent to the client during this session.

Messages may be sequenced numbered to detect replays.

## KERBEROS V5 TICKET FLAGS

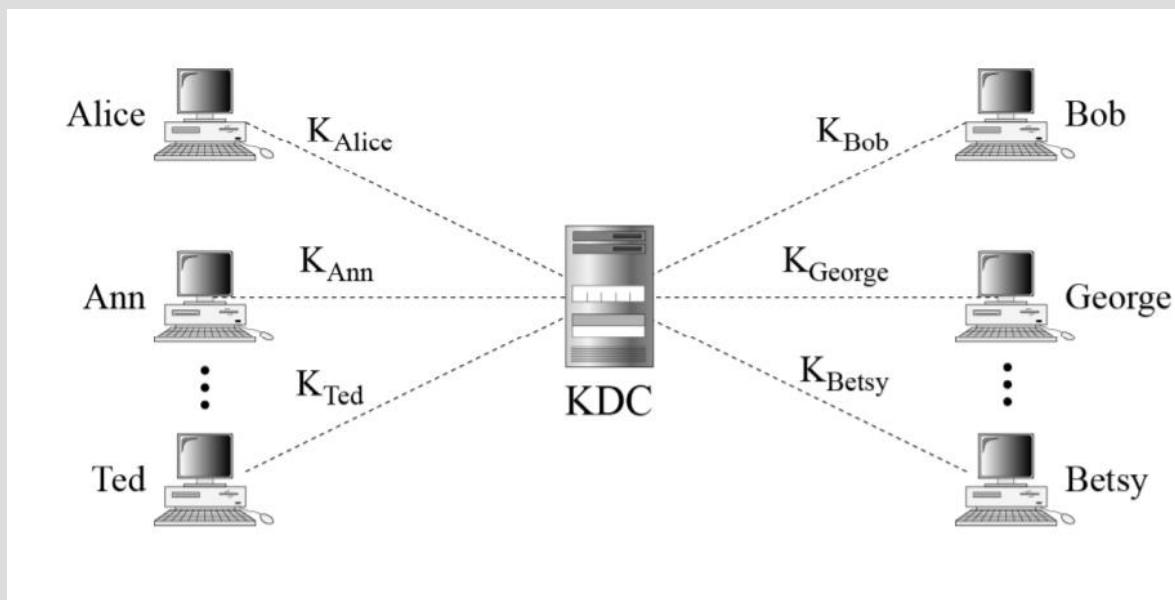
- The flags field was added in Kerberos V5.
- **INITIAL:** This flag indicates that a ticket was issued using the AS protocol and not issued based on a ticket-granting ticket.
- **INVALID:** This flag indicates that a ticket is invalid, which means that application servers must reject tickets which have this flag set.
- **RENEWABLE:** This flag is normally only interpreted by the ticket-granting service, not by application servers, and can be used to obtain a replacement ticket that expires at a later date.
- **POSTDATED:** The POSTDATED flag indicates that a ticket has been postdated.
  - The application server can check the auth-time field in the ticket to see when the original authentication occurred.
  - Some services may choose to reject postdated tickets, or they may only accept them within a certain period after the original authentication.
- **PROXiable:** normally interpreted by the ticket-granting service and ignored by application servers.
  - When set, this flag tells the ticket-granting server that it is OK to issue a new (proxy) 'client' ticket with a different network address based on this ticket.
- **PROXY:** This flag is set in a ticket by the TGS when it issues a proxy ticket.
- **FORWARDABLE:** This flag has an interpretation similar to that of the PROXiable flag, except ticket-granting tickets may also be issued with different network addresses (to be used with remote TGS)

## KERBEROS V4 **VS** V5

- There are a number of noticeable differences between Kerberos versions 4 and 5:
- Protocols used in Kerberos 4 and 5 differ.
- Kerberos 5 supports forwardable, renewable, and post dateable tickets.
- Kerberos 5 is network protocol-independent.
- Kerberos 5 tickets can contain more than one network address.
- Kerberos 5 supports cross-realm authentication.
- Encryption algorithms other than DES can be used in Kerberos 5.

# KDC(KEY DISTRIBUTION CENTER)

- KDC- trusted third party for the distribution of Keys.
- Each person establishes a shared key with the Key-distribution center (KDC).

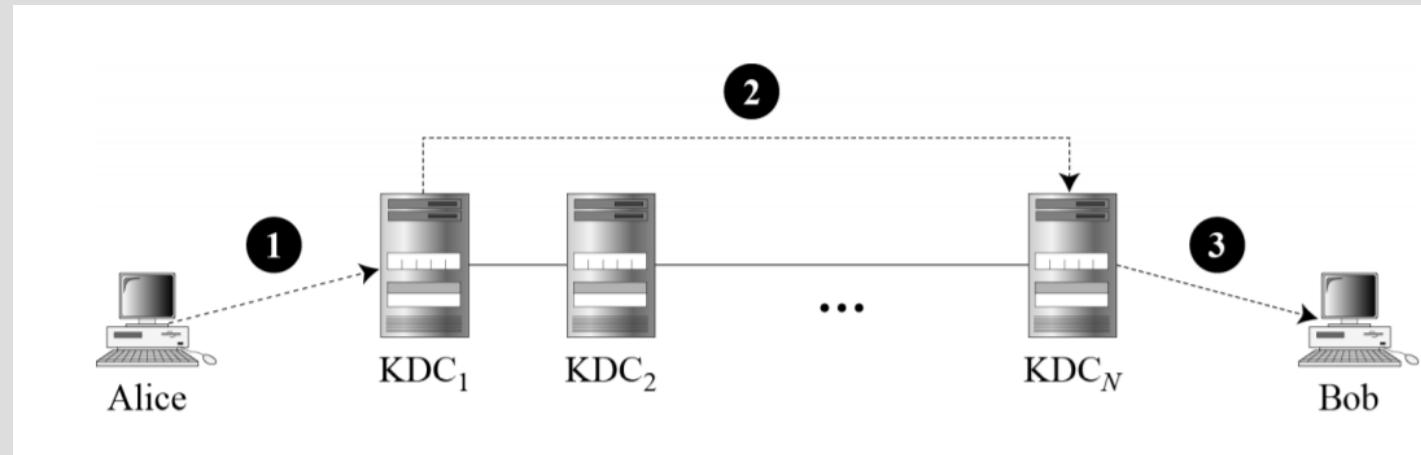


# KDC

- The procedure to get a session key between Alice and Bob is as follows
  - Alice sends a request to KDC stating that she needs a session (temporary) secret key between herself and Bob.
  - Alice uses her secret key with the KDC to authenticate her request and herself to the KDC.
  - The KDC informs Bob about Alice's request.
- If Bob agrees and authenticates himself using his secret key with the KDC, a session key is created between the two.

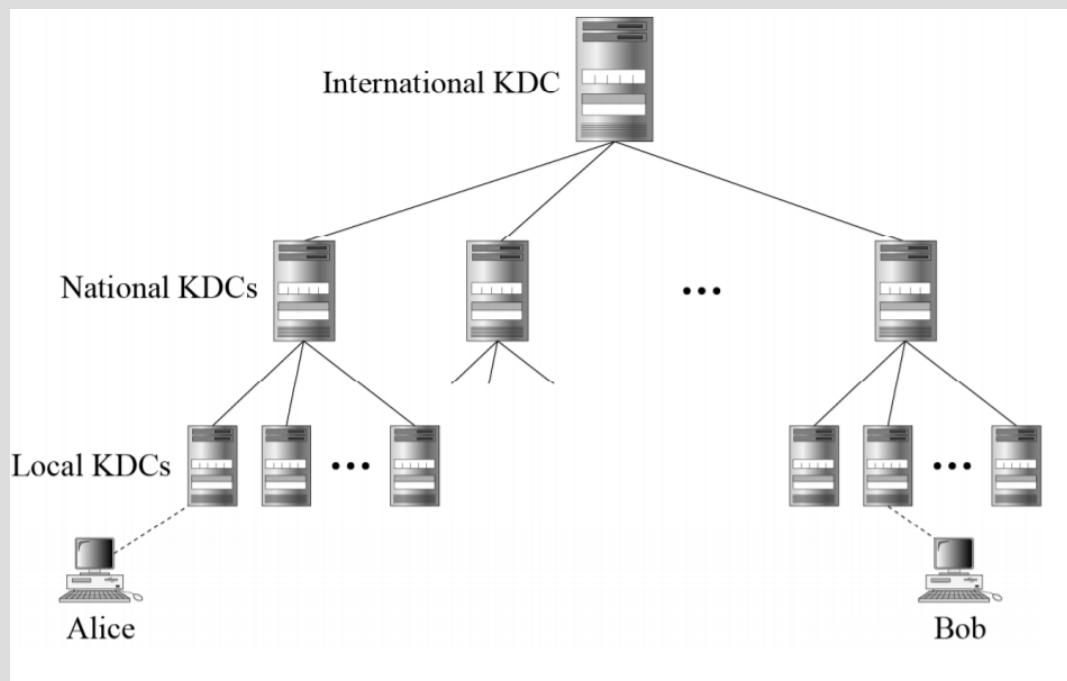
# FLAT MULTIPLE KDCS

- When the number of people using a KDC increases, the system becomes Unmanageable
- Each domain has one KDC (or more if redundancy is desired for fault tolerance).
- If Alice is in one domain and Bob is in another domain, Alice contacts her KDC which in turn contacts the KDC in Bob's domain.
- The two KDC's can create a secret key between Alice and Bob.



# HIERARCHICAL MULTIPLE KDCS

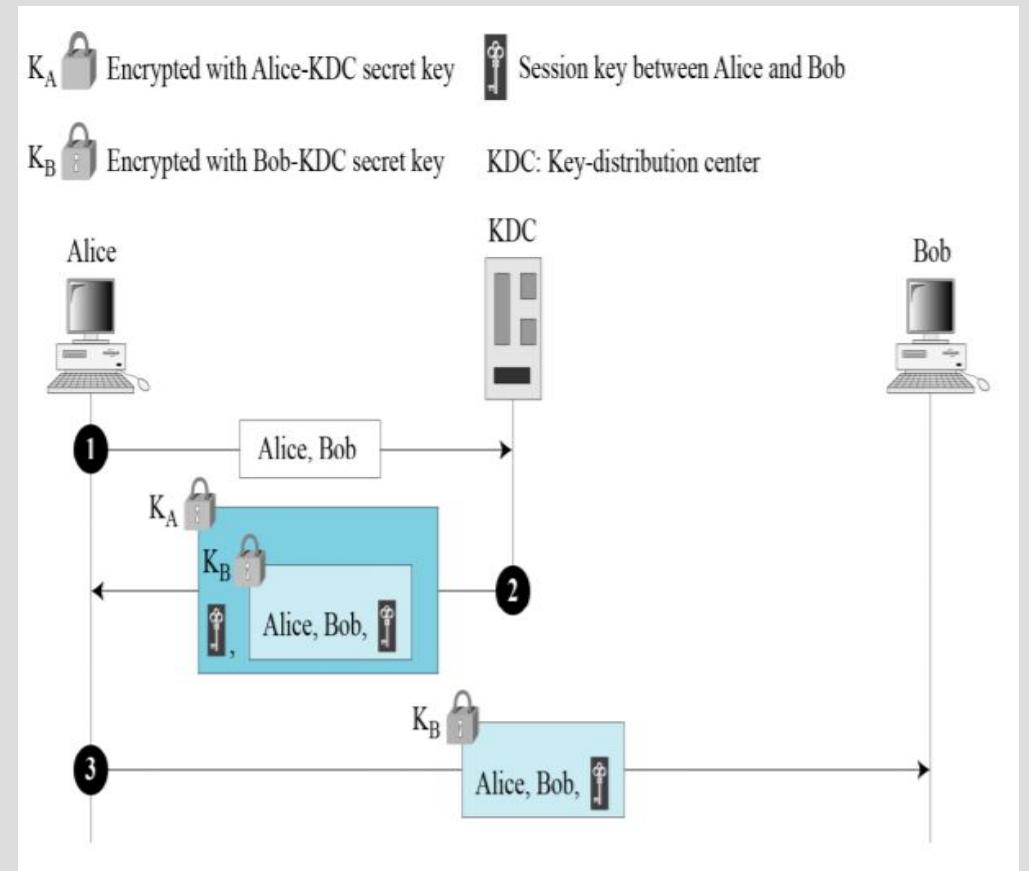
- The hierarchical multiple KDC system has one (or more) KDC at the top of the hierarchy.



# SESSION KEY DISTRIBUTION

- The KDC can help two members (after authenticating their secret key with the KDC) establish a temporary key that can be used by the two members for a single session .
- After communication is terminated, the session key becomes invalid.

A Simple Protocol Using a KDC



## STEPS

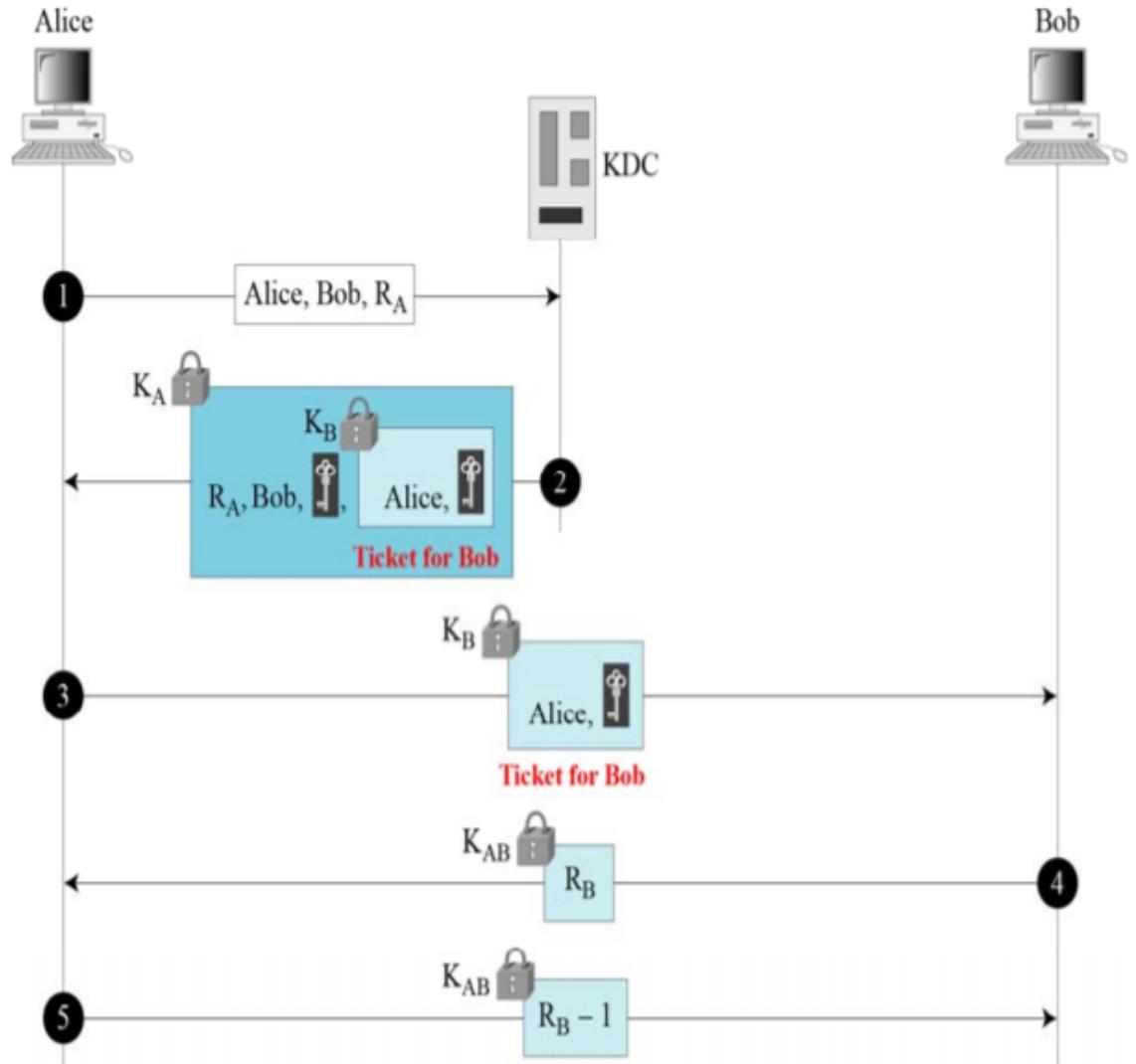
- Alice sends a plaintext message to KDC to request a symmetric session key between herself and Bob.
- The KDC creates a ticket encrypted using Bob's key  $K_B$  containing the session key.
- The ticket and the session key are sent to Alice in a message encrypted using Alice's key  $K_A$ .
- Alice decrypts the message and retrieves the session key and Bob's ticket .
- Alice sends the ticket to Bob who opens (decrypts) the ticket and obtains the value of the session key

# Needham-Schroeder Protocol

1. Alice sends a message to the KDC that includes her nonce,  $R_A$ , her identity, and Bob's identity.
2. The KDC sends an encrypted message to Alice that includes Alice's nonce, Bob's identity, the session key, and an encrypted ticket for Bob. The whole message is encrypted with Alice's key.
3. Alice sends Bob's ticket to him.
4. Bob sends his challenge to Alice ( $R_B$ ), encrypted with the session key.
5. Alice responds to Bob's challenge. Note that the response carries  $R_B - 1$  instead of  $R_B$ .

$K_A$	Encrypted with Alice-KDC secret key
$K_B$	Encrypted with Bob-KDC secret key
$K_{AB}$	Encrypted with Alice-Bob session key
	Session key between Alice and Bob

KDC: Key-distribution center  
 $R_A$ : Alice's nonce  
 $R_B$ : Bob's nonce



## MUTUAL AUTHENTICATION

- It is often necessary for both communicating parties to authenticate themselves to each other.
- For example, in Internet banking, it is imperative that a customer interacts with his/her bank and not some entity posing as the bank.
- Likewise, it is important that a bank to verify the identity of the customer.

## REPLAY ATTACKS

1. The simplest replay attack is one in which the opponent simply copies a message and replays it later
2. An opponent can replay a timestamped message within the valid time window
3. An opponent can replay a timestamped message within the valid time window, but in addition, the opponent suppresses the original message; thus, the repetition cannot be detected
4. Another attack involves a backward replay without modification and is possible if symmetric encryption is used and the sender cannot easily recognize the difference between messages sent and messages received on the basis of content (Reflection Attack)

# APPROACHES TO COPING WITH REPLAY ATTACKS

- Attach a sequence number to each message used in an authentication exchange
  - A new message is accepted only if its sequence number is in the proper order
  - Difficulty with this approach is that it requires each party to keep track of the last sequence number for each claimant it has dealt with
  - Generally not used for authentication and key exchange because of overhead
- Timestamps
  - Requires that clocks among the various participants be synchronized
  - Party A accepts a message as fresh only if the message contains a timestamp that, in A's judgment, is close enough to A's knowledge of current time
- Challenge/response
  - Party A, expecting a fresh message from B, first sends B a nonce (challenge) and requires that the subsequent message (response) received from B contain the correct nonce value
- **Binding** – In all cases, cryptographic means must be used to insure that neither cut-and-paste nor message modification is possible without detection

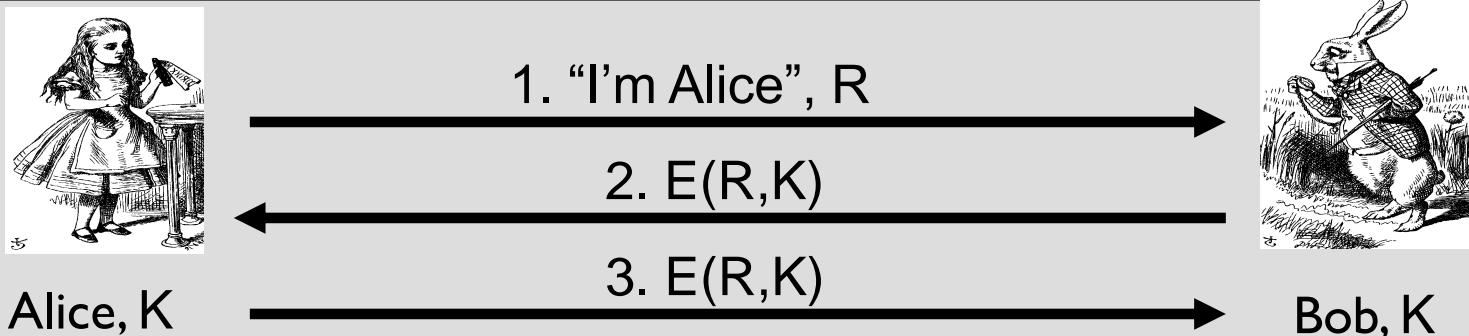
# AUTHENTICATION: SYMMETRIC KEY

- Alice and Bob **share** symmetric key **K**, which is **known only to them**
- Alice will **authenticate** herself to Bob by **proving** that **she knows the key**
- How to accomplish this?
  - Cannot **reveal key**, must **prevent replay (or other) attack**, must be **verifiable**, ...

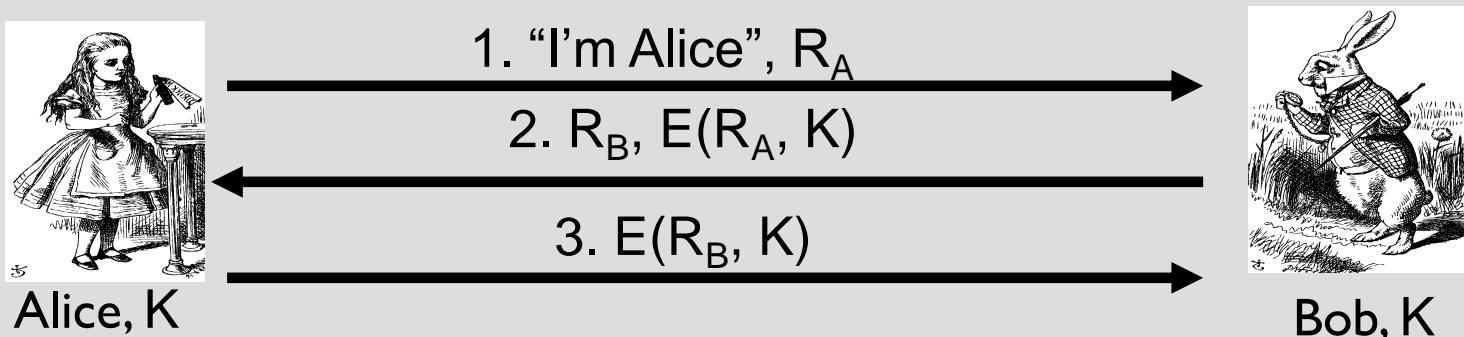


- Secure method for Bob to authenticate Alice (*prevents a replay attack*)
- But, Alice does not authenticate Bob (*lacks mutual authentication*)
- So, how can we achieve mutual authentication?

# MUTUAL AUTHENTICATION?

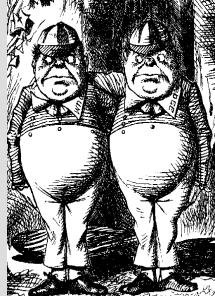


- **What's wrong** with this picture? “Alice” could be Trudy (or anybody else)! (2 vs. 3)
  - Since we have a secure one-way authentication protocol...
  - The obvious thing to do is to use the protocol twice
    - Once for Bob to authenticate Alice and Once for Alice to authenticate Bob



- This provides **mutual authentication**.....or does it?
  - But, subject to **reflection attack**, which is a method of attacking a challenge-response authentication system

# MUTUAL AUTHENTICATION ATTACK: REFLECTION ATTACK



Trudy

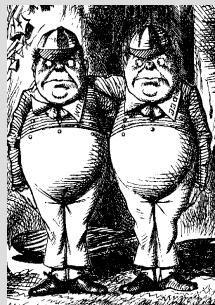
1. "I'm Alice",  $R_A$

2.  $R_B$ ,  $E(R_A, K)$

5.  $E(R_B, K)$



Bob, K



Trudy

3. "I'm Alice",  $R_B$

4.  $R_C$ ,  $E(R_B, K)$



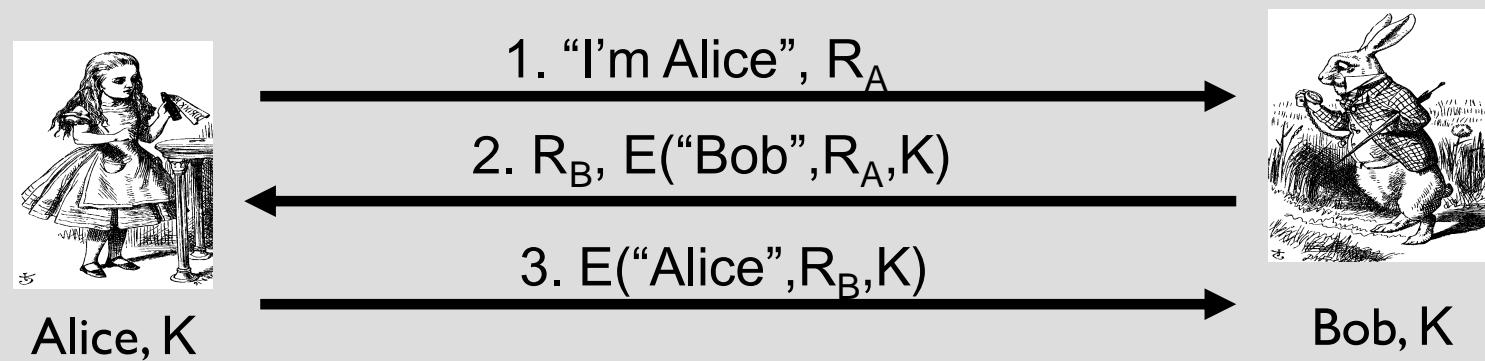
Bob, K

Note:

non-mutual authentication protocol may not be secure for mutual authentication.

# MUTUAL AUTHENTICATION

- Our one-way authentication protocol is **not secure** for **mutual authentication**
  - “obvious” any simple changes to protocols can cause unexpected security problems
  - Also, if assumptions or environment **change**, protocol may not be secure
  - This is a common source of security failure
- **Symmetric Key Mutual Authentication**



- Do these “insignificant” changes help?
- Yes!
  - Encrypting user's identity together with the nonce ( $R$ ) is sufficient to prevent the previous attack since Trudy cannot use a response from Bob for the third message

# **Digital Signature**

# COMPARISON

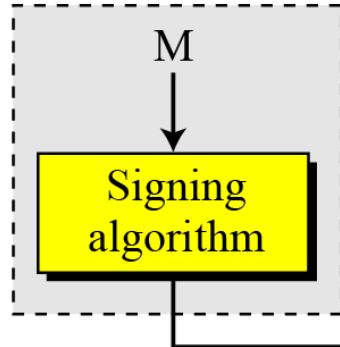
*Let us begin by looking at the differences between conventional signatures and digital signatures.*

- Inclusion
- Verification Method
- Relationship
- Duplicity

# PROCESS

## *Digital signature process*

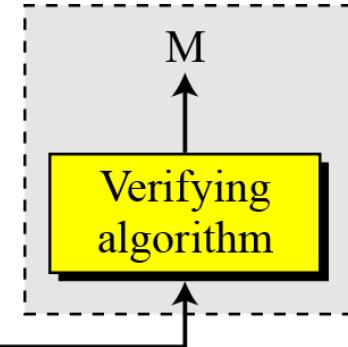
Alice



M: Message  
S: Signature

(M, S)

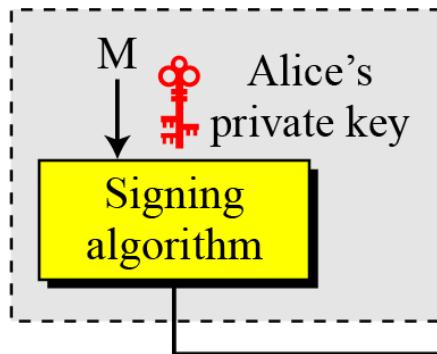
Bob



# Need for Keys

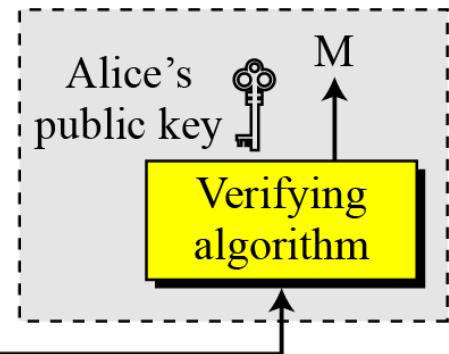
*Adding key to the digital signature process*

Alice



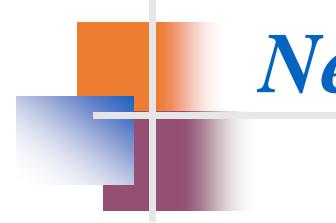
M: Message  
S: Signature

Bob



**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.**



# *Need for Keys*

**Note**

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

# Comparison

## Public key Cryptography

- Public and private keys of the receiver are used
- Sender uses public key of the receiver to encrypt the message
- Receiver uses his own private key to decrypt

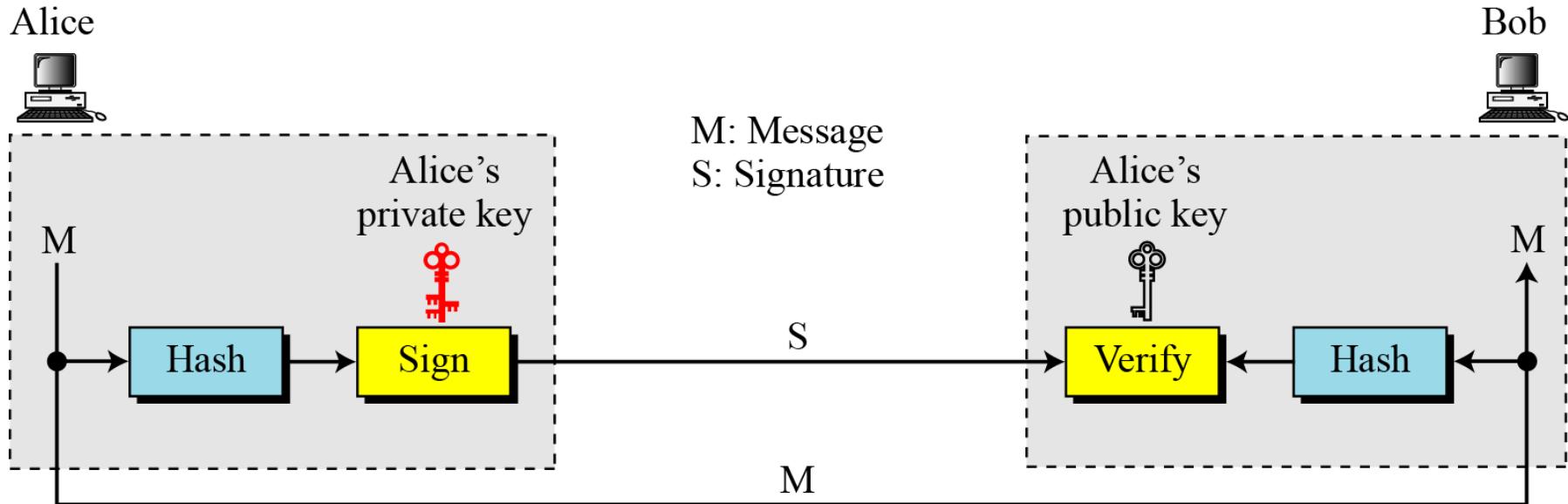
## Digital Signature

- Public and private keys of the sender are used
- Sender uses his own private key of the to sign the message
- Receiver uses public key of the sender to verify

# *Signing the Digest*

*A message digest is a fixed size numeric representation of the contents of a message, computed by a hash function.*

## *Signing the digest*



# SERVICES

- *We discussed several security services including*
  - *message confidentiality,*
  - *message authentication,*
  - *message integrity, and*
  - *nonrepudiation.*
- *A digital signature can directly provide the last three; for message confidentiality we still need encryption/decryption.*

**Message Authentication**

**Message Integrity**

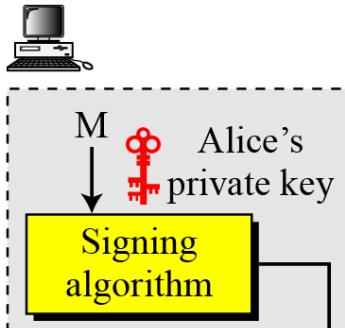
**Nonrepudiation**

**Confidentiality**

# Nonrepudiation

## Using a trusted center for nonrepudiation

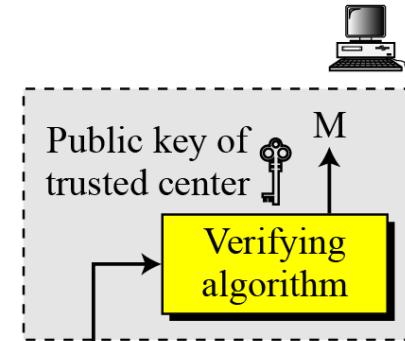
Alice



**Note**

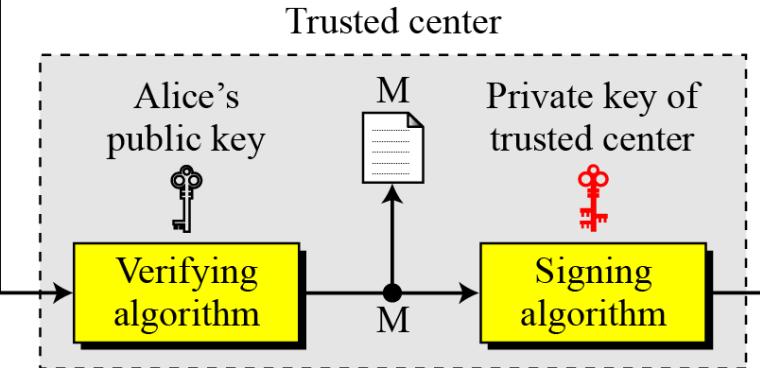
$M$ : Message  
 $S_A$ : Alice's signature  
 $S_T$ : Signature of trusted center

Bob



$(M, S_A)$

Trusted center

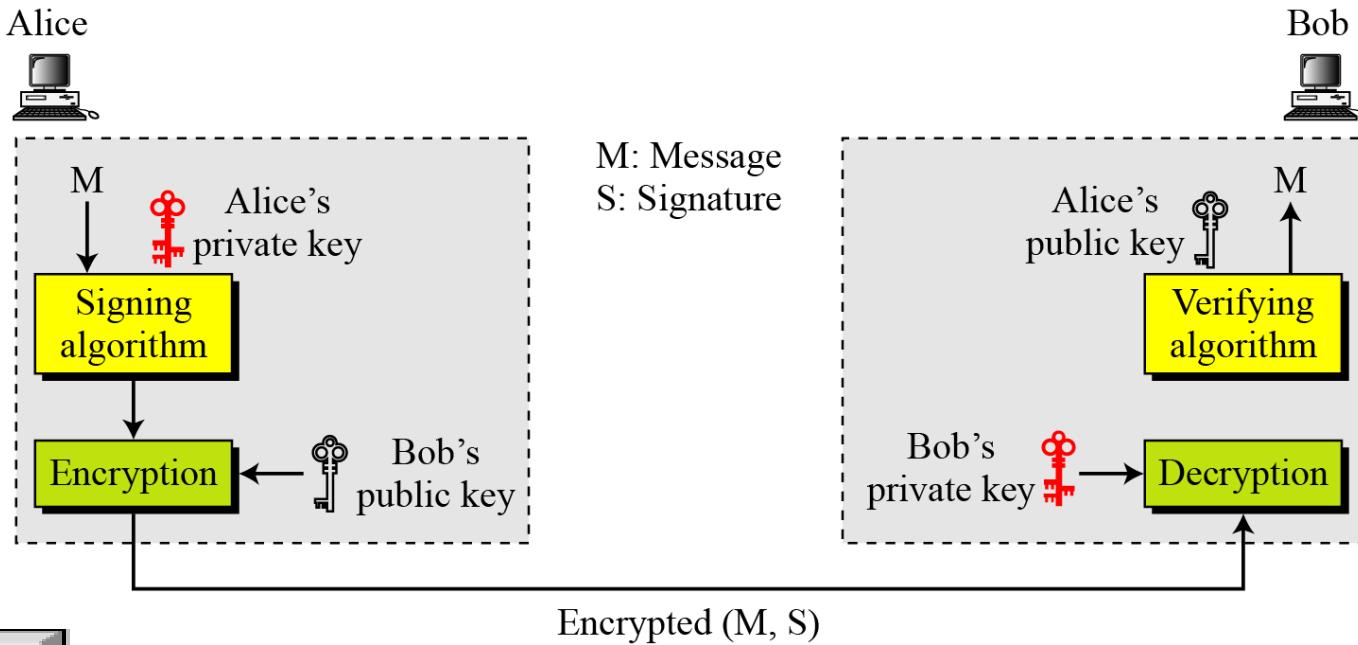


$(M, S_T)$

**Nonrepudiation can be provided using a trusted party.**

# Confidentiality

*Adding confidentiality to a digital signature scheme*



**Note**

**A digital signature does not provide privacy.  
If there is a need for privacy, another layer of  
encryption/decryption must be applied.**

# ATTACKS ON DIGITAL SIGNATURE

- **Attack Types**
  - *Key-Only Attack*
  - *Known-Message Attack*
  - *Chosen-Message Attack*
- **Forgery Types**
  - *Existential Forgery*
  - *Selective Forgery*

# ATTACKS ON DIGITAL SIGNATURE

**(Alice is the signer, Oscar the attacker)**

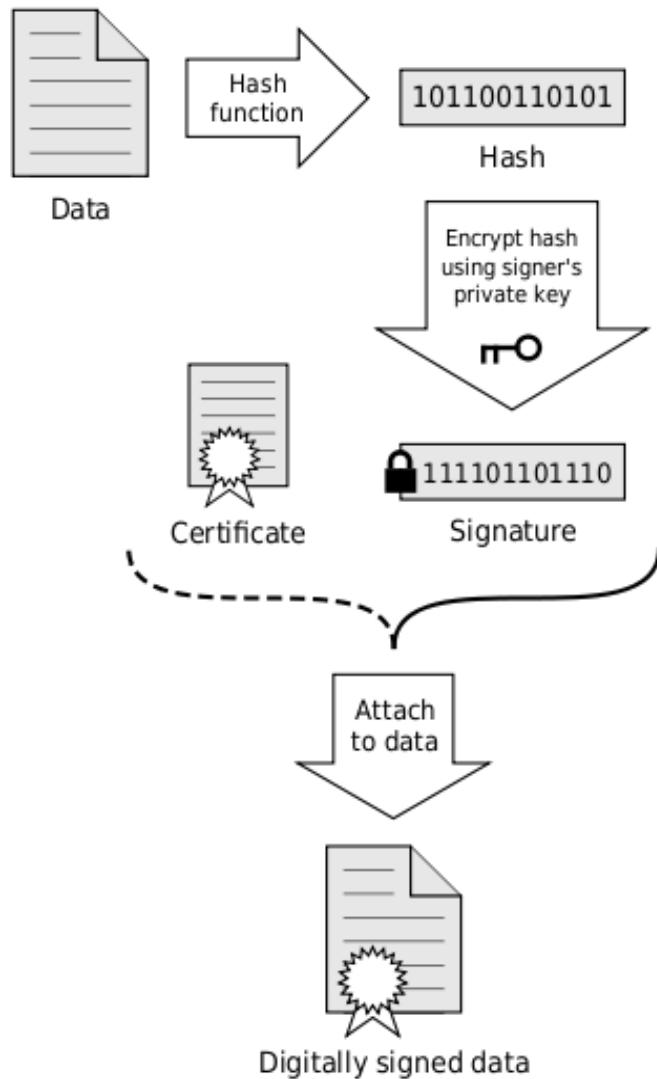
- **Key-only attack** : Oscar possesses Alice's public verification algorithm
- **Known message attack**: Oscar possesses a list of signed messages  $(x_i, y_i)$
- **Chosen message attack**: Oscar queries Alice for the signatures of a list of messages  $x_i$
- **Selective forgery**: C forges a signature for a particular message chosen by C.
- **Existential forgery**: C forges a signature for at least one message. C has no control over the message.

# DIGITAL SIGNATURE SCHEMES

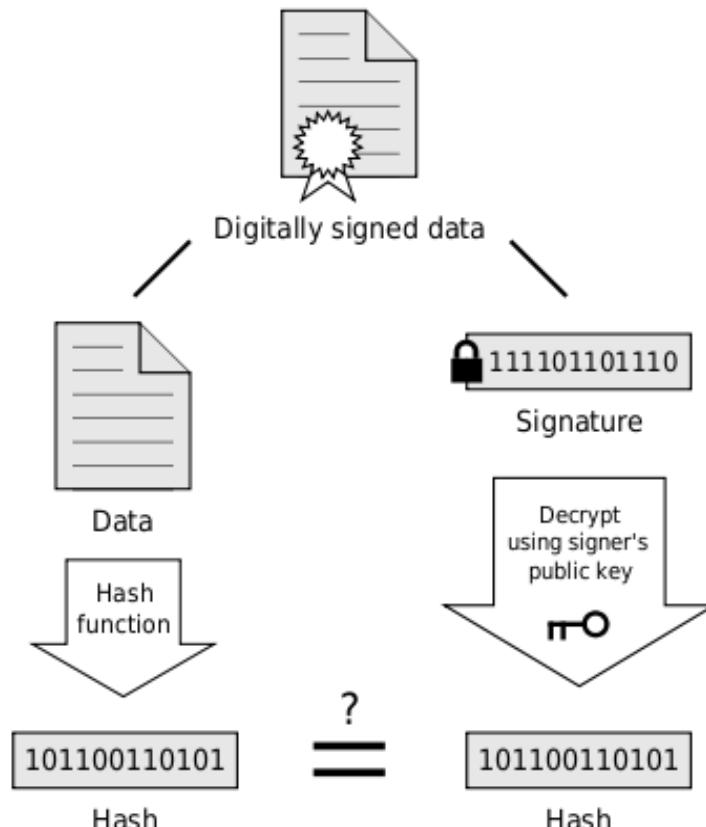
- A digital signature scheme is a mathematical scheme for demonstrating the authenticity of a digital message or document.
- A digital signature scheme typically consists of three algorithms:
  - A key generation algorithm that selects a private key uniformly at random from a set of possible private keys. The algorithm outputs the private key and a corresponding public key.
  - A signing algorithm that, given a message and a private key, produces a signature.
  - A signature verifying algorithm that, given a message, public key and a signature, either accepts or rejects the message's claim to authenticity.

