Cryptography (암호학)

2010. 4. 8.

이 병 천 교수

중부대학교 정보보호학과

Outline

1. Information Security

2. History of Cryptography

3. Modern Cryptography
   3.1 Overview
   3.2 Block cipher
   3.3 Public key encryption
   3.4 Digital signature
   3.5 Hash and MAC

4. Certification and PKI

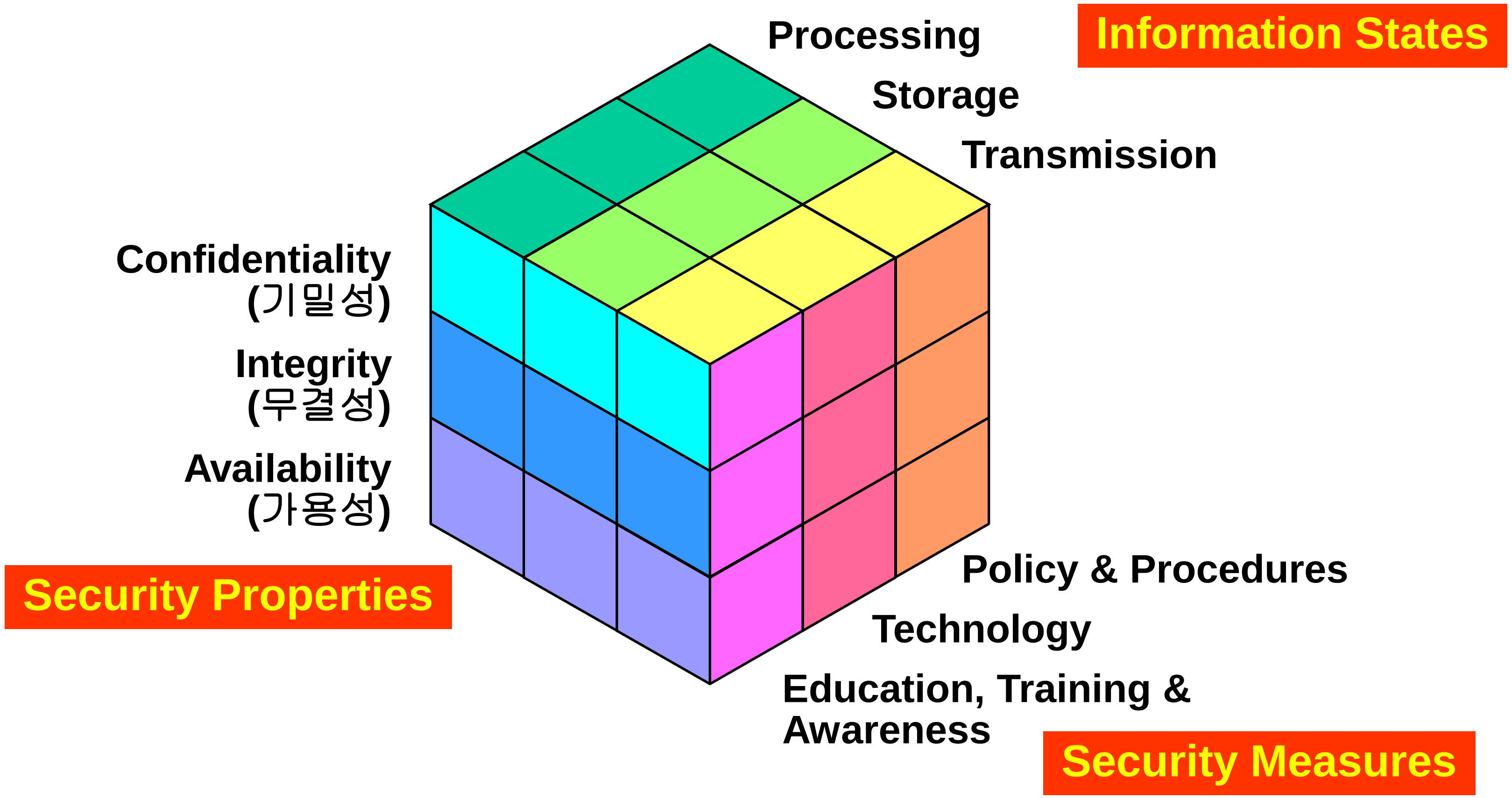
1. Information Security

What is Information Security?

- ❖ **Information Security (정보보호, 정보보안)**
  - ❖ **Information security is the process of protecting information from unauthorized access, use, disclosure, destruction, modification, or disruption**
  - ❖ **Protecting the confidentiality, integrity and availability of information**

  - ❖ **Information security is an essential infrastructure technology to achieve successful information-based society**
  - ❖ **Highly information-based company without information security will lose competitiveness**

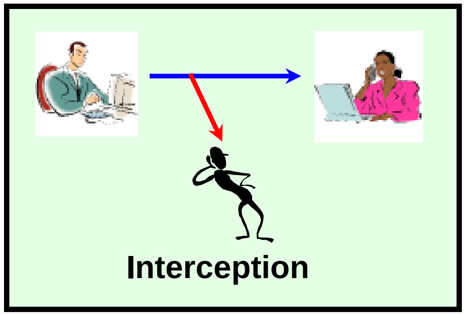
What is Information Security?



NSTISSI 4011: National Training Standard for Information Systems Security Professionals, 1994

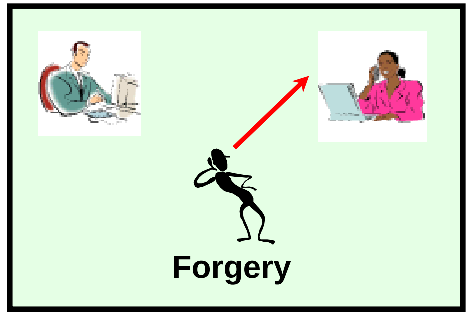
Security Needs for Network Communications

Confidentiality



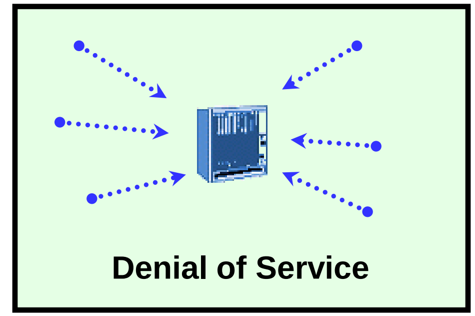
Is Private?

Authentication



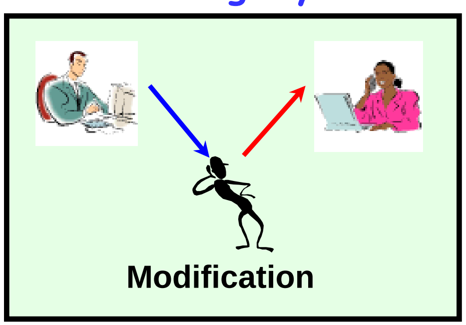
Who am I dealing with?

Availability



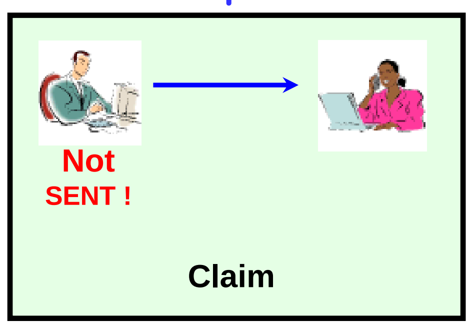
Wish to access!!

Integrity



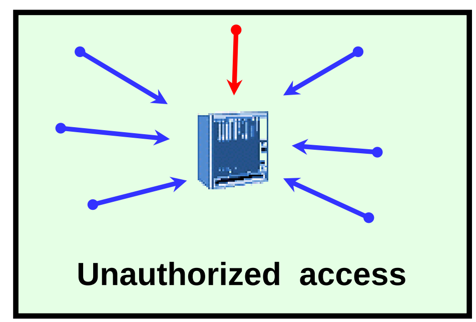
Has been altered?

Non-Repudiation



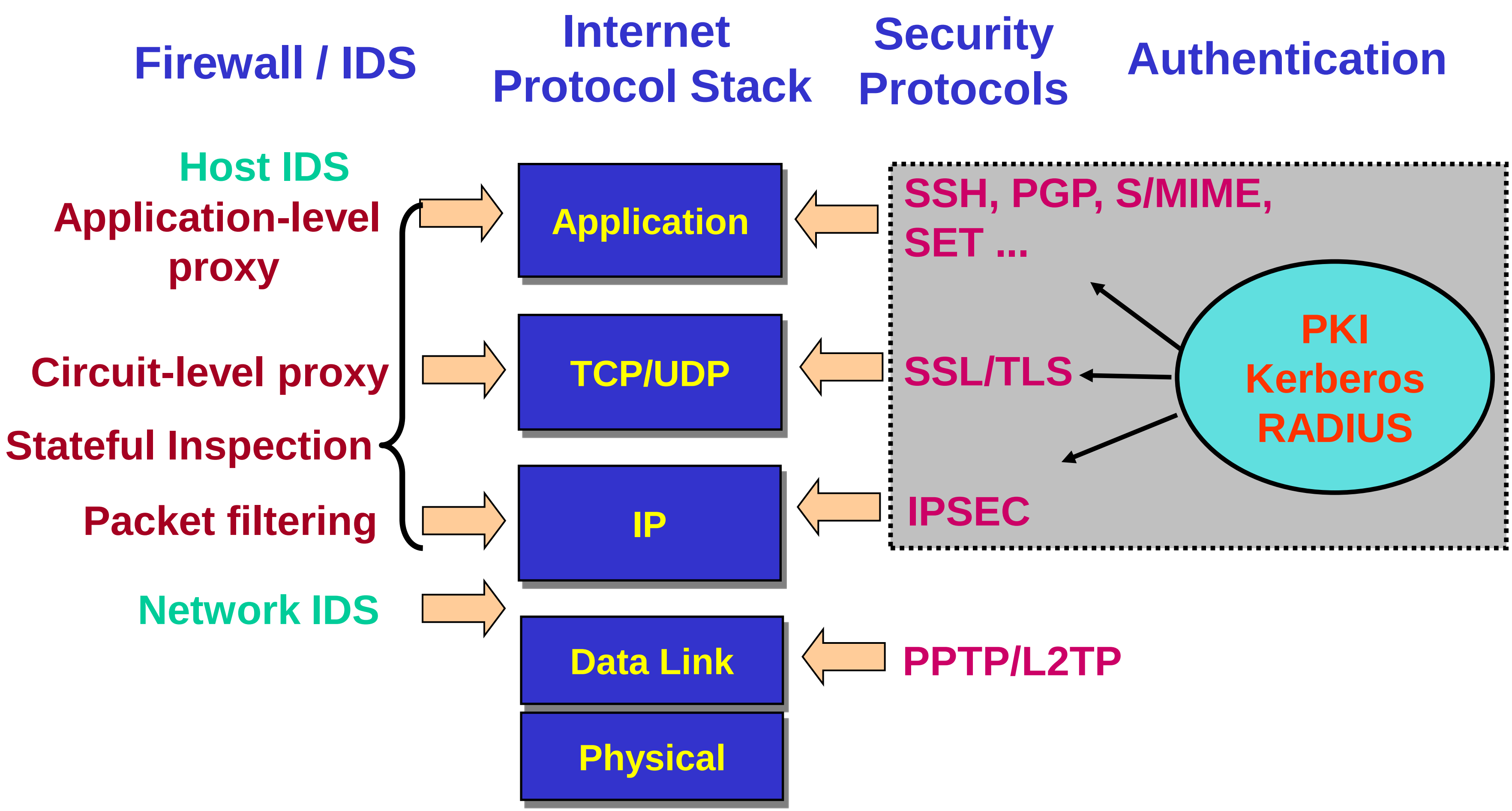
Who sent/received it?

Access Control



Have you privilege?

Major Internet Security Technologies

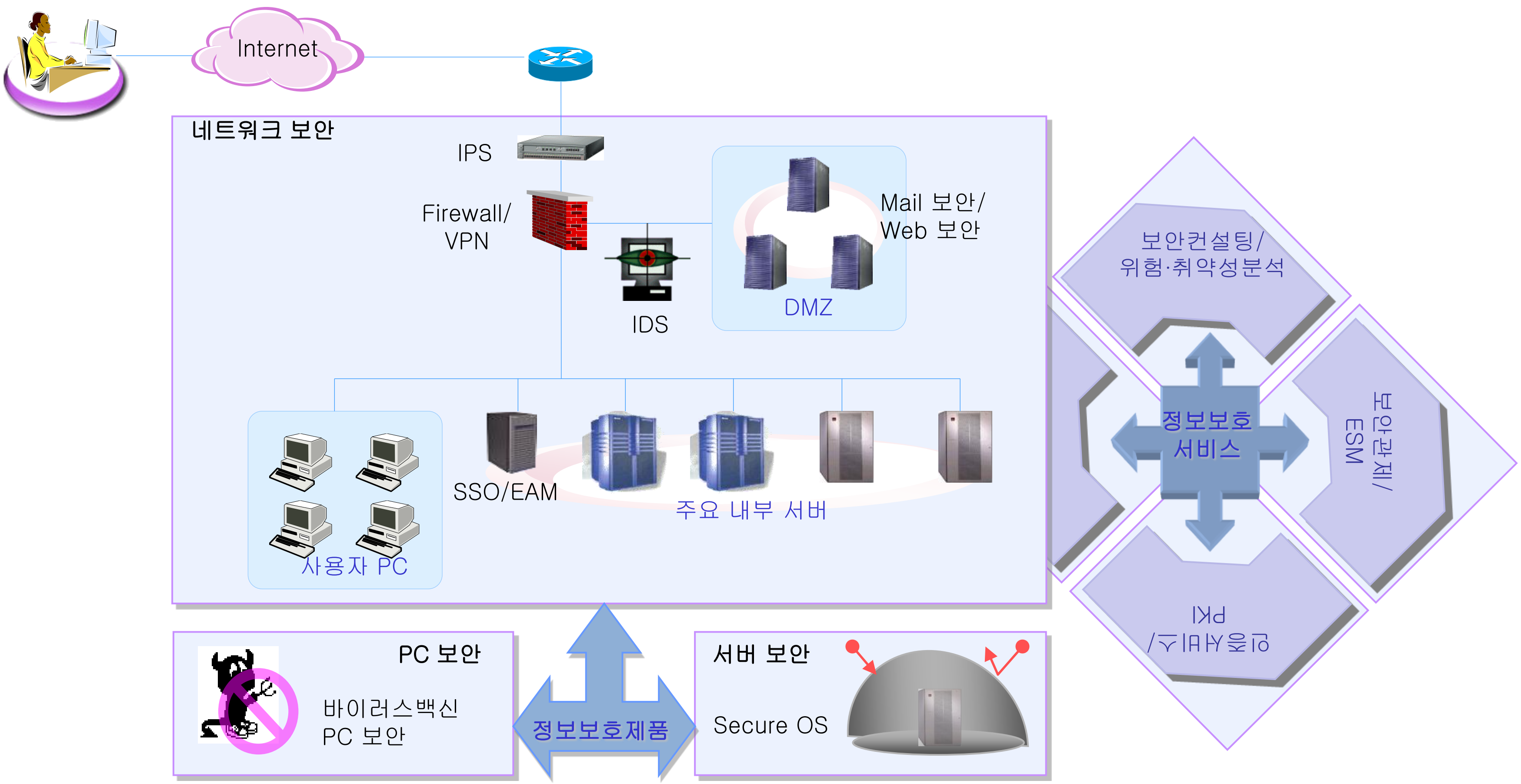


The OSI Security Architecture

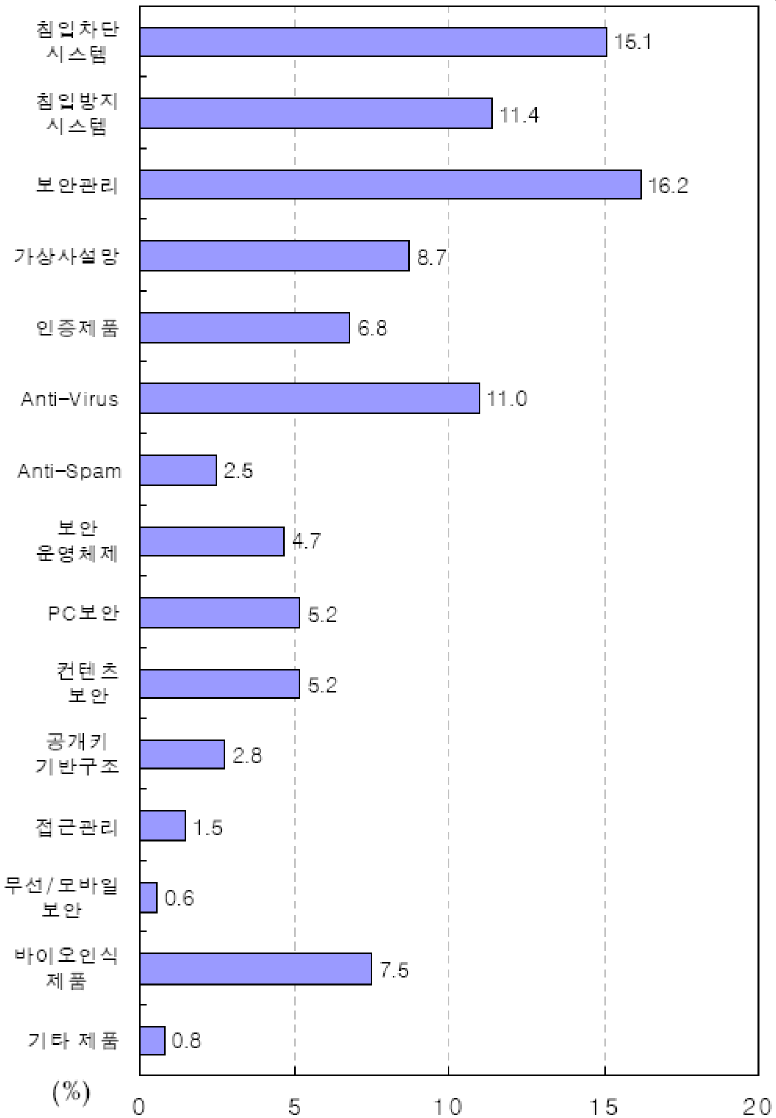
- ITU-T Recommendation X.800, *Security Architecture for OSI*

Security Attacks	Security Mechanisms	Security Services
Interception Forgery Modification Denial of facts Unauthorized access Interruption	Encryption Authentication Digital signature Key exchange Access control Monitoring & Responding	Confidentiality Authentication Integrity Non-repudiation Access control Availability

Information Security Industry



Information Security Industry



[단위 : 백만원]

대분류	소분류	2005년	2006년	증감율(%)
시스템 및 네트워크 정보보호 제품	침입차단(방화벽)시스템	89,108	94,554	6.1
	침입방지시스템(IPS)	62,406	71,469	14.5
	보안관리	87,957	101,038	14.9
	가상사설망(VPN)	53,776	54,558	1.5
	인증제품	56,736	42,691	-24.8
	Anti-Virus	57,935	68,671	18.5
	Anti-Spam	10,756	15,438	43.5
	보안운영체제(Secure OS)	28,652	29,144	1.7
	PC 보안	29,500	32,662	10.7
	컨텐츠 보안	27,329	32,554	19.1
	공개키기반구조(PKI)	16,085	17,735	10.3
	접근관리	11,221	9,087	-19.0
	무선/모바일 보안	2,516	3,646	44.9
	바이오인식 제품	43,195	46,725	8.2
	기타 제품	6,251	5,210	-16.7
	소 계	583,423	625,182	7.2
정보보호 서비스	인증서비스	6,549	7,465	14.0
	보안관제	22,241	26,602	19.6
	보안컨설팅	26,331	31,093	18.1
	유지보수	40,051	43,186	7.8
	기타서비스	2,110	1,264	-40.1
	소 계	97,282	109,610	12.7
합 계		680,705	734,792	7.9

2. History of Cryptography

History of Cryptologic Research

1900BC : Non-standard hieroglyphics

1500BC : Mesopotamian pottery glazes

50BC : Caesar cipher

1518 : Trithemius' cipher book

1558 : Keys invented

1583 : Vigenere's book

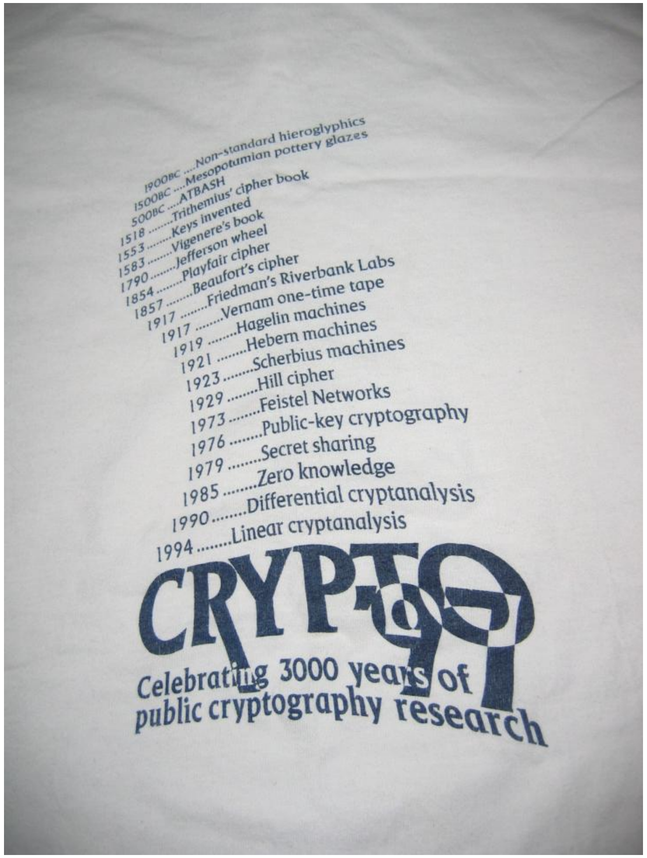
1790 : Jefferson wheel

1854 : Playfair cipher

1857 : Beaufort's cipher

1917 : Friedman's Riverbank Labs

1917 : Vernam one-time pads



History of Cryptologic Research

1919 : Hegelin machines

1921 : Hebern machines

1929 : Hill cipher

1973 : Feistel networks

1976 : Public key cryptography

1979 : Secret sharing

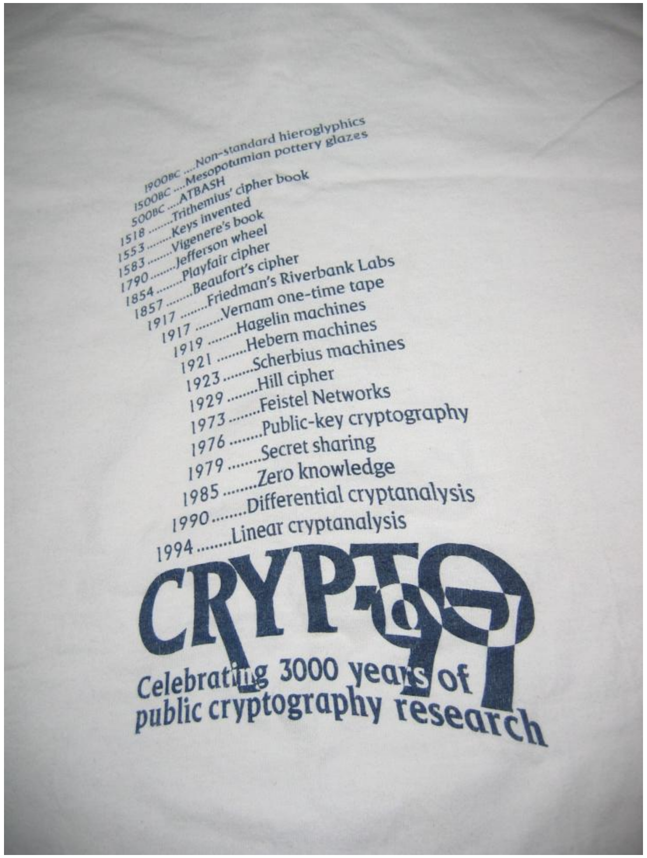
1985 : Zero knowledge

1990 : Differential cryptanalysis

1994 : Linear cryptanalysis

1997 : Triple-DES

1998 ~ 2001 : AES



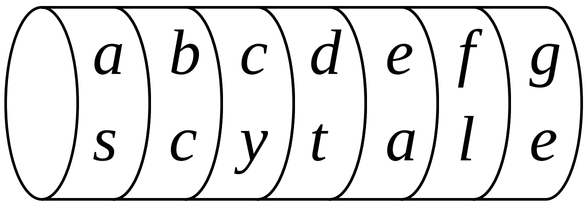
History of Cryptologic Research

	Period	Features	Examples
Manual Crypto	ancient ~ 1920	Substitution Transposition	Scytale, Caesar, Vigenere, Beaufort (USA)
Machine Crypto	1920 ~ 1950	Using complex machine	Enigma (Germany in 2nd WW) M-209 (USA in 2nd WW)
Modern Crypto Computer Crypto	1950 ~ current	Using computer Shannon's theory	DES, SEED, AES RSA, ECC, KCDSA

Using Cryptologic Technology

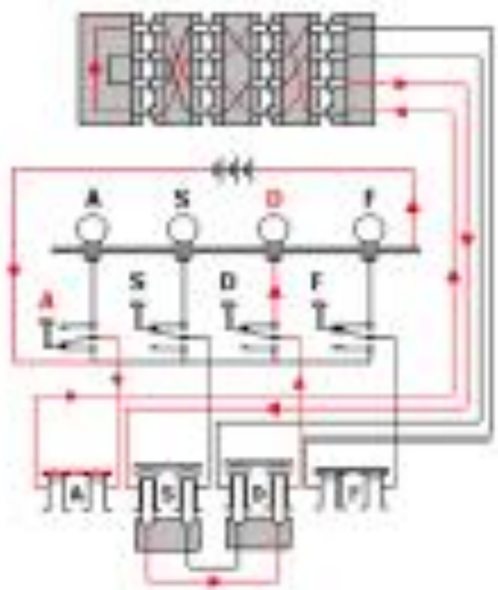
- Before modern crypto : limited usage
  - National security, diplomatic, war
  - Used by limited people
  - Researched by limited people
- Current crypto : widely open, standardized, commerce
  - Internet, e-commerce
  - Anybody is using
  - Research and development by anyone