# *General digital signature scheme*

## Signing



## Verification



If the hashes are equal, the signature is valid.

# **DIGITAL SIGNATURE SCHEMES**

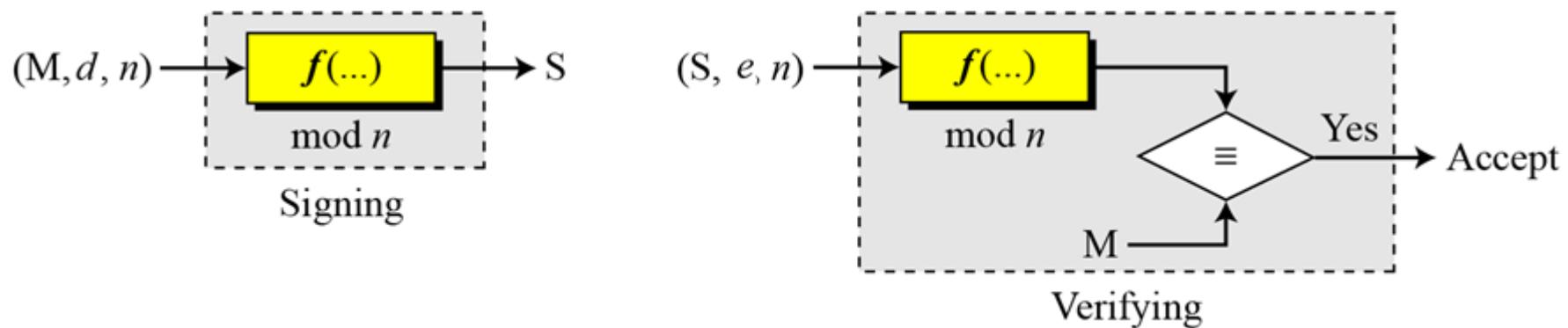
**RSA Digital Signature Scheme  
Digital Signature Standard (DSS)**

# RSA Digital Signature Scheme

*General idea behind the RSA digital signature scheme*

M: Message  
S: Signature

$(e, n)$ : Alice's public key  
 $d$ : Alice's private key



# RSA Digital Signature Scheme

## Key Generation

*Key generation in the RSA digital signature scheme is exactly the same as key generation in the RSA*

- *Alice chooses 2 prime numbers  $p$  and  $q$  and calculates  $n = p * q$*
- *Alice calculates  $\phi(n) = (p-1)(q-1)$*
- *She then chooses  $e$ , the public exponent, and calculates  $d$ , the private exponent such that  $e * d = 1 \text{ mod } \phi(n)$*
- *Alice keeps  $d$  and publicly announces  $e$  and  $n$*

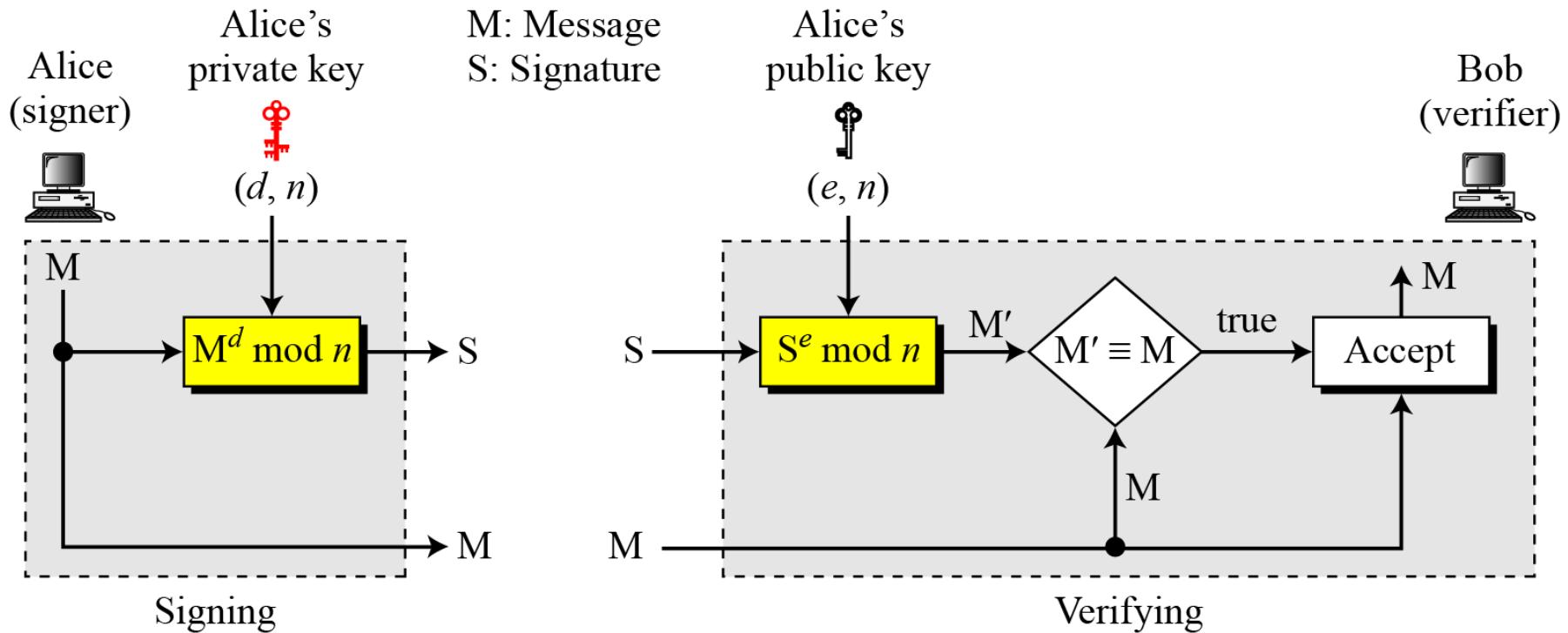
### Note

In the RSA digital signature scheme,  $d$  is private;  
 $e$  and  $n$  are public.

# RSA Digital Signature Scheme

## Signing and Verifying

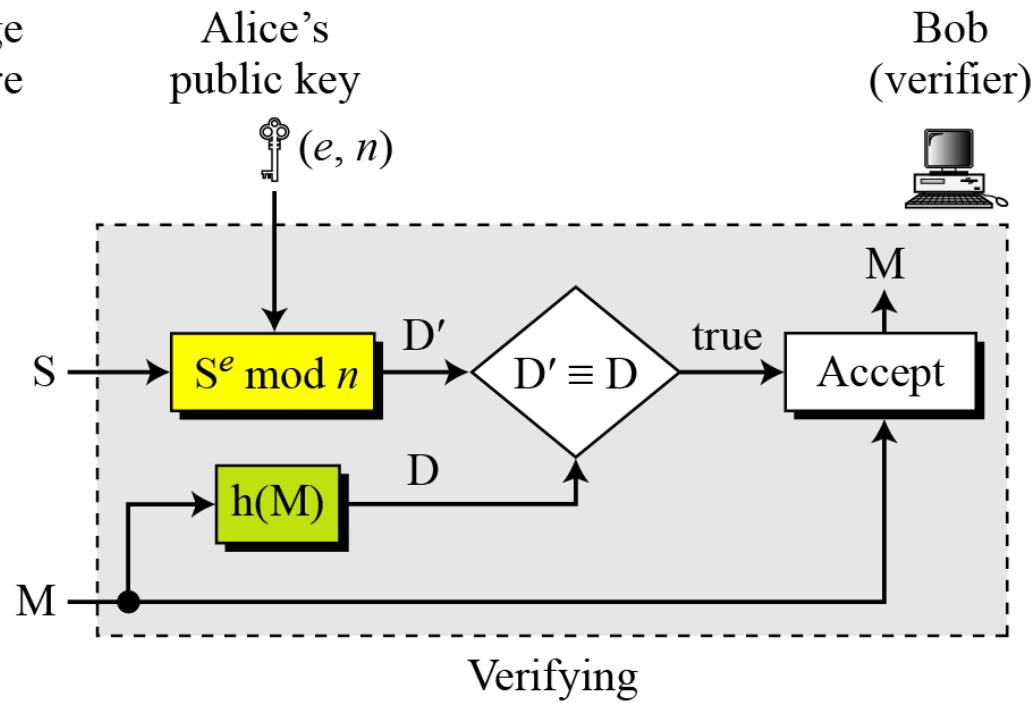
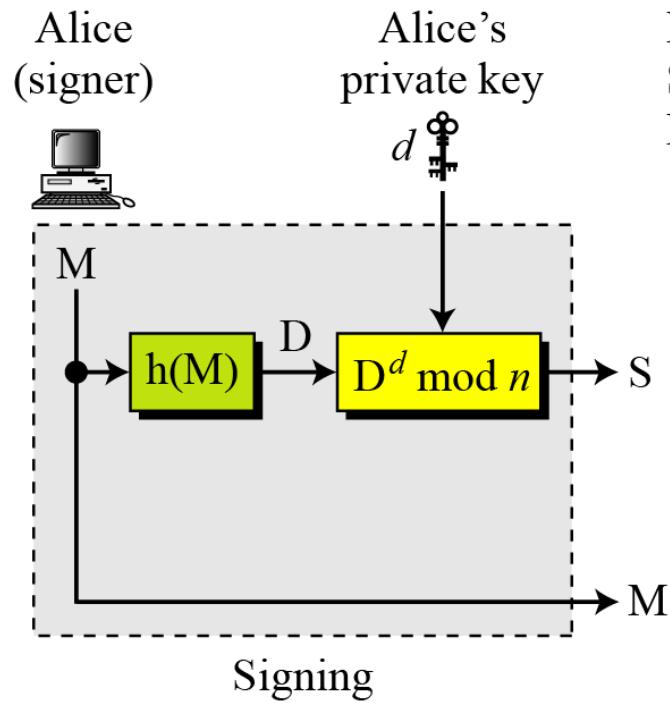
### RSA digital signature scheme



# RSA Digital Signature Scheme

## RSA Signature on the Message Digest

**The RSA signature on the message digest**



# Digital Signature Standard (DSS)

## General idea behind DSS scheme

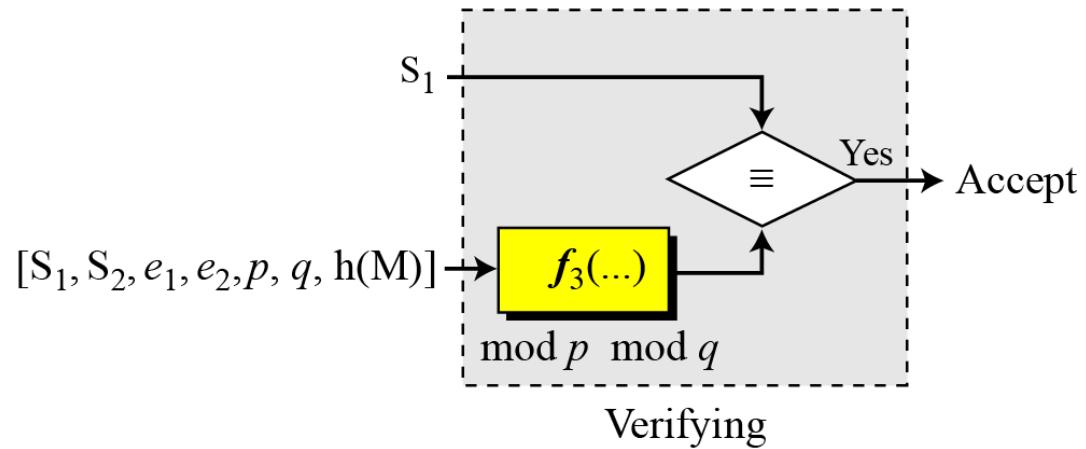
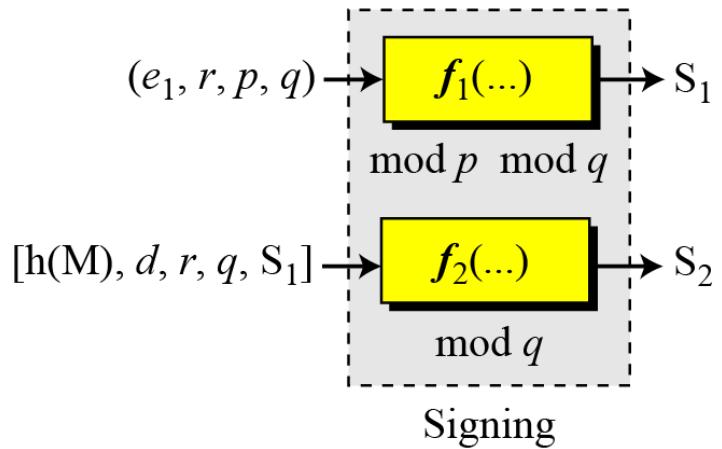
$S_1, S_2$ : Signatures

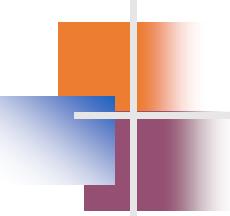
$d$ : Alice's private key

$M$ : Message

$r$ : Random secret

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





# *Digital Signature Standard (DSS)*

## *Key Generation.*

- 1) *Alice chooses primes  $p$ , between 512 and 1024 bits in length. The number of bits in  $p$  must be a multiple of 64.*
- 2) *Alice chooses a 160-bit prime  $q$  in such a way that  $q$  divides  $(p-1)$ .*
- 3) *Alice uses  $\langle \mathbb{Z}_p^*, \times \rangle$  and  $\langle \mathbb{Z}_q^*, \times \rangle$ , second is subgroup of the first.*
- 4) *Alice creates  $e_1$  to be the  $q$ th root of 1 modulo  $p$ . i.e.  
 $e_1^p = 1 \text{ mod } p$*
- 5) *Alice chooses  $d$  and calculates  $e_2 = e_1^d$ .*
- 6) *Alice's public key is  $(e_1, e_2, p, q)$ ; her private key is  $(d)$ .*

# Digital Signature Standard (DSS)

## Verifying and Signing

### DSS scheme

M: Message

$S_1, S_2$ : Signatures

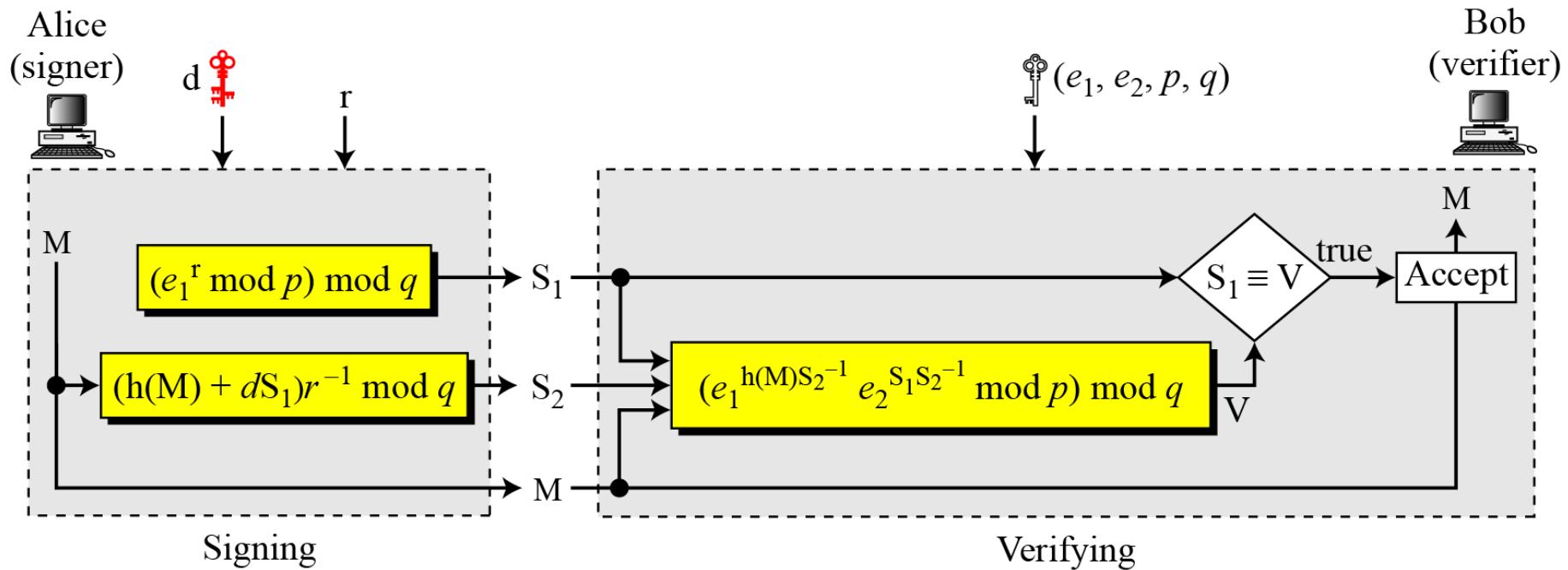
V: Verification

r: Random secret

d: Alice's private key

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

$h(M)$ : Message digest



# VARIATIONS

## *Time Stamped Signatures*

*Sometimes a signed document needs to be time stamped to prevent it from being replayed by an adversary. This is called time-stamped digital signature scheme.*

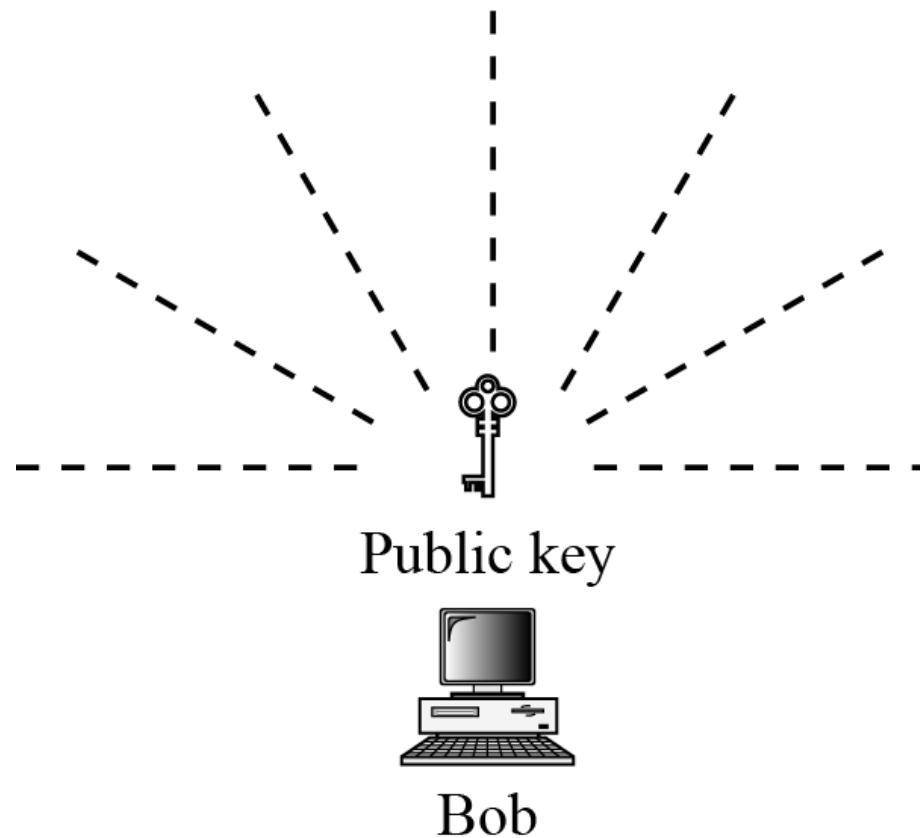
## *Blind Signatures*

*Sometimes we have a document that we want to get signed without revealing the contents of the document to the signer.*

# Public Key Distribution

- I. public announcement
- II. publicly available directory
- III. public-key authority
- IV. public-key certificates

*Announcing a public key*



# Public Announcement

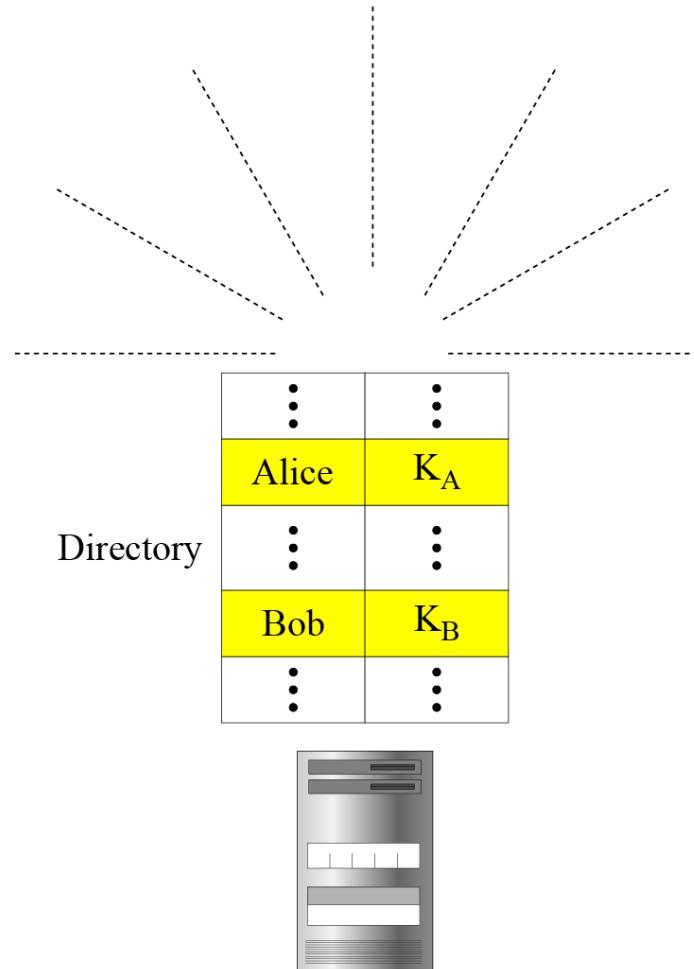
- users distribute public keys to recipients or broadcast to community at large
  - eg. append PGP keys to email messages or post to news groups or email list
- major weakness is forgery
  - anyone can create a key claiming to be someone else and broadcast it
  - until forgery is discovered can masquerade as claimed user

# Publicly Available Directory

- can obtain greater security by registering keys with a public directory
- directory must be trusted with properties:
  - contains {name,public-key} entries
  - participants register securely with directory
  - participants can replace key at any time
  - directory is periodically published
  - directory can be accessed electronically
- still vulnerable to tampering or forgery

# *Trusted Center Publicly available directory*

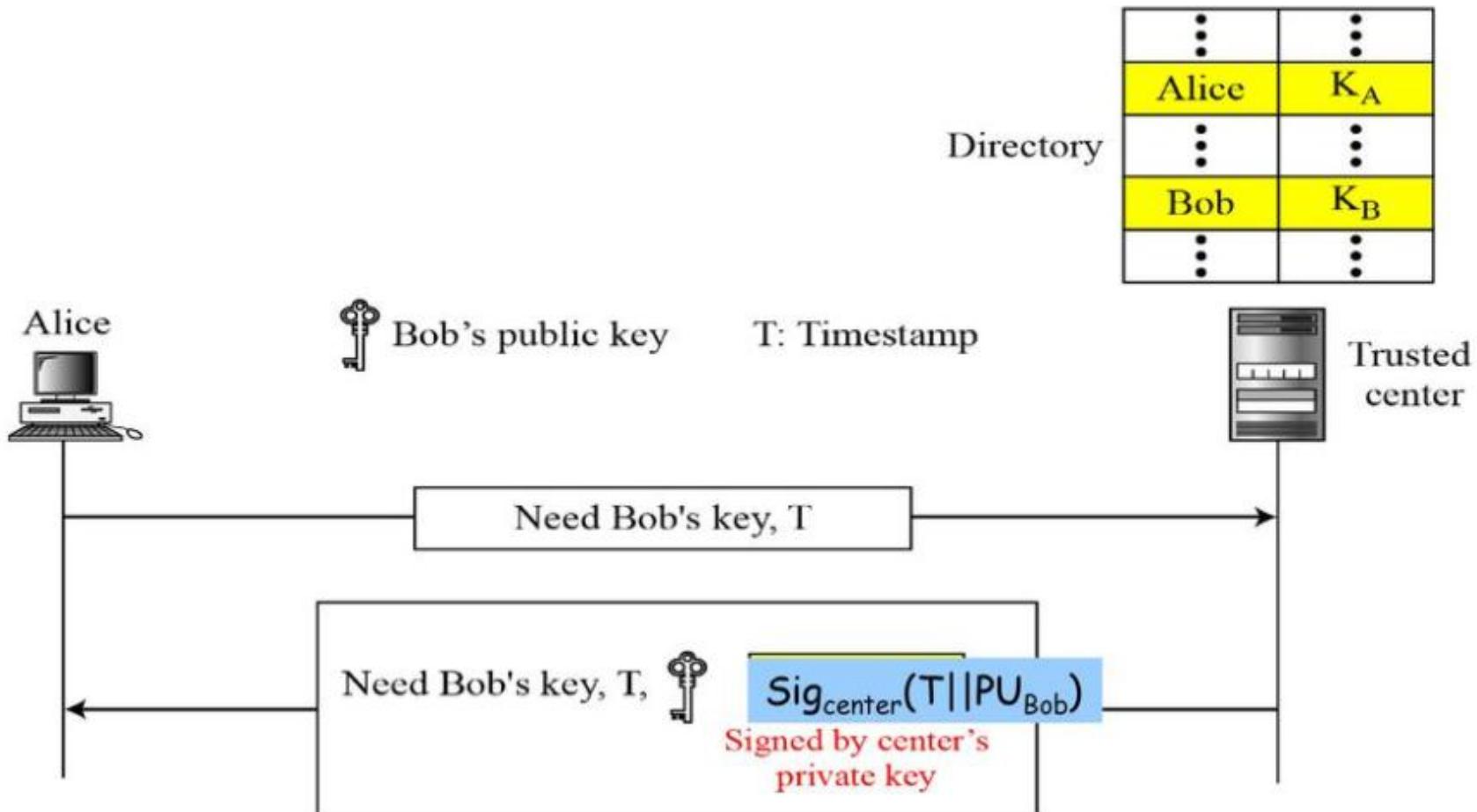
## *Trusted center*



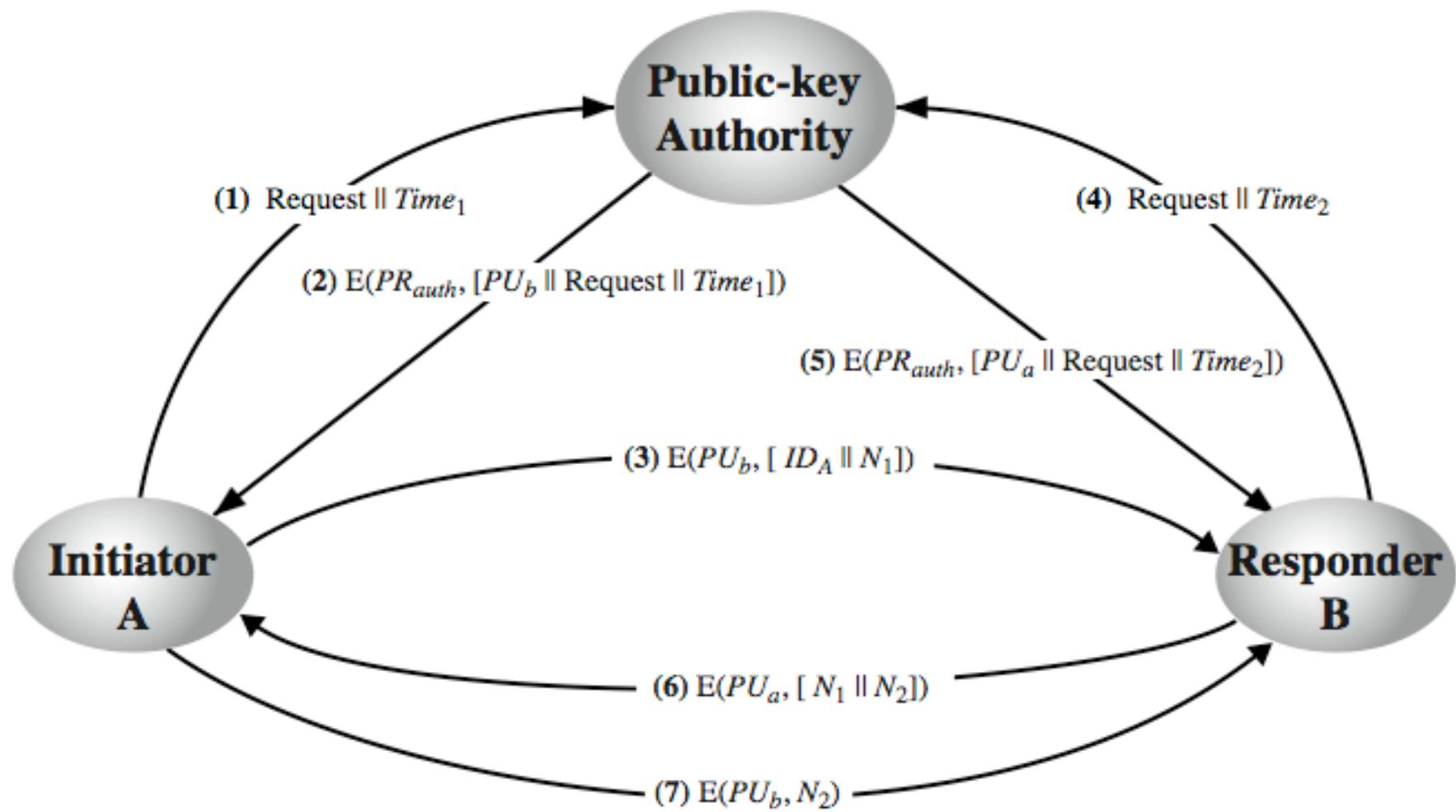
# Public-Key Authority

- improve security by tightening control over distribution of keys from directory
- has properties of directory
- and requires users to know public key for the directory
- then users interact with directory to obtain any desired public key securely
  - does require real-time access to directory when keys are needed
  - may be vulnerable to tampering

## Controlled trusted center



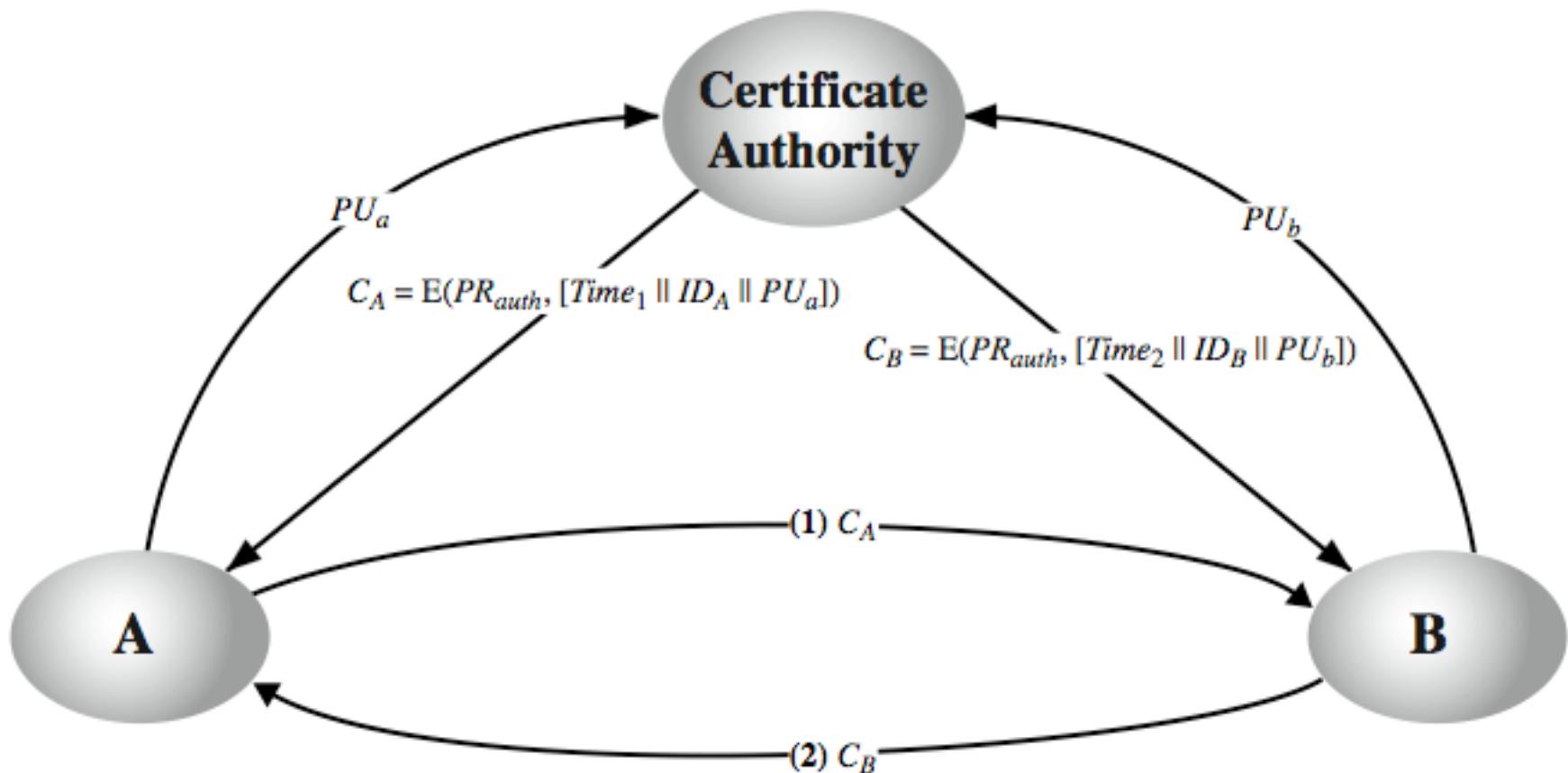
# Public-Key Authority



# Public-Key Certificates

- certificates allow key exchange without real-time access to public-key authority
- a certificate binds **identity** to **public key**
  - usually with other info such as period of validity, rights of use etc
- with all contents **signed** by a trusted Public-Key or Certificate Authority (CA)
- can be verified by anyone who knows the public-key authorities public-key

# Public-Key Certificates



# Certificate Creation Steps

- Parties Involved :
  - ✓ CA
  - ✓ RA: CA is overloaded with a variety of task such as issuing new certificate, revoking, maintaining etc CA delegates some of its task to his third party called **Registration Authority(RA)**
  - ✓ RA is an intermediate entity between the end user and the CA
  - ✓ RA Commonly provides
    - Accepting and verifying registration
    - Generating keys on behalf of user
    - Accepting and authorizing the request for certificate revocation
    - RA cannot issue digital certificate , its CA's responsibility

# Certificate Creation Steps

- Step 1 : Key Generation
- Step 2 : Registration: User sends the public key and the associated registration information all evidences to the RA.
  - ✓ The format for Certificate request is standardized called Certificate Signing Request(CSR).
  - ✓ After this the user usually gets a request identifier for tracking the progress of the certificate request
- Step 3:Verification:
  - ✓ RA need to verify the user's credentials and ensure that they are acceptable
  - ✓ The second Check is to ensure that the user who is requesting the certificate indeed possess the private key corresponding to the public key .Proof of Possession(POP).
  - ✓ RA can demand the user must digitally sign her CSR using Private key and RA verify with Public key.

# Certificate Creation Steps

- Certificate Creation: Assuming that all the steps have been successful. RA Passes all the information to the CA. The CA Creates a digital certificate send it to the user and also retain a copy of certificate in a certificate directory.
- The certificate is signed with the private key of CA, to get the certificate we need the public key of CA .

# X.509 certificate

Although the use of a CA solves the problem of public-key fraud, it has created a side-effect

Each certificate may have a different format

X.509 is used to remove the side-effect

X.509 is a way to describe the certificate in a structured way

# X.509 certificate scheme

- X.509 scheme for generation of a public-key certificate.
- The certificate for Bob's public key includes unique identifying information for Bob, Bob's public key, and identifying information about the CA, plus other information

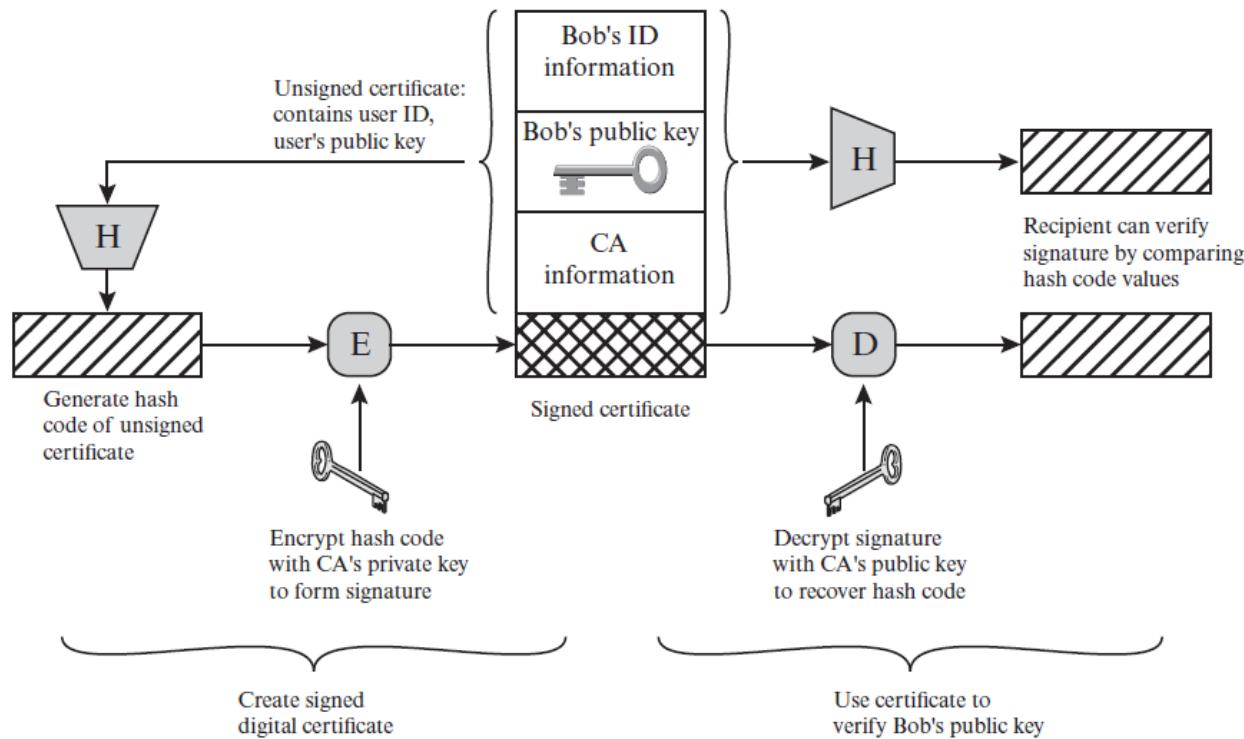
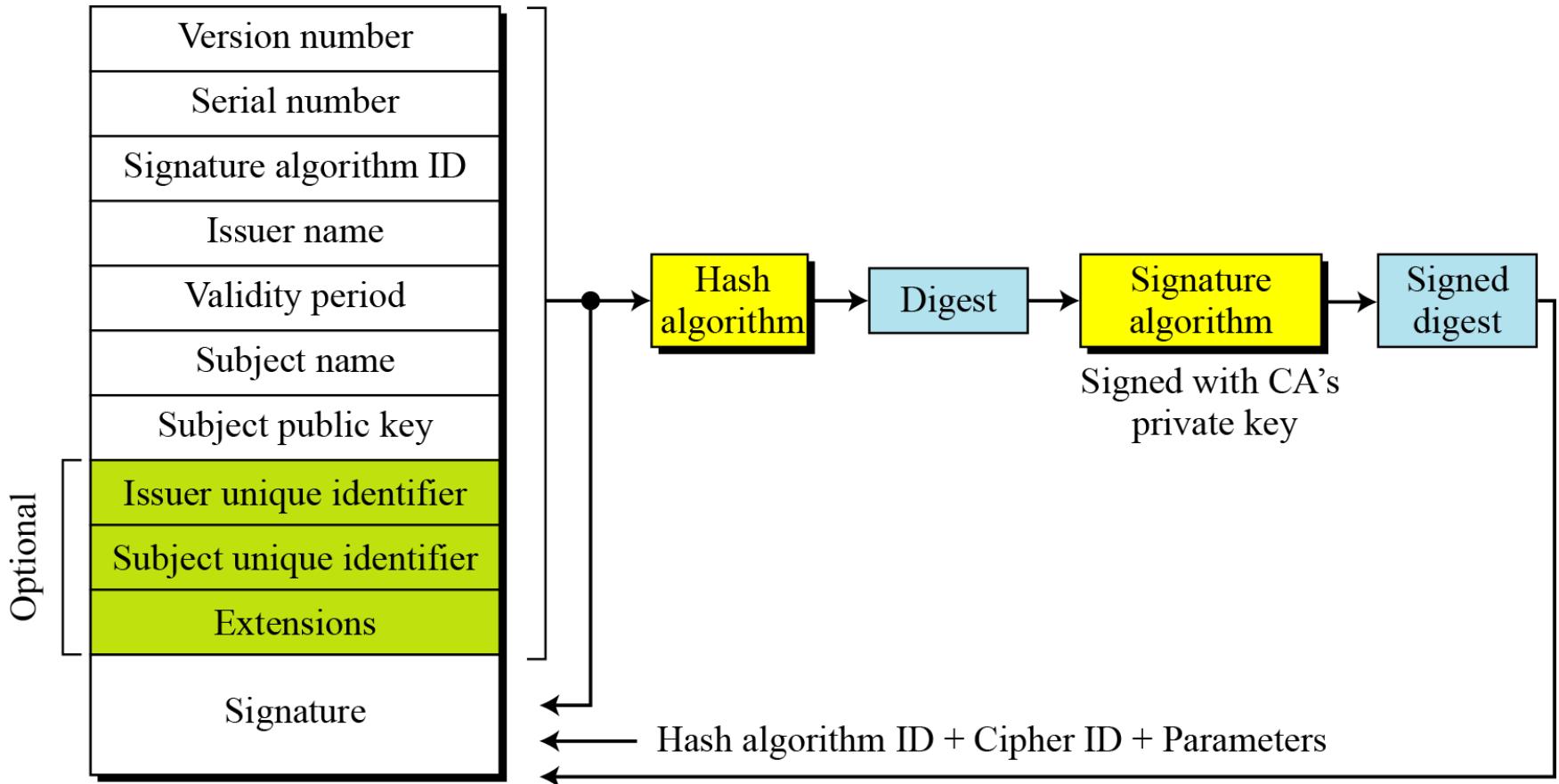


Figure 14.14 X.509 Public-Key Certificate Use

# X.509 Certificates



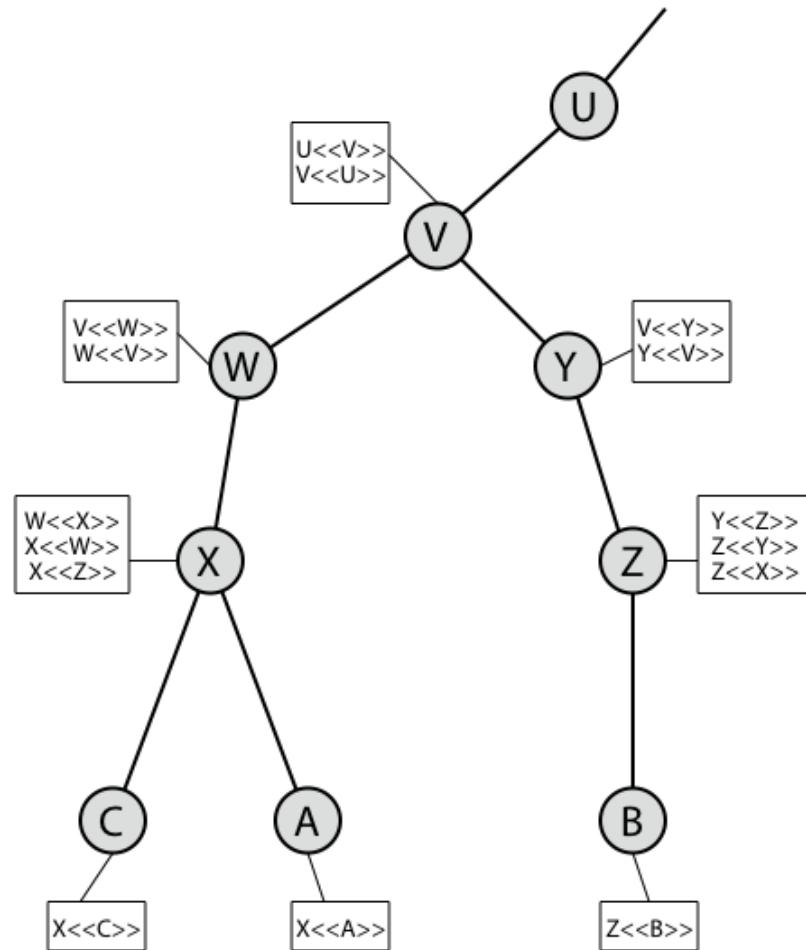
# Certificate

- Version number:The field defines the version of X.509 of the certificate.
- Serial number:The field define a number assigned to each certificate
- Signature algorithm ID:algorithm used to sign the certificate
- Issuer name:identified certification authority that issued the certificate
- Validity Period:Time
- Subject name:entity to which public key belongs
- Subject Public key: heart of certificate
- Signature: This field is made of 3 section. First Section all other fields, second digest of first section encrypted with CA's Private key . The third is algorithm identifier used to create the second section.