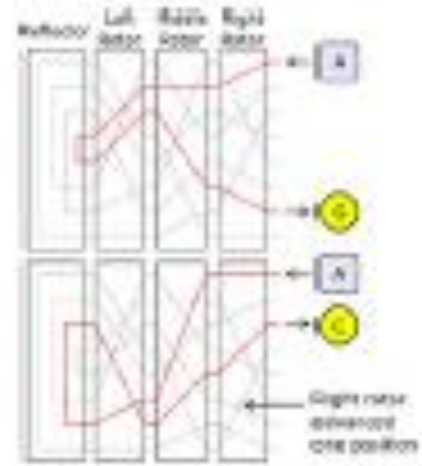
Scytale



as bc cy dt ea fl ge

Enigma





Article at
http://en.wikipedia.org/wiki/Enigma_machine

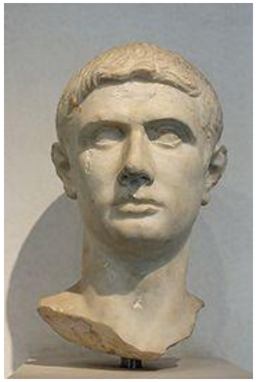
Classical Encryption Techniques

- Basic building blocks of all encryption techniques
  - Substitution(대치,치환): replacement
  - Transposition(전치): relocation

- Substitution ciphers
  - Caesar cipher
  - Monoalphabetic ciphers
  - Playfair cipher
  - Hill cipher
  - Polyalphabetic ciphers: Vigenere cipher
  - Vernam cipher/One-time pad: perfect cipher

- Transposition techniques
  - Rotor machines: Enigma, Purple

Caesar Ciphers



Julius Caesar, the Roman emperor
Also known as shift cipher

Mathematically assign numbers to each alphabet

a	b	c	d	e	f	g	h	i	j	k	...	z
0	1	2	3	4	5	6	7	8	9	10	...	25

Caesar cipher :

$C = E_K(M) = M + K \mod 26$
$K = 3$

$M = D_K(C) = C - K \mod 26$
$K = 3$

Caesar Ciphers

Define transformation as:

a	b	c	d	e	f	g	h	i	j	k	…	z
D	E	F	G	H	I	J	K	L	M	N	…	C

Encryption example

i	**n**	**f**	**o**	**r**	**m**	**a**	**t**	**i**	**o**	**n**
L	**Q**	**I**	**R**	**U**	**P**	**D**	**W**	**L**	**R**	**Q**

Weakness

- **Key space is too short – only 26 possible keys**
- **Brute force search**

Example: Break ciphertext “L ORYH LFX"

Monoalphabetic Substitution Ciphers

Example : 1-1 Substitution rule

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

Example : Encryption

i	n	f	o	r	m	a	t	i	o	n
P	R	N	X	Y	C	E	D	P	X	R

Key space : 26!

Cryptanalysis: Using English character frequency analysis…

English Character Frequencies

Letter	Frequency(%)	Letter	Frequency(%)	Letter	Frequency(%)
e	12.7	d	4.3	p	1.9
t	9.1	l	4.0	b	1.5
a	8.2	c	2.8	v	1.0
o	7.5	u	2.8	k	0.8
i	7.0	m	2.4	j	0.2
n	6.7	w	2.3	x	0.1
s	6.3	f	2.2	q	0.1
h	6.1	g	2.0	z	0.1
r	6.0	y	2.0		

(1) Pr(e)=0.12, (2) Pr(t,a,o,i,n,s,h,r) = 0.06 ~0.09
(3) Pr(d,l)=0.04 (4) Pr(c,u,m,w,f,g,y,p,b)= 0.015~0.023
(5) Pr(v,k,j,x,q,z) <=0.01

Polyalphabetic Substitution Ciphers

Vigenère Ciphers

- **Multiple caesar cipher**

$$k = (k_1, k_2, \ldots, k_d), |k| = 26^d$$

$$c = E_k(m_1, m_2, \ldots, m_d) = (c_1, c_2, \ldots, c_d) = m_i + k_i \bmod 26 \ \text{ for } i = 1, \ldots, d$$

$$m = D_k(c_1, c_2, \ldots, c_d) = (m_1, m_2, \ldots, m_d) = c_i - k_i \bmod 26 \ \text{ for } i = 1, \ldots, d$$

Beauford ciphers (used in US civil war)

$$k = (k_1, k_2, \ldots, k_d), |k| = 26^d$$

$$c = E_k(m_1, m_2, \ldots, m_d) = (c_1, c_2, \ldots, c_d) = k_i - m_i \bmod 26 \ \text{ for } i = 1, \ldots, d$$

$$m = D_k(c_1, c_2, \ldots, c_d) = (m_1, m_2, \ldots, m_d) = k_i - c_i \bmod 26 \ \text{ for } i = 1, \ldots, d$$

Vigenère Ciphers

Look-up table for Vigenère Ciphers

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

Vigenère Ciphers

Plaintext	t h i s c r y p t o s y s t e m i s n o t s e c u r e
Keyword	S E C U R I T Y S E C U R I T Y S E C U R I T Y S E C
Ciphertext	L L K M T Z R N L S U S J B X K A W P I K A X A M V G

One-time Pad (Vernam cipher)

- **Use a random key as long as the message size and use the key only once**
- **Unbreakable**
  - **Since ciphertext bears no statistical relationship to the plaintext**
  - **Since for any plaintext & any ciphertext there exists a key mapping one to other**
- **Have the problem of safe distribution of key**

Ex) Binary alphabet

P :	o	n	e	t	i
P':	01101111	01101110	01100101	01110100	01101001
K :	01011100	01010001	11100000	01101001	01111010
C :	00110011	00111111	10000101	00011101	00010011

Perfect Cipher : $p(x|y) = p(x)$ for all $x \in P$, $y \in C$

Impossible COA

KAIST ICC

KIPO

Transposition Ciphers

- Rearrange characters of plaintext to produce ciphertext
- Frequency distribution of the characters is not changed by encryption

- Example:

Encryption permutation

1	2	3	4	5	6
3	5	1	6	4	2

Decryption permutation

1	2	3	4	5	6
3	6	1	5	2	4

plaintext	i n f o r m a t i o n s e c u r i t y x y z a b
ciphertext	F R I M O N I N A S O T U I E T R C Y A Y B Z X

ADFGVX

- **Product of substitution and permutation**

Substitution table

	A	D	F	G	V	X
A	f	x	a	9	u	1
D	n	g	0	l	d	o
F	5	b	k	2	h	z
G	m	j	s	y	t	v
V	7	4	3	e	8	i
X	c	w	q	6	r	p

c → XA

Substitution result

c	o	n	v	e	n	t	i	o	n	a	l
X	D	D	G	V	D	G	V	D	D	A	D
A	X	A	X	G	A	V	X	X	A	F	G

c	r	y	p	t	o	g	r	a	p	h	y
X	X	G	X	G	D	D	X	A	X	F	G
A	V	G	X	V	X	D	V	F	X	V	G

ADFGVX

Permutation table

C	I	P	H	E	R
1	**4**	**5**	**3**	**2**	**6**
X	A	D	X	D	A
G	X	V	G	D	A
G	V	V	X	D	X
D	A	A	F	D	G
X	A	X	V	G	G
X	X	G	V	D	X
D	D	X	V	A	F
X	X	F	V	G	G

Keyword

permutation

Ciphertext

XGGDXXDX

DDDDGDAG

XGXFVVVV

AXVAAXDX

DVVAXGXF

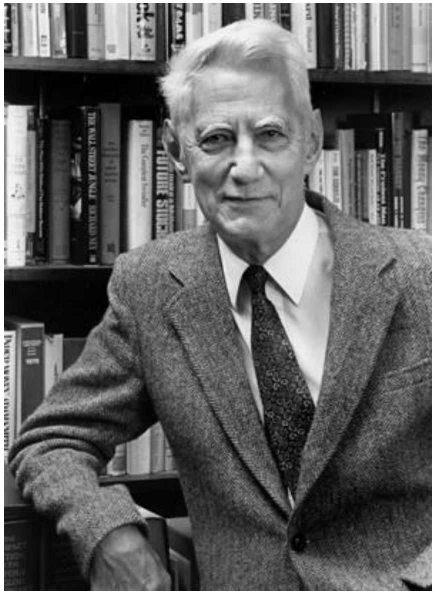
AAXGFXFG

Shannon's Proposal

- C. Shannon, "Communication Theory for Secrecy Systems", 1949
  - Compose different kind of simple and insecure ciphers to create complex and secure cryptosystems → called "product cipher"
  - Incorporate confusion and diffusion
  - Substitution-Permutation Network

http://www.bell-labs.com/news/2001/february/26/1.html

http://cm.bell-labs.com/cm/ms/what/shannonday/paper.html



Claude Shannon

Confusion and Diffusion

- **Confusion (substitution) :**
  - **The ciphertext statistics should depend on the plaintext statistics in a manner too complicated to be exploited by the enemy cryptanalyst**
  - **Makes relationship between ciphertext and key as complex as possible**

- **Diffusion (permutation) :**
  - **Each digit of the plaintext should influence many digits of the ciphertext, and/or**
  - **Each digit of the secret key should influence many digits of the the ciphertext.**
  - **Dissipates statistical structure of plaintext over bulk of ciphertext**

Kerckhoff's Principle

- **Auguste Kerckhoff, 1883**
  - **A cryptosystem should be secure even if everything about the system, except the key, is public knowledge.**
  - **Eric Raymond extends this principle in support of open source software, saying "Any security software design that doesn't assume the enemy possesses the source code is already untrustworthy; therefore, never trust closed source".**
  - **The majority of civilian cryptography makes use of publicly-known algorithms. By contrast, ciphers used to protect classified government or military information are often kept secret**

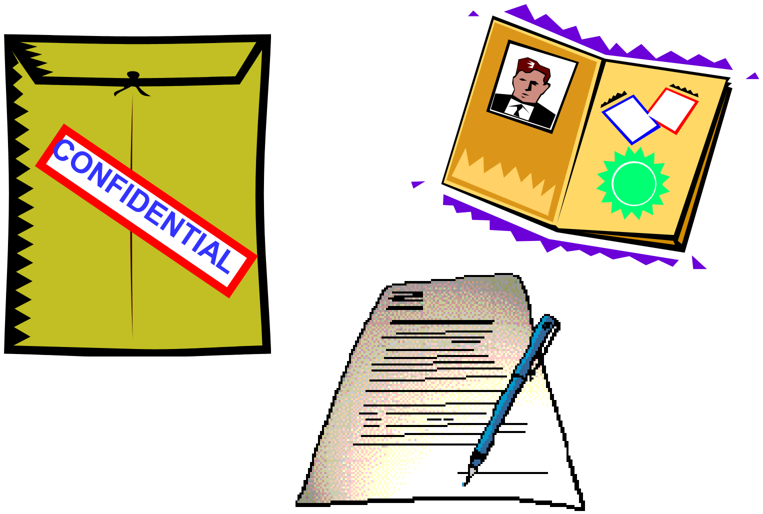
3. Modern Cryptography

3.1 Overview
3.2 Block Cipher
3.3 Public Key Encryption
3.4 Digital Signature
3.5 Hash Function

3.1 Overview

- ❖ Cryptography is a basic tool to implement information security
- ❖ Security goals
  - ❖ Secrecy (confidentiality)
  - ❖ Authentication
  - ❖ Integrity
  - ❖ Non-repudiation
  - ❖ Verifiability
  - ❖ More application-specific security goals
- ❖ Achieve these security goals using cryptography
  - ❖ Without cryptography …. ???