# CA Hierarchy

- if both users share a common CA then they are assumed to know its public key
- otherwise CA's must form a hierarchy(chain of trust)
- use certificates linking members of hierarchy to validate other CA's
  - each CA has certificates for clients (forward) and parent (backward)
- each client trusts parents certificates
- enable verification of any certificate from one CA by users of all other CAs in hierarchy
- .The directory entry for each CA includes two types of certificates:
  - Forward certificates: Certificates of X generated by other CAs
  - Reverse certificates: Certificates generated by X that are the certificates of other CAs.

# CA Hierarchy Use

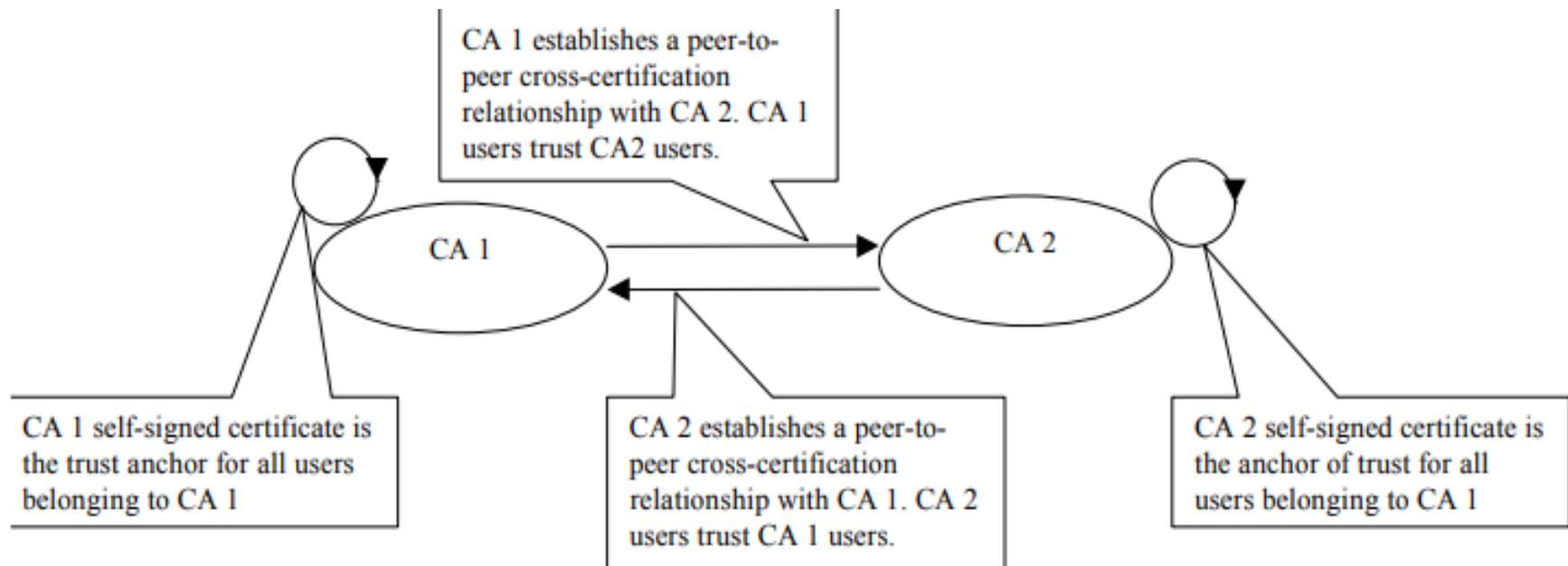


A acquires B certificate using chain: X<<W>>W<<V>>V<<Y>>Y<<Z>>Z<<B>>  
B acquires A certificate using chain: Z<<Y>>Y<<V>>V<<W>>W<<X>>X<<A>>

# Cross Certificate

- Root CA's could be different
- No single root CA
- Cross Certification:Single monolithic CA Certifying every possible user in the world is quite unlikely. Instead , the concept of decentralized CA's for different Countries, business etc
- It allows CA and end user from different domain to interact
- The root CA's cross certify each other. Technically Alice's root CA has obtained a certificate for itself from Bob's root CA.Similarly bob's CA has obtained a certificate from Alice's root CA.

# Cross Certification



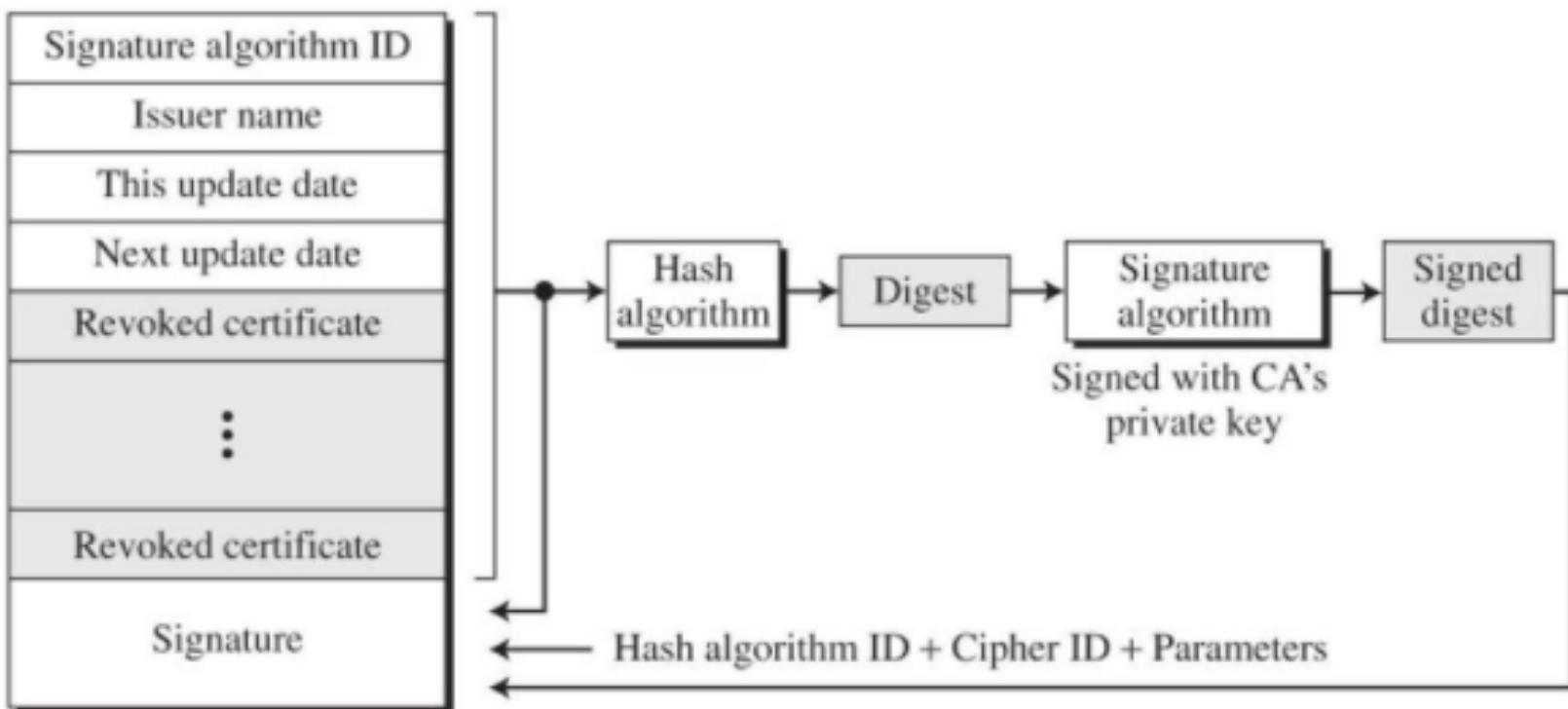
## Certificate Renewal

- Each certificate has a period of validity
- If there is no problem with the certificate, the CA issues a new certificate before the old one expires.
- In some cases a certificate must be revoked before its expiration

# Certificate Revocation

- certificates have a period of validity
- may need to revoke before expiry, eg:
  1. user's private key is compromised
  2. user is no longer certified by this CA
  3. CA's certificate is compromised
- CA's maintain list of revoked certificates
  - the Certificate Revocation List (CRL)
- users should check certificates with CA's CRL

# Certificate Revocation List Format



# Certificate Revocation List

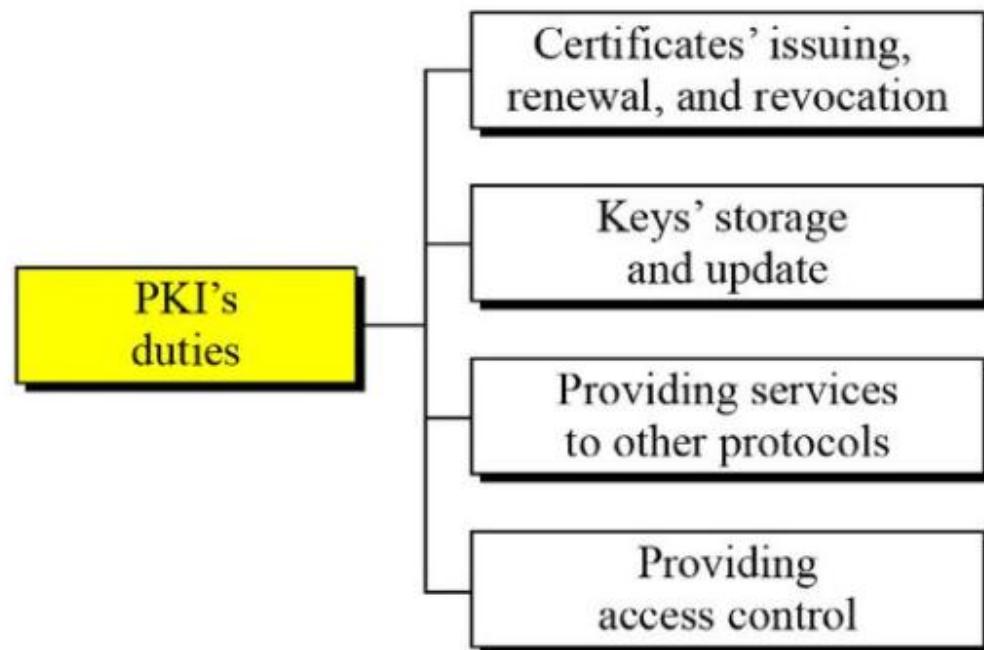
- CRL is a list of certificates published regularly by each CA, identifying all the certificates that have been revoked.
- It does not include certificates whose validity period is over.
- It's a sequential file that grows over time, it lists the serial number, the date and time on which the certificate was revoked.
- The CRL can become really quite big over time.
- The bottleneck is solved using Delta CRL (change since last update).
- This mechanism makes the CRL file size small and therefore its transmission is easier.
- The changes to base CRL is called Delta CRL.

# PKI

- **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.
- Its principal is to enable secure, convenient, and efficient acquisition of public keys.
- The IETF Public Key Infrastructure X.509 (PKIX) working group has setup a formal (and generic) model based on X.509 that is suitable for deploying a certificate-based architecture on the Internet.

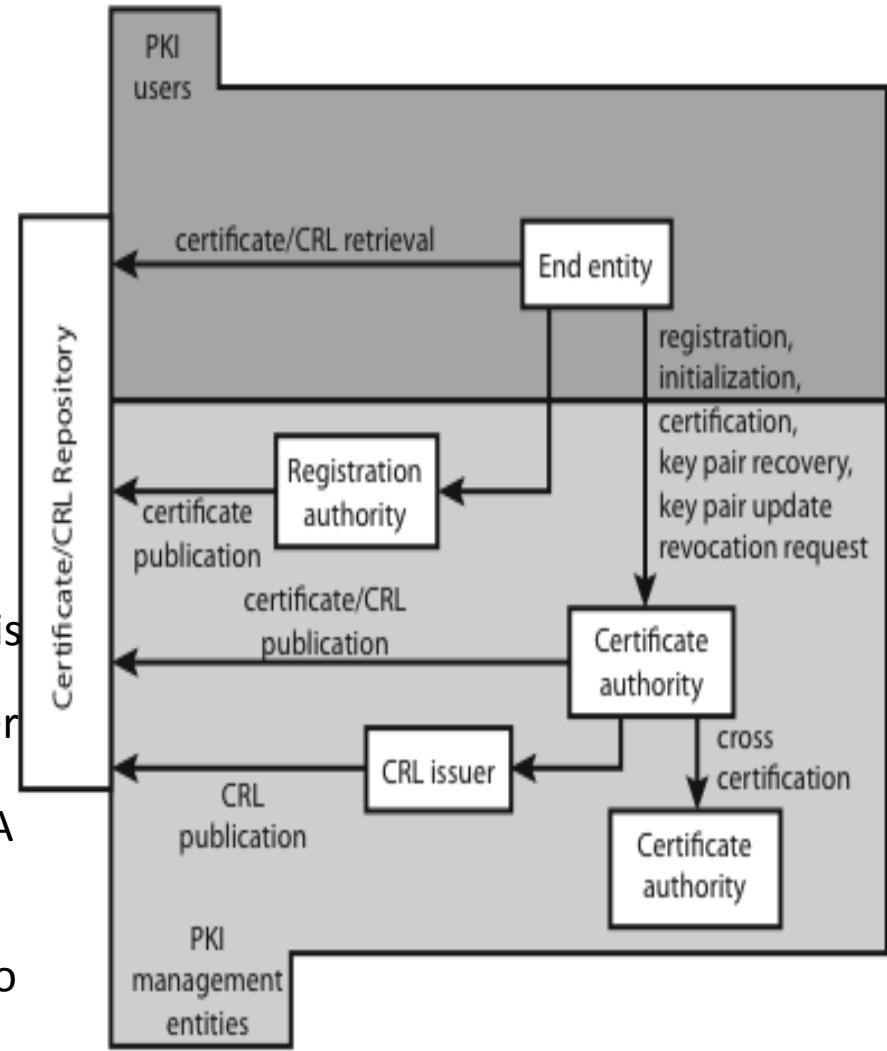
# Public Key Infrastructure

- ❑ PKI is a model for creating, distributing and revoking certificates based on X.509



# Public Key Infrastructure

- **End entity:** A generic term used to denote end users, devices (e.g., servers, routers), or any other entity that can be identified in the subject field of a public key certificate. End entities can consume and/or support PKI-related services.
- **Certification authority (CA):** The issuer of certificates and (usually) certificate revocation lists (CRLs). It may also support a variety of administrative functions, although these are often delegated to Registration Authorities.
- **Registration authority (RA):** An optional component that can assume a number of administrative functions from the CA. The RA is often associated with the End Entity registration process, but can assist in a number of other areas as well.
- **CRL issuer:** An optional component that a CA can delegate to publish CRLs.
- **Repository:** A generic term used to denote any method for storing certificates and CRLs so that they can be retrieved by End Entities.



# PKIX Management

➤ functions:

- registration
- initialization
- certification
- key pair recovery
- key pair update
- revocation request
- cross certification

➤ protocols: CMP, CMC

# **Digital Signature**

# COMPARISON

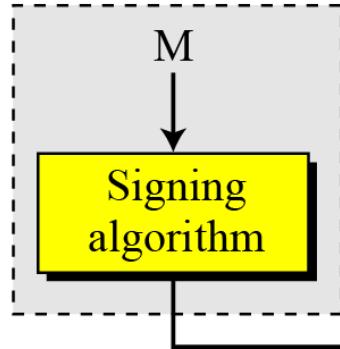
*Let us begin by looking at the differences between conventional signatures and digital signatures.*

- Inclusion
- Verification Method
- Relationship
- Duplicity

# PROCESS

## *Digital signature process*

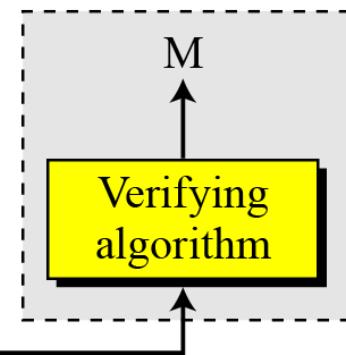
Alice



M: Message  
S: Signature

(M, S)

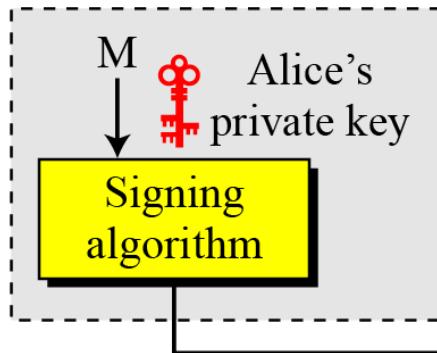
Bob



# Need for Keys

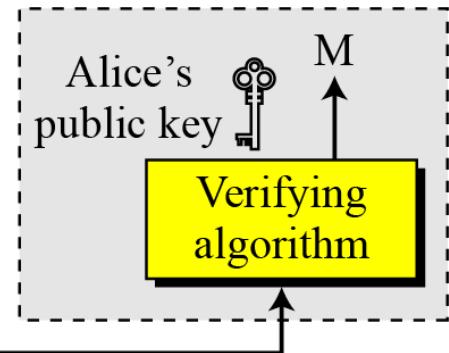
*Adding key to the digital signature process*

Alice



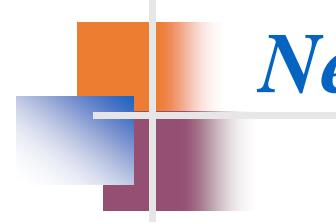
M: Message  
S: Signature

Bob



**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.**



# *Need for Keys*

**Note**

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

# Comparison

## Public key Cryptography

- Public and private keys of the receiver are used
- Sender uses public key of the receiver to encrypt the message
- Receiver uses his own private key to decrypt

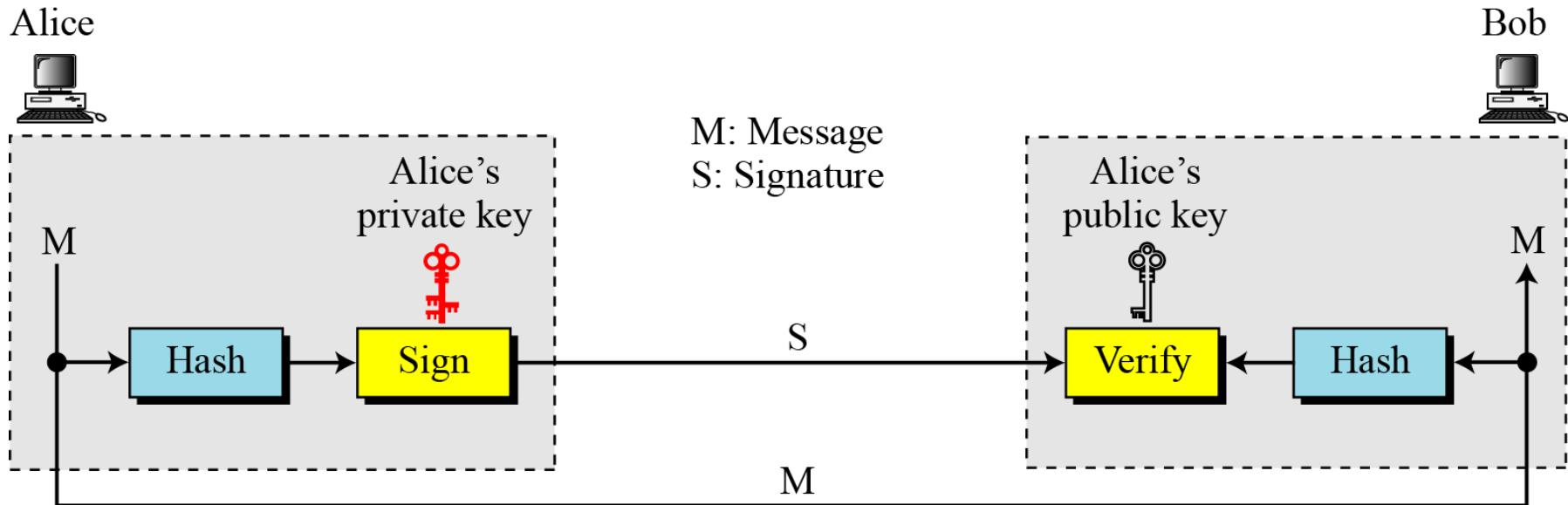
## Digital Signature

- Public and private keys of the sender are used
- Sender uses his own private key of the to sign the message
- Receiver uses public key of the sender to verify

# *Signing the Digest*

*A message digest is a fixed size numeric representation of the contents of a message, computed by a hash function.*

## *Signing the digest*



# SERVICES

- *We discussed several security services including*
  - *message confidentiality,*
  - *message authentication,*
  - *message integrity, and*
  - *nonrepudiation.*
- *A digital signature can directly provide the last three; for message confidentiality we still need encryption/decryption.*

**Message Authentication**

**Message Integrity**

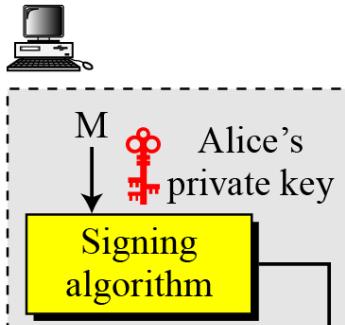
**Nonrepudiation**

**Confidentiality**

# Nonrepudiation

## Using a trusted center for nonrepudiation

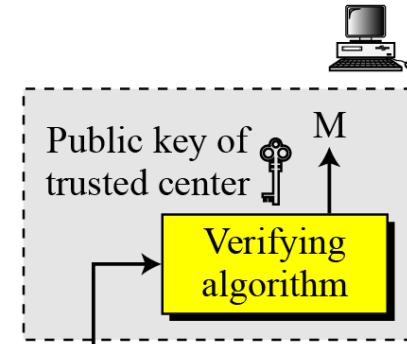
Alice



**Note**

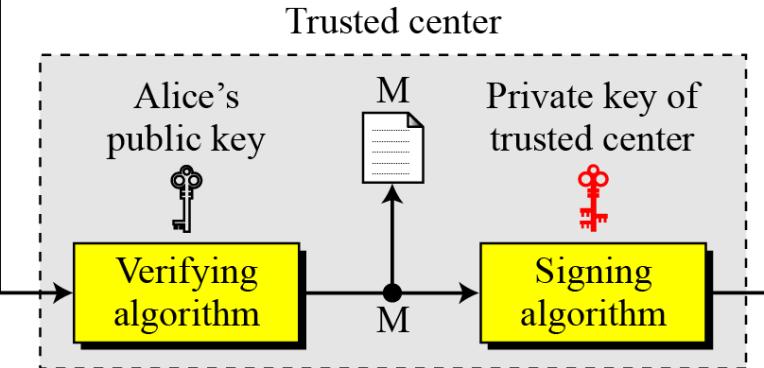
$M$ : Message  
 $S_A$ : Alice's signature  
 $S_T$ : Signature of trusted center

Bob



$(M, S_A)$

Trusted center

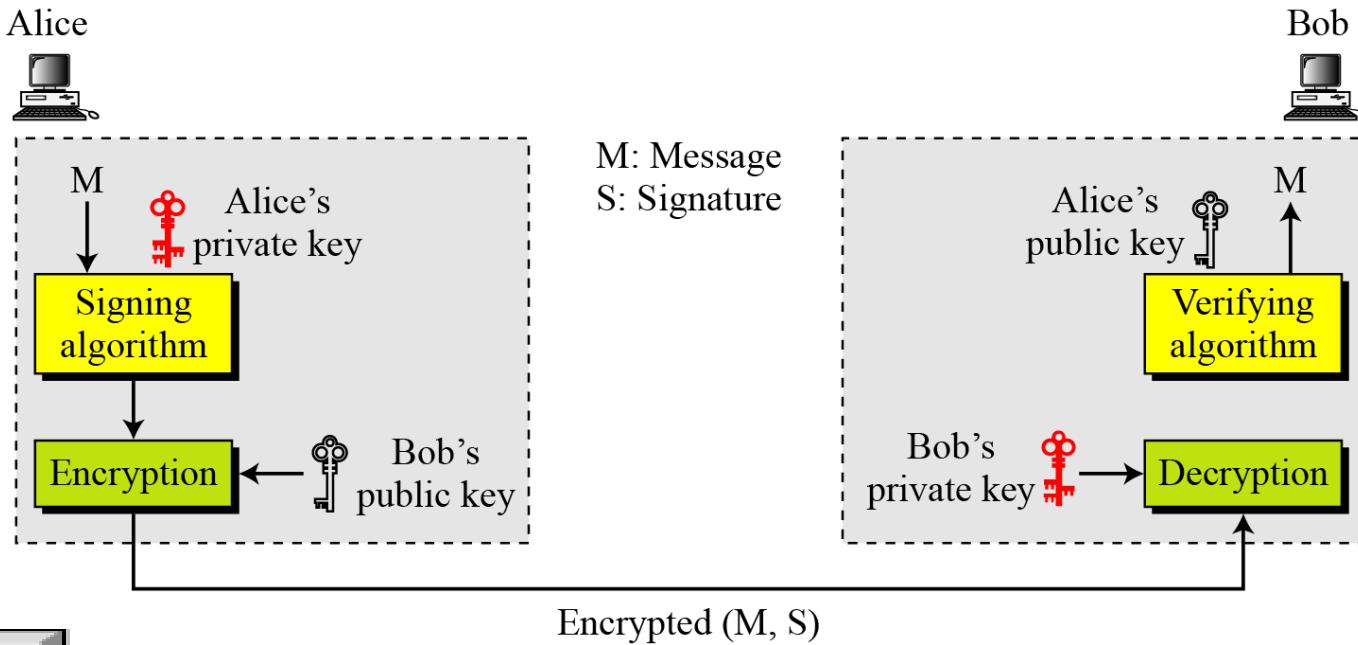


$(M, S_T)$

**Nonrepudiation can be provided using a trusted party.**

# Confidentiality

*Adding confidentiality to a digital signature scheme*



**Note**

**A digital signature does not provide privacy.  
If there is a need for privacy, another layer of  
encryption/decryption must be applied.**

# ATTACKS ON DIGITAL SIGNATURE

- **Attack Types**
  - *Key-Only Attack*
  - *Known-Message Attack*
  - *Chosen-Message Attack*
- **Forgery Types**
  - *Existential Forgery*
  - *Selective Forgery*

# ATTACKS ON DIGITAL SIGNATURE

**(Alice is the signer, Oscar the attacker)**

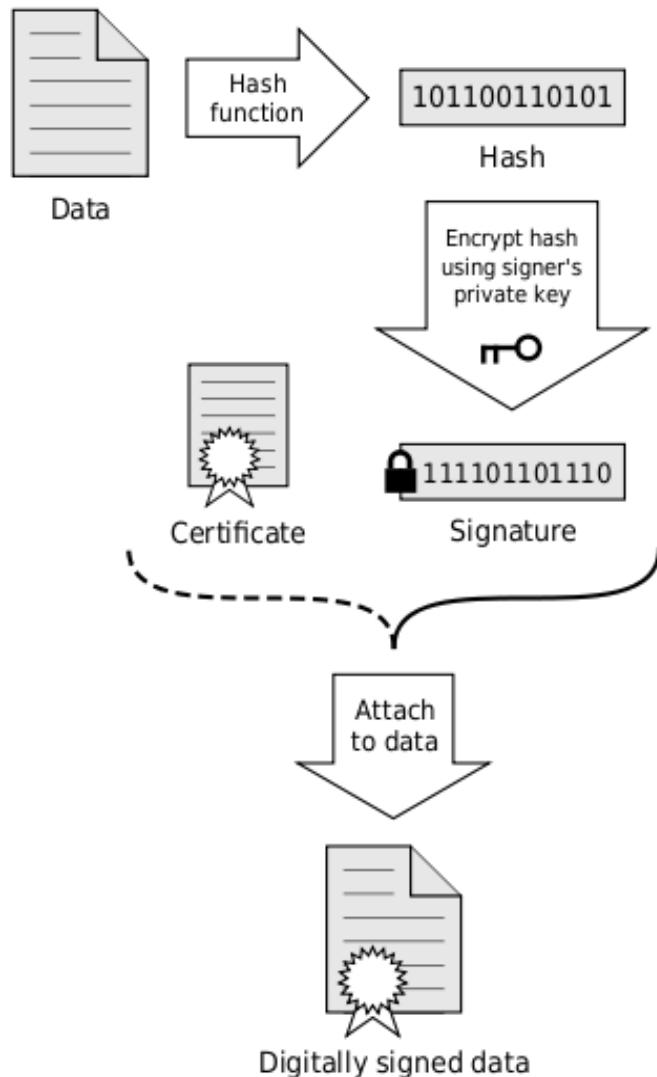
- **Key-only attack** : Oscar possesses Alice's public verification algorithm
- **Known message attack**: Oscar possesses a list of signed messages  $(x_i, y_i)$
- **Chosen message attack**: Oscar queries Alice for the signatures of a list of messages  $x_i$
- **Selective forgery**: C forges a signature for a particular message chosen by C.
- **Existential forgery**: C forges a signature for at least one message. C has no control over the message.

# DIGITAL SIGNATURE SCHEMES

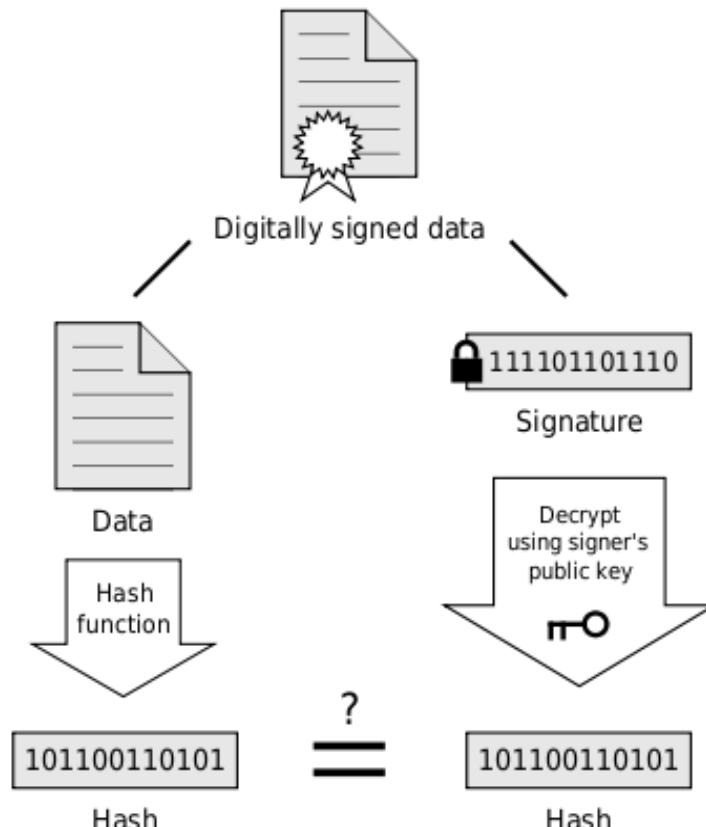
- A digital signature scheme is a mathematical scheme for demonstrating the authenticity of a digital message or document.
- A digital signature scheme typically consists of three algorithms:
  - A key generation algorithm that selects a private key uniformly at random from a set of possible private keys. The algorithm outputs the private key and a corresponding public key.
  - A signing algorithm that, given a message and a private key, produces a signature.
  - A signature verifying algorithm that, given a message, public key and a signature, either accepts or rejects the message's claim to authenticity.

# *General digital signature scheme*

## Signing



## Verification



If the hashes are equal, the signature is valid.

# **DIGITAL SIGNATURE SCHEMES**

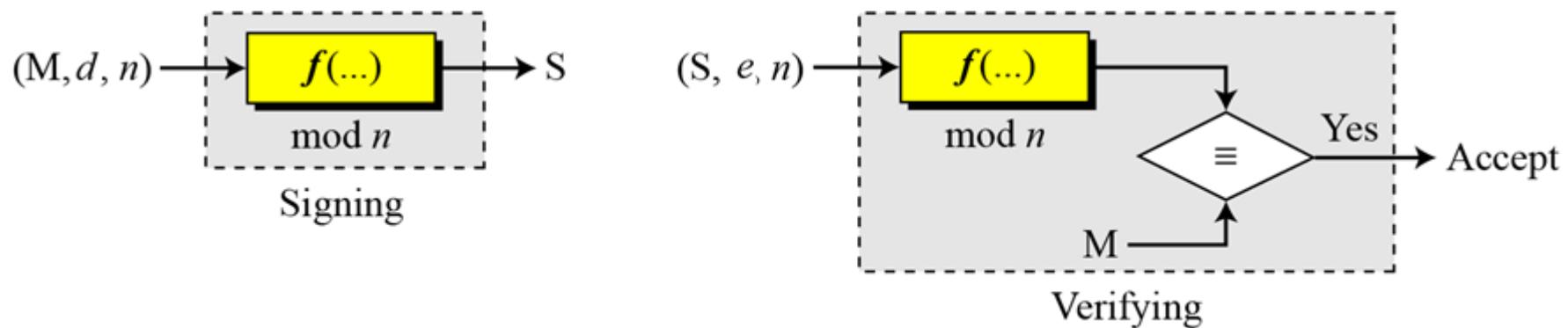
**RSA Digital Signature Scheme  
Digital Signature Standard (DSS)**

# RSA Digital Signature Scheme

*General idea behind the RSA digital signature scheme*

M: Message  
S: Signature

$(e, n)$ : Alice's public key  
 $d$ : Alice's private key



# RSA Digital Signature Scheme

## Key Generation

*Key generation in the RSA digital signature scheme is exactly the same as key generation in the RSA*

- *Alice chooses 2 prime numbers  $p$  and  $q$  and calculates  $n = p * q$*
- *Alice calculates  $\phi(n) = (p-1)(q-1)$*
- *She then chooses  $e$ , the public exponent, and calculates  $d$ , the private exponent such that  $e * d = 1 \text{ mod } \phi(n)$*
- *Alice keeps  $d$  and publicly announces  $e$  and  $n$*

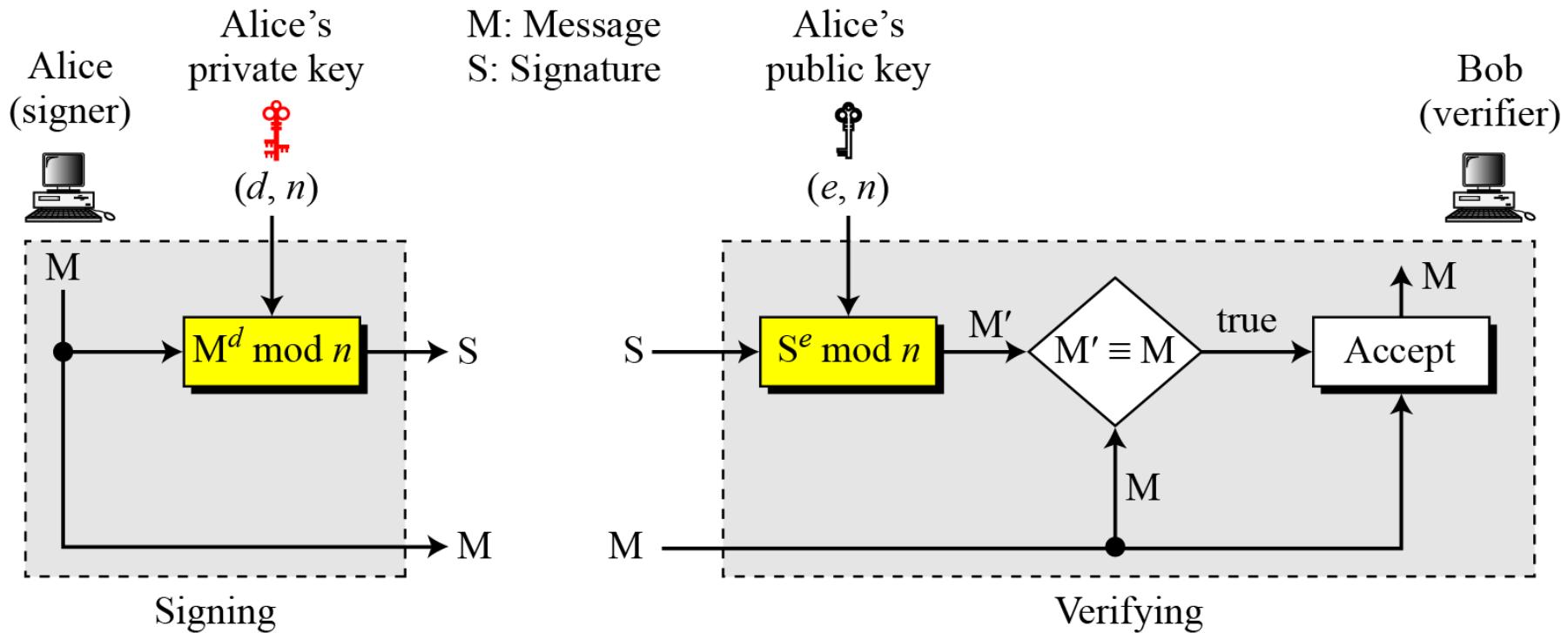
### Note

In the RSA digital signature scheme,  $d$  is private;  
 $e$  and  $n$  are public.