Physical vs. Cryptographic Solutions



Physical Solutions

- Temper-evident sealed envelope
- ID-card, Passport, Drivers license
- Signature

Cryptographic Solutions (for communications over open network)

- **Encryption with MAC :** Confidentiality, Authentication, Integrity Protection
- **Digital Certificate :** Identification
- **Digital Signature :** Authentication, Integrity Protection, Non-Repudiation
- **Security mechanisms are combined to provide a security service**
  - ✓ **VPN, Firewall, IDS, etc.**

Cryptology = Cryptography + Cryptanalysis

- ❖ **Cryptography(암호설계) : designing secure cryptosystems**
  - ❖ **Cryptography (from the Greek kryptós and gráphein, "to write") was originally the study of the principles and techniques by which information could be concealed in ciphers and later revealed by legitimate users employing the secret key.**

- ❖ **Cryptanalysis(암호해독) : analyzing the security of cryptosystems**
  - ❖ **Cryptanalysis (from the Greek kryptós and analýein, "to loosen" or "to untie") is the science (and art) of recovering or forging cryptographically secured information without knowledge of the key.**

- ❖ **Cryptology(암호학) : science dealing with information security**
  - ❖ **Science concerned with data communication and storage in secure and usually secret form. It encompasses both cryptography and cryptanalysis.**

Common Terms

- Cipher: Block cipher, Stream cipher, Public key cipher
- Plaintext/Cleartext (평문), Ciphertext (암호문)
- Encryption/Encipherment(암호화)
- Decryption/Decipherment(복호화)
- Key (or Cryptographic key)
  - Secret key
  - Private key / Public key
- Hashing (해쉬)
- Authentication (인증)
  - Message authentication
  - User authentication
- Digital signature (전자서명)

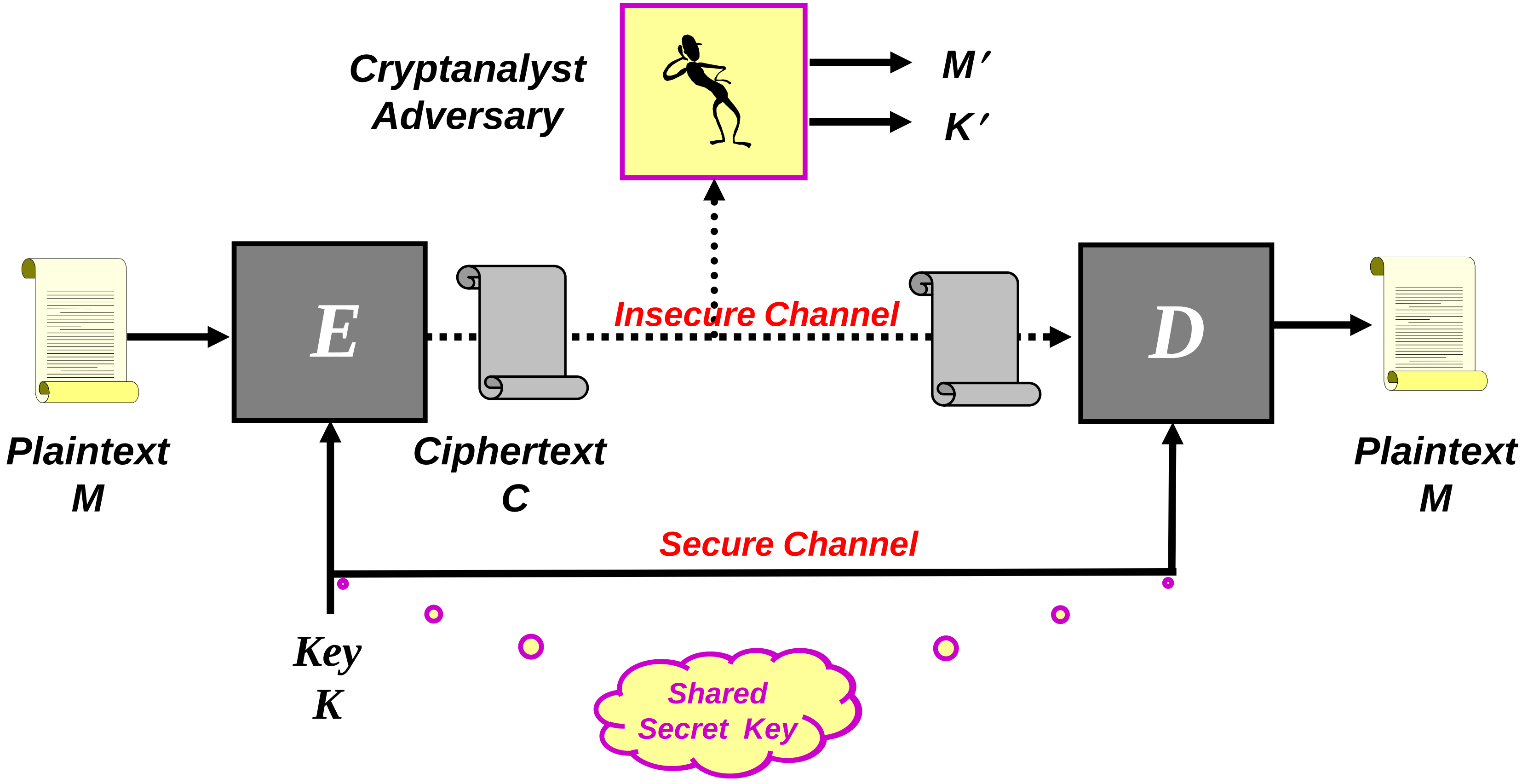
Cryptographic Primitives

- Block / Stream Cipher
  - 3DES, AES, SEED, RC2, RC5, … / RC4, SEAL …
- Hash Function
  - MD5, SHA1/SHA2, HAS160, RMD160, Tiger …
- Message Authentication Code (MAC)
  - HMAC, CBC-MAC, UMAC

- Public Key Encryption
  - RSA, ElGamal
- Digital Signature
  - RSA, DSA/ECDSA, KCDSA/EC-KCDSA …
- Key Exchange
  - Diffie-Hellman, ECDH, RSA key transport

3.2 Block Cipher

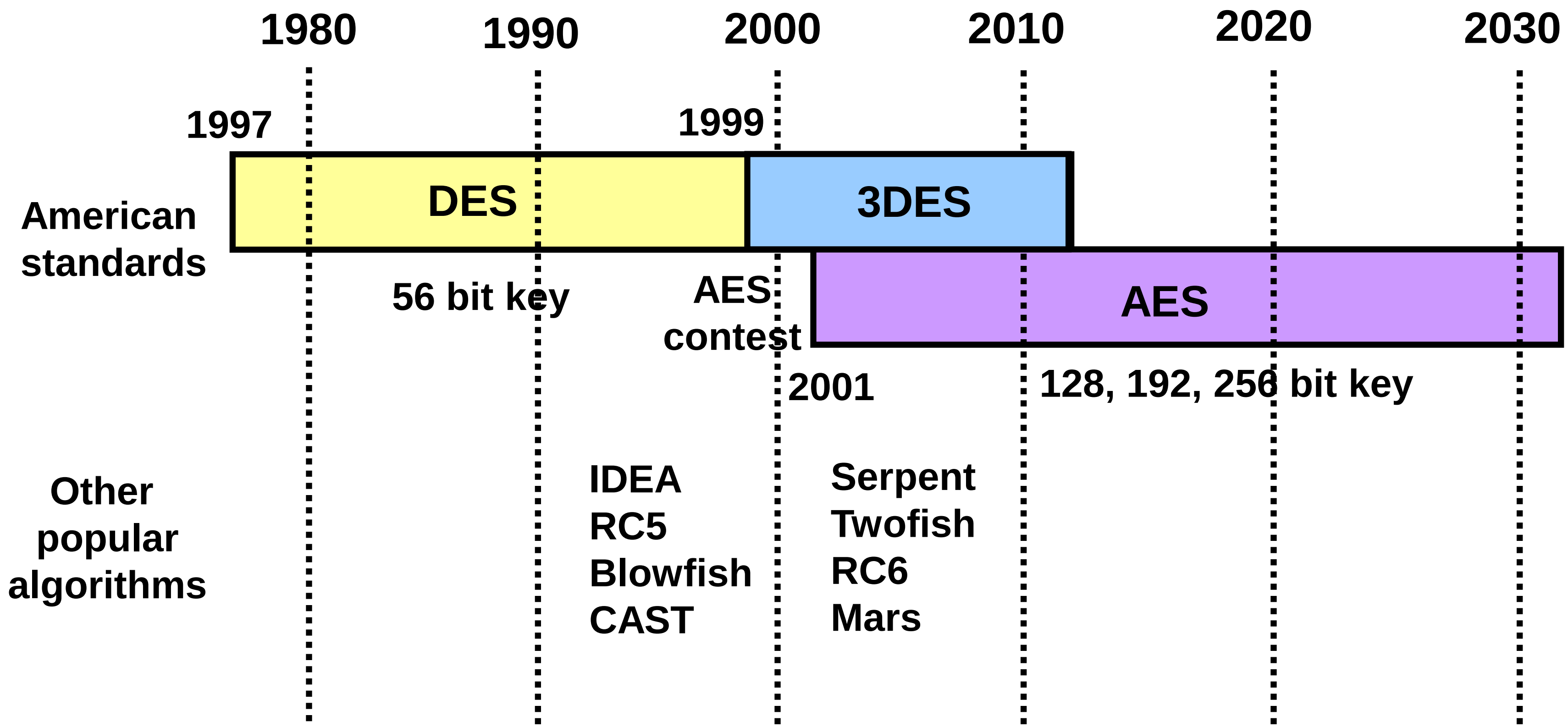
Symmetric Encryption Model



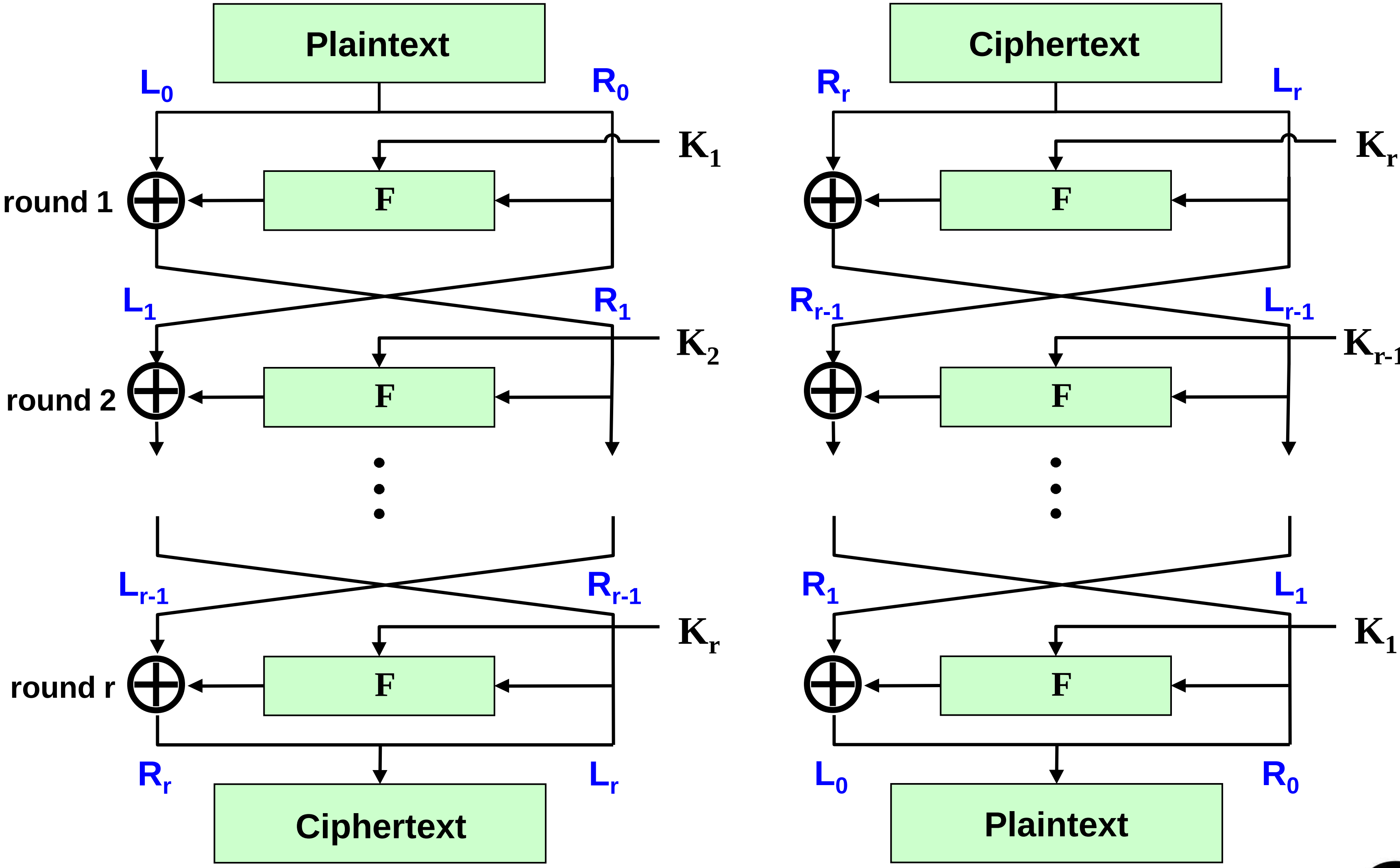
$C = E_K(M)$

$D_K(C) = M$

History of Symmetric Ciphers



Block Cipher Architecture : Feistel-type



Design of Feistel-type Ciphers

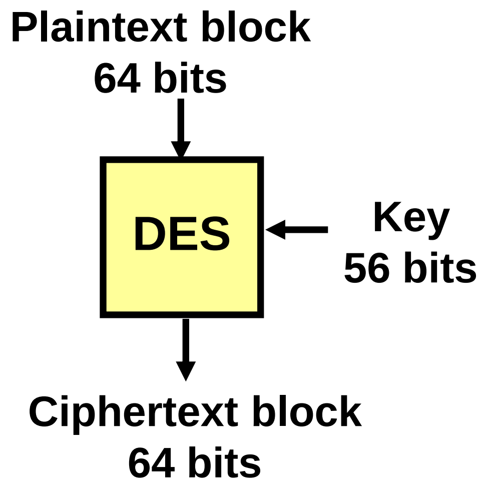
- **Design of F-function**
  - ✓ The only non-linear part in the Feistel-type cipher
  - ✓ Need not be invertible
  - ✓ Typically uses S-boxes (Substitution boxes) for non-linearity
  - ✓ May also contain mixing (permutation) part of the S-box outputs
  - ✓ Determines the ultimate security
- **Design of Key scheduling algorithm**
  - ✓ Algorithm for deriving as many round keys as necessary from a fixed user key
  - ✓ On-the-fly vs. off-line calculation
- **Number of rounds**
  - ✓ Depends on the strength of round function (F-function)
  - ✓ A safety margin should be considered for long-term security
  - ✓ Determined through the analysis of the whole algorithm against most powerful known cryptanalysis techniques