# RSA Digital Signature Scheme

## Signing and Verifying

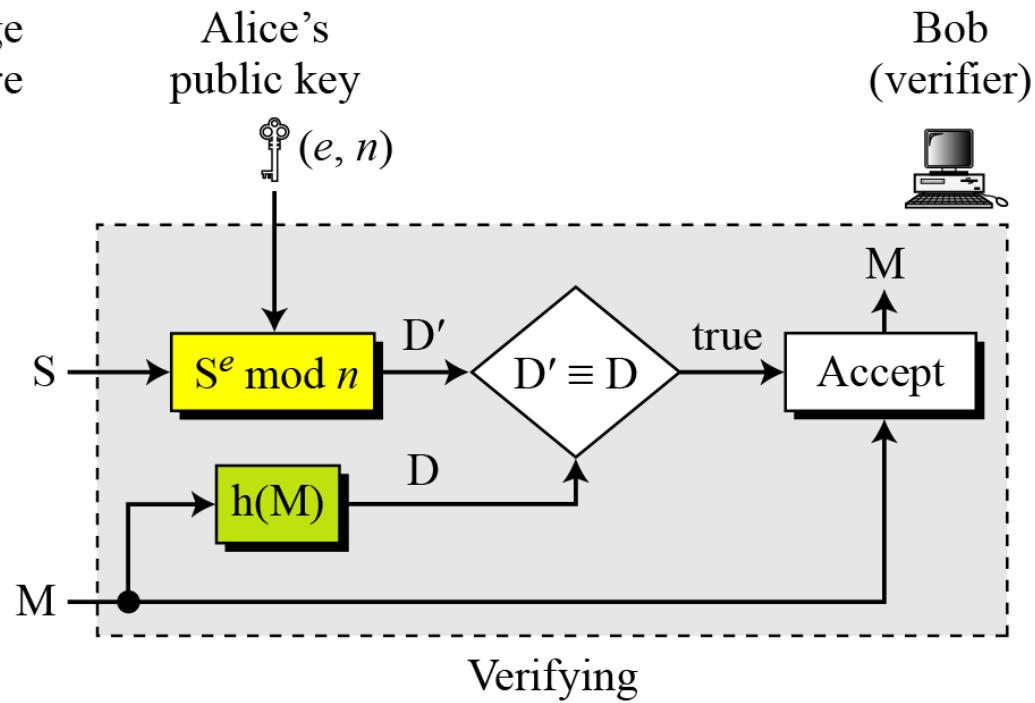
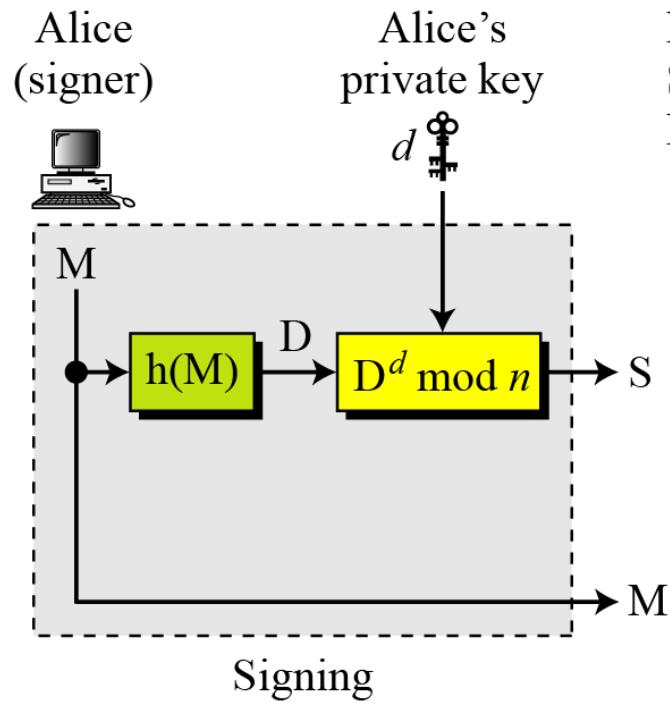
### RSA digital signature scheme



# RSA Digital Signature Scheme

## RSA Signature on the Message Digest

**The RSA signature on the message digest**



# Digital Signature Standard (DSS)

## General idea behind DSS scheme

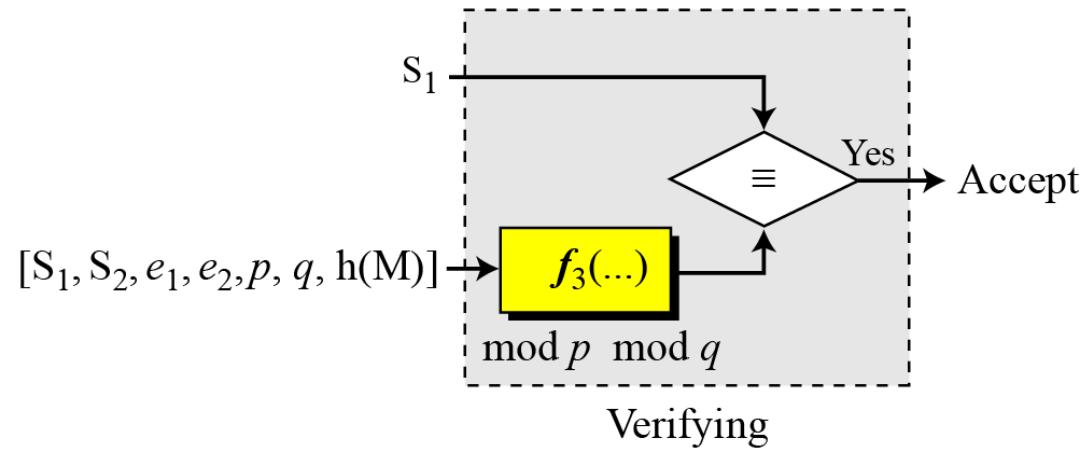
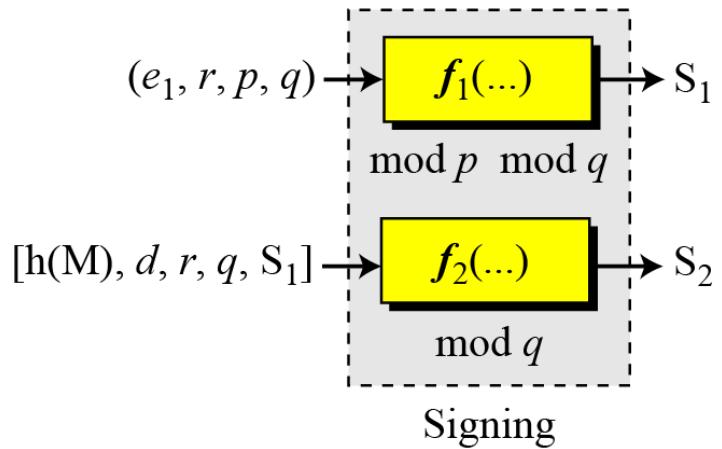
$S_1, S_2$ : Signatures

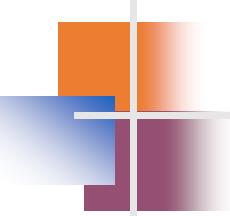
$d$ : Alice's private key

$M$ : Message

$r$ : Random secret

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





# *Digital Signature Standard (DSS)*

## *Key Generation.*

- 1) *Alice chooses primes  $p$ , between 512 and 1024 bits in length. The number of bits in  $p$  must be a multiple of 64.*
- 2) *Alice chooses a 160-bit prime  $q$  in such a way that  $q$  divides  $(p-1)$ .*
- 3) *Alice uses  $\langle \mathbb{Z}_p^*, \times \rangle$  and  $\langle \mathbb{Z}_q^*, \times \rangle$ , second is subgroup of the first.*
- 4) *Alice creates  $e_1$  to be the  $q$ th root of 1 modulo  $p$ . i.e.  
 $e_1^p = 1 \text{ mod } p$*
- 5) *Alice chooses  $d$  and calculates  $e_2 = e_1^d$ .*
- 6) *Alice's public key is  $(e_1, e_2, p, q)$ ; her private key is  $(d)$ .*

# Digital Signature Standard (DSS)

## Verifying and Signing

### DSS scheme

M: Message

$S_1, S_2$ : Signatures

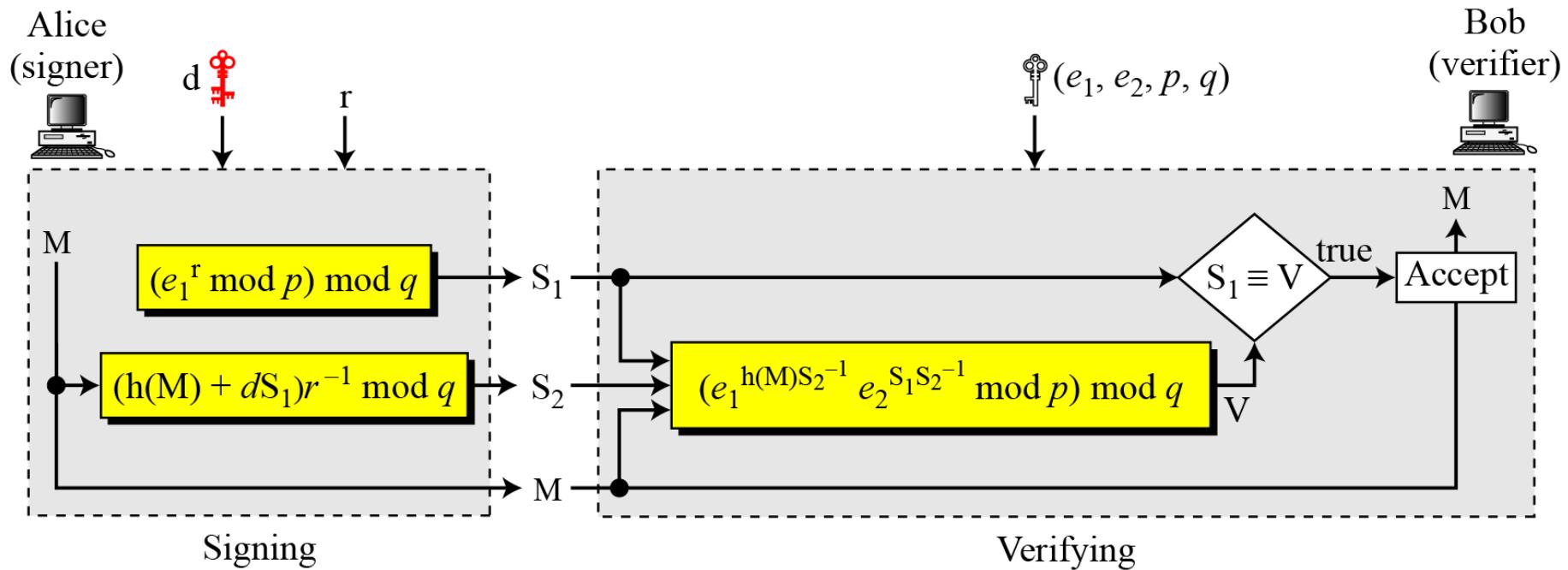
V: Verification

r: Random secret

d: Alice's private key

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

$h(M)$ : Message digest



# VARIATIONS

## *Time Stamped Signatures*

*Sometimes a signed document needs to be time stamped to prevent it from being replayed by an adversary. This is called time-stamped digital signature scheme.*

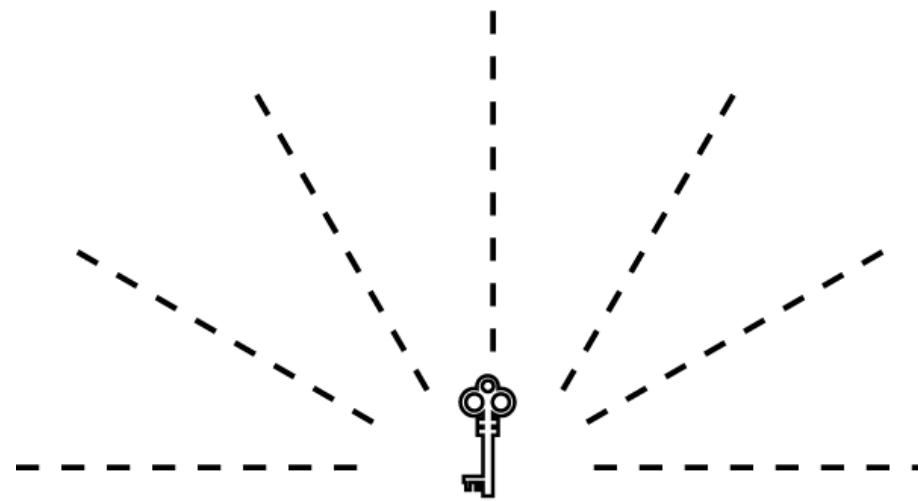
## *Blind Signatures*

*Sometimes we have a document that we want to get signed without revealing the contents of the document to the signer.*

# Public Key Distribution

- I. public announcement
- II. publicly available directory
- III. public-key authority
- IV. public-key certificates

*Announcing a public key*



Bob

# Public Announcement

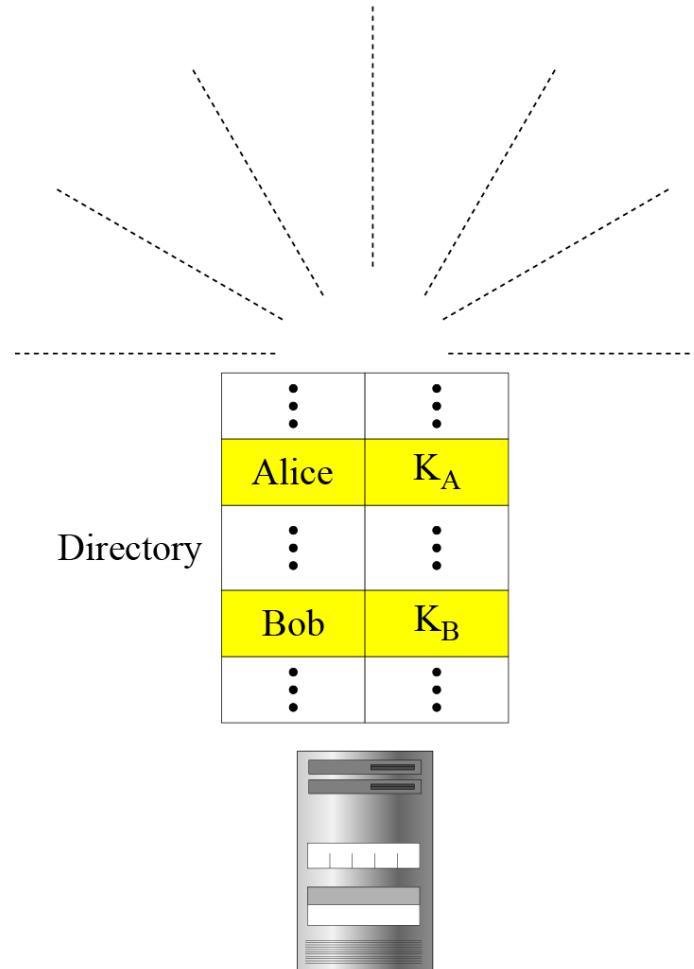
- users distribute public keys to recipients or broadcast to community at large
  - eg. append PGP keys to email messages or post to news groups or email list
- major weakness is forgery
  - anyone can create a key claiming to be someone else and broadcast it
  - until forgery is discovered can masquerade as claimed user

# Publicly Available Directory

- can obtain greater security by registering keys with a public directory
- directory must be trusted with properties:
  - contains {name,public-key} entries
  - participants register securely with directory
  - participants can replace key at any time
  - directory is periodically published
  - directory can be accessed electronically
- still vulnerable to tampering or forgery

# *Trusted Center Publicly available directory*

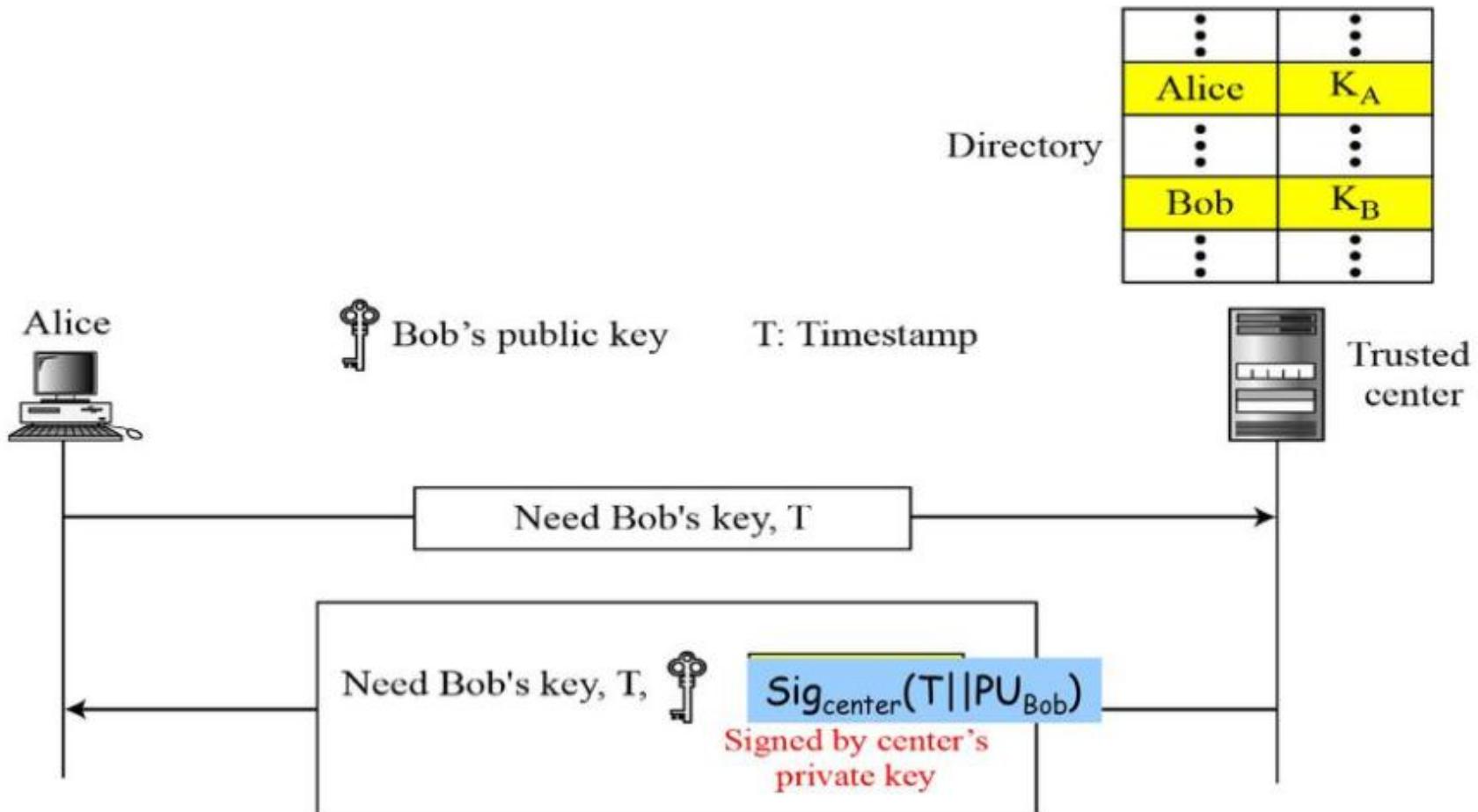
## *Trusted center*



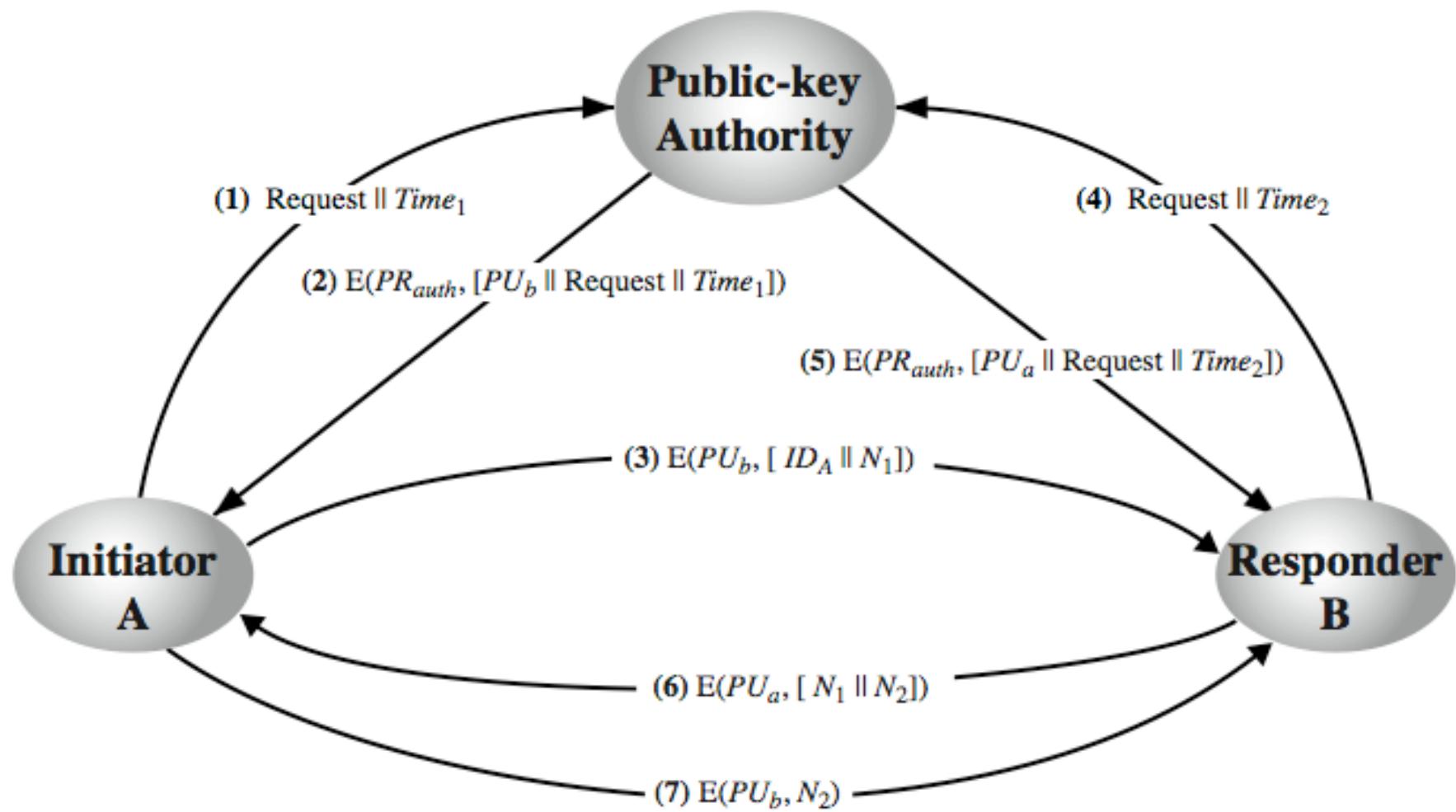
# Public-Key Authority

- improve security by tightening control over distribution of keys from directory
- has properties of directory
- and requires users to know public key for the directory
- then users interact with directory to obtain any desired public key securely
  - does require real-time access to directory when keys are needed
  - may be vulnerable to tampering

## Controlled trusted center



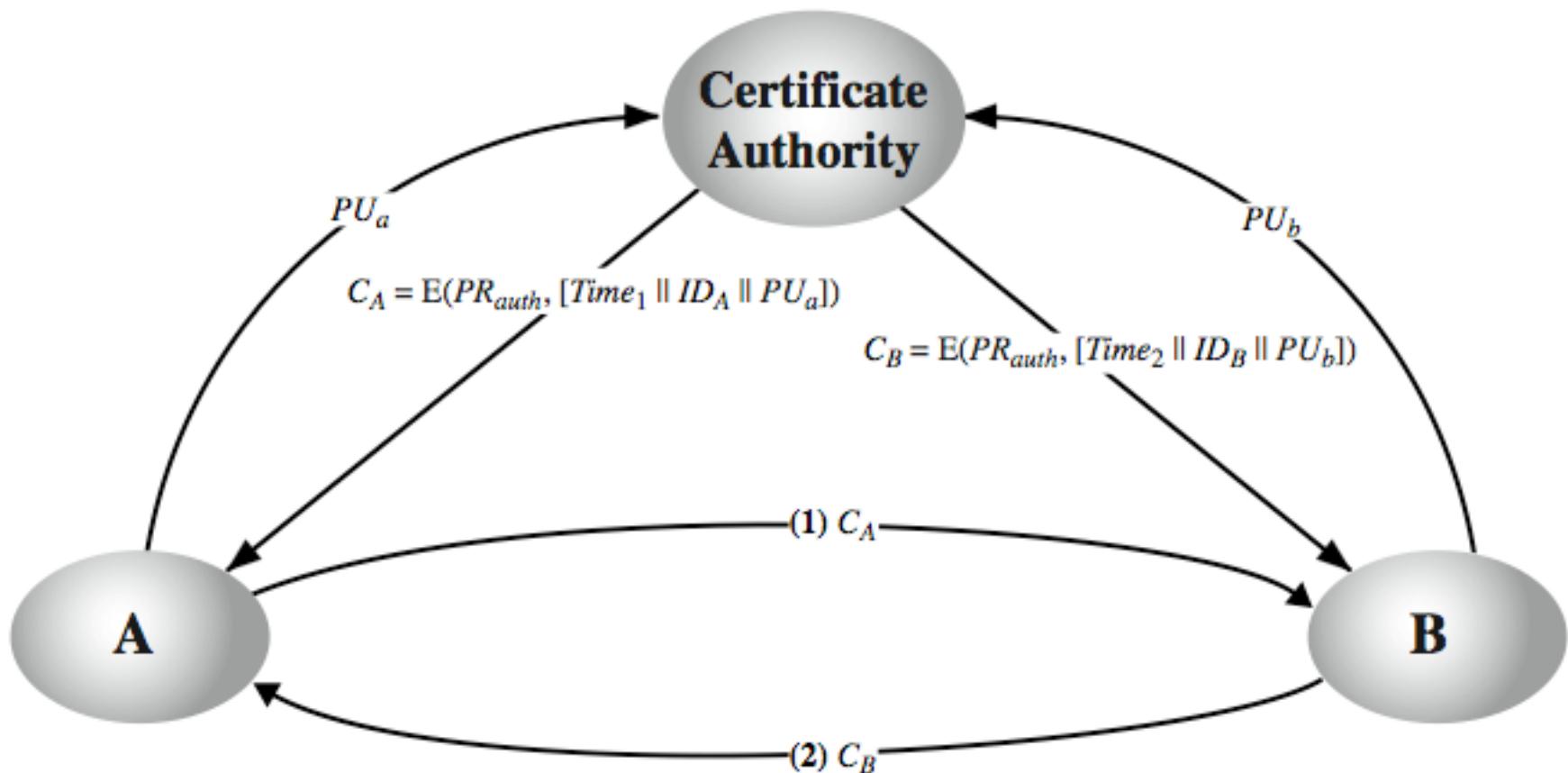
# Public-Key Authority



# Public-Key Certificates

- certificates allow key exchange without real-time access to public-key authority
- a certificate binds **identity** to **public key**
  - usually with other info such as period of validity, rights of use etc
- with all contents **signed** by a trusted Public-Key or Certificate Authority (CA)
- can be verified by anyone who knows the public-key authorities public-key

# Public-Key Certificates



# Certificate Creation Steps

- Parties Involved :
  - ✓ CA
  - ✓ RA: CA is overloaded with a variety of task such as issuing new certificate, revoking, maintaining etc CA delegates some of its task to his third party called **Registration Authority(RA)**
  - ✓ RA is an intermediate entity between the end user and the CA
  - ✓ RA Commonly provides
    - Accepting and verifying registration
    - Generating keys on behalf of user
    - Accepting and authorizing the request for certificate revocation
    - RA cannot issue digital certificate , its CA's responsibility

# Certificate Creation Steps

- Step 1 : Key Generation
- Step 2 : Registration: User sends the public key and the associated registration information all evidences to the RA.
  - ✓ The format for Certificate request is standardized called Certificate Signing Request(CSR).
  - ✓ After this the user usually gets a request identifier for tracking the progress of the certificate request
- Step 3:Verification:
  - ✓ RA need to verify the user's credentials and ensure that they are acceptable
  - ✓ The second Check is to ensure that the user who is requesting the certificate indeed possess the private key corresponding to the public key .Proof of Possession(POP).
  - ✓ RA can demand the user must digitally sign her CSR using Private key and RA verify with Public key.

# Certificate Creation Steps

- Certificate Creation: Assuming that all the steps have been successful. RA Passes all the information to the CA. The CA Creates a digital certificate send it to the user and also retain a copy of certificate in a certificate directory.
- The certificate is signed with the private key of CA, to get the certificate we need the public key of CA .

# X.509 certificate

Although the use of a CA solves the problem of public-key fraud, it has created a side-effect

Each certificate may have a different format

X.509 is used to remove the side-effect

X.509 is a way to describe the certificate in a structured way

# X.509 certificate scheme

- X.509 scheme for generation of a public-key certificate.
- The certificate for Bob's public key includes unique identifying information for Bob, Bob's public key, and identifying information about the CA, plus other information

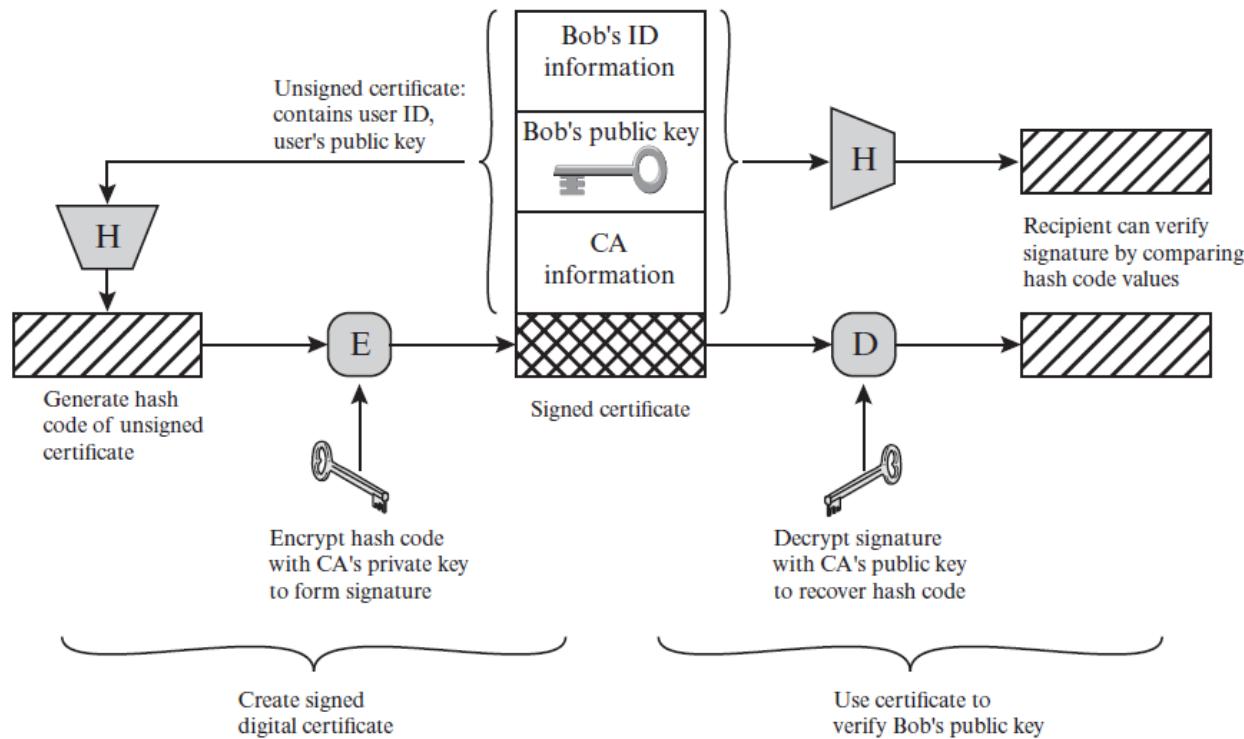
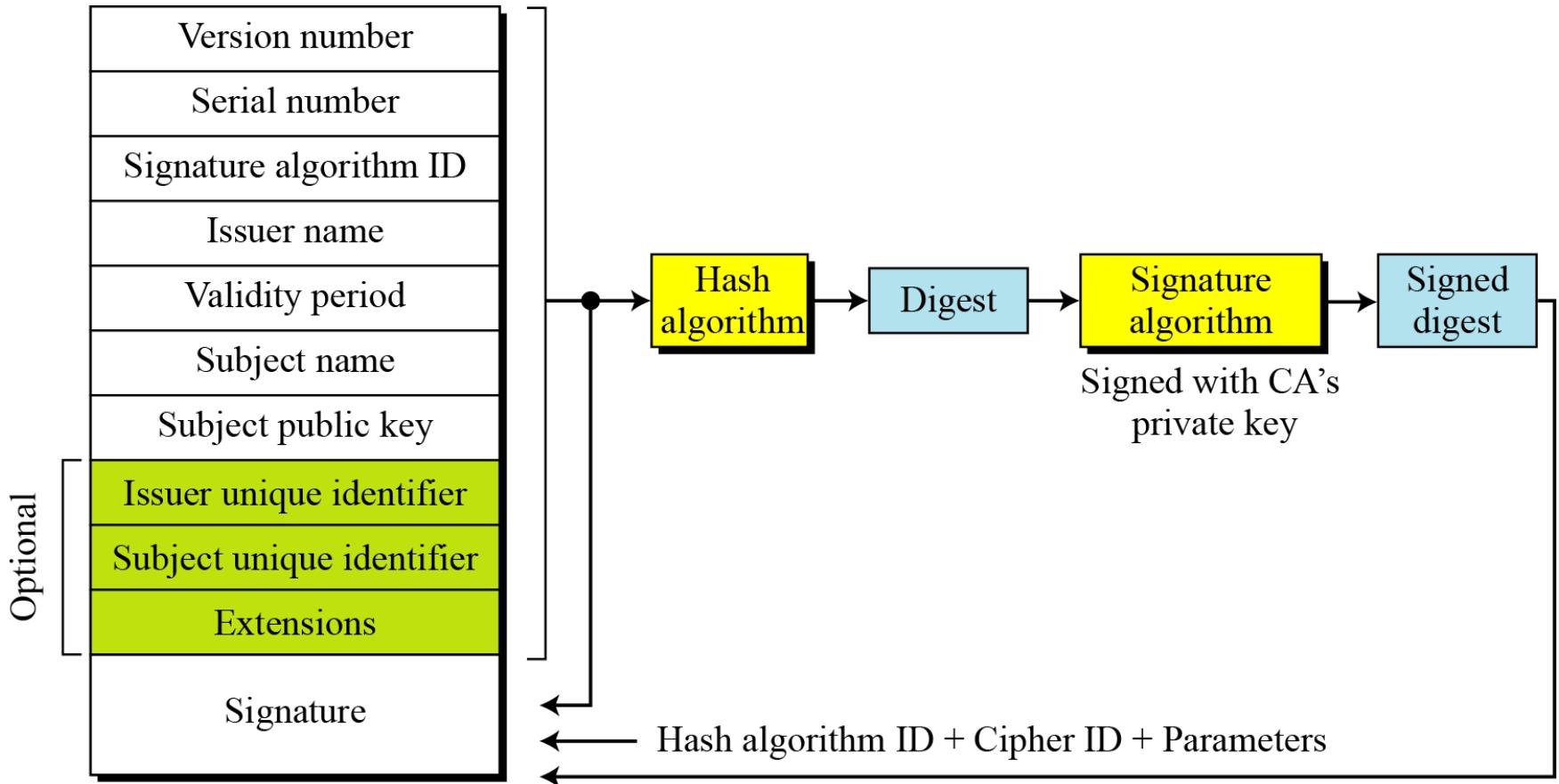


Figure 14.14 X.509 Public-Key Certificate Use

# X.509 Certificates



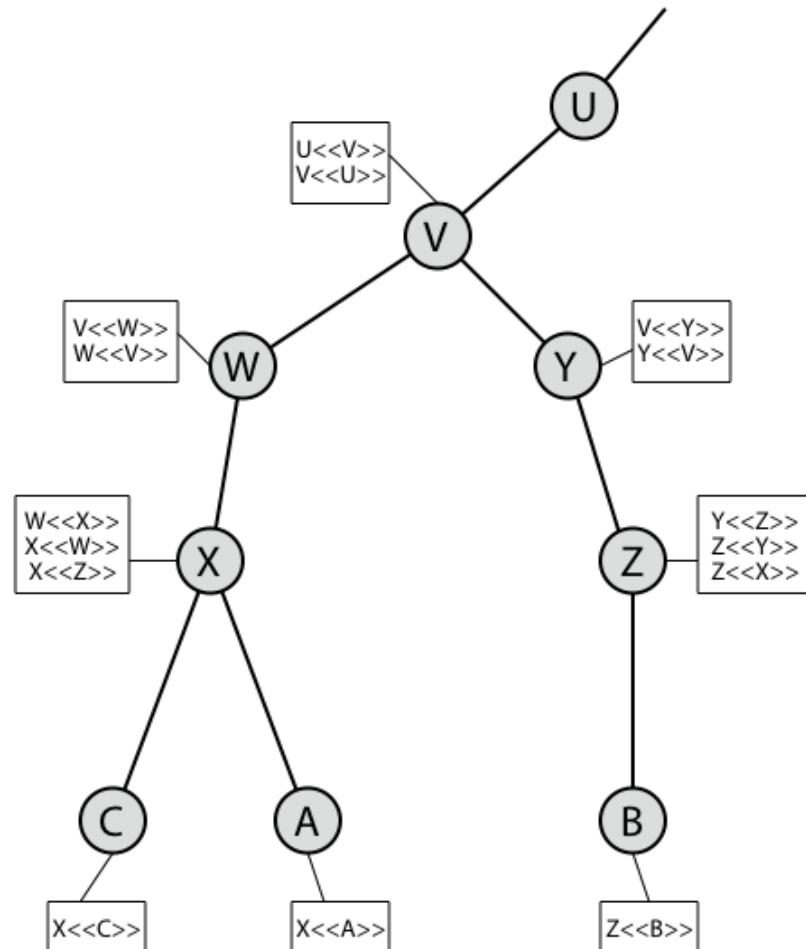
# Certificate

- Version number:The field defines the version of X.509 of the certificate.
- Serial number:The field define a number assigned to each certificate
- Signature algorithm ID:algorithm used to sign the certificate
- Issuer name:identified certification authority that issued the certificate
- Validity Period:Time
- Subject name:entity to which public key belongs
- Subject Public key: heart of certificate
- Signature: This field is made of 3 section. First Section all other fields, second digest of first section encrypted with CA's Private key . The third is algorithm identifier used to create the second section.

# CA Hierarchy

- if both users share a common CA then they are assumed to know its public key
- otherwise CA's must form a hierarchy(chain of trust)
- use certificates linking members of hierarchy to validate other CA's
  - each CA has certificates for clients (forward) and parent (backward)
- each client trusts parents certificates
- enable verification of any certificate from one CA by users of all other CAs in hierarchy
- .The directory entry for each CA includes two types of certificates:
  - Forward certificates: Certificates of X generated by other CAs
  - Reverse certificates: Certificates generated by X that are the certificates of other CAs.

# CA Hierarchy Use

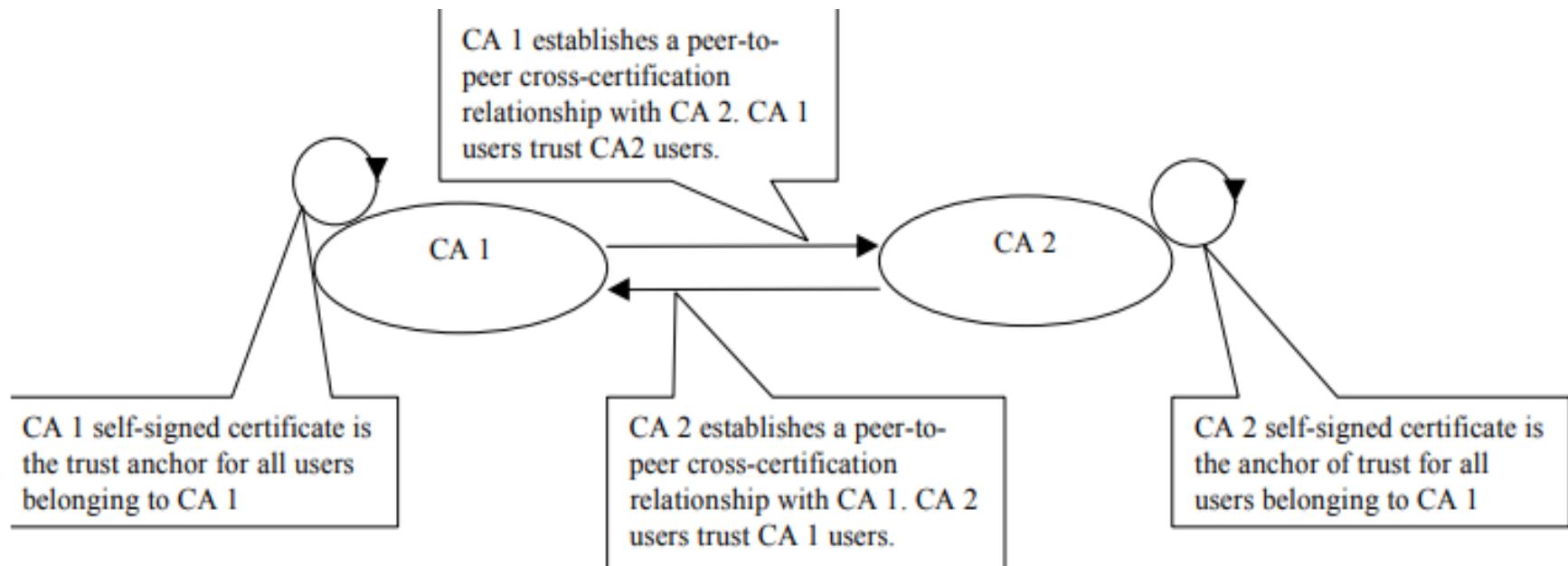


A acquires B certificate using chain: X<<W>>W<<V>>V<<Y>>Y<<Z>>Z<<B>>  
B acquires A certificate using chain: Z<<Y>>Y<<V>>V<<W>>W<<X>>X<<A>>

# Cross Certificate

- Root CA's could be different
- No single root CA
- Cross Certification:Single monolithic CA Certifying every possible user in the world is quite unlikely. Instead , the concept of decentralized CA's for different Countries, business etc
- It allows CA and end user from different domain to interact
- The root CA's cross certify each other. Technically Alice's root CA has obtained a certificate for itself from Bob's root CA.Similarly bob's CA has obtained a certificate from Alice's root CA.

# Cross Certification



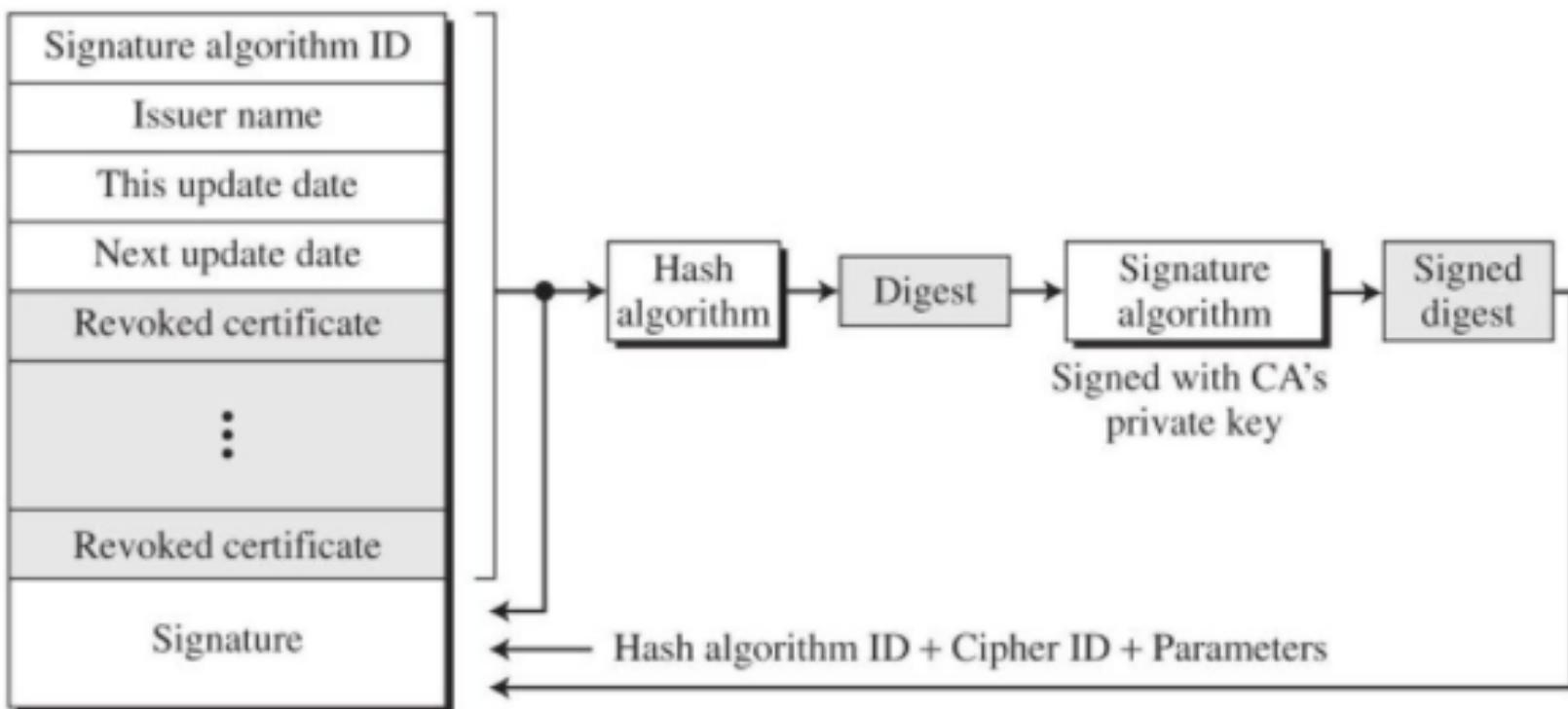
## Certificate Renewal

- Each certificate has a period of validity
- If there is no problem with the certificate, the CA issues a new certificate before the old one expires.
- In some cases a certificate must be revoked before its expiration

# Certificate Revocation

- certificates have a period of validity
- may need to revoke before expiry, eg:
  1. user's private key is compromised
  2. user is no longer certified by this CA
  3. CA's certificate is compromised
- CA's maintain list of revoked certificates
  - the Certificate Revocation List (CRL)
- users should check certificates with CA's CRL

# Certificate Revocation List Format



# Certificate Revocation List

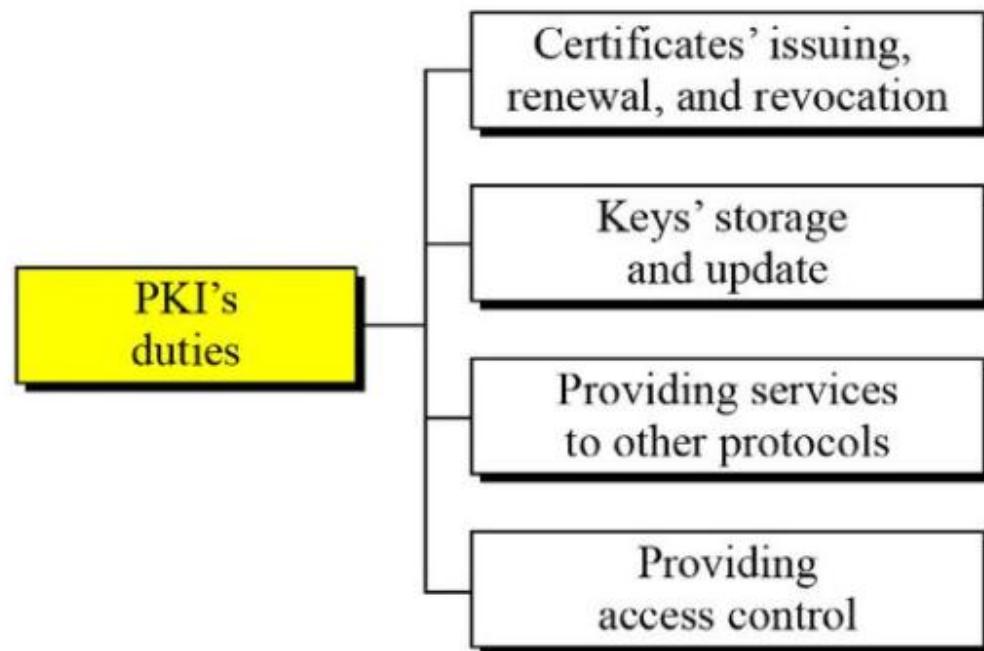
- CRL is a list of certificates published regularly by each CA, identifying all the certificates that have been revoked.
- It does not include certificates whose validity period is over.
- It's a sequential file that grows over time, it lists the serial number, the date and time on which the certificate was revoked.
- The CRL can become really quite big over time.
- The bottleneck is solved using Delta CRL (change since last update).
- This mechanism makes the CRL file size small and therefore its transmission is easier.
- The changes to base CRL is called Delta CRL.

# PKI

- **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.
- Its principal is to enable secure, convenient, and efficient acquisition of public keys.
- The IETF Public Key Infrastructure X.509 (PKIX) working group has setup a formal (and generic) model based on X.509 that is suitable for deploying a certificate-based architecture on the Internet.

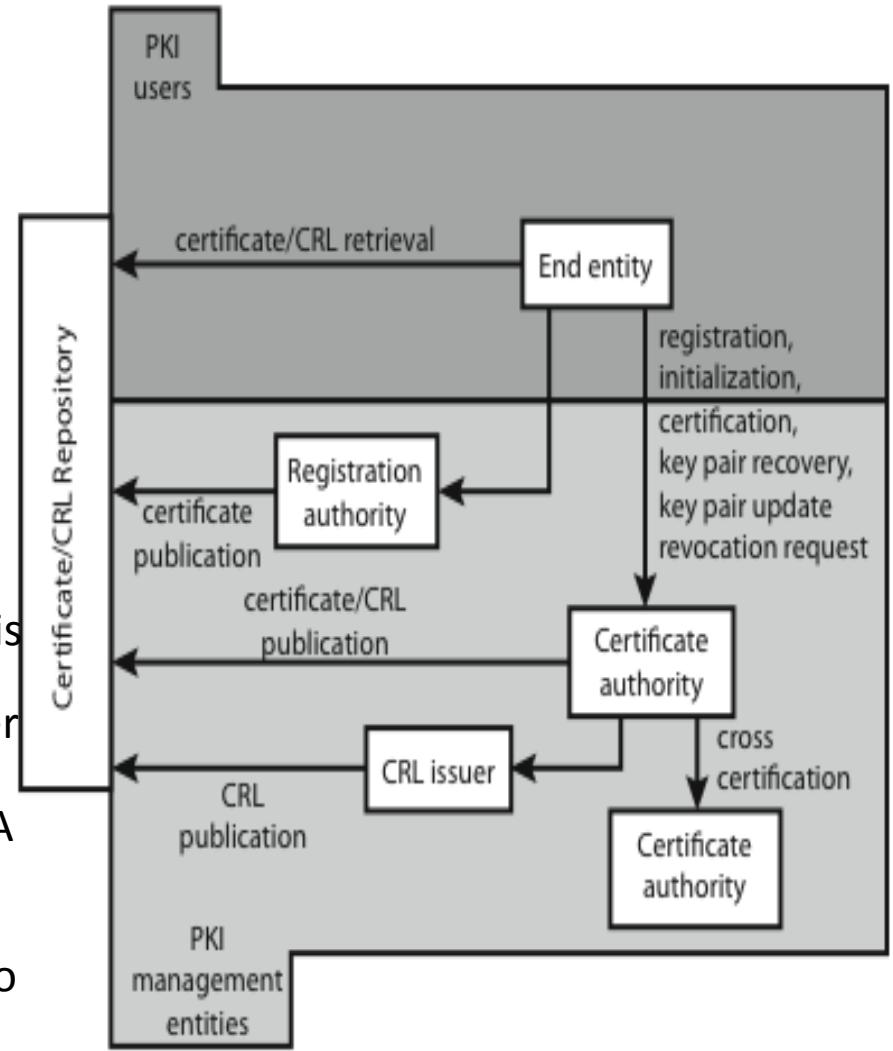
# Public Key Infrastructure

- ❑ PKI is a model for creating, distributing and revoking certificates based on X.509



# Public Key Infrastructure

- **End entity:** A generic term used to denote end users, devices (e.g., servers, routers), or any other entity that can be identified in the subject field of a public key certificate. End entities can consume and/or support PKI-related services.
- **Certification authority (CA):** The issuer of certificates and (usually) certificate revocation lists (CRLs). It may also support a variety of administrative functions, although these are often delegated to Registration Authorities.
- **Registration authority (RA):** An optional component that can assume a number of administrative functions from the CA. The RA is often associated with the End Entity registration process, but can assist in a number of other areas as well.
- **CRL issuer:** An optional component that a CA can delegate to publish CRLs.
- **Repository:** A generic term used to denote any method for storing certificates and CRLs so that they can be retrieved by End Entities.



# PKIX Management

➤ functions:

- registration
- initialization
- certification
- key pair recovery
- key pair update
- revocation request
- cross certification

➤ protocols: CMP, CMC

# UNIT 4

## INTERNET AND WEB SECURITY

# Contents

- ▶ SSL
- ▶ IPSec
- ▶ Email Security- PGP, Email attacks
- ▶ Web services Security:
  - ▶ web app versus web service concept, WS-Security, SOAP web service, SAML assertion, Browser attacks, web attacks targeting users, obtaining user or website data.

# SSL (Secure Socket Layer)

# Web Security

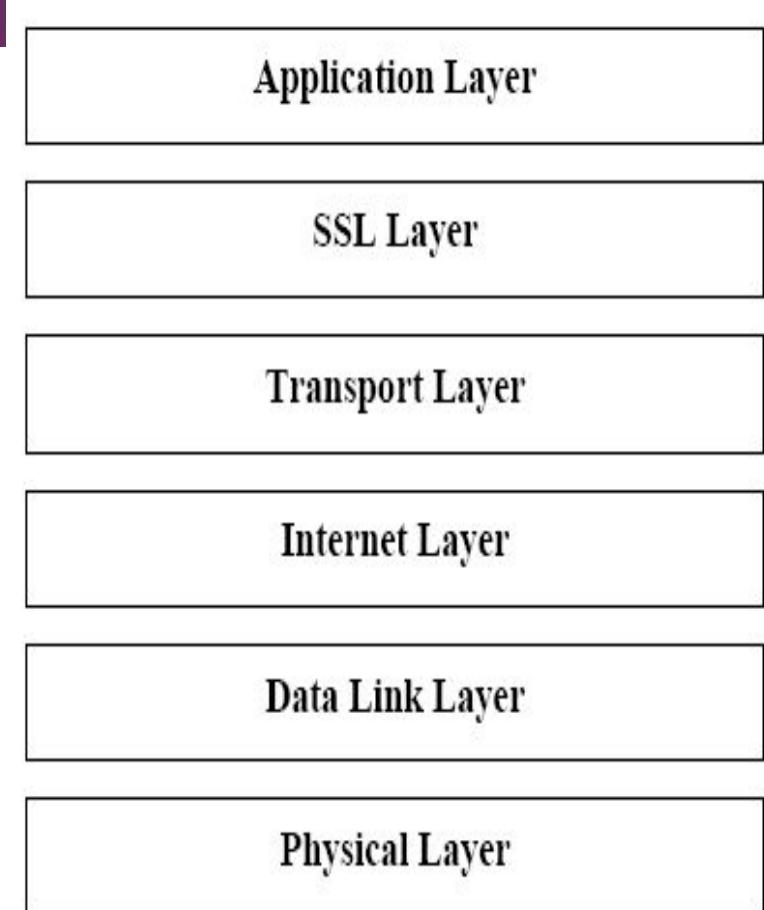
- ▶ Web now widely used by business, government, individuals
- ▶ but Internet & Web are vulnerable
- ▶ have a variety of threats
  - ▶ integrity
  - ▶ confidentiality
  - ▶ denial of service
  - ▶ authentication
- ▶ need added security mechanisms

# SSL (Secure Socket Layer)

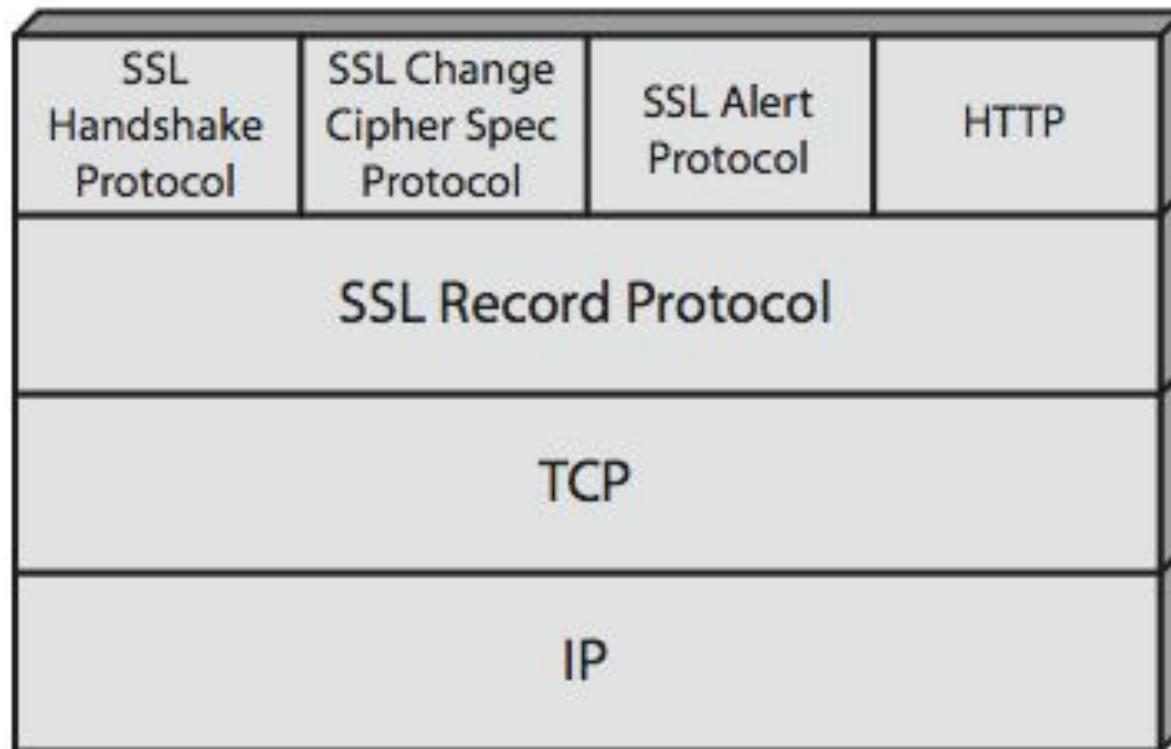
- ▶ SSL probably most widely used Web security mechanism.
- ▶ This is an internet protocol for secure exchange of information between a web browser and a web server.
- ▶ It provides a secure pipe between the web browser and the web server.
- ▶ Its implemented at the Transport layer
- ▶ It is originally developed by Netscape
- ▶ uses TCP to provide a reliable end-to-end service
- ▶ SSL has two layers of protocols

# Position of SSL in TCP/IP

- ▶ Hypertext Transfer Protocol (HTTP), which provides the transfer service for Web client/server interaction, operate on top of SSL.



# SSL Architecture



# SSL Architecture

- ▶ SSL has two layers of protocols .
- ▶ SSL Record Protocol provides basic security services to various higher-layer protocols.
- ▶ Three higher-layer protocols of SSL:
  1. Handshake Protocol
  2. Change Cipher Spec Protocol
  3. Alert Protocol.
- ▶ These SSL-specific protocols are used in the management of SSL exchanges.

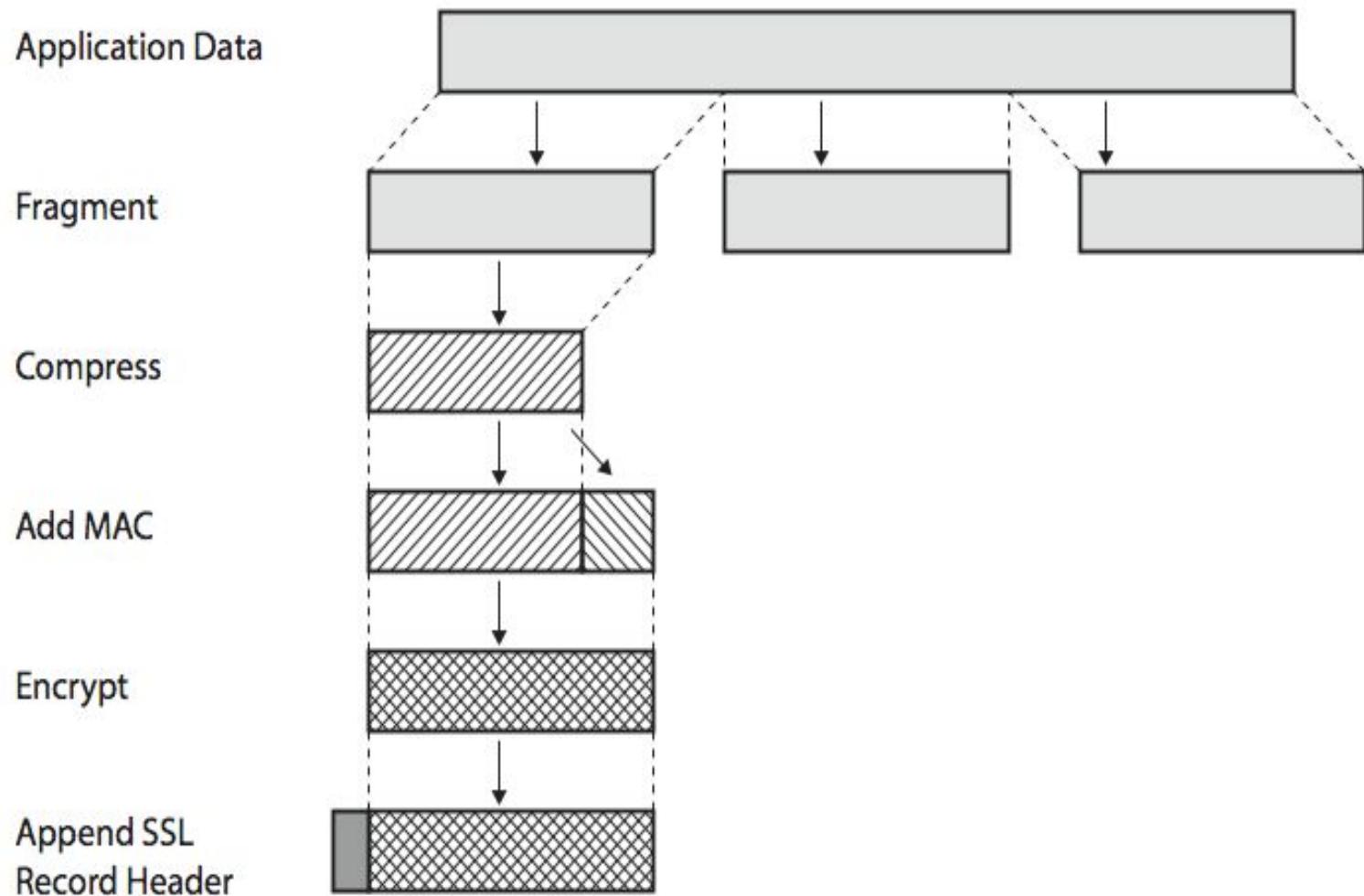
# SSL Architecture

- ▶ **SSL connection**
  - ▶ A connection is a transport that provides a suitable type of service, peer-to-peer relationships, associated with one session
- ▶ **SSL session**
  - ▶ An association between client & server
  - ▶ Created by the handshake protocol

# SSL Record Protocol Services

- ▶ SSL Record Protocol provides two services for SSL connections.
- ▶ **confidentiality**
  - ▶ The Handshake Protocol defines a shared secret key that is used for encryption.
  - ▶ message is compressed before encryption
- ▶ **message integrity**
  - ▶ The Handshake Protocol also defines a shared secret key that is used to form a message authentication code (MAC), which is similar to HMAC

# SSL Record Protocol Operation



# SSL Record Protocol Operation

## 1. **Fragmentation:**

- ▶ It takes an application message to be transmitted and fragments it into manageable blocks. These block are  $2^{14} = 16,384$  bytes or less.

## 2. **Compression:**

- ▶ These blocks are then optionally compressed which must be lossless and may not increase the content length by more than 1024 bytes.

## 3. **Addition of MAC:**

- ▶ A message authentication code is then computed over the compressed data using a shared secret key.
- ▶ This is then appended to the compressed (or plaintext) block.

# SSL Record Protocol Operation

## 4. **Encryption:**

- ▶ The compressed message plus MAC are then encrypted using symmetric encryption.
- ▶ Encryption may not increase the content length by more than 1024 bytes.
- ▶ A number of different encryption algorithms are permitted.

## 5. **Append Header:**

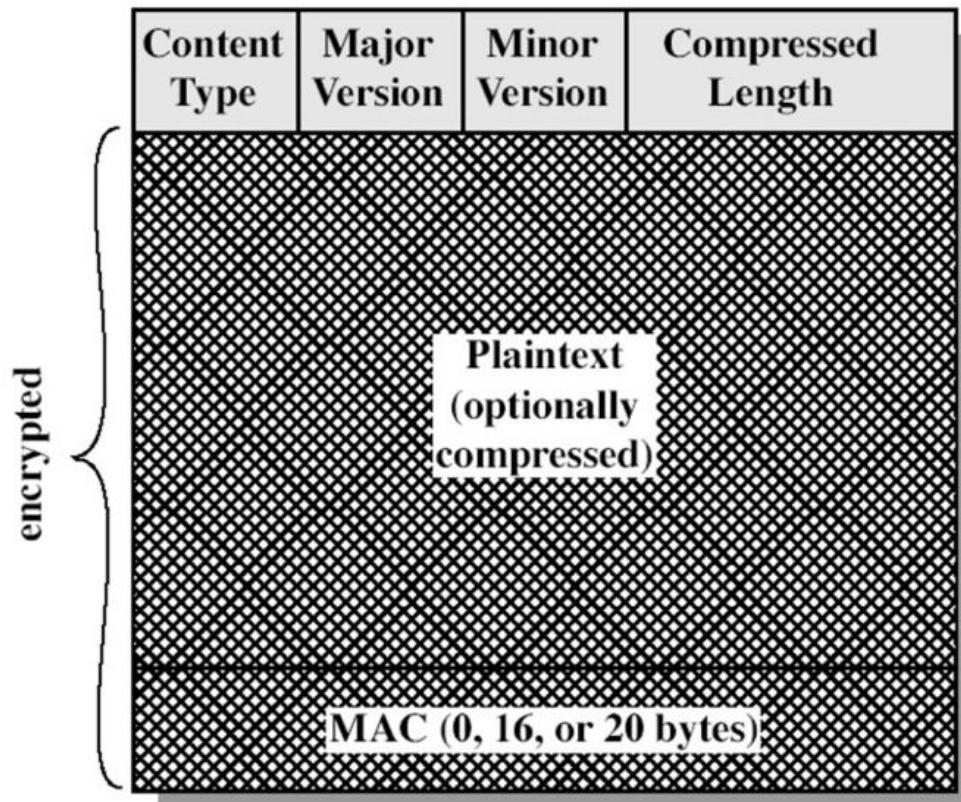
- ▶ The final step is to append a header to encrypted block.

# SSL Record Protocol Operation

**The header consists of the following fields:**

- ▶ **Content type (8 bits)** - The higher layer protocol used to process the enclosed fragment.( handshake, alert, change cipher)
- ▶ **Major Version (8 bits)** - Indicates major version of SSL in use. For SSLv3, the value is 3.
- ▶ **Minor Version (8 bits)** - Indicates minor version in use. For SSLv3, the value is 0.
- ▶ **Compressed Length (16 bits)** - The length in bytes of the compressed (or plaintext) fragment.

# SSL record format



# SSL Handshake Protocol

- ▶ allows server & client to:
  - ▶ authenticate each other
  - ▶ to negotiate encryption & MAC algorithms
  - ▶ to negotiate cryptographic keys to be used
- ▶ **Phases of Handshake Protocol**
  1. Establish Security Capabilities
  2. Server Authentication and Key Exchange
  3. Client Authentication and Key Exchange
  4. Finalizing the handshake protocol

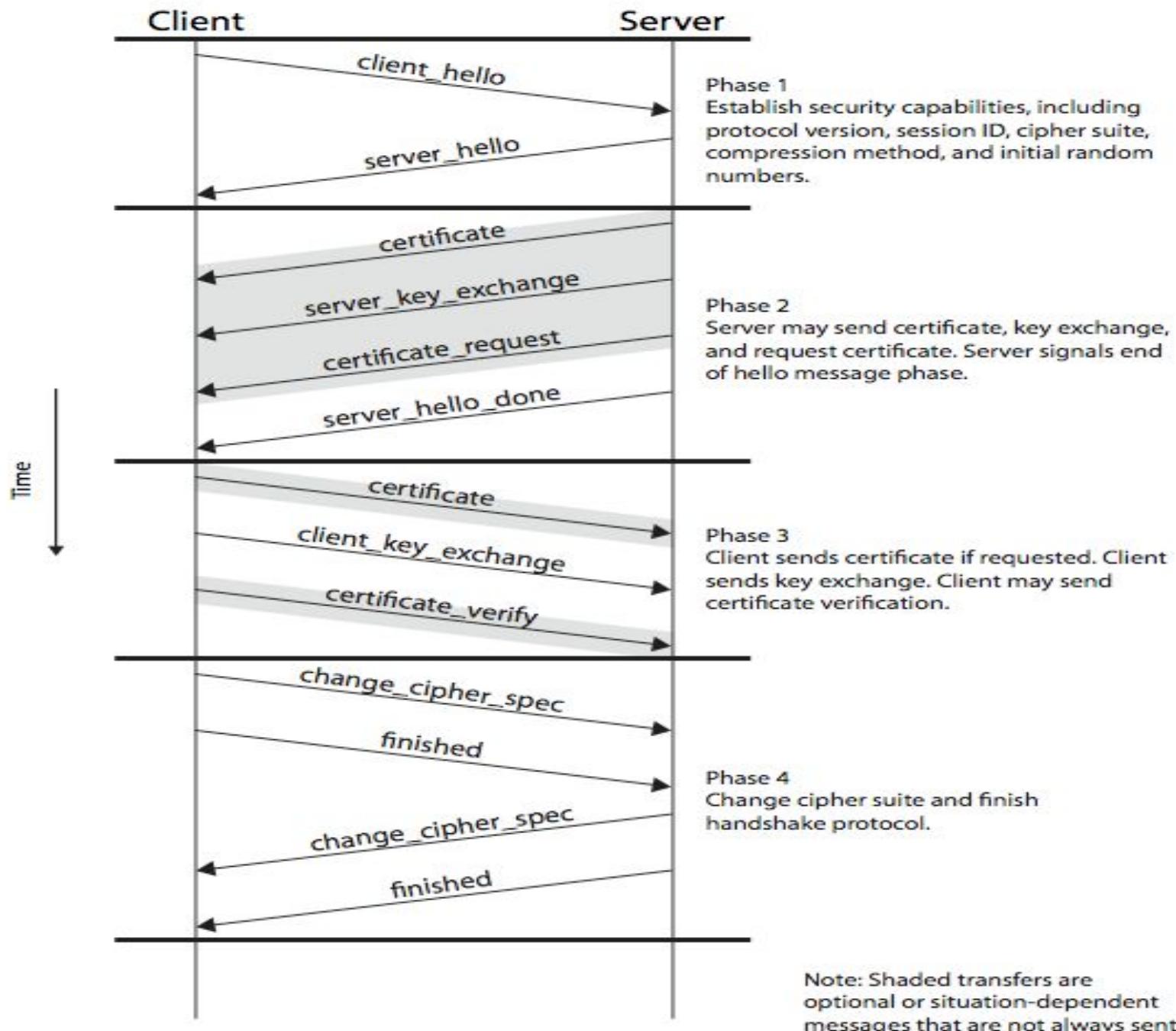
# SSL Handshake Protocol

- ▶ Each message has three fields:
  1. Type (1 byte): Indicates one of 10 messages
  2. Length (3 bytes): The length of the message in bytes.
  3. Content( 0 byte): The parameters associated with message

1 byte	3 bytes	$\geq 0$ bytes
Type	Length	Content

# SSL Handshake Protocol

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



# Client Hello

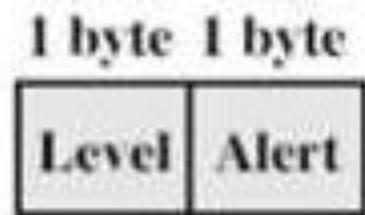
- ▶ It contains
  1. The Highest SSL Version the client supports
  2. A 32-byte random number from the client used for master secret generation
  3. A session ID that defines the session
  4. A cipher Suite that defines the list of algorithm the client can support
  5. A list of compression methods that client can support

# Server Hello

- ▶ It Contains
  1. An SSL Version number.- This number is the lower of two version number the highest supported by the client and highest supported by server
  2. A 32 byte random number
  3. A session ID
  4. The selected Cipher Set from the client list
  5. The selected Compression method from the client list

# SSL Alert Protocol

- ▶ This protocol is used to convey SSL-related alerts to the peer entity.
- ▶ It consists of two bytes the first of which takes the values 1 (warning) or 2 (fatal).
- ▶ If the level is fatal SSL immediately terminates the connection.
- ▶ The second byte contains a code that indicates the specific alert.



# Alert types

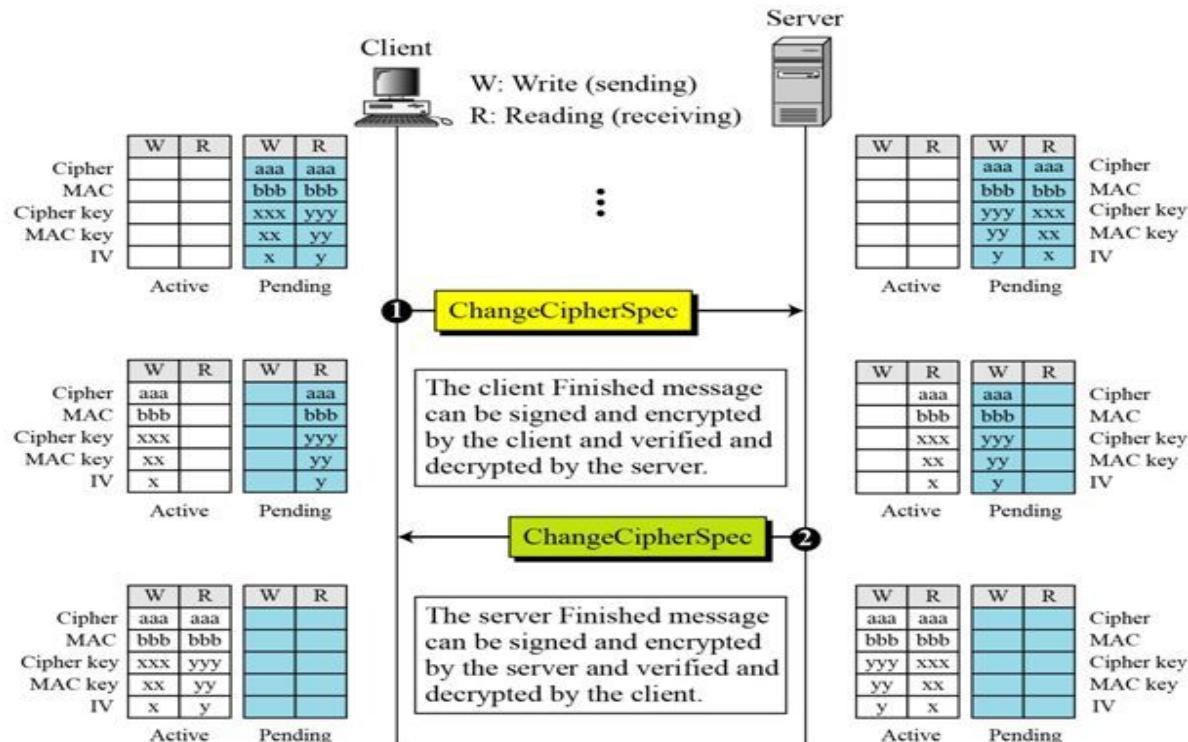
<i>Value</i>	<i>Description</i>	<i>Meaning</i>
0	<i>CloseNotify</i>	Sender will not send any more messages.
10	<i>UnexpectedMessage</i>	An inappropriate message received.
20	<i>BadRecordMAC</i>	An incorrect MAC received.
30	<i>DecompressionFailure</i>	Unable to decompress appropriately.
40	<i>HandshakeFailure</i>	Sender unable to finalize the handshake.
41	<i>NoCertificate</i>	Client has no certificate to send.
42	<i>BadCertificate</i>	Received certificate corrupted.
43	<i>UnsupportedCertificate</i>	Type of received certificate is not supported.
44	<i>CertificateRevoked</i>	Signer has revoked the certificate.
45	<i>CertificateExpired</i>	Certificate expired.
46	<i>CertificateUnknown</i>	Certificate unknown.
47	<i>IllegalParameter</i>	An out-of-range or inconsistent field.

# SSL Change Cipher Spec Protocol

- ▶ This consists of a single message which consists of a single byte with the value 1.
- ▶ This is used to cause the pending state to be copied into the current state which updates the cipher suite to be used on this connection.



## Change Cipher Spec Protocol



# *IP Security*

# *IP Security Issues*

- ▶ Many solutions are application-specific:
  - ▶ S/MIME, PGP for electronic mail , Kerberos for client/server, Secure Sockets Layer for secure Web access .
- ▶ However routing protocol directly work at Internet Layer.
- ▶ IP Packet contain data in plain text format, anyone can modify the packet.
- ▶ Attacker can insert forged source IP addresses in IP header.
- ▶ Attacker may be able to corrupt routing tables on routers by sending false updates.

# *IP Security*

Application Layer

*Second level of security*

Transport Layer

Internet Layer

*First level of security*

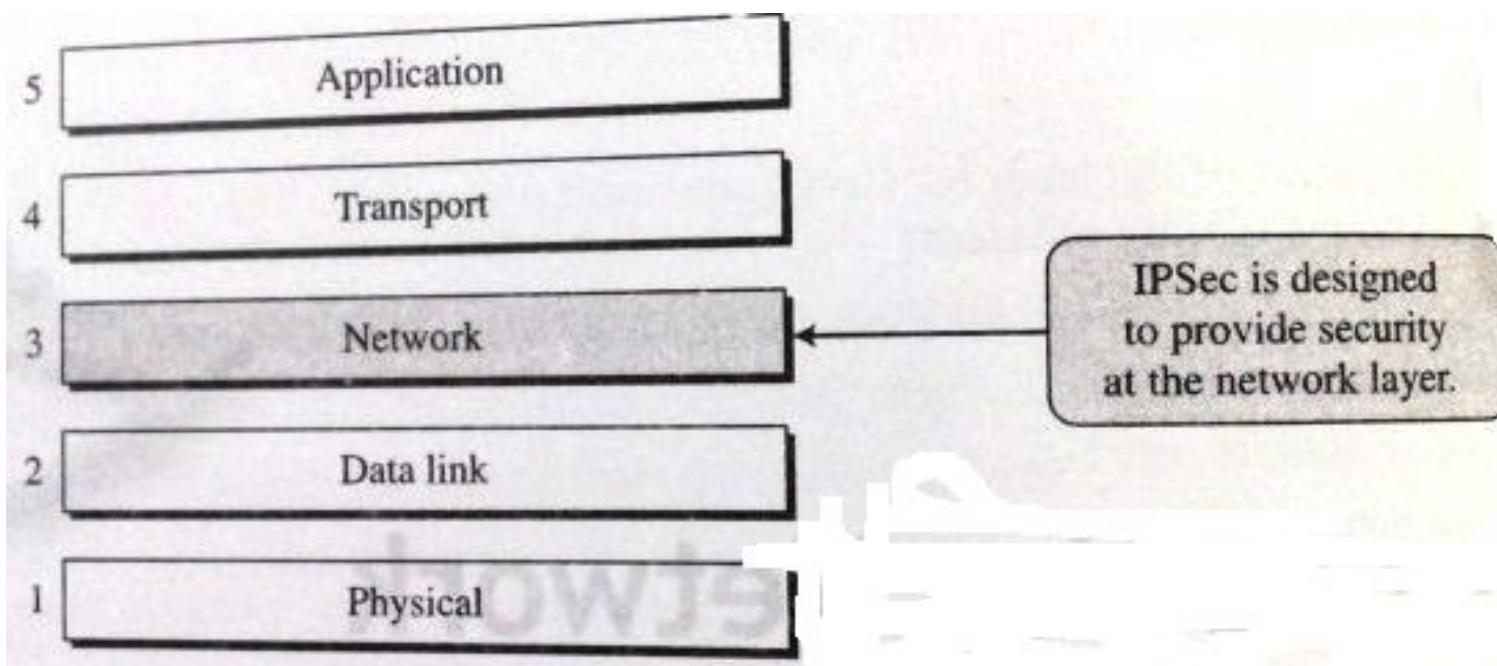
Data Link Layer

Physical Layer

# *IPSec overview*

- ▶ General IP Security mechanisms provides
  - ▶ authentication
  - ▶ confidentiality
  - ▶ key management
- ▶ The authentication mechanism assures that a received packet was transmitted by the party identified as the source in the packet header, and that the packet has not been altered in transit.
- ▶ The confidentiality facility enables communicating nodes to encrypt messages to prevent eavesdropping by third parties.
- ▶ The key management facility is concerned with the secure exchange of keys.