Data Encryption Standard (DES)

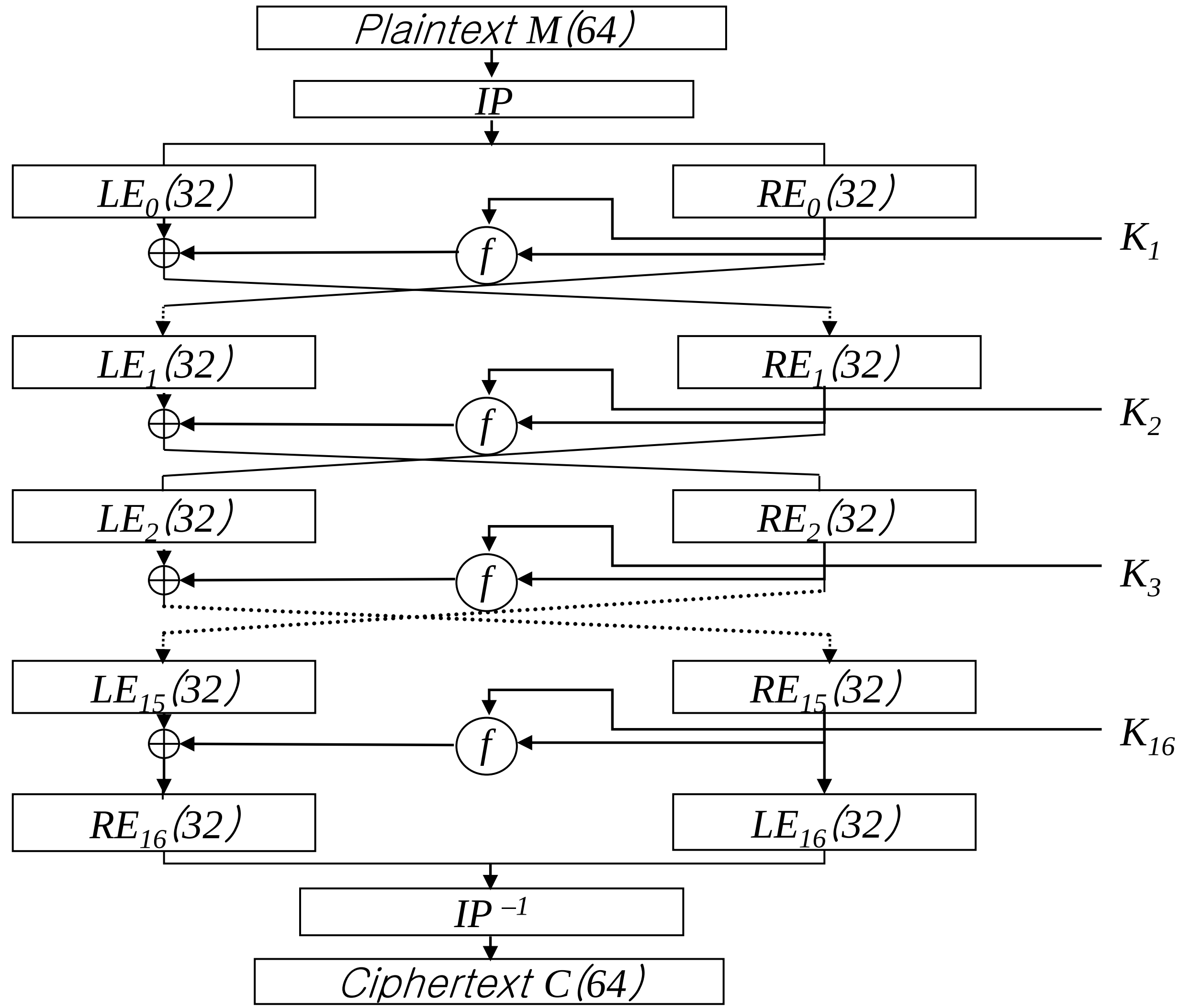
- **DES - History**
  - ✓ 1976 – adopted as a federal standard
  - ✓ 1977 – official publication as FIPS PUB 46
  - ✓ 1983, 1987, 1993 – recertified for another 5 years

* Federal Information Processing Standards

- **Design Criteria of DES**
  - ✓ Provide a high level of security
  - ✓ Completely specify and easy to understand
  - ✓ Security must depend on hidden key, not algorithm
  - ✓ Available to all users
  - ✓ Adaptable for use in diverse applications
  - ✓ Economically implementable in electronic device
  - ✓ Able to be validated
  - ✓ Exportable



DES Overview



Initial Permutation and Final Permutation

IP (Initial permutation)

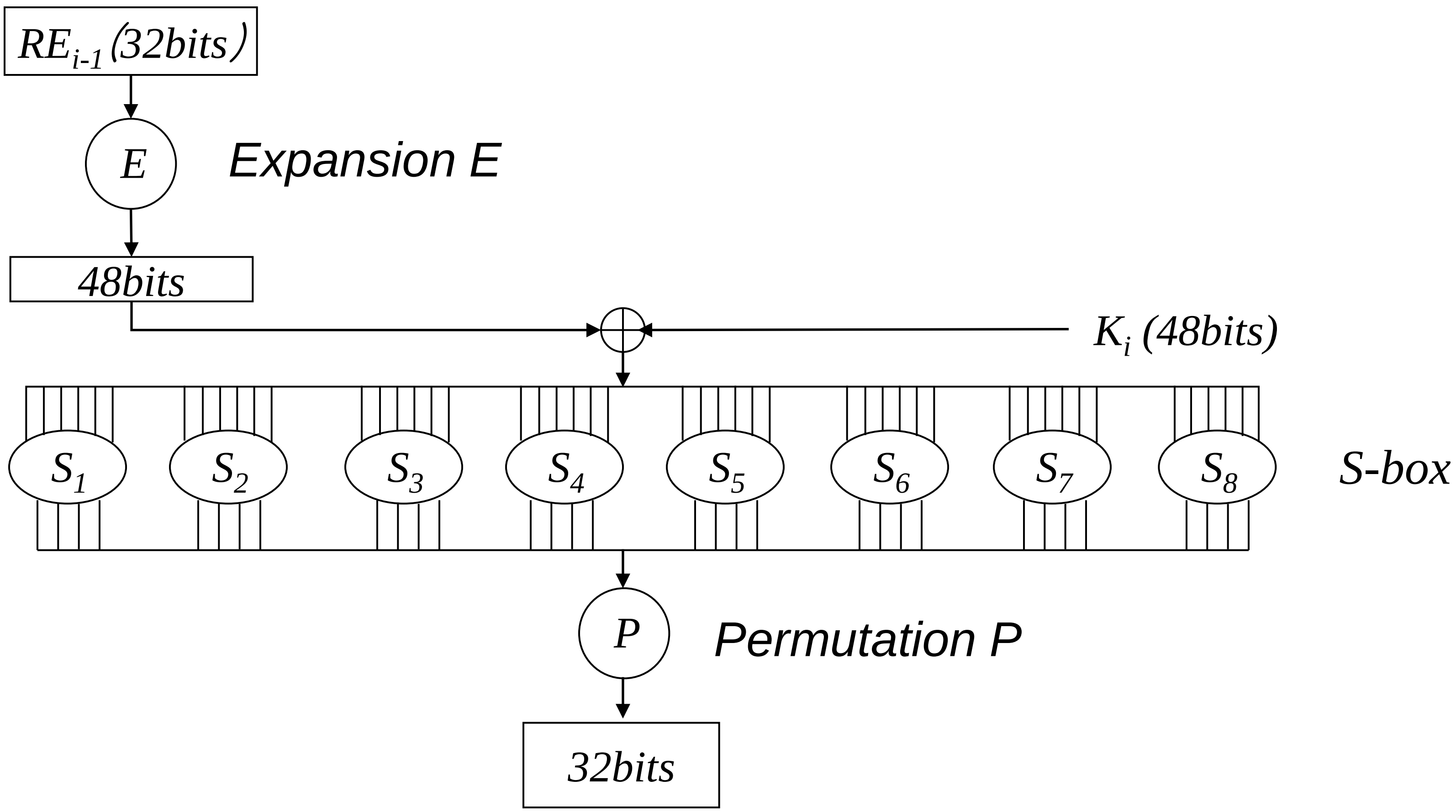
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

IP $^{-1}$ (Final permutation)

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

cf.) The 58th bit of x is the first bit of IP(x)

Function $f(k_i, RE_{i-1})$



Expansion E and Permutation P

Expansion E

32	1	2	3	4	5
4	5	6	7	8	9
8	9	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	30	31	32	1

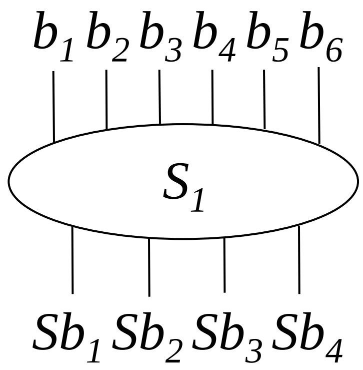
cf.) 32-bits are expanded into 48-bits.
Some bits are selected more than once.

Permutation P

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

32-bit → 32-bit
permutation

S-box (substitution box)



Look-up a value from the table using

$b_1 b_6$: *row*

$b_2 b_3 b_4 b_5$: *column*

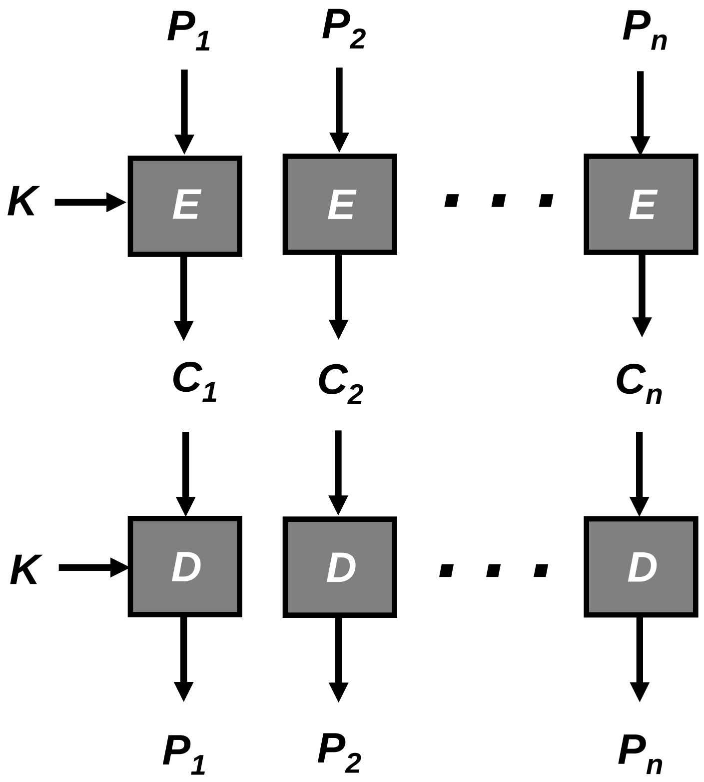
$b_1 b_6$: *row*

S_1-box table

$b_1 b_6$ \ $b_2 b_3 b_4 b_5$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	Sb_1															
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