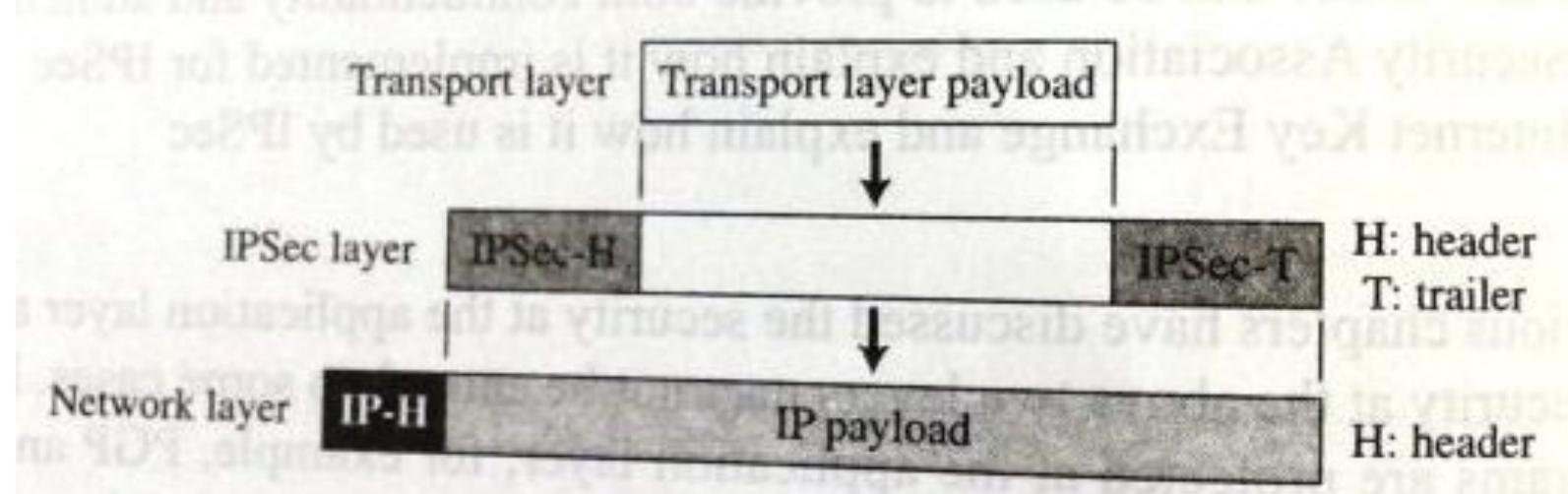
# *TCP/IP Protocol Suite & IPsec*



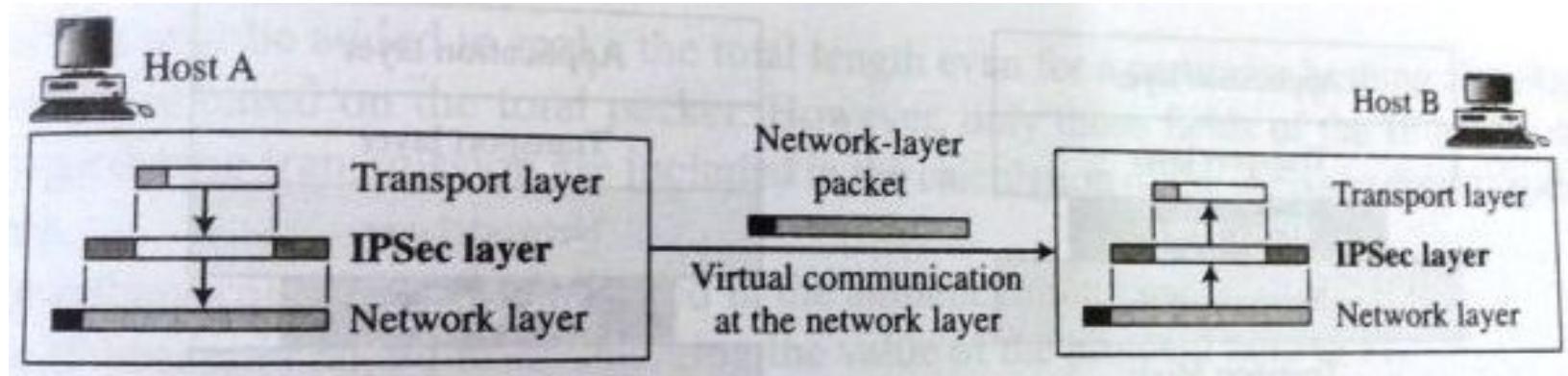
# *IPSec modes of operation*

1. Transport mode
2. Tunnel mode

## IPSec mode: Transport Mode

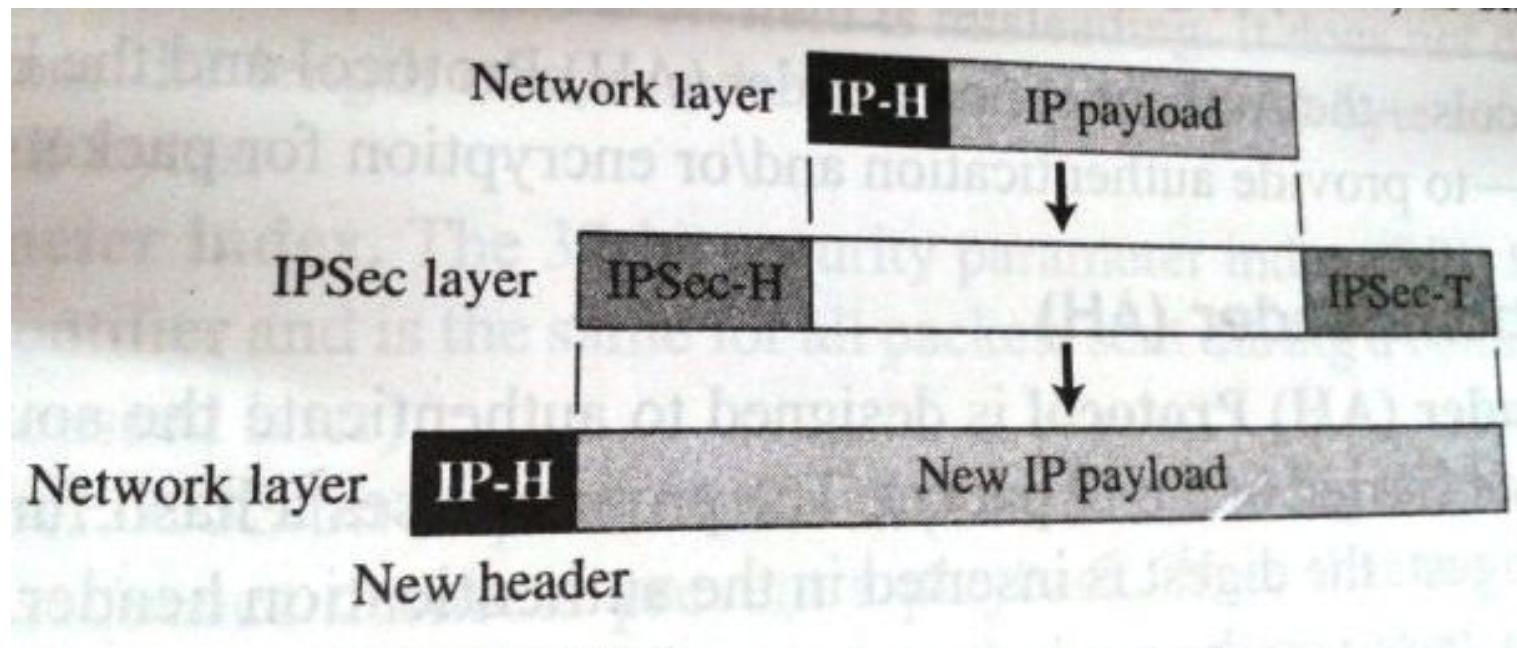


## *IPSec mode: Transport Mode*

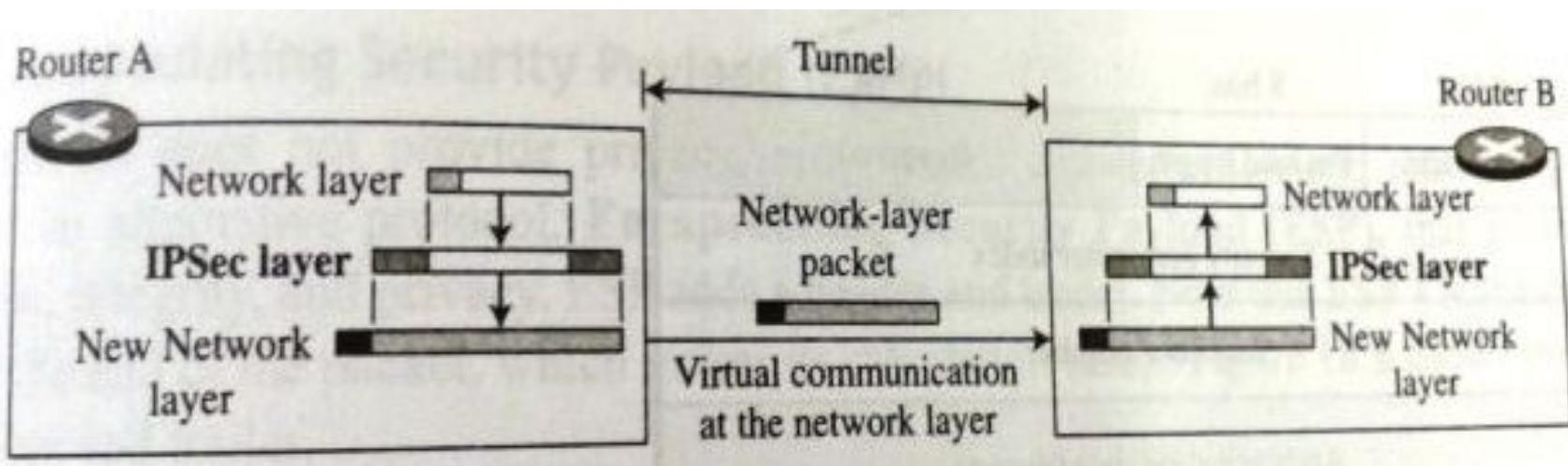


Transport mode is normally used when we need host- to –host (end to end ) protection of data

## IPSec mode: Tunnel Mode



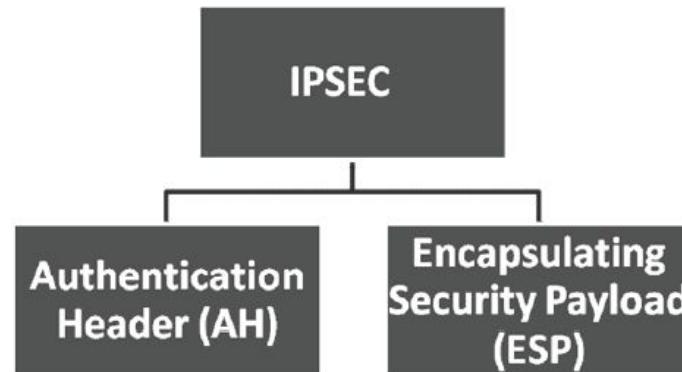
## IPSec mode: Tunnel Mode



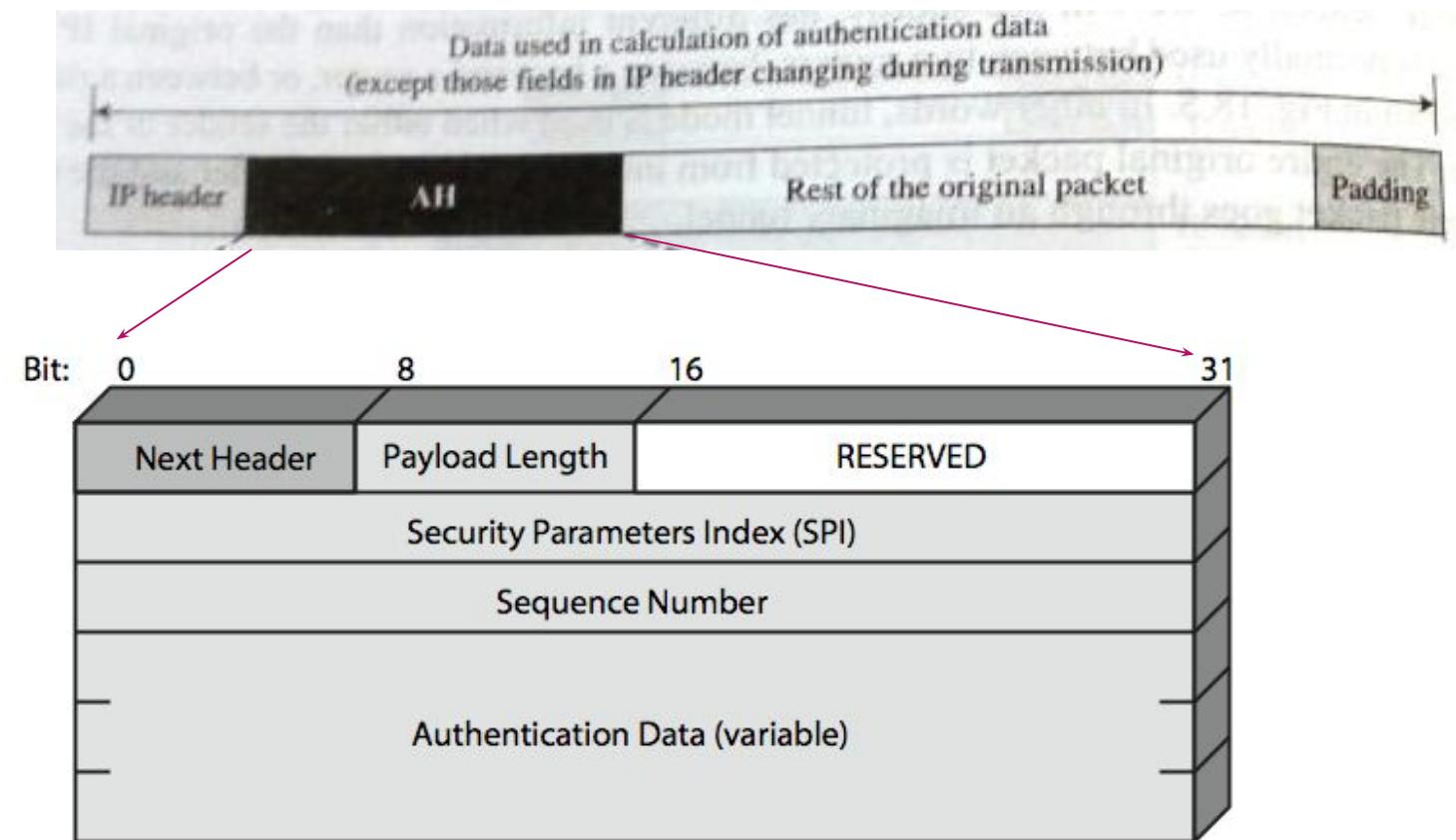
Tunnel mode is normally used between 2 routers, between a host and a router or between a router and a host.

# *IPSec Protocols*

- ▶ Two protocols are used to provide security:
- ▶ **Authentication Header (AH):** An authentication protocol provides support for data integrity & authentication of IP packets
- ▶ **Encapsulating Security Payload (ESP):** ESP provides support for data integrity & authentication and confidentiality of IP packets



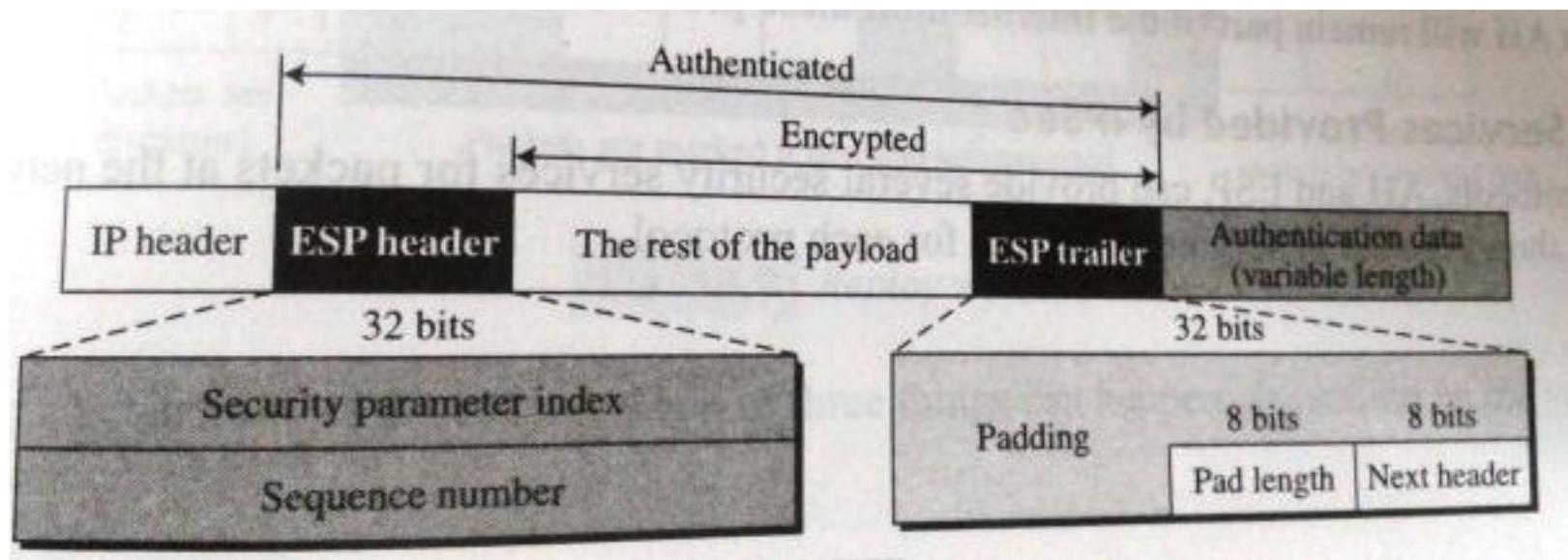
# Authentication Header



## *Authentication Header*

- ▶ **Next Header (8 bits):** Identifies the type of payload carried by the IP datagram (like TCP, UDP, ICMP etc.)
- ▶ **Payload Length (8 bits):** Length of Authentication Header in 32-bit words, minus 2.
- ▶ **Reserved (16 bits):** For future use
- ▶ **Security Parameters Index (32 bits):** Identifies a security association
- ▶ **Sequence Number (32 bits):** A monotonically increasing counter value which provides ordering information for a sequence of datagrams.
- ▶ **Authentication Data (variable):** A variable-length field (must be an integral number of 32-bit words) that contains MAC, for this packet

# Encapsulating Security Payload



1. An ESP trailer is added to the payload.
2. The payload and the trailer are encrypted.
3. The ESP header is added.
4. The ESP header, payload, and ESP trailer are used to create the authentication data.
5. The authentication data are added to the end of the ESP trailer.
6. The IP header is added after changing the protocol value to 50.

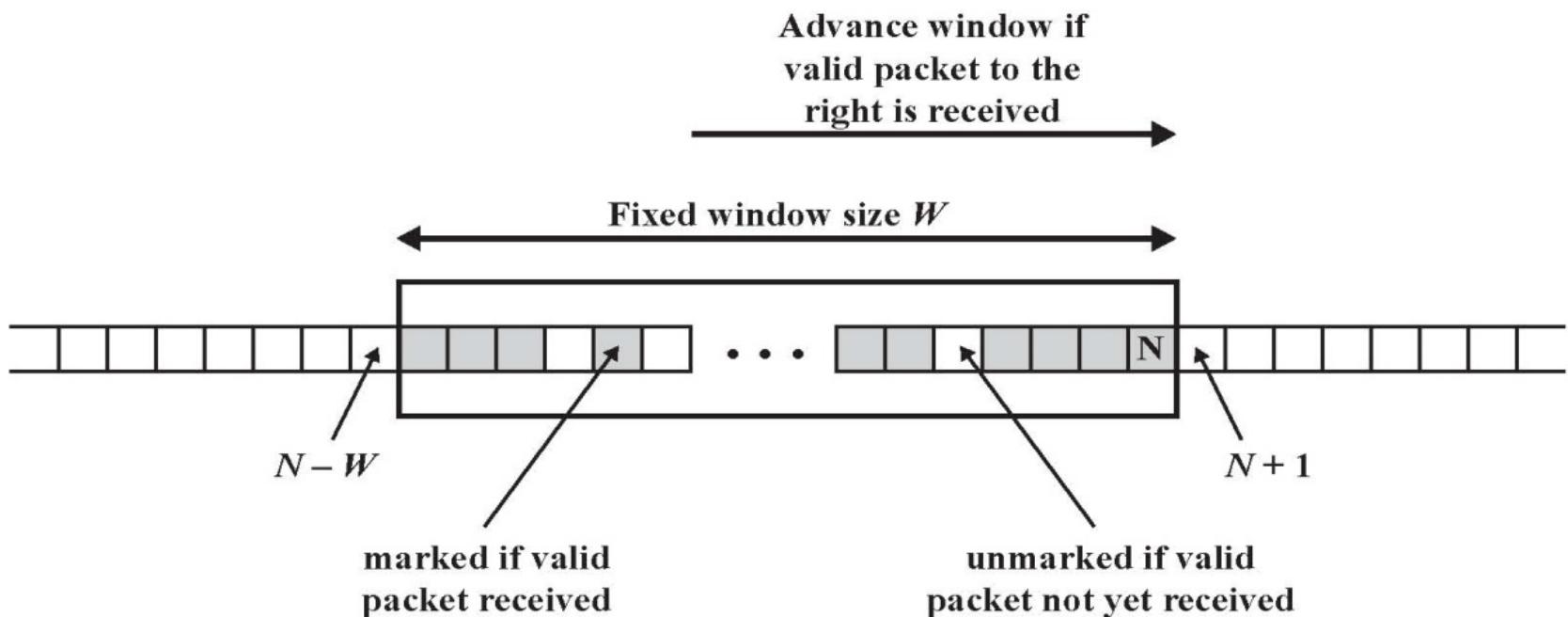
# *IPSec Services*

<i>Services</i>	<i>AH</i>	<i>ESP</i>
Access control	yes	yes
Message authentication (message integrity)	yes	yes
Entity authentication (data source authentication)	yes	yes
Confidentiality	<b>no</b>	yes
Replay attack protection	yes	yes

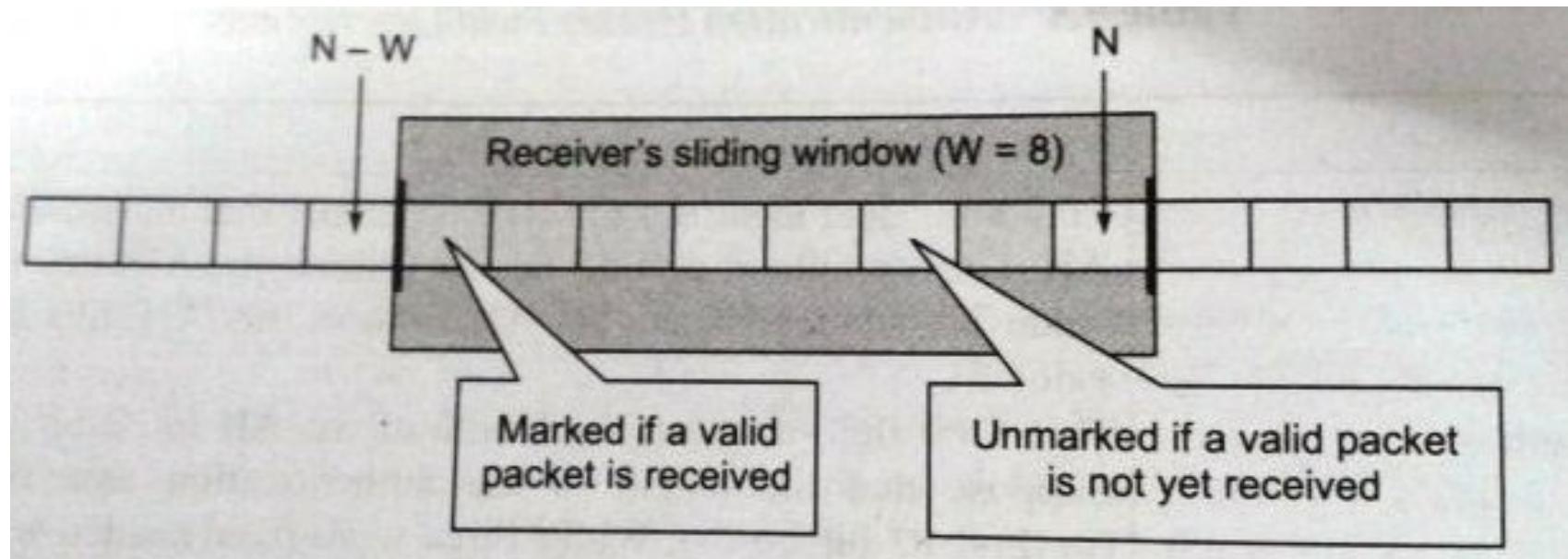
## *Replay Attack Protection*

- ▶ Replay is when attacker resends a copy of an authenticated packet
- ▶ Use sequence number to prevent this attack
- ▶ Sender initializes sequence number to 0 when a new SA is established
  - ▶ Increment for each packet
  - ▶ Must not exceed limit of  $2^{32} - 1$
- ▶ Receiver then accepts packets with seq no within window of  $(N-W+1)$  to N

# Replay Attack Protection



# Replay Attack Protection



# Web Services Security

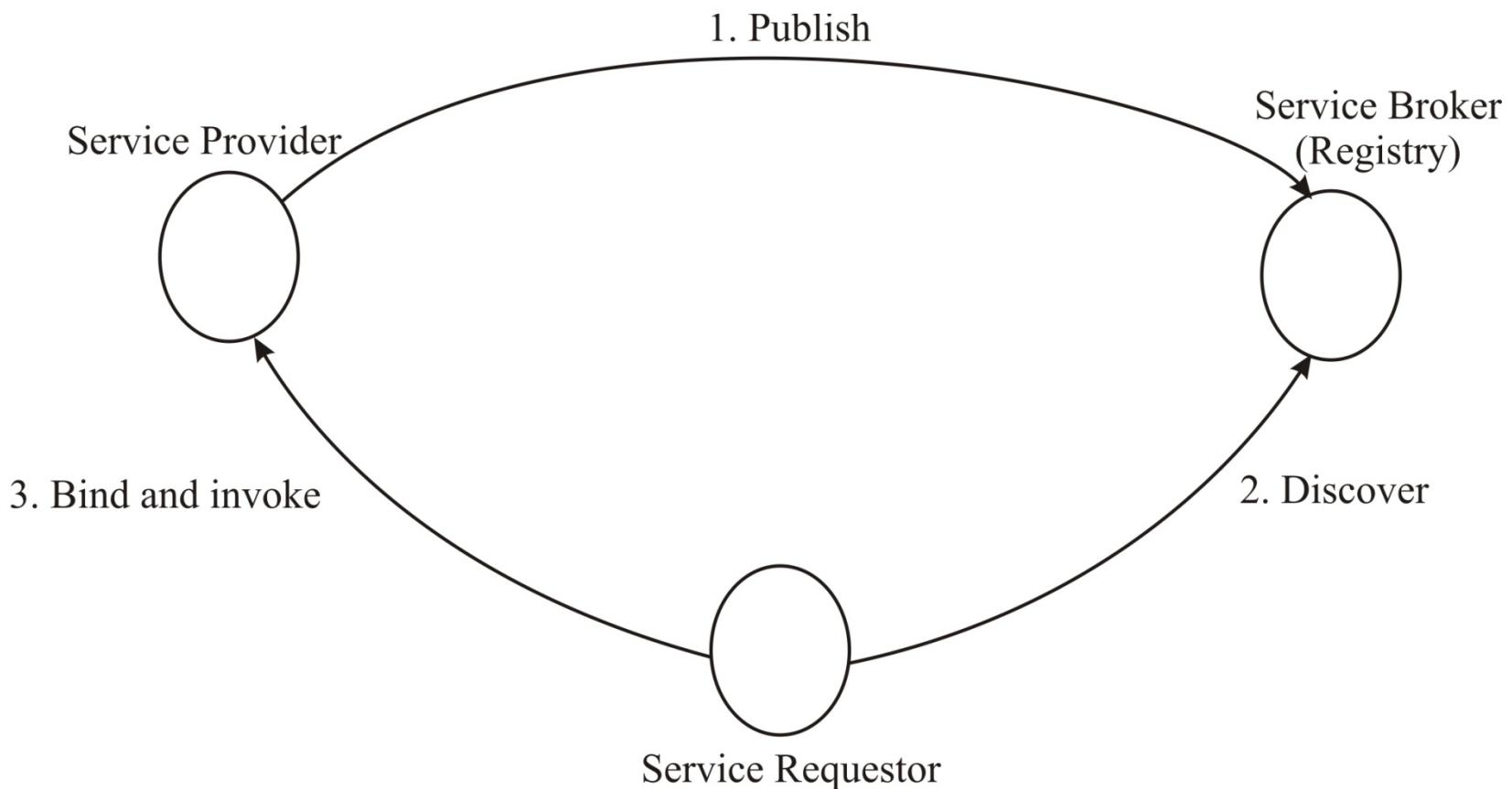
# Web apps versus web services

- Many of the earlier web applications (such as *internet banking*) involved human-to-program interaction. However, applications such as *supply chain management* differ from traditional web applications in several significant respects:
- Programs communicate with each other over the web with little or no human intervention.

# Web apps versus web services (contd.)

- Services might have a *composite nature*. Such “composite services” necessitate the involvement of multiple providers, each providing an “atomic service”.(web based travel planning)
- There are potentially a large number of “atomic service” providers offering a given service. So clients have a choice and can **dynamically change providers**.

# Entities involved in a web service

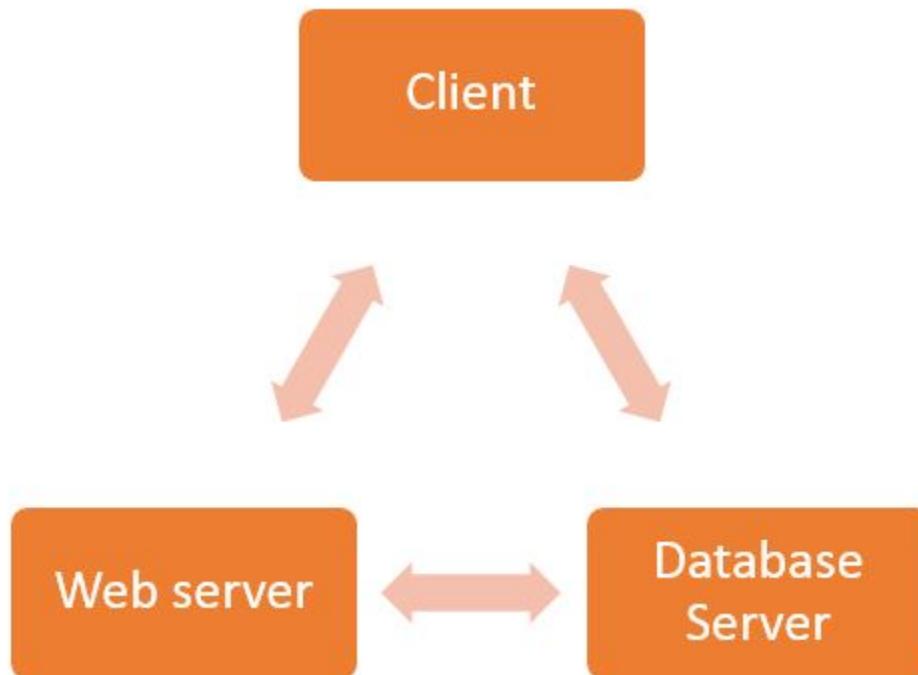


# WS Security

- WS Security is a standard that addresses security when data is exchanged as part of a Web service.
- HTTPS is the secure way of communication between the client and the server over the web.
- HTTPS makes use of the Secure Sockets layer or SSL for secure communication.
- Both the client and the server will have a digital certificate to identify themselves as genuine when any communication happens between the client and the server.

# WS security

- But the above type of security will not work in all situations.



# SOAP Header

- If the message within the SOAP body has been signed with any security key, that key can be defined in the header element.
- If any element within the SOAP Body is encrypted, the header would contain the necessary encryptions keys so that the message can be decrypted when it reaches the destination.

# Web Service Security Standards

- WS-Security standard revolves around having the security definition included in the SOAP Header. The credentials in the SOAP header is managed in 2 ways.
- First, it defines a special element called UsernameToken. This is used to pass the username and password to the web service.
- The other way is to use a Binary Token via the BinarySecurityToken. This is used in situations in which encryption techniques such as Kerberos or X.509 is used.

# WS-Security

- Security related information is contained within <security> element in a SOAP header. It specify which operation are performed and in what order
- The <security> element include security tokens,keys,signature,timestamp and security meta-information.

# WS-Sec Token Example

```
< UsernameToken >
    < Username > Shivani < /Username >
    < Password Type = "PasswordDigest" >
        4u%h&+q:L
    < /Password >
    < Nonce > . . . < /Nonce >
    < Created > . . . < Created >
< /UsernameToken >
```

# WS-Sec Binary Token

```
< BinarySecurityToken  
  ValueType = “ . . . X509v3”  
  EncodingType = “ . . . Base64Binary” >  
    Lp9tba4Pc7G . . .  
< / BinarySecurityToken >
```

# XML Encryption

- The XML encryption standard was developed by W3C in 2002.
- It defines XML elements for representing encrypted data and keys used for encryption.
- It allows encryption at different levels of granularity
  - Entire document
  - Complete XML element
  - Content of an XML element

# Example

- Smith's credit card number is sensitive information! If the application wishes to keep that information confidential, it can encrypt the CreditCard element:
- <?xml version='1.0'?>
- <PaymentInfo xmlns='http://example.org/paymentv2'>
- <Name>John Smith</Name>
- <EncryptedData Type='http://www.w3.org/2001/04/xmlenc#Element'  
•       xmlns='http://www.w3.org/2001/04/xmlenc#'>
- <CipherData>
- <CipherValue>A23B45C56</CipherValue>
- </CipherData>
- </EncryptedData>
- </PaymentInfo>

# XML signature

- XML Signature involves computing the hash of a document followed by encryption using signers private key.
- The first step in generating an XML signature is cononicalization . The part of a document that need to be signed are first transformed into a canonical form before computing their hashes
- Signature are included in SOAP header in the <signature> element .The major sub element are:
  - <SignedInfo>
  - <Signature Value>
  - <KeyInfo>

# SAML

- Security Assertion Markup Language (SAML) is an open standard that allows identity providers (IdP) to pass authorization credentials to service providers (SP)

# SAML

- Client C      Service Provider SP
- Each time C request a service from SP, he need to be authenticated
- SP can store a cookie in C's browser which would be dispatched to SP .
- The cookie may be stored in encrypted form
- C wishes service from another service provider
- If SP2 knows that SP1 trust C.

# Security Assertion Markup Language (SAML)

- Designed to support single sign-on and propagate authorization information
- Example :Principal C, Service provider SP1 (asserting party Identity Provider(I)),Service provider SP2 (relying party, Consumer of assertion). (Cookie-“SP1 trusts C”)cookies cannot be dispatched from one domain to another.
- SP1 might make the following assertion
  - SP1 authenticate C

Using password based authentication  
On 1<sup>st</sup> feb 2017  
At 9:25:15 hours.

# What is a SAML Assertion

- A SAML Assertion is the XML document that the identity provider sends to the service provider that contains the user authorization. There are three different types of SAML Assertions – authentication, attribute, and authorization decision

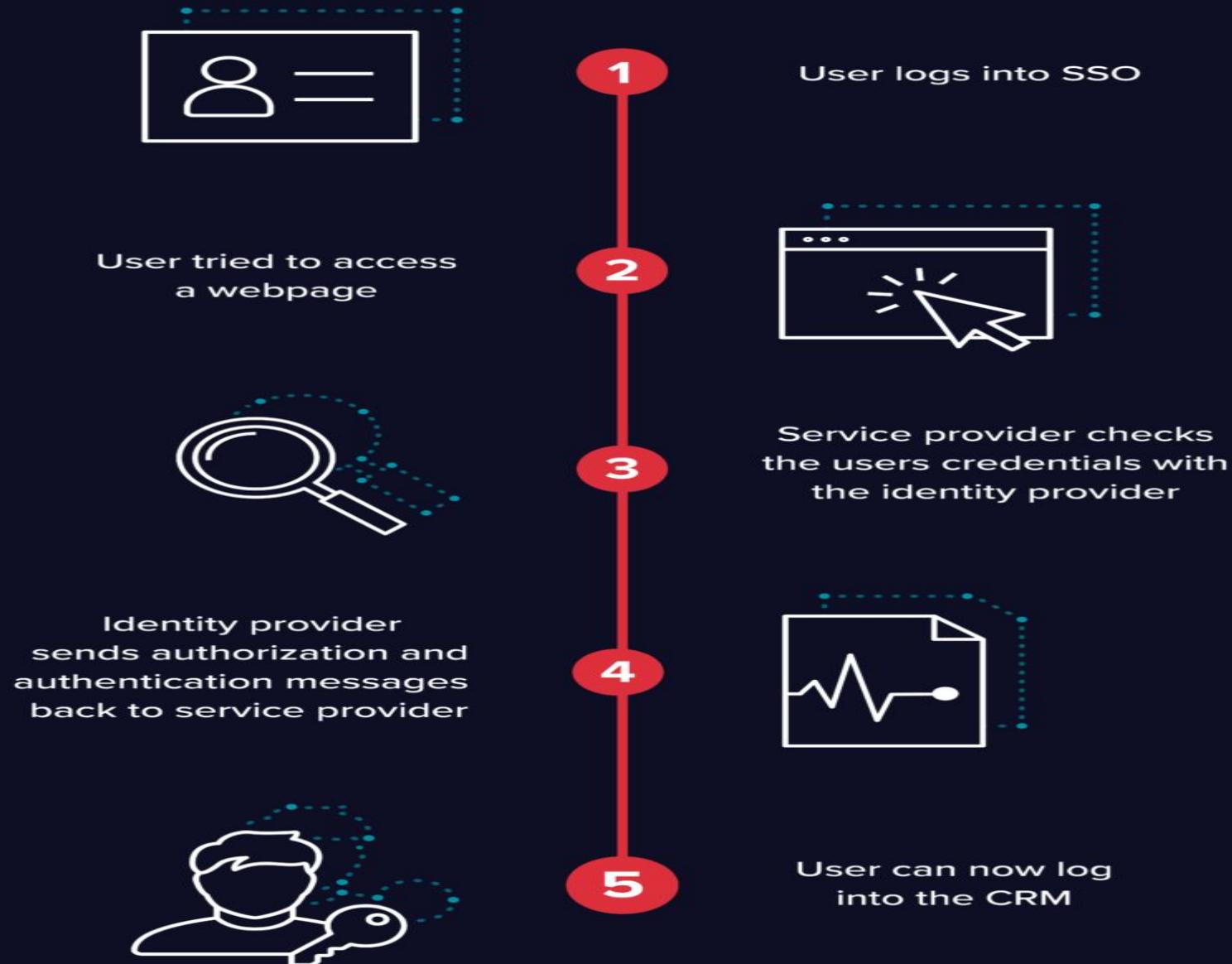
# SAML

- Authentication assertions prove identification of the user and provide the time the user logged in and what method of authentication they used (I.e., Kerberos, 2 factor, etc.)
- The attribution assertion passes the SAML attributes to the service provider – SAML attributes are specific pieces of data that provide information about the user.
- An authorization decision assertion says if the user is authorized to use the service or if the identify provider denied their request due to a password failure or lack of rights to the service.

# SAML working steps

- Each user logs in once to Single Sign On with the identify provider, and then the identify provider can pass SAML attributes to the service provider when the user attempts to access those services.

# SAML Example Steps



# SAML Authenticating Assertion

```
<saml:Assertion . . .
  <saml:AuthenticationStatement
    AuthMethod="password"
    AuthInstant="2008- . .
  <saml:Subject>
    <saml:NameID
      SecurityDomain="iitb.ac.in"
      Name="Rajesh" />
```

...

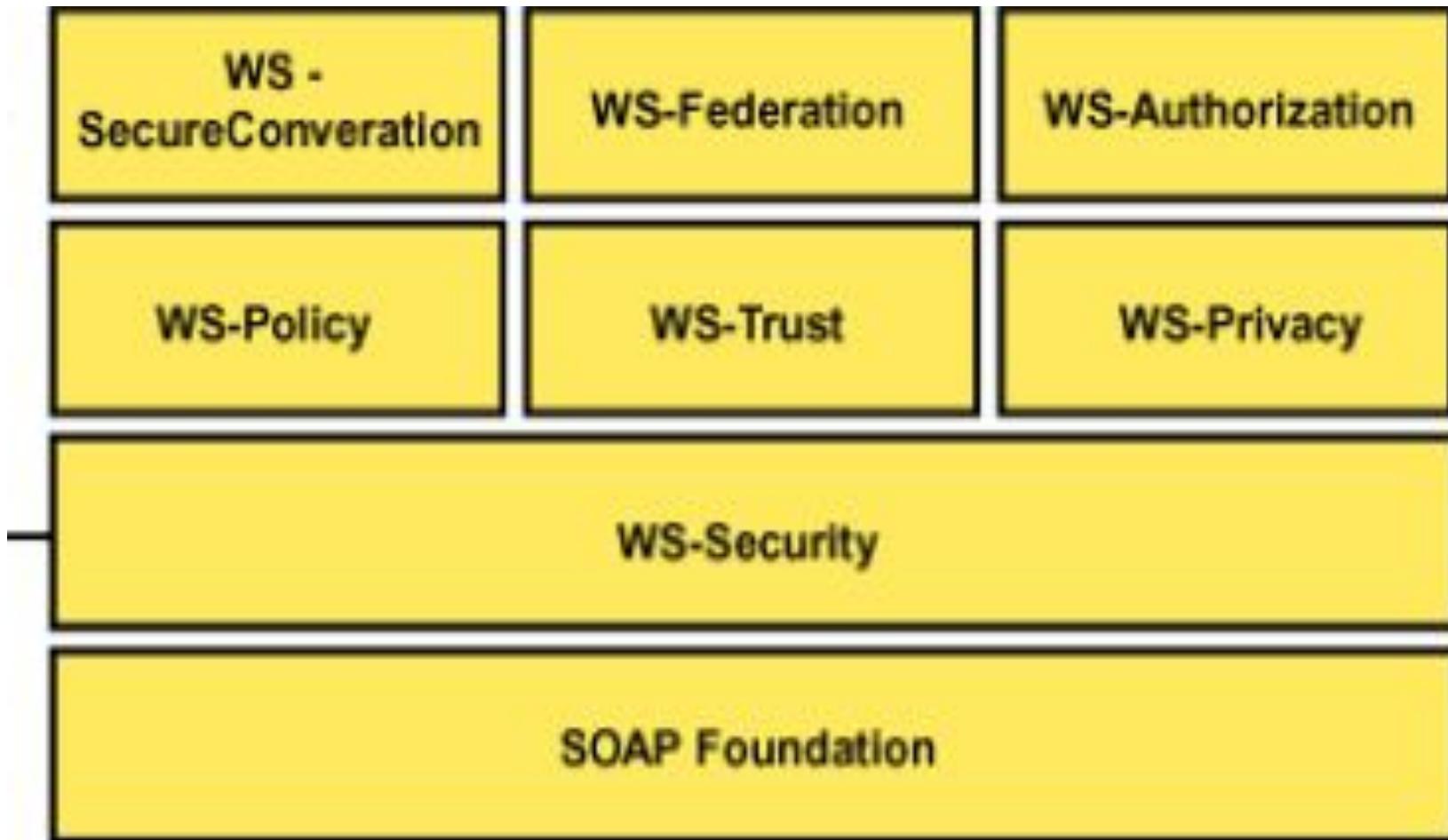
...

Browser attacks are very common and are likely to succeed against systems that have not been hardened against them specifically.

Some of the more commonly used browsers, such as Microsoft's Internet Explorer and Mozilla Firefox, now include at least a rudimentary form of protection against such attacks.

Browser security plug-ins, such as NoScript<sup>1</sup> for Firefox and GuardedID<sup>2</sup> for Internet Explorer, can also help to foil such attacks.

# WS-Security Specifications



# Unit 5

Firewall and IDS

# Contents

- **Firewall:** Introduction, Characteristic ,Types :Packet Filter, Stateful and Stateless Packet Filter, Attacks of Packet Filter, Circuit Level and Application Level Firewall, Bastion Host, Firewall Configurations.
- **Intrusion:** What is Intrusion, Intruders, Intrusion Detection, Behavior of Authorized user and Intruder, Approaches for Intrusion Detection: Statistical Anomaly Detection and Rule based Detection. Audit Record and Audit Record Analysis.

# *Introduction*

- A firewall is basically the **first line of defence** for your network.
- Generally is placed at the perimeter of the network to act as the **gatekeeper** for all incoming and outgoing traffic.
- A firewall can be a hardware device or a software application

# *Introduction*

A firewall allows you to establish certain rules to determine what traffic should be allowed in or out of your private network.