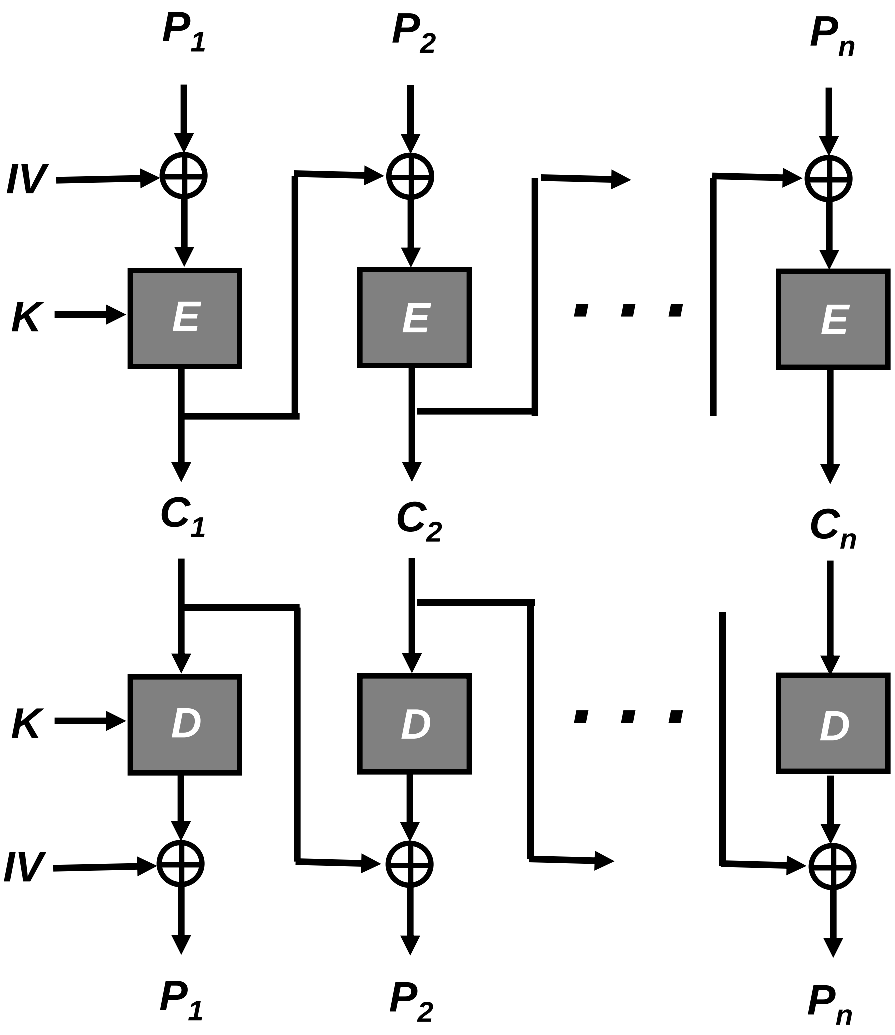
$b_2 b_3 b_4 b_5$: *column*

Modes of Operation – ECB Mode



- **Electronic Code Book Mode**
  - ✓ Break a message into a sequence of plaintext blocks
  - ✓ Each plaintext block is encrypted (or decrypted) independently
  - ✓ The same plaintext block always produces the same ciphertext block
  - ✓ May not be secure; e.g., a highly structured message
  - ✓ Typically used for secure transmission of single vales (e.g., encryption key)

Modes of Operation – CBC Mode



- **Cipher Block Chaining Mode**
  - ✓ **Each ciphertext block is affected by previous blocks**
  - ✓ **No fixed relationship between the plaintext block and its input to the encryption function**
  - ✓ **The same plaintext block, if repeated, produces different ciphertext blocks**
  - ✓ **IV(Initializing Vector) must be known to both ends**
  - ✓ **Most widely used for block encryption**

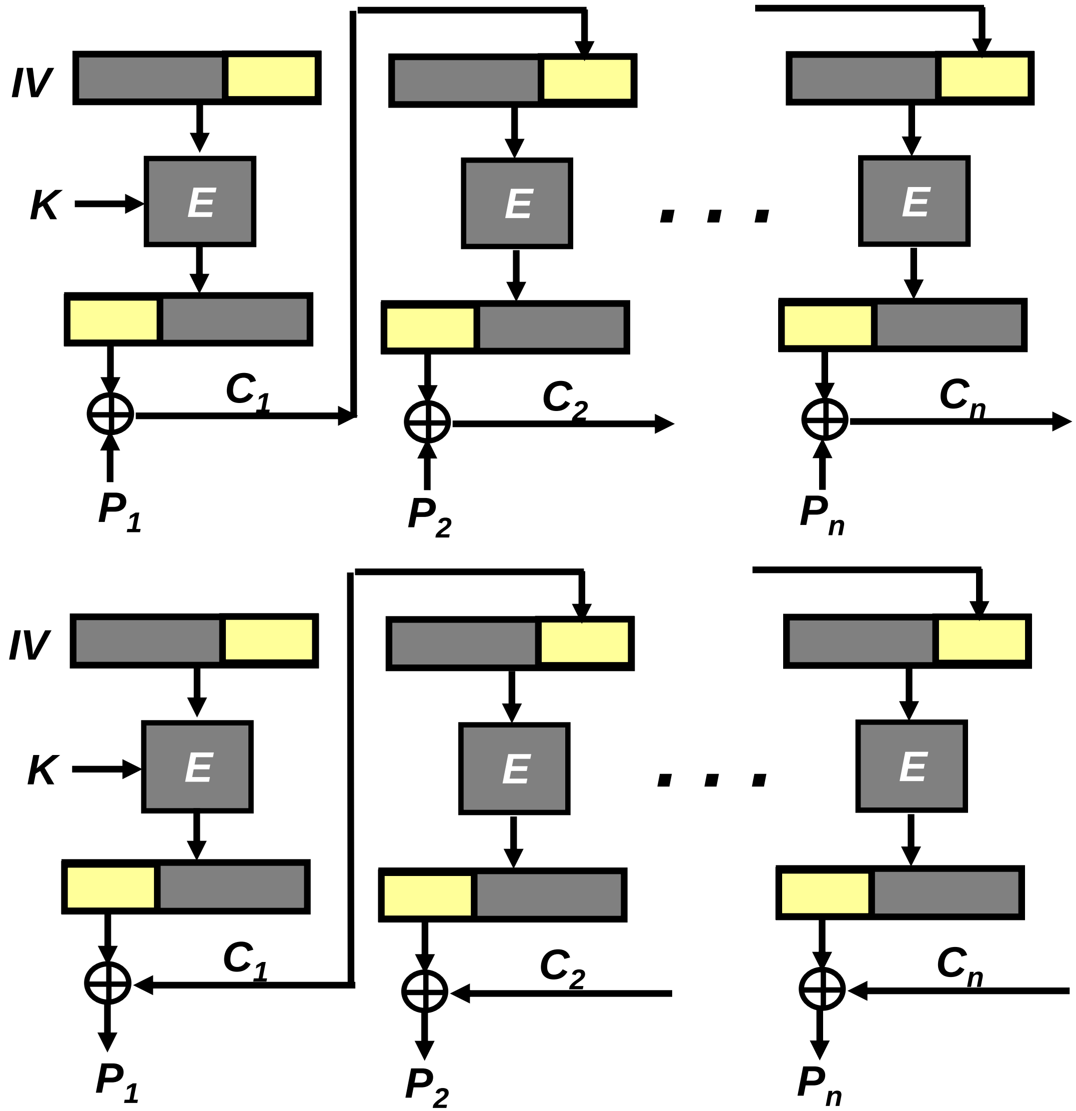
$C_1 = E_K(P_1 \oplus IV)$ $\quad$ $C_3 = E_K(P_3 \oplus C_2)$

$P_1 = IV \oplus D_K(C_1)$ $\quad$ $P_3 = C_2 \oplus D_K(C_3)$

$C_2 = E_K(P_2 \oplus C_1)$ $\quad$ $C_4 = E_K(P_4 \oplus C_3)$

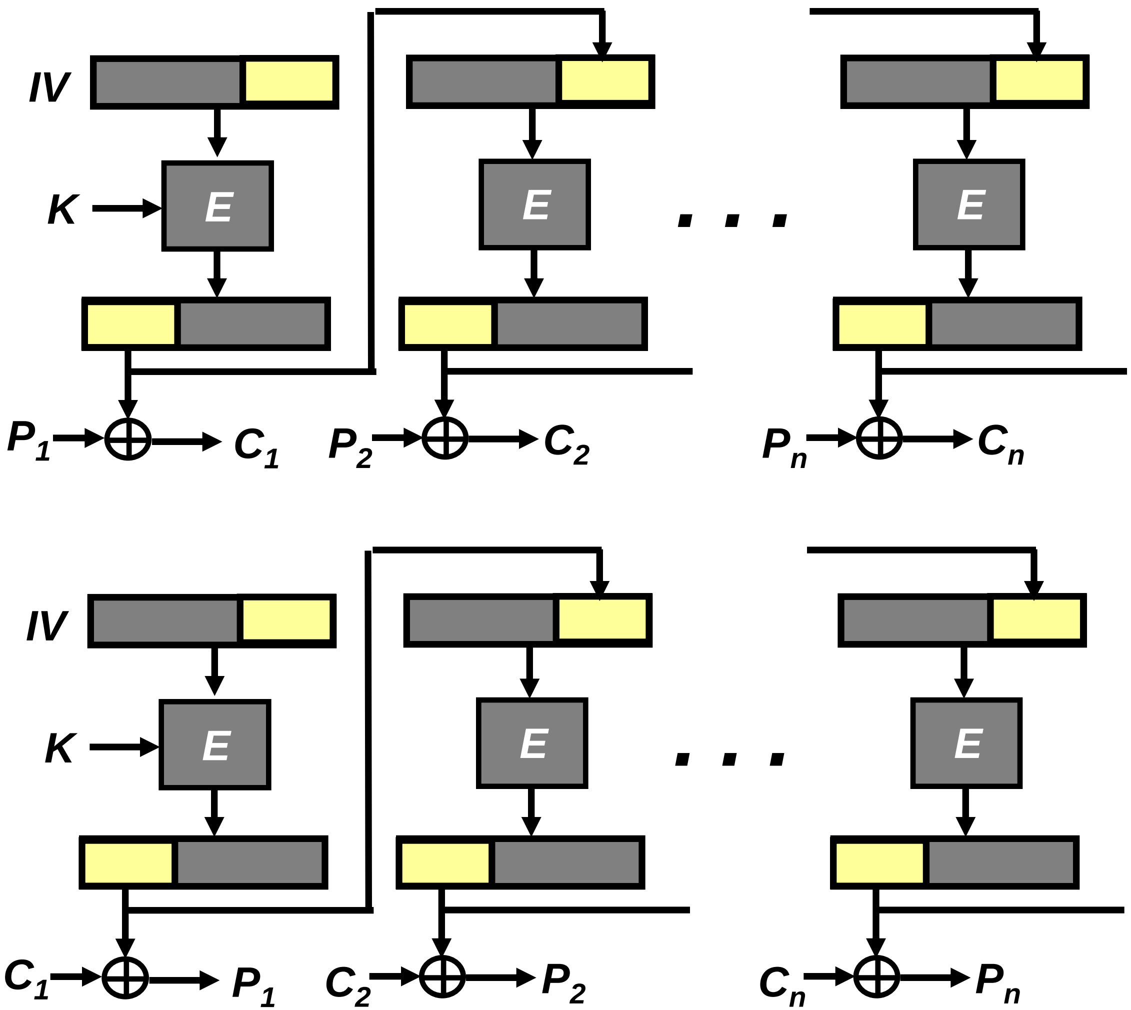
$P_2 = C_1 \oplus D_K(C_2)$ $\quad$ $P_4 = C_3 \oplus D_K(C_4)$

Modes of Operation – CFB Mode



- **Cipher Feedback Mode**
  - ✓ A way of using a block cipher as a stream cipher
  - ✓ A shift register of block size maintains the current state of the cipher operation, initially set to some IV
  - ✓ The value of the shift register is encrypted using key K and the leftmost j bits of the output is XORed with j-bit plaintext P_i to produce j-bit ciphertext C_i
  - ✓ The value of the shift register is shifted left by j bits and the C_i is fed back to the rightmost j bits of the shift register
  - ✓ Typically j = 8, 16, 32, 64 …
  - ✓ Decryption function D_K is never used

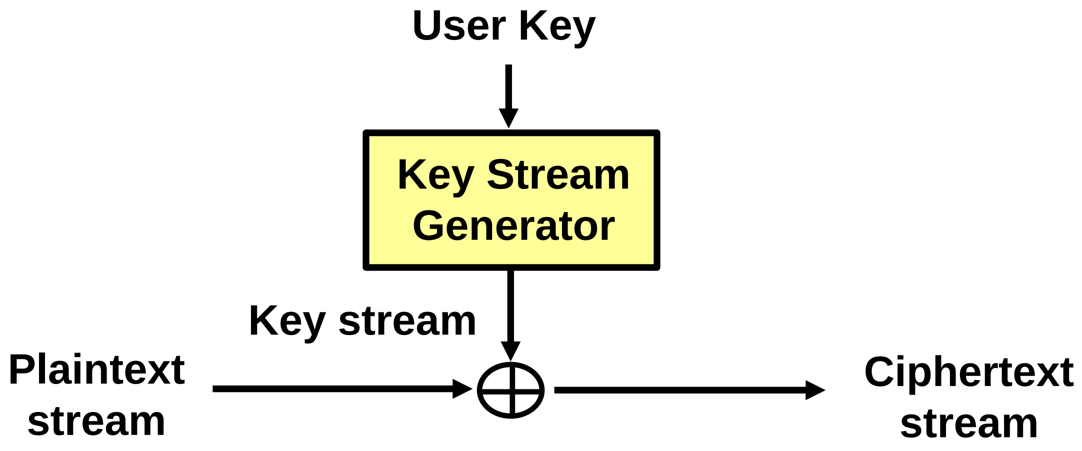
Modes of Operation – OFB Mode



- **Output Feedback Mode**
  - ✓ **The structure is similar to that of CFB, but**
    - **CFB: Ciphertext is fed back to the shift register**
    - **OFB: Output of E is fed back to the shift register**
  - ✓ **For security reason, only the full feedback (j = block size) mode is used**
  - ✓ **No error propagation**
  - ✓ **More vulnerable to a message stream modification attack**
  - ✓ **May useful for secure transmission over noisy channel (e.g., satellite communication)**

Stream Cipher

- Plaintext stream is bit-by-bit XORed with key stream to generate ciphertext stream
- The encryption function may vary as plaintext is processed; stateful
- Encryption depends not only on the key and plaintext, but also on the current state
- Advantages
  - No need for padding: the length of ciphertext = the length of plaintext
  - Real-time transmission: encrypt and transmit character by character