Depending on the type of firewall implemented you could restrict access to only certain IP addresses ,domain names, or you can block certain types of traffic by blocking the TCP/IP ports they use.

## *Firewall Characteristics*

- .All traffic (inside to outside and vise versa) must pass through the firewall
- .Only authorized traffic , as defined by security policies will be allowed.
- .Firewall itself must be strong enough, to avoid attacks by using trusted system with secure operating system.

# *Firewall Control Mechanisms*

*.Service control:* which

- .types of accessible Internet services,
- .filters on basis of IP address and TCP port number

*.Direction control:* where

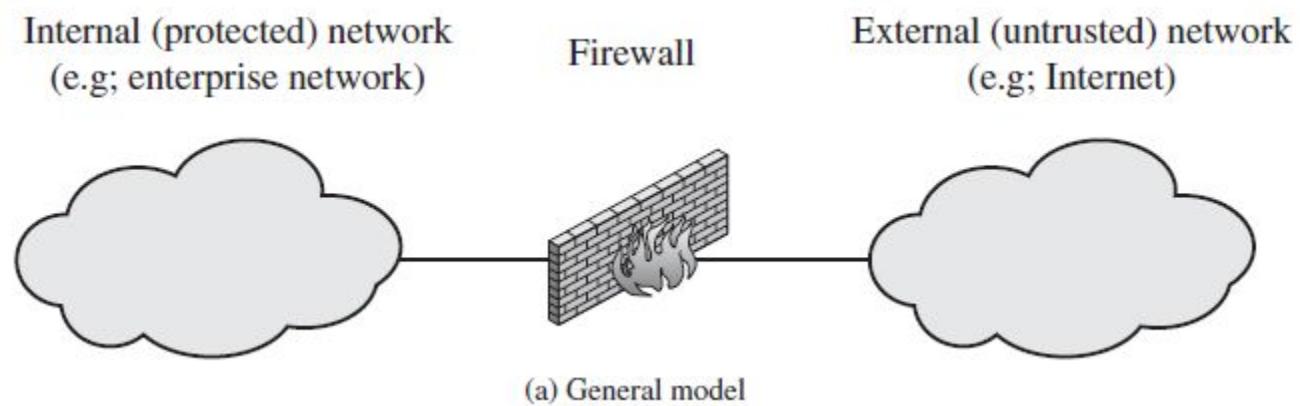
- .determines direction in which service requests allowed to flow

*.User control:* who

- .controls access to service depending on who requests it
- .typically applied to local users (with IPSec also to externals)

*.Behavior control:* what

- .controls how particular services are used (e.g. eliminate spam)



# *Types of Firewalls*

.There are three common types of firewalls:

**Packet-filtering routers:**

Filters out packets that pass through it

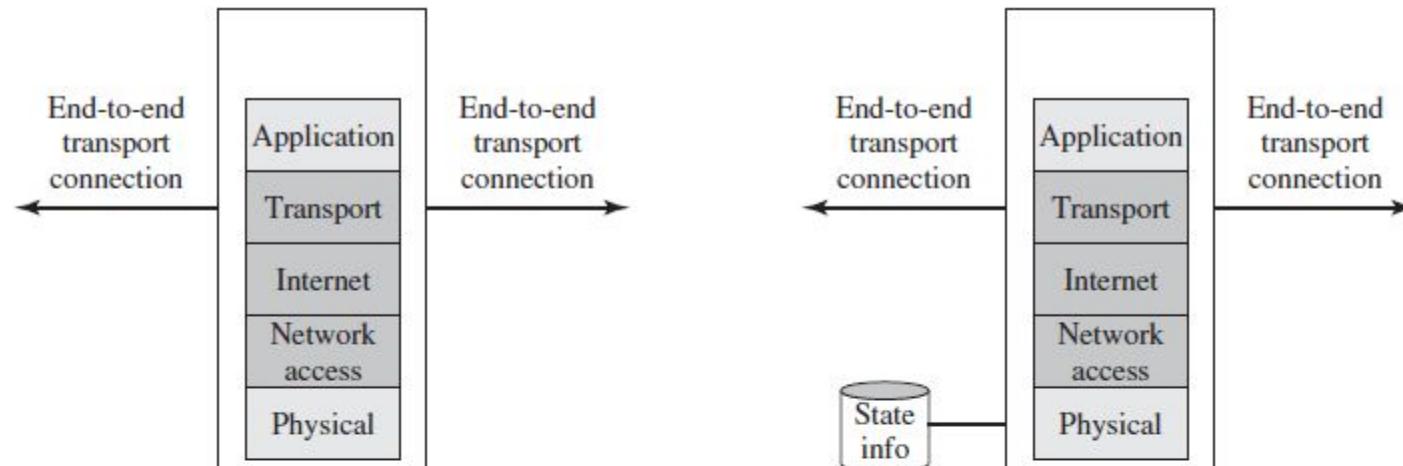
**Stateful Inspection Firewalls**

**Application-level gateway:**

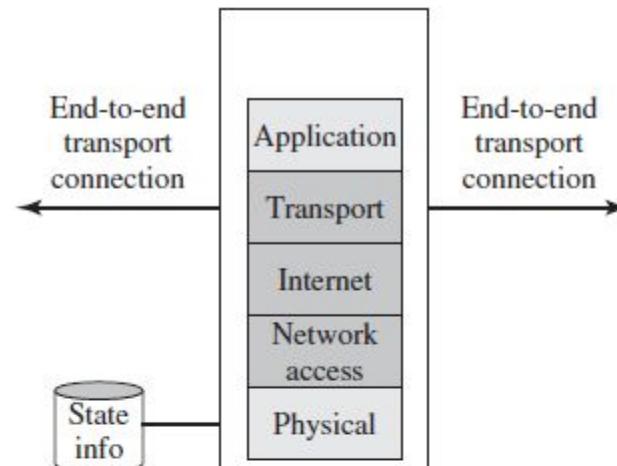
Proxy for remote services

**Circuit-level gateway:**

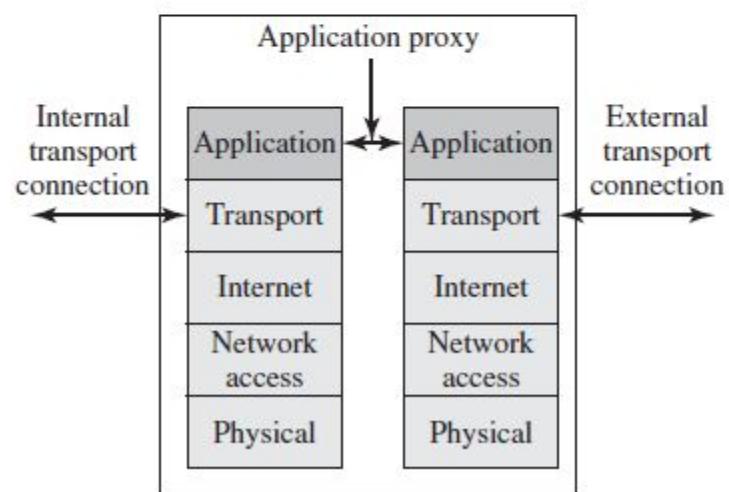
Allows only certain TCP connections



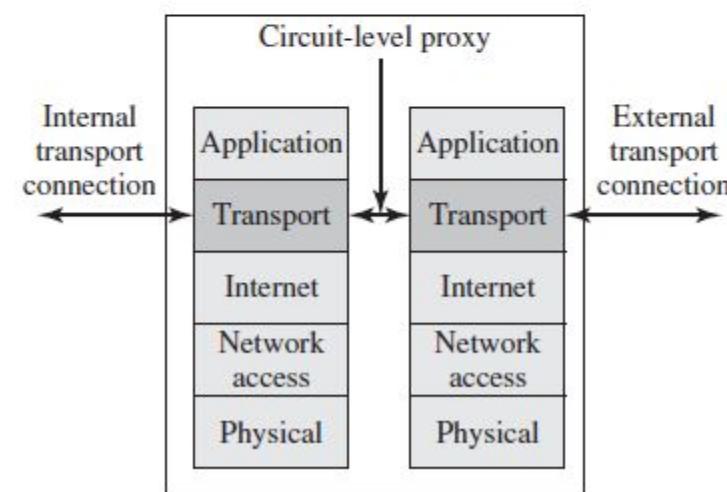
(b) Packet filtering firewall



(c) Stateful inspection firewall

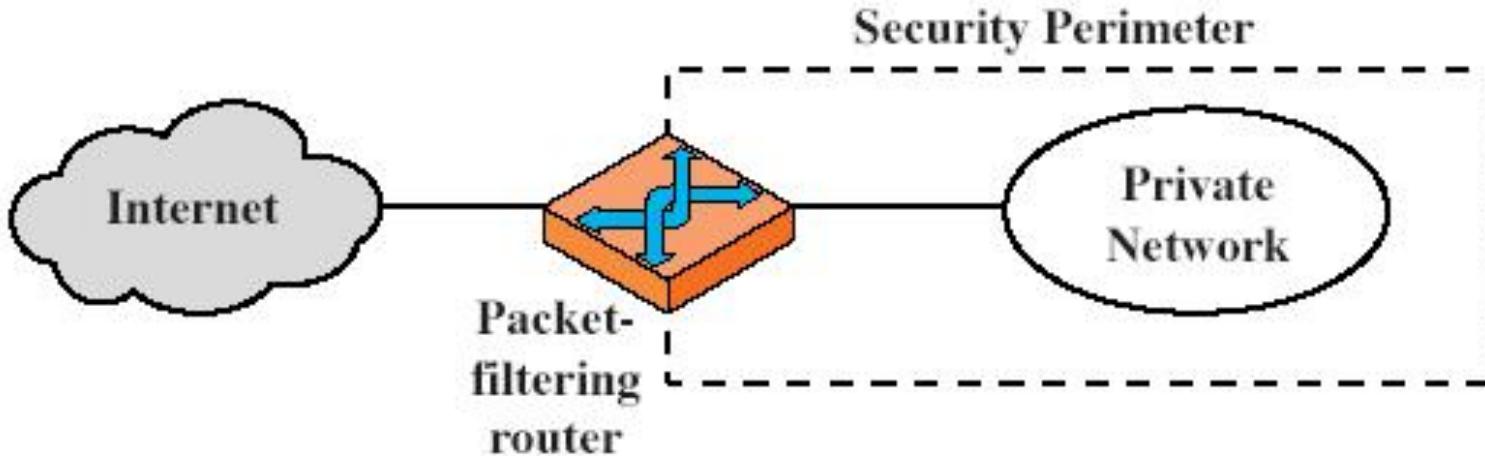


(d) Application proxy firewall



(e) Circuit-level proxy firewall

# *Packet Filters*



- .Apply set of rule to packet.
- .Examines each IP packet and decides to forward or to discard the packet.
- .Packet filters also called screening router or screening filter.

# **Packet Filters**

Filtering rules are based on information in the packet.

- **Source IP address**
- **Destination IP address**
- **IP protocol field** : Defines transport protocol (TCP or UDP).
- **Source and Destination transport level address:** TCP or UDP port number, which identify application which is using this packet such as email, file transfer or WWW.
- **Interface:** For a firewall with three or more ports, which interface of the firewall the packet came from or which interface of the firewall the packet is destined for

# **Packet Filters : steps**

- Receive each packet as it arrives.
- Apply the set of rules, based on contents of IP and Transport headers. If there is match with one of the rules decide whether to accept or discard.(e.g rules: disallow traffic from IP address 157.29.19.10 or disallow all traffics that uses UDP as transport layer protocol)
- If there is no match with any rule, take default action (default = discard, default = forward). Default = discard policy is more conservative. Initially everything is blocked, and services are added on case-by-case basis. Default=forward policy increases ease of use but reduce security. Security administrator has to react with each new security threat.

# *Packet Filters – Example*

- based on matches to fields in IP or TCP header
- mail from SPIGOT blocked, because bad history of emails.
- inbound mail allowed (port 25 for SMTP incoming), only to gateway host

<b>action</b>	<b>ourhost</b>	<b>port</b>	<b>theirhost</b>	<b>port</b>	<b>Comment</b>
Block	*	*	SPIGOT	*	We don't trust these people
Allow	OUR-GW	25	*	*	Connection to our SMTP port

<b>action</b>	<b>ourhost</b>	<b>port</b>	<b>theirhost</b>	<b>port</b>	<b>comment</b>
block	*	*	*	*	default

#### **Rule Set C**

<b>action</b>	<b>ourhost</b>	<b>port</b>	<b>theirhost</b>	<b>port</b>	<b>comment</b>
allow	*	*	*	25	connection to their SMTP port

#### **Rule Set D**

<b>action</b>	<b>src</b>	<b>port</b>	<b>dest</b>	<b>port</b>	<b>flags</b>	<b>comment</b>
allow	{our host}	*	*	25		our packets to their SMTP port
allow	*	25	*	*	ACK	their replies

#### **Rule Set E**

<b>action</b>	<b>src</b>	<b>port</b>	<b>dest</b>	<b>port</b>	<b>flags</b>	<b>comment</b>
allow	{our hosts}	*	*	*		our outgoing calls
allow	*	*	*	*	ACK	replies to our calls
allow	*	*	*	>1024		traffic to nonservers

# *Attacks on Packet Filters*

- IP address spoofing
- Source routing attacks
- Tiny fragment attacks

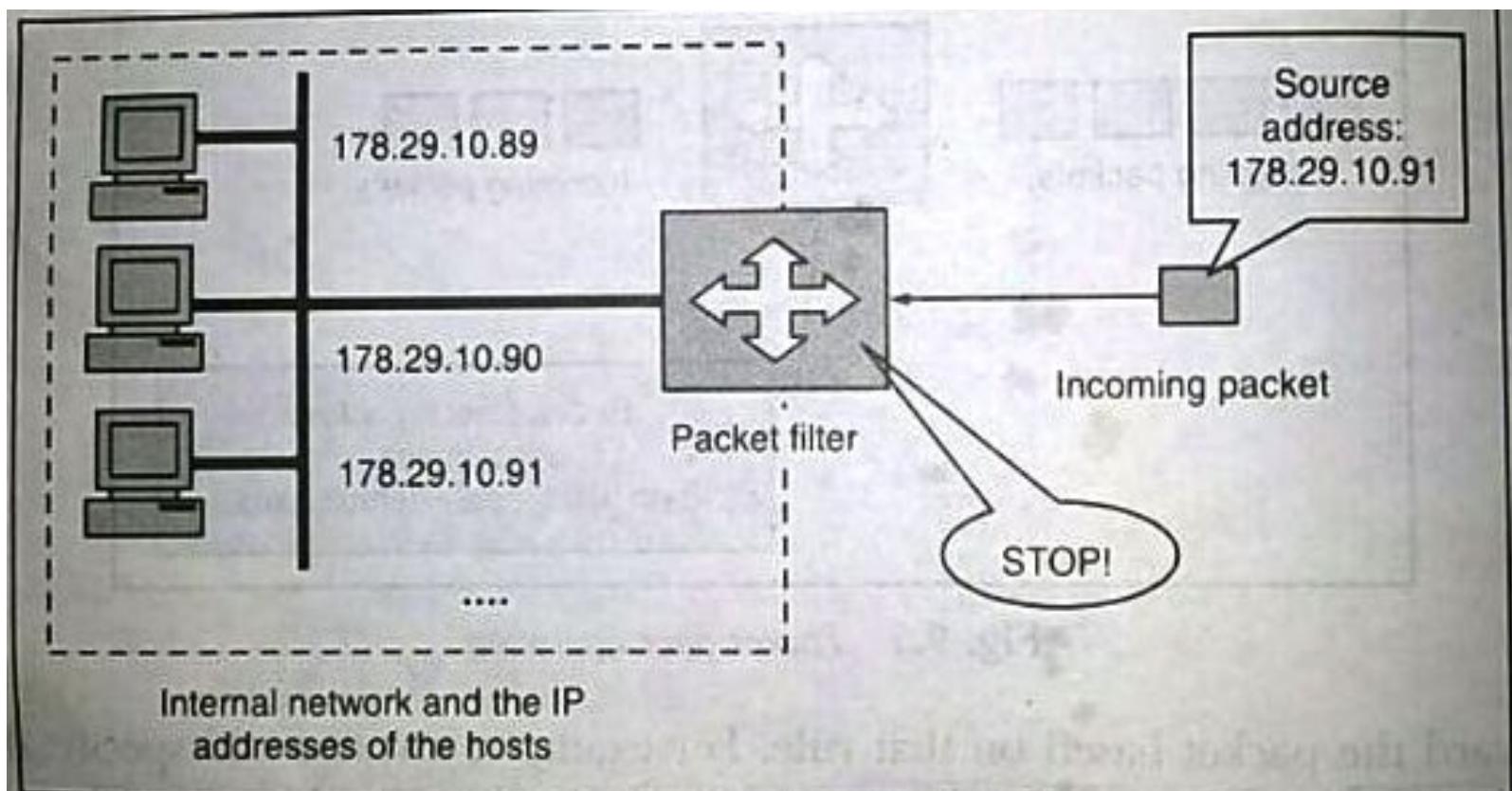
# *Attacks on Packet Filters*

## **IP address spoofing**

- An intruder can attempt to send a packet towards internal network , with source IP address set to equal IP address of internal users.
- This attack is defeated by discarding packets with internal source address arriving on external surface
- This countermeasure is often implemented at the router external to the firewall.

# *Attacks on Packet Filters*

## IP address spoofing



# *Attacks on Packet Filters*

## **Source routing attacks**

- Attacker specifies a route that packet should follow to move along network with the hope that this will bypass security measures that do not analyze the source routing information
- Counter Measure is to Block all source routed packets that use this option.

# *Attacks on Packet Filters*

## **Tiny fragment attacks**

- IP packets pass through various physical networks (e.g. Ethernet, X.25 etc.)
- Each network has predefined Maximum Transfer Unit (MTU).
- IP packet larger than MTU needs to be fragmented.
- Attacker attempt to use this characteristic and intentionally fragments IP packet and sends it.
- Attacker feels that after fragmentation packet filter will check first fragment and will not check further fragments.
- This attack can be defeated by enforcing a rule that first fragment of a packet must contain a predefined minimum amount of the transport header.
- If the first fragment is rejected, the filter can remember the packet and discard all subsequent fragments.

## PACKET-FILTERING FIREWALLS

Advantages	Disadvantages	Protection Level	Who is it for:
<ul style="list-style-type: none"><li>- Fast and efficient for filtering headers.</li><li>- Don't use up a lot of resources.</li><li>- Low cost.</li></ul>	<ul style="list-style-type: none"><li>- No payload check.</li><li>- Vulnerable to IP spoofing.</li><li>- Cannot filter application layer protocols.</li><li>- No user authentication.</li></ul>	<ul style="list-style-type: none"><li>- Not very secure as they don't check the packet payload.</li></ul>	<ul style="list-style-type: none"><li>- A cost-efficient solution to protect devices within an internal network.</li><li>- A means of isolating traffic internally between different departments.</li></ul>

## *Stateful Packet Filters*

- Stateful filter allows examination based on current state of the network.
- It adopts the current exchange information.
- whereas normal packet filter has routing rules hardcoded.
- Stateful packet filter has to maintain a list of currently open connections and outgoing packets to deal with rules.

# *Stateful Packet Filters*

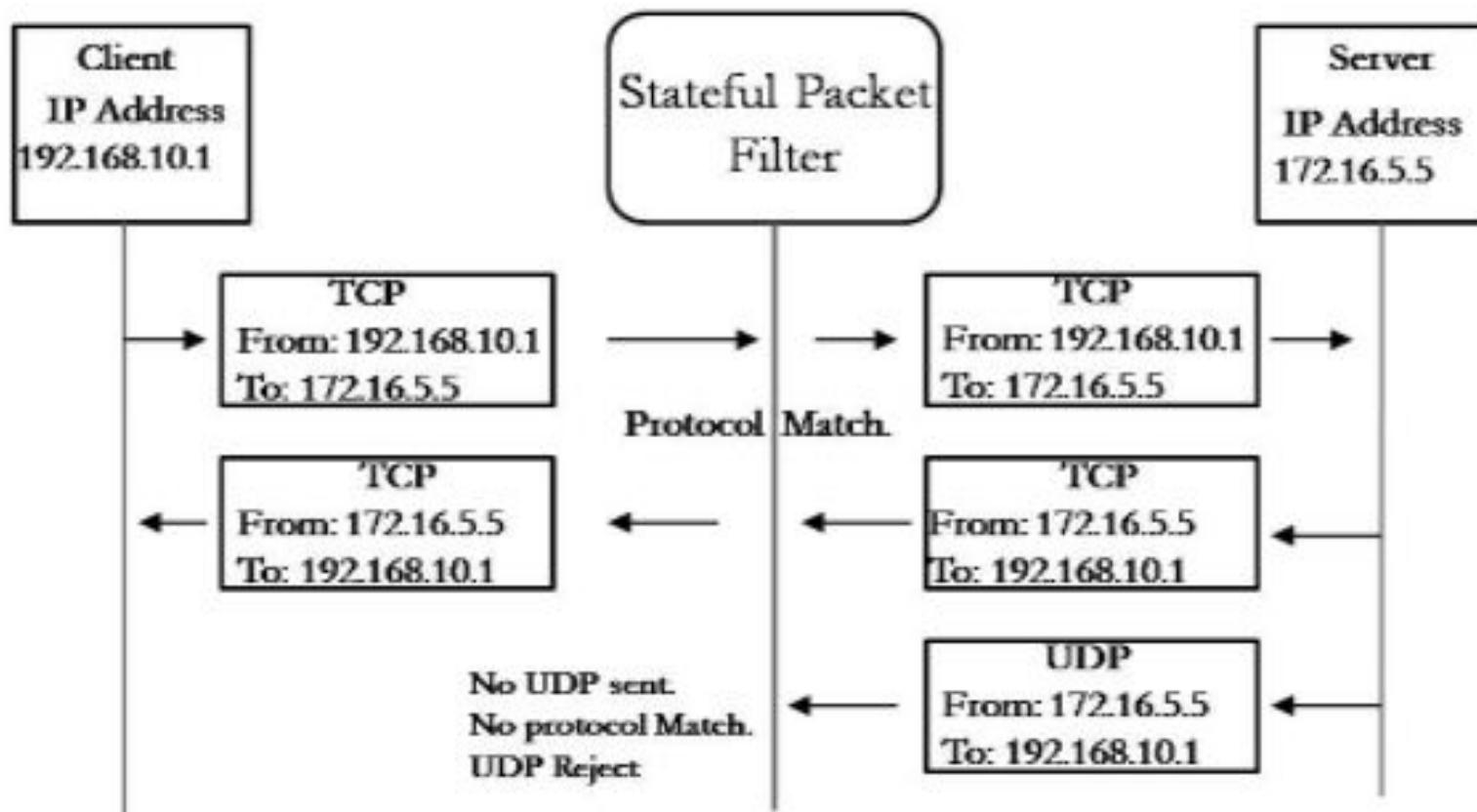
**For example:**

- Internal client sends TCP packet to external server.
- Stateful maintains this information and sends this packet.
- In response server sends another TCP packet.
- Packet filter examines and realizes that it is a response to the clients request.
- Next time if server sends UDP packet, packet filter rejects it, because previous exchange was using TCP.
- It is against the rule previously established hence filter discards the packet.

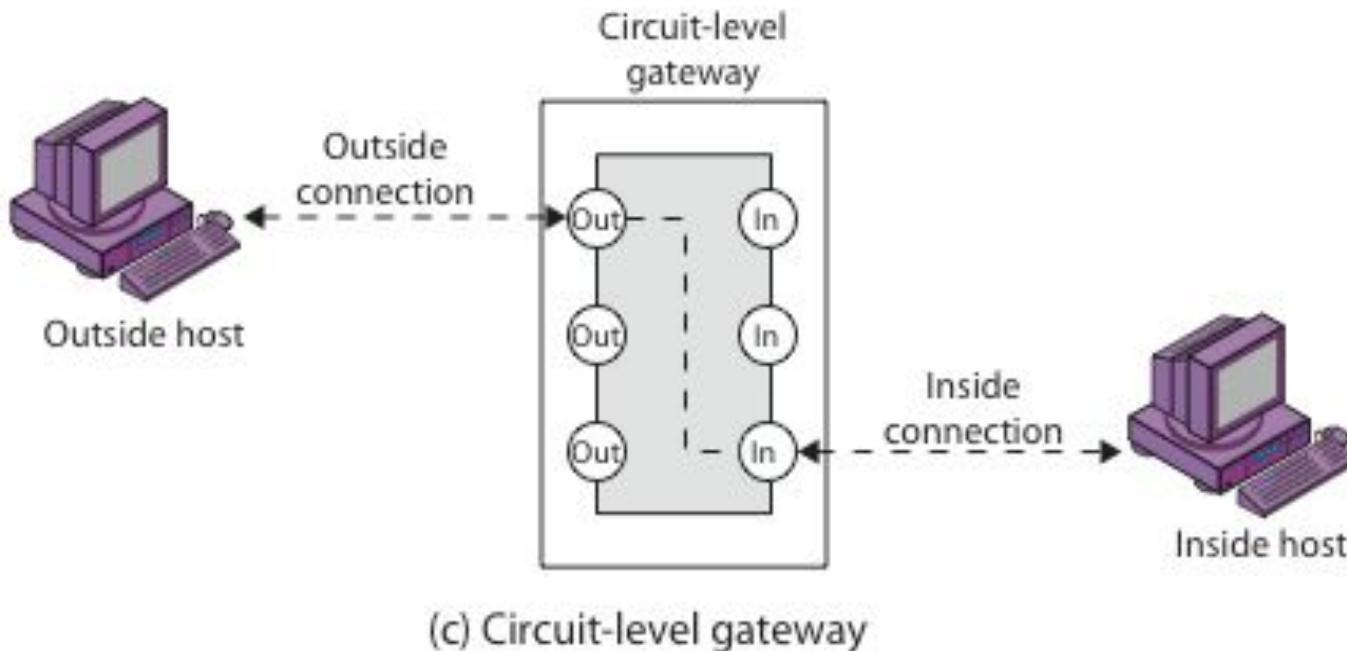
# *Stateful Packet Filters*

Source Address	Source Port	Destination Address	Destination Port	Connection State
192.168.1.100	1030	210.9.88.29	80	Established
192.168.1.102	1031	216.32.42.123	80	Established
192.168.1.101	1033	173.66.32.122	25	Established
192.168.1.106	1035	177.231.32.12	79	Established
223.43.21.231	1990	192.168.1.6	80	Established
219.22.123.32	2112	192.168.1.6	80	Established
210.99.212.18	3321	192.168.1.6	80	Established
24.102.32.23	1025	192.168.1.6	80	Established
223.212.212	1046	192.168.1.6	80	Established

# *Stateful Packet Filters*



# *Circuit Level Gateway*



- Circuit level firewall operate at the transport and session layer of the OSI
- It validate TCP and UDP Session before opening a connection through firewall
- Firewall maintain a table of valid connection and let data pass through when session info matches an entry in table
- When session terminate the table entry is removed

# *Circuit Level Gateway*

- To validate the session and ensure that it follows a legitimate handshake the firewall store a virtual circuit table which stores
  - A unique session id
  - Handshake info
  - Sequence info
  - Source and destination IP
  - MAC address of source and destination
- Once a connection is allowed all packet associated with the connection are allowed without further security checks

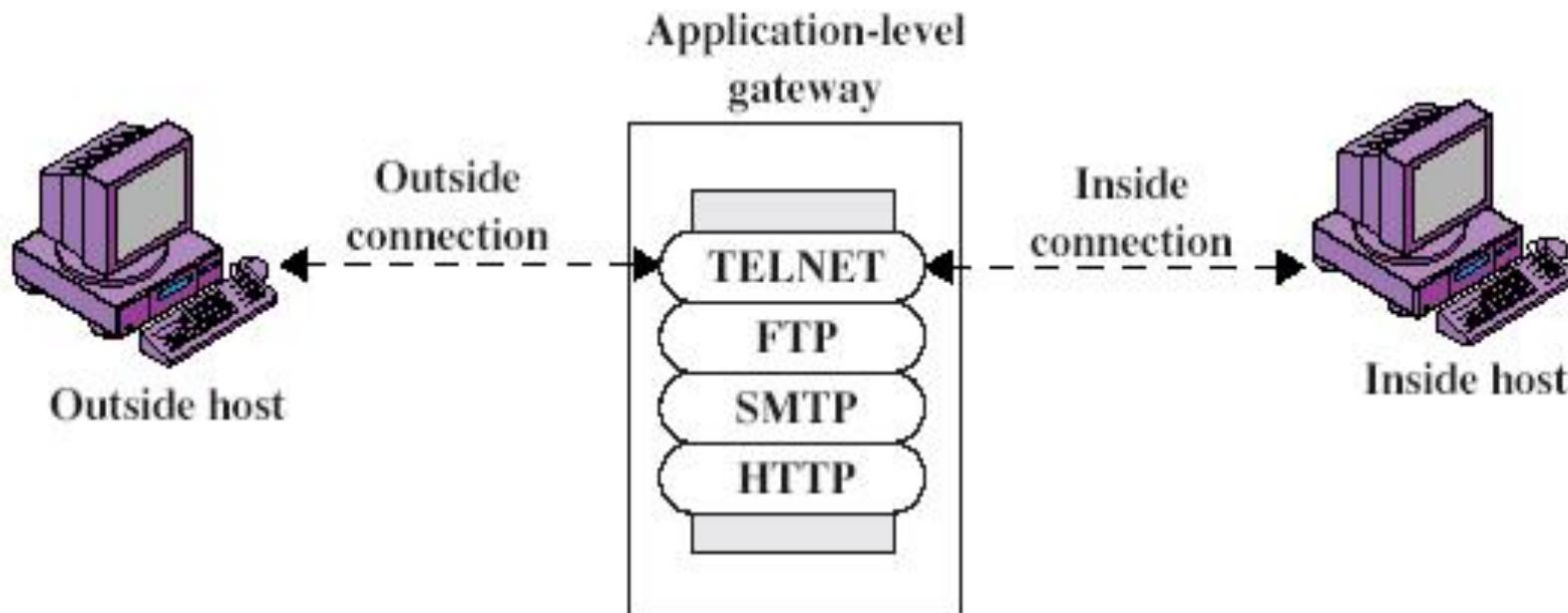
# *Circuit Level Gateway*

- Circuit gateway creates new connection between itself and remote host.
- User is not aware of this.
- Circuit gateway changes source IP address of internal host and replace with own IP address.
- Source IP addresses are hidden from outside world.

## CIRCUIT-LEVEL GATEWAYS

Advantages	Disadvantages	Protection Level	Who is it for:
<ul style="list-style-type: none"><li>- Resource and cost-efficient.</li><li>- Provide data hiding and protect against address exposure.</li><li>- Check TCP handshakes.</li></ul>	<ul style="list-style-type: none"><li>- No content filtering.</li><li>- No application layer security.</li><li>- Require software modifications.</li></ul>	<ul style="list-style-type: none"><li>- Moderate protection level (higher than packet filtering, but not completely efficient since there is no content filtering).</li></ul>	<ul style="list-style-type: none"><li>- They should not be used as a stand-alone solution.</li><li>- They are often used with application-layer gateways.</li></ul>

# *Application Level Gateway (or Proxy)*



# *Application Level Gateway (or Proxy)*

- Third generation firewall that evaluate network packets at application layer before allowing connection
- Uses special proxy service to manage data Transfer such as ftp/http.
- Proxy service do not allow direct connection between real service and user.
- It inspect ever communication between them
- Proxy service as two Component
- Proxy server
- Proxy client

# *Application Level Gateway*

- When a real client want to communicate to an external service (telnet)the request is directed to proxy Server(because the users default gateway is set to proxy server) which deny or allow depending on set of rules
- They also perform auditing, user authentication and caching service which were not performed by packet Filter Or circuit level firewall

# *Application Level Gateway (or Proxy)*

Relay of application-level traffic:

- User contacts gateway using TCP/IP application such as TELNET, FTP.
- Gateway asks name of remote host(IP address, domain name etc.)
- Gateway also asks for UserId and Authentication information.
- User provides this information.
- Gateway contacts with remote host and transmits packet to the remote host.
- If the gateway does not implement proxy code for a specific application, service is not supported and can not be forwarded across firewall.

# *Application Level Gateway (or Proxy)*

Advantage:

- Just concerned with allowable applications
- Easy to log and audit incoming traffic

Disadvantage:

- Additional processing overhead on each connection
- Slow and lead to performance degradation.
- Not scalable as each new network service adds onto the number of proxy services required.
- They require replacing of native network stack on the firewall server.

# Bastion Host

**Bastion host is a kind of reception area in office premise. Every outsider first need to register themselves at reception and then only entry inside office area is permitted.**

**On the Internet, a bastion host is the only host computer that a company allows to be addressed directly from the public network and that is designed to protect the rest of its network from exposure.**

**Bastion host are heavily fortified against attack.**

**Common characteristics of a bastion host are as follows:**

**-Its Operating system is hardened, in the sense that only essential services are installed on it.**

**-System should have all the unnecessary services disabled, unneeded ports closed, unused applications removed, unnecessary administrative tools removed i.e vulnerabilities to be removed to the extent possible**

# ***Bastion Host (Application Gateway)***

- A bastion host is a system identified by the firewall administrator as a critical strong point in the network's security.
- Typical platform for application level (proxy) or circuit-level gateway

## **Characteristics of Bastion Host:**

- Executes secure version of its OS
- Only essential services are installed: Telnet, FTP, SMTP,...
- May require additional authentication to access proxy services.
- Each proxy maintains detailed audit information by logging all traffic, each connection, and the duration of each connection. The audit log is an essential tool for discovering and terminating intruder attacks.

# ***Bastion Host***

- Each proxy is independent of other proxies on the bastion host. If there is a problem with the operation of any proxy, or if a future vulnerability is discovered, it can be uninstalled without affecting the operation of the other proxy applications.
- A proxy generally performs no disk access other than to read its initial configuration file. Hence, the portions of executable code can be made read only. This makes it difficult for an intruder to install Trojan horse or other dangerous files on the bastion host.

# *Firewall Configurations*

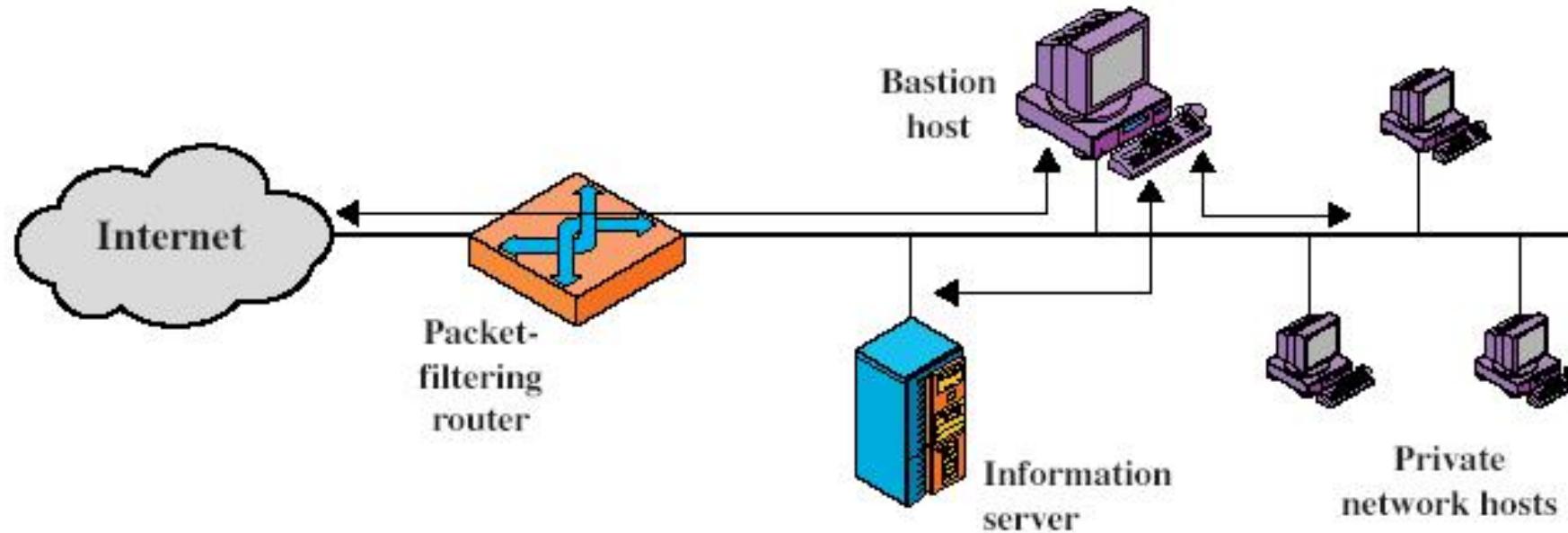
- Firewall is usually combination of packet filter and application (or circuit) gateways.

Based on this three possible configurations of firewall.

- Screened Host firewall system (Single-Homed Bastion Host)
- Screened Host firewall system (Dual-homed Bastion Host)
- Screened- Subnet Firewall

# *Single-Homed Bastion Host*

- Consists of Packet-filtering router plus bastion host(application gateway)
- Ensures that Only packets destined for bastion host allowed in
- Ensures that Only packets from bastion host allowed out

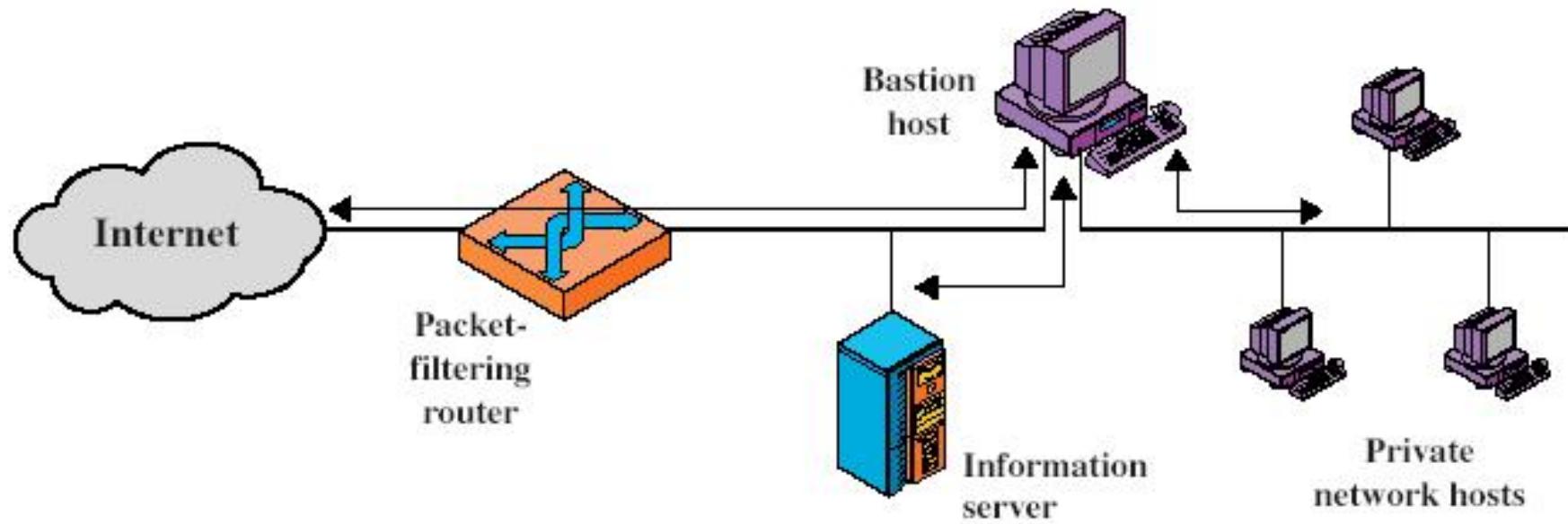


# *Single-Homed Bastion Host*

- Increases security by performing checks at both packet and application level.
- Disadvantage:
  - Internal users are connected to bastion host as well as packet filter.
  - If packet filter is somehow successfully attacked, then whole network is exposed to attacker.

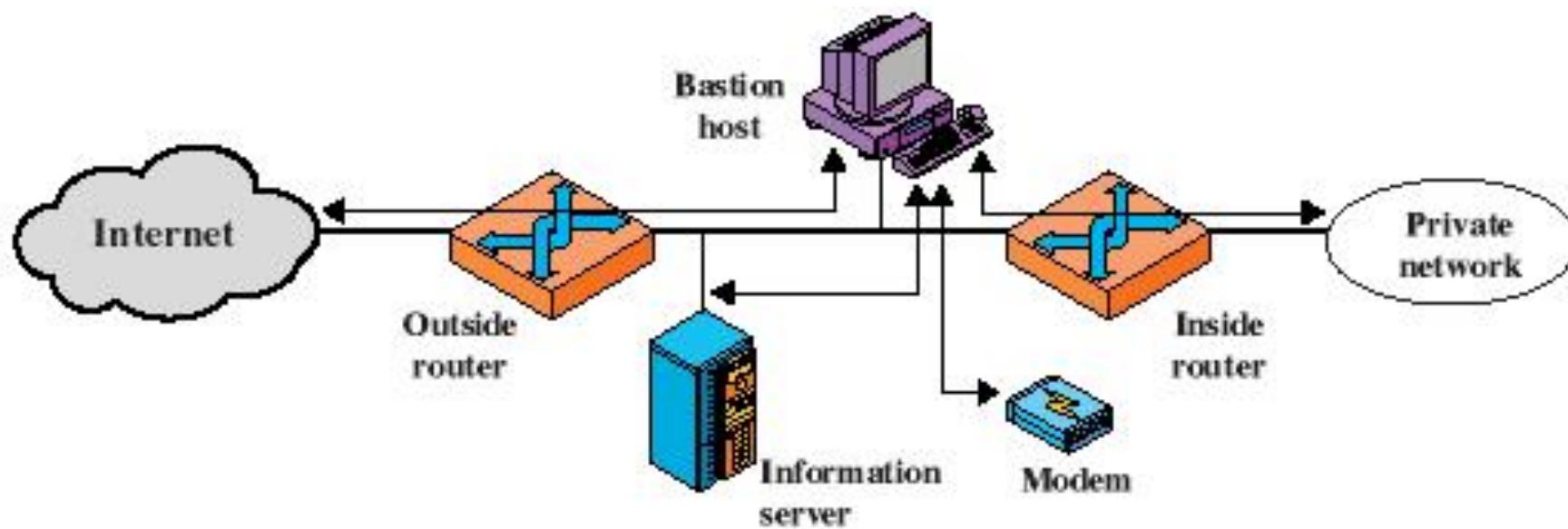
# *Dual-homed Bastion Host*

- Direct connection between internal hosts and packet filter is avoided.
- Packet filter is connected to Bastion host and Bastion host is separately connected internal hosts.
- Thus internal hosts are protected.