Binary Additive Stream Cipher

Stream Cipher - Example

- *RC4*
- *SEAL*

Plaintext		0	1	1	0	1	0	0	1	1	1
Key Stream	$\oplus$	1	1	0	0	1	0	0	0	1	0
Ciphertext		1	0	1	0	0	0	0	1	0	1
Key Stream	$\oplus$	1	1	0	0	1	0	0	0	1	0
Plaintext		0	1	1	0	1	0	0	1	1	1

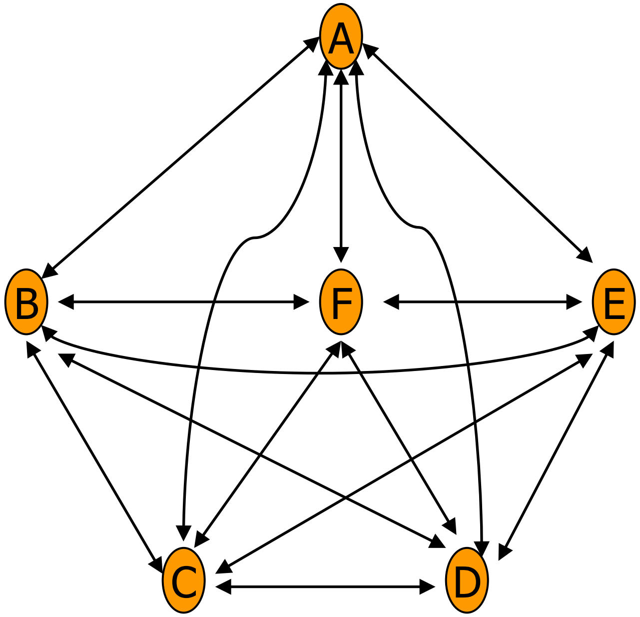
- **Extremely fast Encryption/Decryption; Easy to implement in HW**
  BUT
- **Vulnerable to traffic analysis (|Plaintext| = |Ciphertext|)**
- **Vulnerable to various attacks without integrity check**
- **Using two Ciphertexts from the same key stream,**
  we can recover the XOR of the Plaintexts
  $\Rightarrow$ **NEVER reuse a key stream !!**

Stream Cipher vs. Block Cipher

- **Stream Cipher**
  - ✓ Encrypt individual character (often 1 bit)
  - ✓ Have memory; stateful cipher
  - ✓ Generally extremely faster than block ciphers
  - ✓ Suitable for multimedia streaming data (audio, video)
  - ✓ Limited / No error propagation
  - ✓ Problem : Re-sync. of key stream generator state
  - ✓ Problem : insertion/deletion, replay of ciphertext digits
    - → Need additional Integrity Check
- **Block Cipher**
  - ✓ Encrypt simultaneously a group of characters (64 / 128 bits)
    - → Need Padding
  - ✓ Memoryless
  - ✓ Substitution Permutation Networks(SPN) or Feistel-type
  - ✓ Various modes of operation : ECB, CBC, OFB, CFB, CTS, etc…

3.3 Public Key Cryptography

Key distribution problem

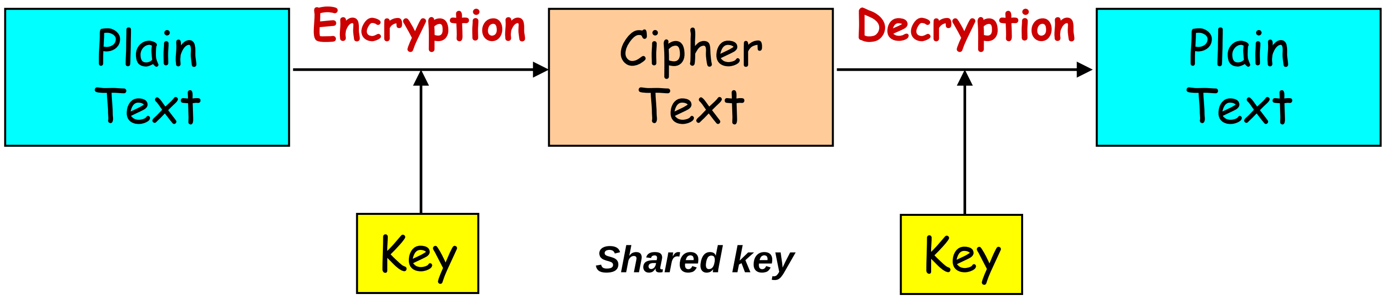


- **Symmetric Key Cryptosystem**
  - ✓ **${}_{n}C_{2}$ = n(n-1)/2 secret keys**
  - ✓ **Each user has n-1 keys**

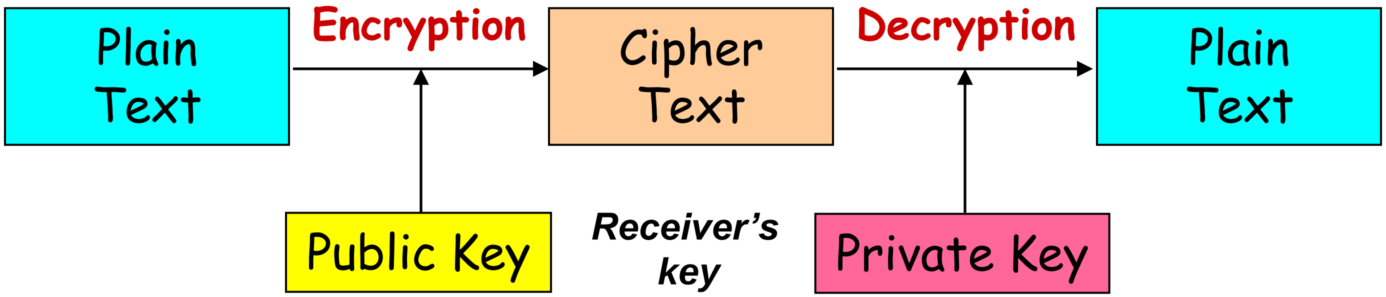
- **Public Key Cryptosystem**
  - ✓ **n private keys**
  - ✓ **Each user has 1 private key**

Symmetric Key vs. Public Key Systems

- Symmetric Key Cryptosystem

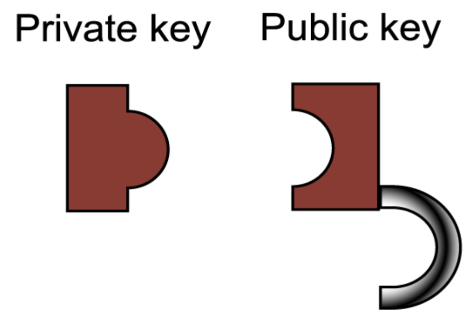


- Public Key Cryptosystem



Public Key Cryptography - Concept

Using a pair of keys which have special mathematical relation.
Each user needs to keep securely only his private key.
All public keys of users are published.



In Encryption

Anyone can lock (using the public key)
Only the receiver can unlock (using the private key)

In Digital Signature

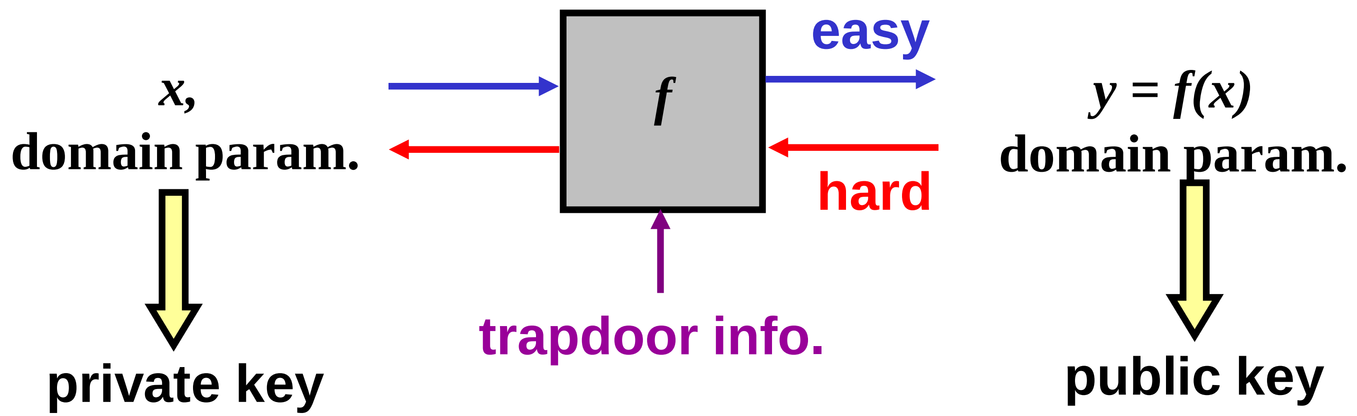
Only the signer can sign (using the private key)
Anyone can verify (using the public key)

Public Key Cryptography

- Keys
  - A pair of (Public Key, Private Key) for each user
  - Public keys must be publicly & reliably available
- Public Encryption
  - Encrypt with peer's Public Key; Decrypt with its own Private Key
  - RSA, ElGamal
- Digital signature
  - Sign with its own Private Key; verify with peer's Public Key
  - RSA, DSA, KCDSA, ECDSA, EC-KCDSA …
- Key exchange
  - Key transport or key agreement for secret-key crypto.
  - RSA; DH(Diffie-Hellman), ECDH

Public Key Cryptography

- Invented by Diffie and Hellman in 1976
- Solve the secret key sharing problem in symmetric cryptosystems
- Two keys used: public key & private key
- Also known as two-key cryptography or asymmetric cryptography
- Based on (trapdoor) one-way function



But, easy if trapdoor info. is given.

Public Key Cryptosystems – History

- RSA scheme (1978)
  - R.L.Rivest, A.Shamir, L.Adleman, “A Method for Obtaining Digital Signatures and Public Key Cryptosystems”,CACM, Vol.21, No.2, pp.120-126,Feb,1978
- McEliece scheme (1978)
- Rabin scheme (1979)
- Knapsack scheme (1979-): Merkle-Hellman, Chor-Rivest
- ElGamal scheme (1985)
- Elliptic Curve Cryptosystem (1985): Koblitz, Miller
- Non-Abelian group Cryptography (2000): Braid group

Integer Factorization Problem (IFP)

- **Problem: Given a composite number n, find its prime factors**

$$\text{Primes } p, q \quad \underset{\text{hard}}{\overset{\text{easy}}{\rightleftarrows}} \quad n = pq$$

- **Application: Used to construct RSA-type public key cryptosystems**

- **Algorithms to solve IFP (probabilistic sub-exponential algorithms)**
  - **Quadratic sieve**
  - **General Number Field Sieve**

Discrete Logarithm Problem (DLP)

- **Problem:**
  Given g, y, and prime p, find an integer x, if any, such that
  $y = g^x \bmod p$ ($x=\log_g y$)

$$\text{Given } g, x, p \xrightarrow{\text{easy}} y = g^x \bmod p$$

$$x = \log_g y \xleftarrow[\text{hard}]{} \text{Given } g, y, p$$

- **Application: Used to construct Diffie-Hellman & ElGamal-type public key systems: DH, DSA, KCDSA …**

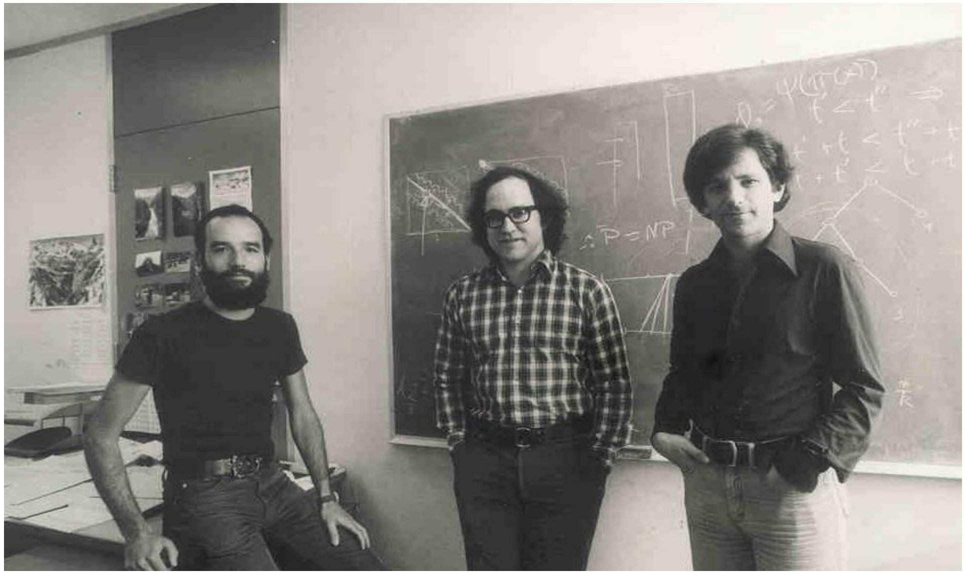
- **Algorithms to solve DLP:**
  - **Shank's Baby Step Giant Step**
  - **Index calculus**