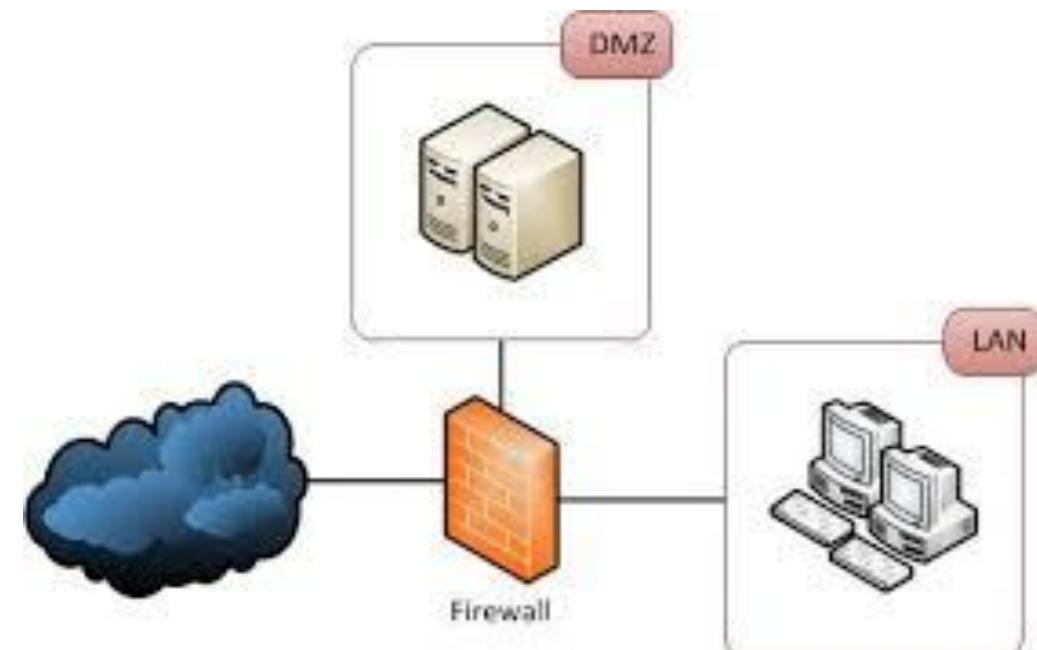
# *Screened Subnet Firewall*

- Two packet-filtering routers creating isolated sub-network
- One packet filter in between Bastion host and internal network.
- Another filter is between Bastion host and Internet.
- Three levels of defense, internal network invisible to Internet, systems on inside cannot construct direct routes to Internet



# Demilitarized Zone

- Organization has server that it need to make available to the outside world.
- Systems that are externally accessible but need some protections are usually located on DMZ networks.
- Typically, the systems in the DMZ require or foster external connectivity, such as a corporate Web site, an e-mail server, or a DNS (domain name system) server.
- Web server/limit traffic to port 80 and 443
- All other traffic can be filtered



# Limitation of firewall

- Insider intrusion
- Direct internet traffic
- Virus attack

THANK YOU

# Intrusion Detection

Process of monitoring the events that occur in a CS

# *Definitions*

## .Intrusion

.A set of actions aimed to compromise the security goals, namely

.Integrity, confidentiality, or availability, of a computing and networking resource

## .Intruders

## .Intrusion detection

.The process of identifying and responding to intrusion activities

## Intrusion Prevention

detecting the signs of intrusion and attempting to stop the intrusion effect.

# *Intruders*

- significant issue for networked systems is unwanted access either via network or local
- One of the two most publicized threats to security is the intruder (the other is viruses), often referred to as a hacker or cracker.
- Basic classes of intruders:
  - masquerader
  - misfeasor
  - clandestine user

# *Intruders*

- **Masquerader:** An individual who is not authorized to use the computer and who penetrates a system's access controls to exploit a legitimate user's account.
- **Misfeasor:** A legitimate user who accesses data, programs, or resources for which such access is not authorized, or who is authorized for such access but misuses his or her privileges
- **Clandestine user:** An individual who seizes supervisory control of the system and uses this control to evade auditing and access controls or to suppress audit collection

# ***Intrusion Techniques***

- .Objective of the intruder is to gain access to a system or to increase the range of privileges accessible on a system.
- .Intruder attempts to acquire information that should have been protected.
- .In some cases, this information is in the form of a user password.
- .With knowledge of some other user's password, an intruder can log in to a system and exercise all the privileges accorded to the legitimate user.

# ***Intrusion Detection***

A system's second line of defence is intrusion detection

## **.Need of intrusion detection:**

- If an intrusion is detected quickly enough, the intruder can be identified and ejected from the system before any damage is done or any data are compromised. Even if the detection is not sufficiently timely to pre-empt the intruder, the sooner that the intrusion is detected, the less the amount of damage and the more quickly that recovery can be achieved.
- An effective intrusion detection system can serve as a deterrent, so acting to prevent intrusions.
- Intrusion detection enables the collection of information about intrusion techniques that can be used to strengthen the intrusion prevention facility.

# IDS

IDSs can be classified as follows:

**Host-based IDS** : Monitors the characteristics of a single host and the events occurring within that host for suspicious activity

**Network-based IDS** : Monitors network traffic for particular network segments or devices and analyzes network, transport, and application protocols to identify suspicious activity

# IDS

An IDS comprises three logical components:

## **Sensors:**

Sensors are responsible for collecting data.

Types of input to a sensor includes network packets, log files, and system call traces.

Sensors collect and forward this information to the analyzer.

## **Analyzers:**

Analyzers receive input from one or more sensors or from other analyzers.

The analyzer is responsible for determining if an intrusion has occurred

The analyzer may provide guidance about what actions to take as a result of the intrusion.

## **User interface:**

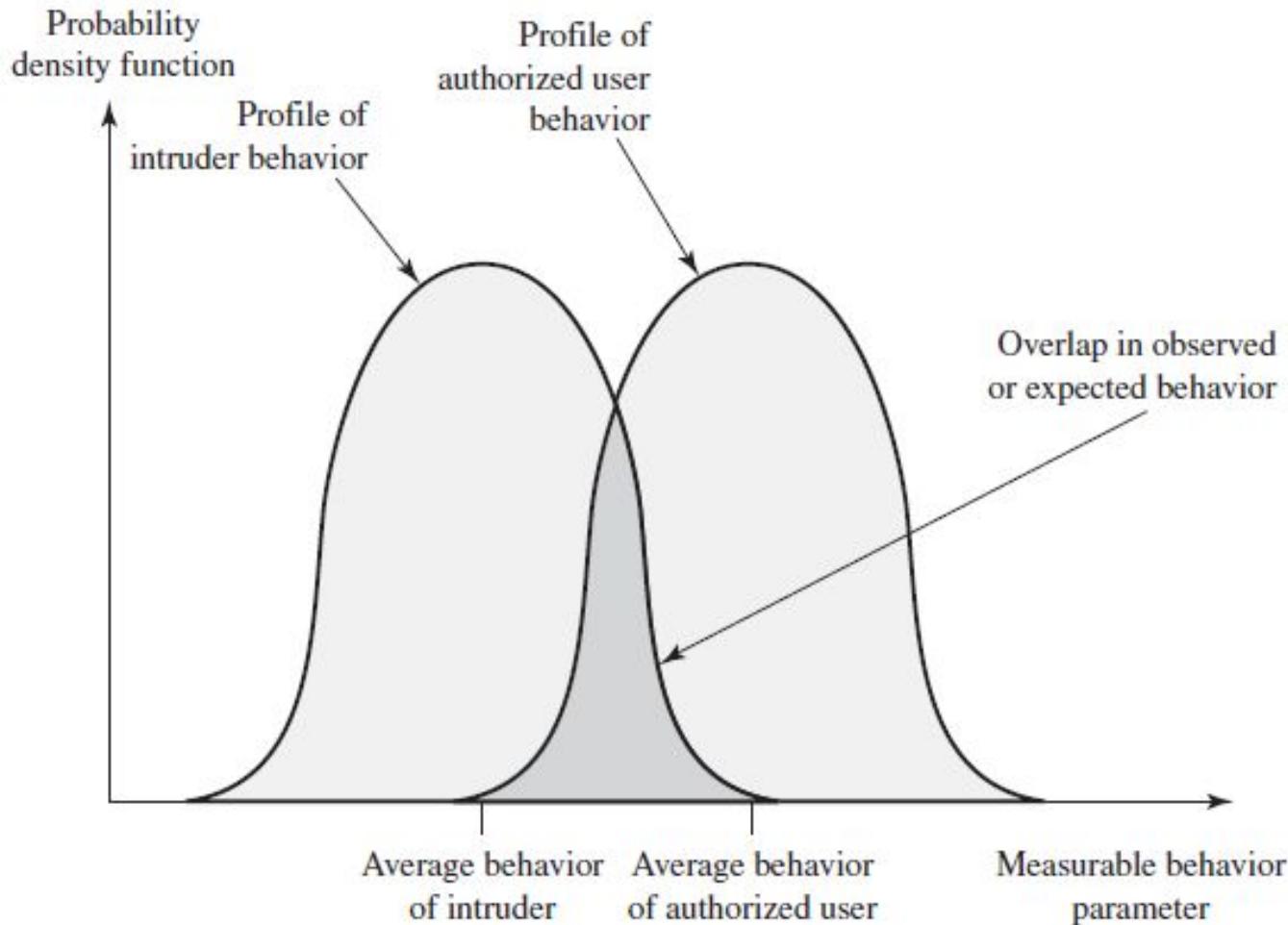
The user interface to an IDS enables a user to view output from the system or control the behavior of the system.

In some systems, the user interface may equate to a manager, director, or console component.

# Intrusion Detection

- It is based on the assumption that the behavior of the intruder differ from that of a legitimate user in ways that can be quantified.
- Although the typical behavior of an intruder differ from that of authorized user , there is some overlapping in the behavior.
- This may lead to false positive- **authorized users identified as intruders**.
- or False negative-**intruders not identified as intruders**.
- In Anderson study , it postulated that one could, with reasonable confidence, distinguish between a masquerader and a legitimate user
- Patterns of legitimate user behavior can be established by observing past history, and significant deviation from such patterns can be detected.

# Behavior of authorized user and intruder

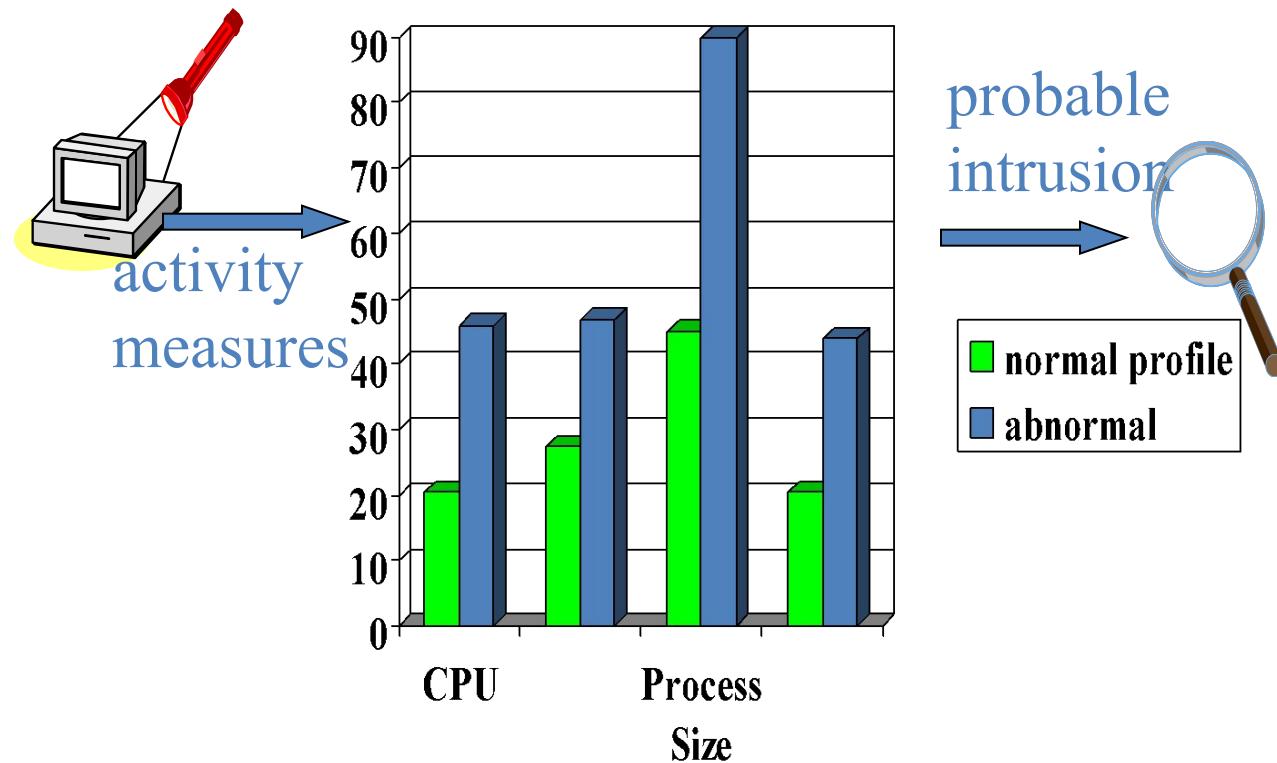


# *Approaches to Intrusion Detection*

The host-based IDS monitors activity on the system in a variety of ways to detect suspicious behavior.

1. **Statistical anomaly detection:** collect data relating to the behavior of legitimate users, then use statistical tests to determine with a high level of confidence whether new behavior is not a legitimate user behavior
  - a. **Threshold detection:** define thresholds, independent of user, for the frequency of occurrence of events.
  - b. **Profile based:** develop profile of activity of each user and use to detect changes in the behavior
2. **Signature detection:** Involves an attempt to define a set of rules or attack patterns that can be used to decide that a given behavior is that of an intruder.

# Anomaly Detection



Relatively high false positive rate -  
anomalies can just be new normal activities.

# *Approaches to Intrusion Detection*

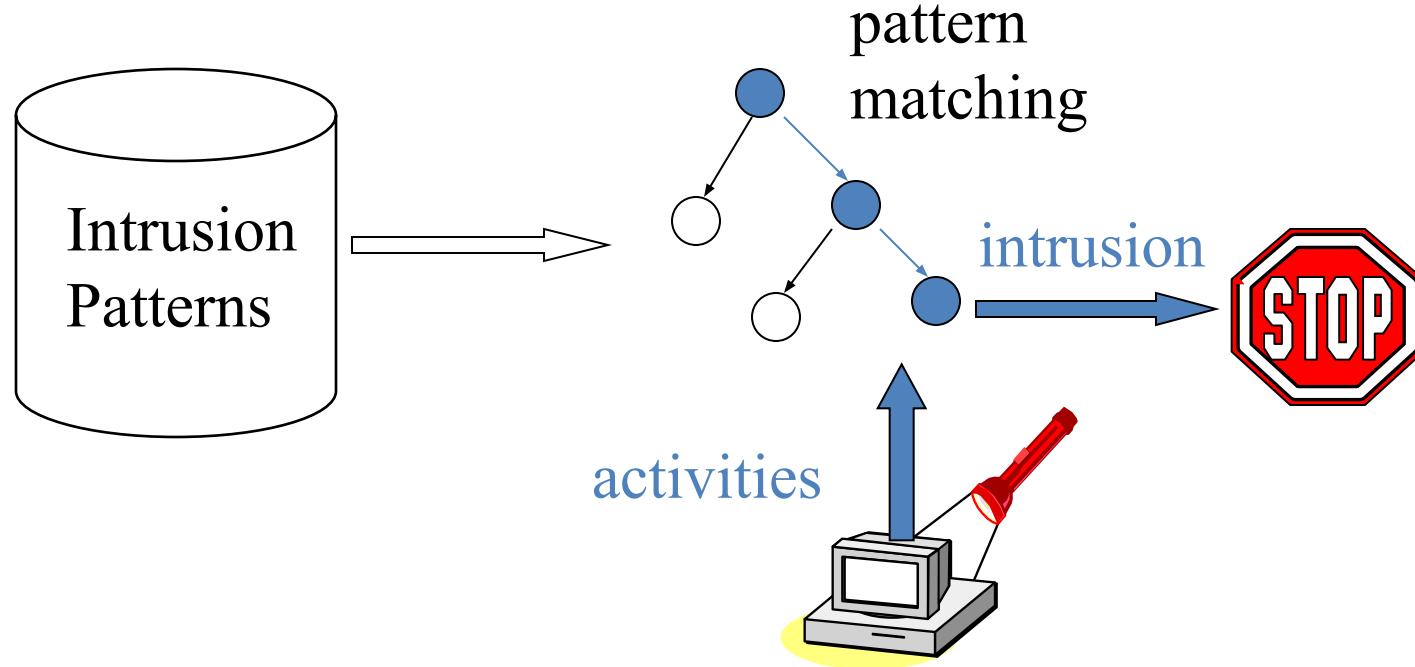
**Rule-based detection:** define a set of rules that can be used to decide that a given behavior is that of an intruder.

**a. Anomaly detection:** Rules are developed to detect deviation from previous usage patterns.

**b. Penetration identification:** An expert system approach that searches for suspicious behaviour.

In a nutshell, statistical approaches attempt to define normal or expected behaviour whereas rule based approaches attempt to define proper behaviour.

# Rule base Detection



Example: *if*(src\_ip == dst\_ip) *then* “land attack”

Can't detect new attacks

# *Audit Records*

Fundamental tool for intrusion detection

- Native audit records:  
OS include accounting software that collects information on user activity part of all common multi-user O/S already present for use  
The advantage of using this information is that no additional collection software is needed.  
The disadvantage is that they may not have info wanted in desired form
- Detection-specific audit records  
A collection facility can be used to implement that generate audit record containing only the information required by the intrusion detection system.

Advantage is it could be made vendor independent and ported to a variety of systems.

Disadvantage is cost of additional overhead on system

# Audit Record Analysis

A example of detection specific audit record is developed by Denning. Each audit record contain the following fields:

Subject: Initiator of action

Action: Operation performed . for example, login, read, perform I/O, execute

Object: Receptor of action Examples include files, programs, messages, records, terminals, printers, and user- or program-created structures.

Exception condition : if any

Resource Usage: A list of quantitative elements in which each element gives the amount used of some resource (e.g., number of lines printed or displayed, number of records read or written, processor time, I/O units used, session elapsed time).

Timestamp: Unique time-and-date stamp identifying when the action took place.

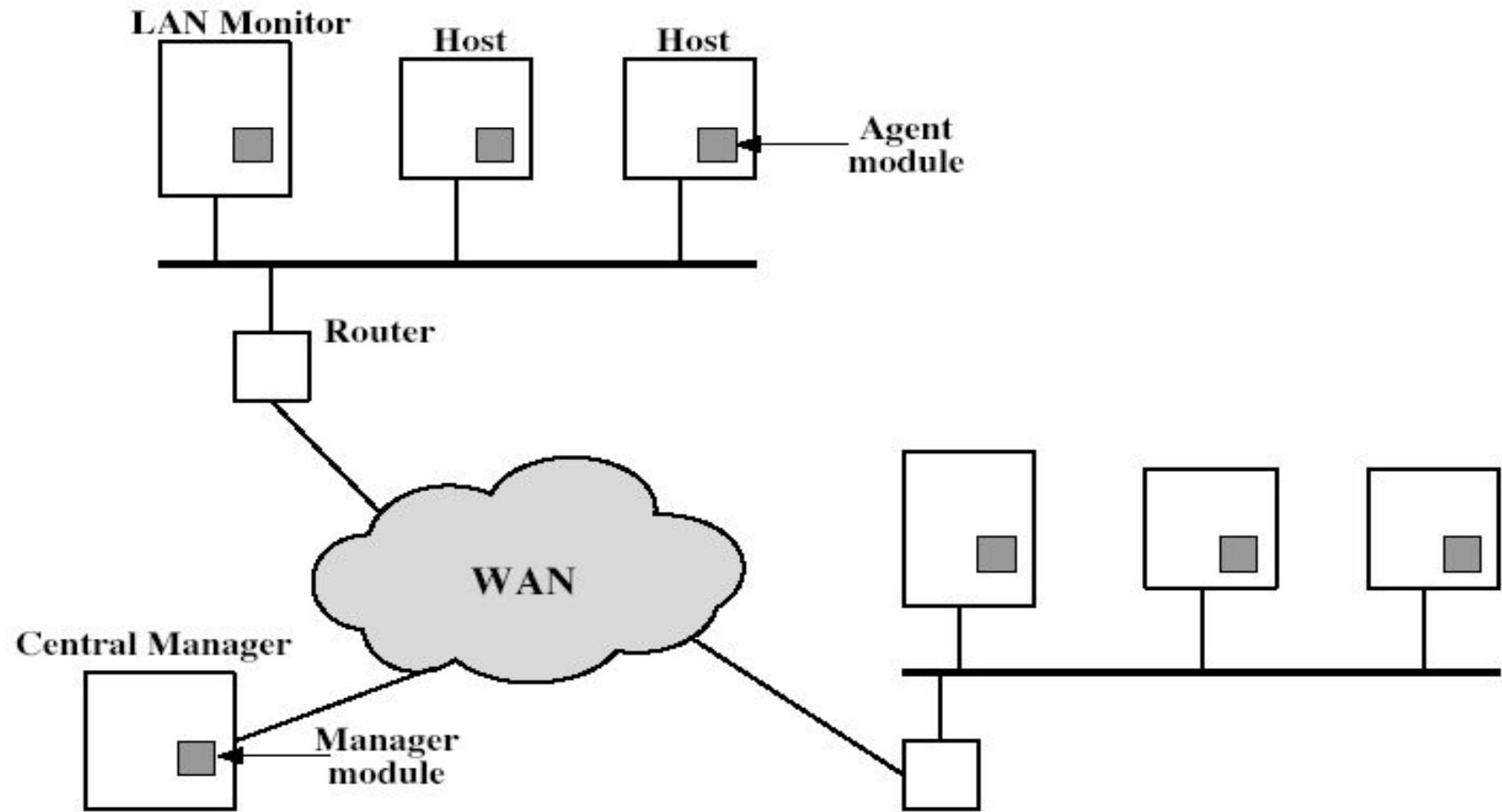
# Base-Rate Fallacy

- practically an intrusion detection system needs to detect a substantial percentage of intrusions with few false alarms
  - if too few intrusions detected -> false security
  - if too many false alarms -> ignore / waste time
- this is very hard to do
- existing systems seem not to have a good record

# Distributed Intrusion Detection

- traditional focus is on single systems
- but typically have networked systems or distributed system
  - dealing with varying audit record formats
  - integrity & confidentiality of networked data
  - centralized or decentralized architecture

# Distributed Intrusion Detection – Architecture



# DIDS Components

The components are:

**Host agent module:** audit collection module operating as a background process on a monitored system

**LAN monitor agent module:** like a host agent module except it analyzes LAN traffic

**Central manager module:** Receives reports from LAN monitor and host agents and processes and correlates these reports to detect intrusion

# General Approach

- The agent captures each native O/S audit record, & applies a filter that
- retains only records of security interest.
- These records are then reformatted into a standardized format (HAR).
- Then a template-driven logic module analyzes the records for suspicious
- activity.
- When suspicious activity is detected, an alert is sent to the central
- manager.
- The central manager includes an expert system that can draw
- inferences from received data.
- The manager may also query individual systems for copies of HARs to
- correlate with those from other agents.

# Distributed Intrusion Detection – Agent Implementation

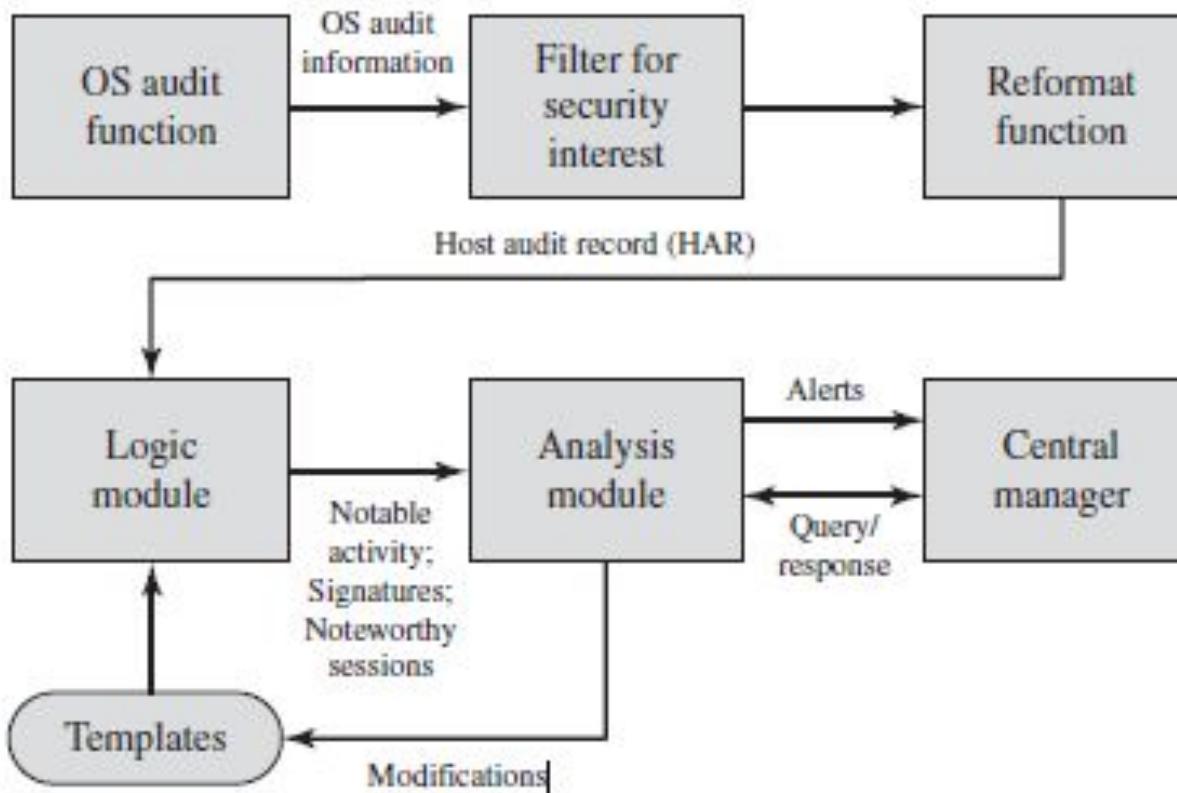


Figure 8.3 Agent Architecture

# Honeypots

- decoy systems to lure attackers
  - away from accessing critical systems
  - to collect information of their activities
  - to encourage attacker to stay on system so administrator can respond
- are filled with fabricated information
- instrumented to collect detailed information on attackers activities
- may be single or multiple networked systems

# THE NEED FOR DATABASE SECURITY

Organizational databases tend to concentrate sensitive information in a single logical system. Examples include:

1. Corporate financial data
2. Confidential phone records
3. Customer and employee information, such as name, Social Security number, bank account information, credit card information  
Proprietary product information Health care information and medical records

# SQL Based Access Definition

SQL Provide two commands for managing access right Grant and Revoke

- GRANT { privileges | role }
- [ON table]
- TO { user | role | PUBLIC }
- [IDENTIFIED BY password]
- [WITH GRANT OPTION]

GRANT SELECT ON ANY TABLE TO ricflair

This statement enables user ricflair to query any table in the database.

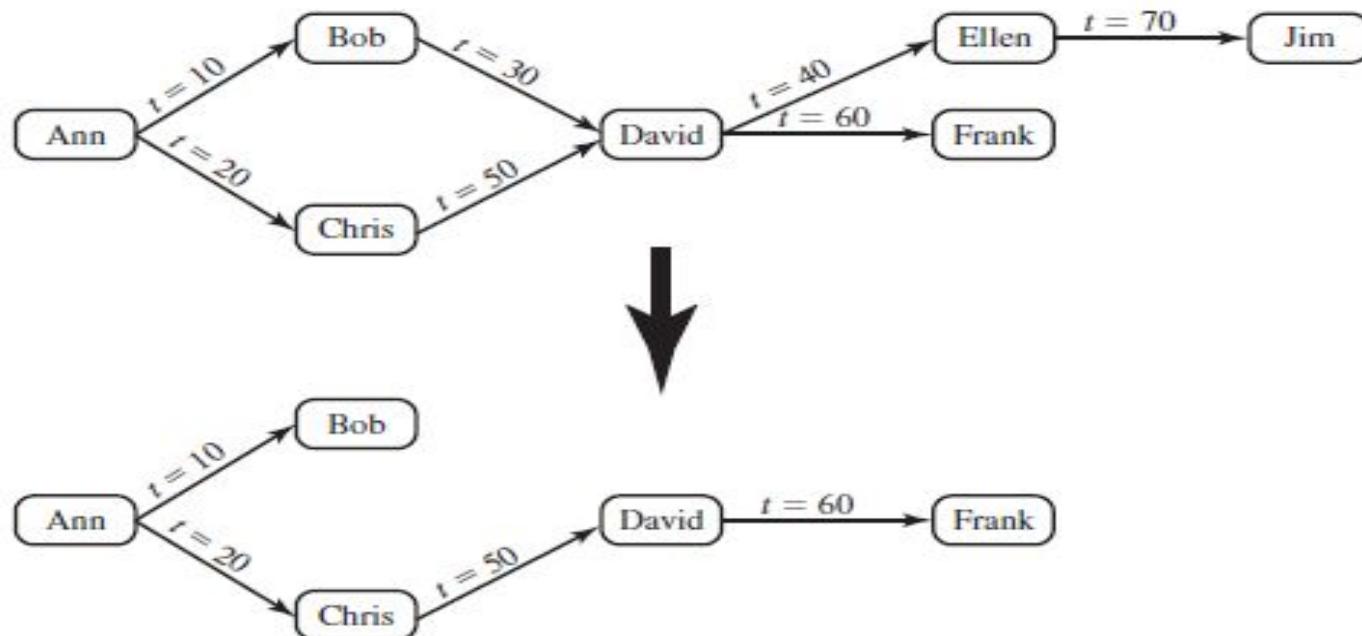
# Revoke

The REVOKE command has the following syntax:

```
REVOKE { privileges | role }  
[ON table]  
FROM { user | role | PUBLIC }
```

# Cascading Authorization

- The grant option enables an access right to cascade through a number of users



# The issue

- Just as the granting of privileges cascades from one user to another using the grant option, the revocation of privileges also cascaded. Thus, if Ann revokes the access right to Bob and Chris, then the access right is also revoked to David, Ellen, Jim, and Frank. A complication arises when a user receives the same access right multiple times, as happens in the case of David.
- Because David granted the access right to Frank after David was granted the access right with grant option from Chris, the access right to Frank remains

# **Role-Based Access Control**

A role-based access control (RBAC) scheme is a natural fit for database access control.

we can classify database users in three broad categories:

**1.Application owner**

**2.End user other than application owner**

**3.Administrator**

- An application has associated with it a number of tasks, with each task requiring specific access rights to portions of the database.
- The application owner may assign roles to end users
- Administrators are responsible for Security mechanism

# RBAC

A database RBAC facility needs to provide the following capabilities:

- 1.Create and delete roles.
- 2.Define permissions for a role.
- 3.Assign and cancel assignment of users to roles.

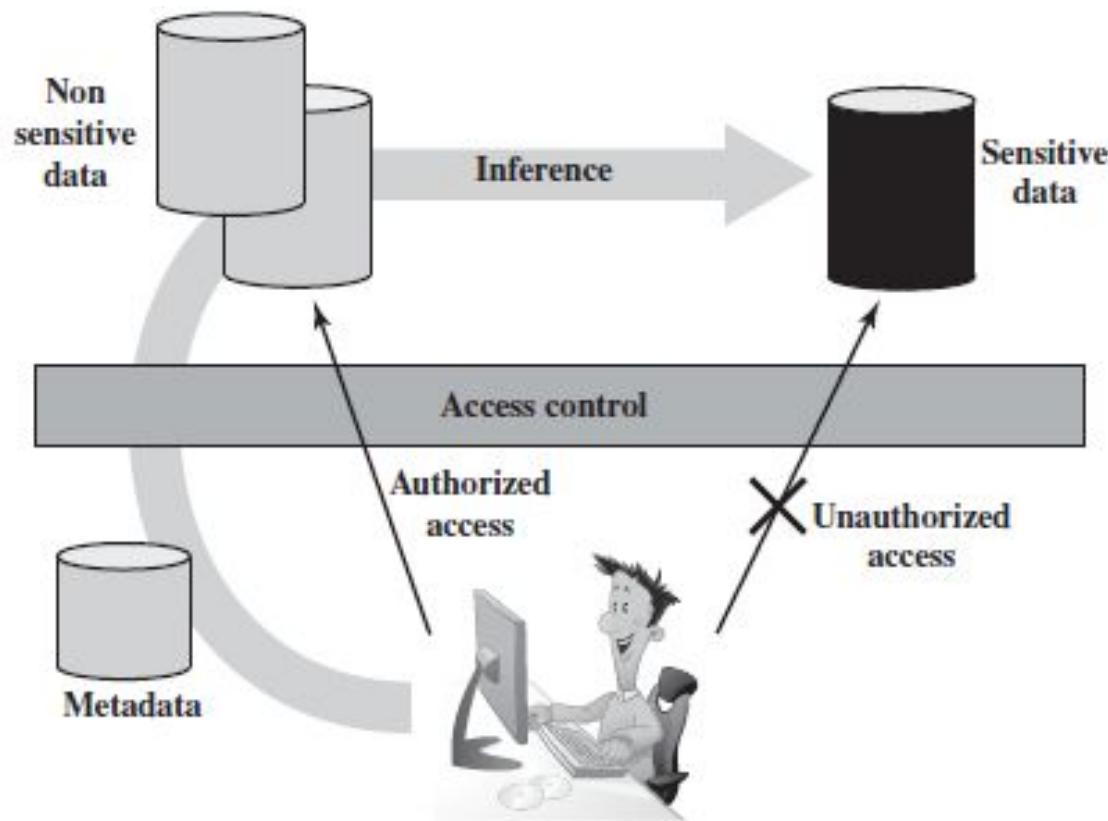
•SQL Server supports three types of roles: server roles, database roles, and user-defined roles

# Interface

- Inference, as it relates to database security, is the process of performing authorized queries and deducing unauthorized information from the legitimate responses received.
- The inference problem arises when the combination of a number of data items is more sensitive than the individual items, or when a combination of data items can be used to infer data of a higher sensitivity

# Interface Channel

- The information transfer path by which unauthorized data is obtained is referred to as an **inference channel**.



# Interface Problem

- CREATE view V1 AS SELECT Availability, Cost FROM Inventory WHERE Department "hardware"
- CREATE view V2 AS SELECT Item, Department FROM Inventory WHERE Department "hardware"
- suppose the two views are created with the access constraint that Item and Cost cannot be accessed together.

Item	Availability	Cost (\$)	Department
Shelf support	in-store/online	7.99	hardware
Lid support	online only	5.49	hardware
Decorative chain	in-store/online	104.99	hardware
Cake pan	online only	12.99	housewares
Shower/tub cleaner	in-store/online	11.99	housewares
Rolling pin	in-store/online	10.99	housewares

(a) Inventory table

Availability	Cost (\$)
in-store/online	7.99
online only	5.49
in-store/online	104.99

Item	Department
Shelf support	hardware
Lid support	hardware
Decorative chain	hardware

(b) Two views

Item	Availability	Cost (\$)	Department
Shelf support	in-store/online	7.99	hardware
Lid support	online only	5.49	hardware
Decorative chain	in-store/online	104.99	hardware

(c) Table derived from combining query answers

- A user who knows the structure of the Inventory table and who knows that the view tables maintain the same row order as the Inventory table is then able to merge the two views to construct the table.
- This violates the access control policy that the relationship of attributes Item and Cost must not be disclosed.
- In general terms, there are two approaches to dealing with the threat of disclosure by inference:

**Inference detection during database design**

**Inference detection at query time**

# Example Interface

- Employees (Emp#, Name, Address)
  - Salaries (S#, Salary)
  - Emp-Salary (Emp#, S#)
  - The Employees table and the Salaries table are accessible to the Clerk role, but the Emp-Salary table is only available to the Administrator role. In this structure, the sensitive relationship between employees and salaries is protected from users assigned the Clerk role
  - Employees (Emp#, Name, Address)
  - Salaries (S#, Salary, Start-Date)
  - Emp-Salary (Emp#, S#)
- An employee's start date is an easily observable or discoverable attribute of an employee. Thus a user in the Clerk role should be able to infer (or partially infer) the employee's name.
- A straightforward way to remove the inference channel is to add the start-date column to the Employees table rather than to the Salaries table.

# Statistical Database (SDB)

- **Pure statistical database:** This type of database only stores statistical data. An example is a census database. Only certain user is allowed to access entire database.
- **Ordinary database with statistical access:** This type of database contains individual entries. supports a population of non statistical users who are allowed access to selected portions of the database.

*Objective for an SDB system is to provide users with the aggregate information without compromising the confidentiality of any individual entity represented in the database.*

# Abstract Model

- There are  $N$  individuals, or entities, in the table and  $M$  attributes.  
 $(\text{Sex Male}) \bullet ((\text{Major CS}) (\text{Major EE}))$   
 $C = \text{Female} \bullet \text{CS, X}(C)$

## Inference from a Statistical Database

- The inference problem in this context is that a user may infer confidential information about individual entities represented in the SDB. Such an inference is called a **compromise**.

# Interface in SDB

The statistic **sum** (EE • Female, GP) 2.5 compromises the database if the user knows that Baker is the only female EE student.

In some cases, a sequence of queries may reveal information. For example, suppose a questioner knows that Baker is a female EE student but does not know if she is the only one.

Consider the following sequence of two queries:

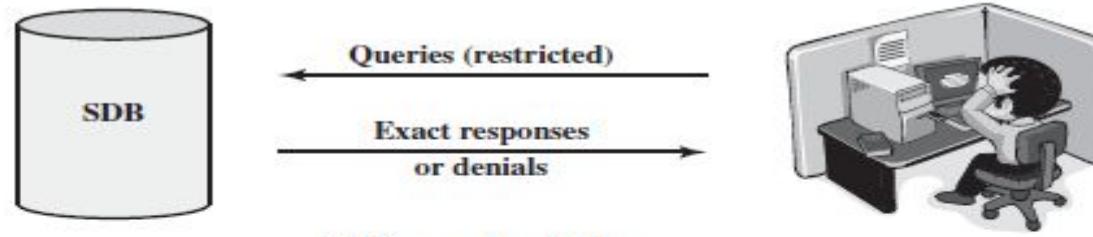
**count** (EE • Female) 1

**sum** (EE • Female, GP) 2.5

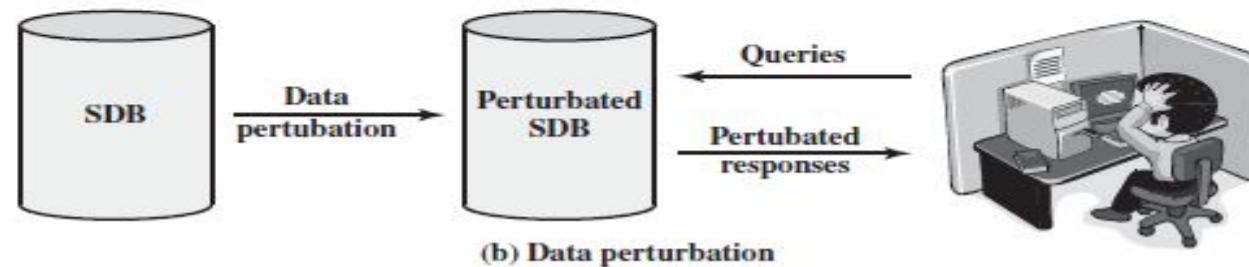
This sequence reveals the sensitive information.

# Query Restriction

1. **Query restriction:** Rejects a query that can lead to a compromise. The answers provided are accurate.
2. **Perturbation:** Provides answers to all queries, but the answers are approximate.



(a) Query set restriction



(b) Data perturbation

# Perturb

- Perturb the original values such that confidential individual data become useless for a snooper while statistical properties of attribute are preserved.
- The manipulated data is stored in the second database and is then freely accessible for the user.
- Noise addition means adding distributing term in each value
  - $Y=x+e$

X is the orginal value and e is the propbability distribution with the mean zero.

# Data Perturbation Techniques

- Data Swapping: In this method, attribute values are exchanged (swapped) between records in sufficient quantity so that nothing can be deduced from the disclosure of individual records. The swapping is done in such a way that the accuracy of at least low-order statistics is preserved.

Record	D			D'		
	Sex	Major	GP	Sex	Major	GP
1	Female	Bio	4.0	Male	Bio	4.0
2	Female	CS	3.0	Male	CS	3.0
3	Female	EE	3.0	Male	EE	3.0
4	Female	Psy	4.0	Male	Psy	4.0
5	Male	Bio	3.0	Female	Bio	3.0
6	Male	CS	4.0	Female	CS	4.0
7	Male	EE	4.0	Female	EE	4.0
8	Male	Psy	3.0	Female	Psy	3.0

# Data Perturbation Techniques

Another method is to generate a modified database using the estimated underlying probability distribution of attribute values. The following steps are used:

- 1.** For each confidential or sensitive attribute, determine the probability distribution function that best matches the data and estimate the parameters of the distribution function.
- 2.** Generate a sample series of data from the estimated density function for each sensitive attribute.
- 3.** Substitute the generated data of the confidential attribute for the original data in the same rank order. That is, the smallest value of the new sample should replace the smallest value in the original data, and so on.