KAIST ICC

KIPO

RSA Public Key Systems

- RSA is the first public key cryptosystem
- Proposed in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman at MIT
- It is believed to be secure and still widely used
- Patent : *US Patent 4,405,829, expired on 21 September 2000*



Shamir *Rivest* *Adleman*

RSA Public Key Systems

- **Key generation**
  - Choose two large (512 bits or more) primes p & q
  - Compute modulus $n = pq$, and $\phi(n) = (p-1)(q-1)$
  - Pick an integer e relatively prime to $\phi(n)$, $\gcd(e, \phi(n))=1$
  - Compute d such that $ed = 1 \bmod \phi(n)$
  - Public key (n, e) : publish
  - Private key d : keep secret (may discard p & q)
- **Encryption / Decryption**
  - E: $c = m^e \bmod n$ for $0 < m < n$
  - D: $m = c^d \bmod n$
  - Proof) $C^d = (M^e)^d = M^{ed} = M^{k\phi(n)+1} = M\{M^{\phi(n)}\}^k = M$

KAIST ICC

KIPO

RSA Public Key Systems

❖ ***Example:***

Key Generation

- $p=3$, $q=11$
- $n = pq = 33$, $\phi(n) = (p-1)(q-1) = 2 \times 10 = 20$
- $e = 3$ s.t. $\gcd(e, \phi(n)) = (3,20) = 1$
- Choose d s.t. $ed = 1 \bmod \phi(n)$, $3d = 1 \bmod 20$, $d=7$
- Public key $=\{e,n\}=\{3,33\}$, private key $=\{d\}=\{7\}$

Encryption

- $M = 5$
- $C = M^e \bmod n = 5^3 \bmod 33 = 26$

Decryption

- $M = C^d \bmod n = 26^7 \bmod 33 = 5$

RSA Challenge Solution

RSA-160

Date: Tue, 1 Apr 2003 14:05:10 +0200
From: Jens Franke
Subject: RSA-160

We have factored RSA160 by gnfs. The prime factors are:
p=45427892858481394071686190649738831\ 656137145778469793250959984709250004157335359
q=47388090603832016196633832303788951\ 973268922921040957944741354648812028493909367

http://www.loria.fr/~zimmerma/records/rsa160

RSA-200

Date: Mon, 9 May 2005 18:05:10 +0200 (CEST)
From: Thorsten Kleinjung
Subject: rsa200

We have factored RSA200 by GNFS. The factors are
p=35324619344027701212726049781984643686711974001976\ 25023649303468776121253679423200058547956528088349
and
q=79258699544783330333470858414800596877379758573642\ 19960734330341455767872818152135381409304740185467

http://www.loria.fr/~zimmerma/records/rsa200

Diffie-Hellman / ElGamal-type Systems

- **Domain parameter generation**
  - **Based on the hardness of DLP**
  - **Generate a large (1024 bits or more) prime p**
  - **Find a generator g that generates the cyclic group Z_p^***
  - **Domain parameter = {p, g}**

- **Key generation**
  - **Pick a random integer $x \in [1, p-1]$**
  - **Compute $y = g^x \bmod p$**
  - **Public key (p, g, y) : publish**
  - **Private key x : keep secret**

- **Applications**
  - **Public key encryption**
  - **Digital signatures**
  - **Key agreement**

ElGamal Encryption Scheme

- **Keys & parameters**
  - Domain parameter = {p, g}
  - Choose $x \in [1, p-1]$ and compute $y = g^x \bmod p$
  - Public key (p, g, y)
  - Private key x
- **Encryption**: $m \rightarrow (C_1, C_2)$
  - Pick a random integer $k \in [1, p-1]$
  - Compute $C_1 = g^k \bmod p$
  - Compute $C_2 = m \times y^k \bmod p$
- **Decryption**
  - $m = C_2 \times C_1^{-x} \bmod p$
  - $C_2 \times C_1^{-x} = (m \times y^k) \times (g^k)^{-x} = m \times (g^x)^k \times (g^k)^{-x} = m \bmod p$

ElGamal Encryption Scheme -- Example

- **Key Generation**
  - **Let p=23, g=7**
  - **Private key x=9**
  - **Public key $y = g^x \bmod p = 7^9 \bmod 23 = 15$**

- **Encryption: $m \rightarrow (C_1, C_2)$**
  - **Let m=20**
  - **Pick a random number k=3**
  - **Compute $C_1 = g^k \bmod p = 7^3 \bmod 23 = 21$**
  - **Compute $C_2 = m \times y^k \bmod p = 20 \times 15^3 \bmod 23 = 20 \times 17 \bmod 23 = 18$**
  - **Send $(C_1, C_2) = (21,18)$ as a ciphertext**

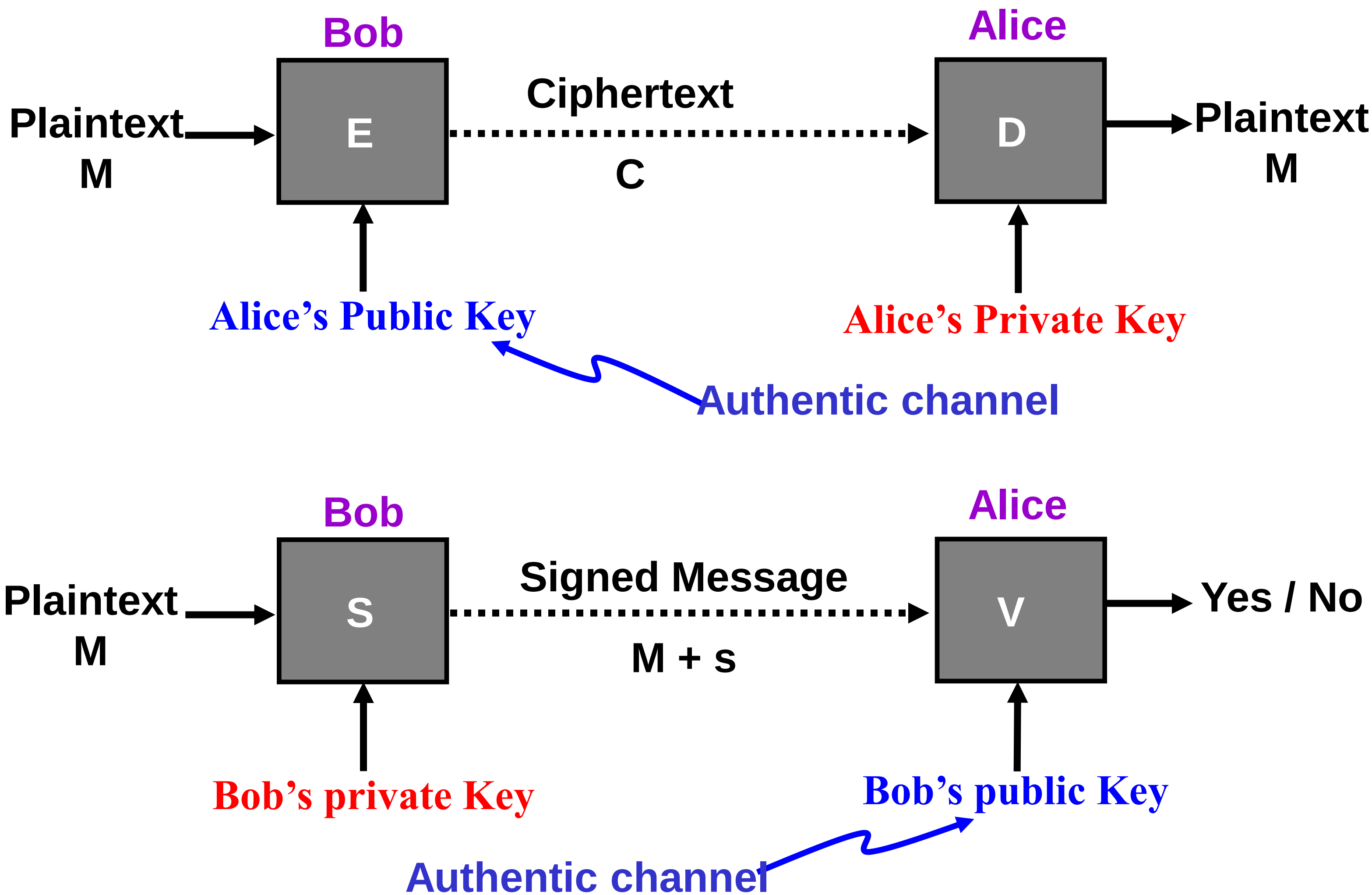
- **Decryption**
  - **$m = C_2 / C_1^x \bmod p = 18 / 21^9 \bmod 23 = 18 / 17 \bmod 23 = 20$**

3.4 Digital Signature

- ❖ **Digital Signature**
  - ➢ **Electronic version of handwritten signature on electronic document**
  - ➢ **Signing using private key (only by the signer)**
  - ➢ **Verification using public key (by everyone)**

- ❖ **Hash then sign: sig(h(m))**
  - ❖ **Efficiency in computation and communication**

Public Key Encryption vs. Signature



Digital Signature

- ❖ Security requirements for digital signature
  - ➢ Unforgeability (위조 방지)
  - ➢ User authentication (사용자 인증)
  - ➢ Non-repudiation (부인 방지)
  - ➢ Unalterability (변조 방지)
  - ➢ Non-reusability (재사용 방지)

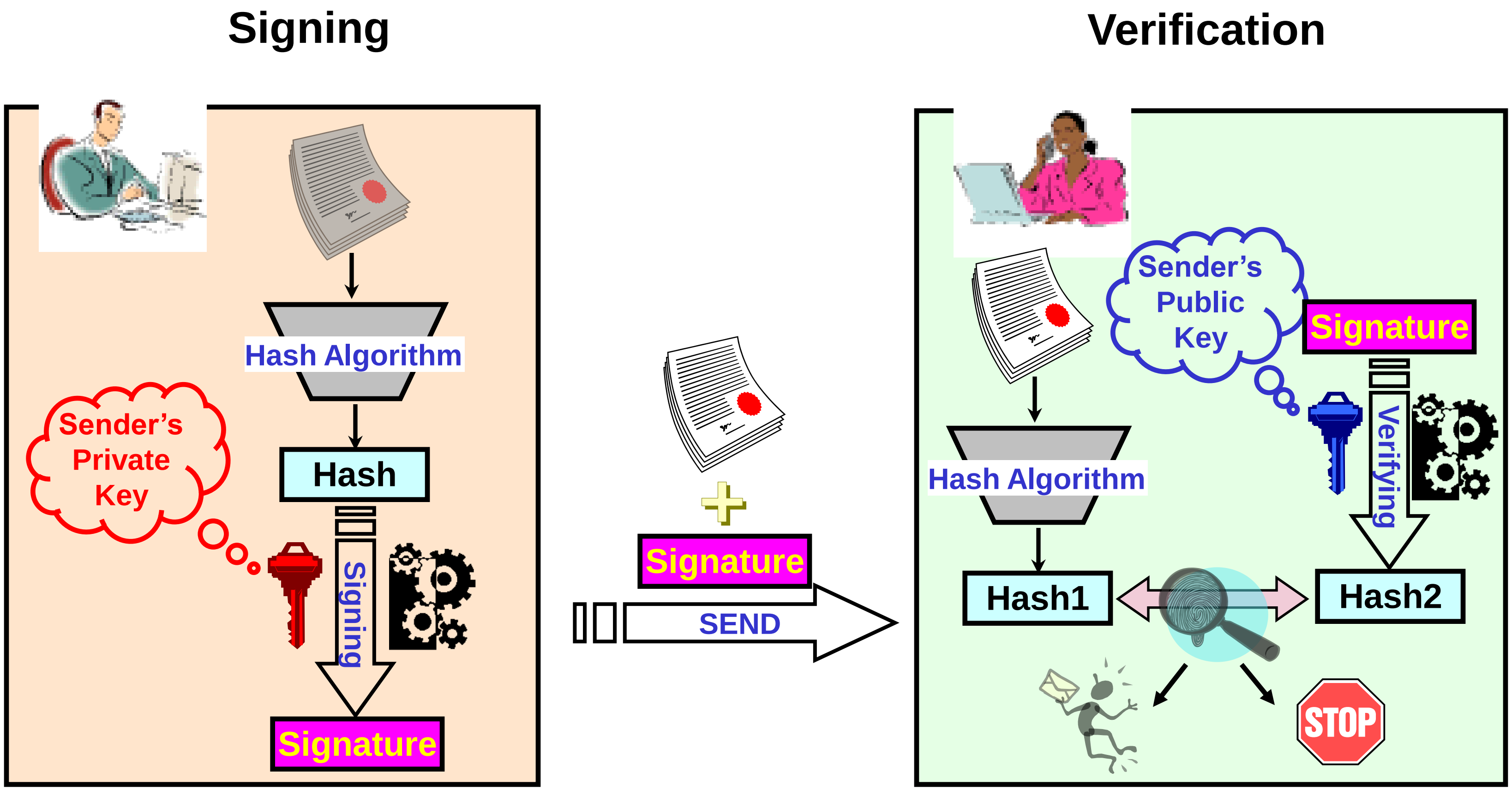
- ❖ Services provided by digital signature
  - ❖ Authentication
  - ❖ Data integrity
  - ❖ Non-Repudiation

RSA Signature

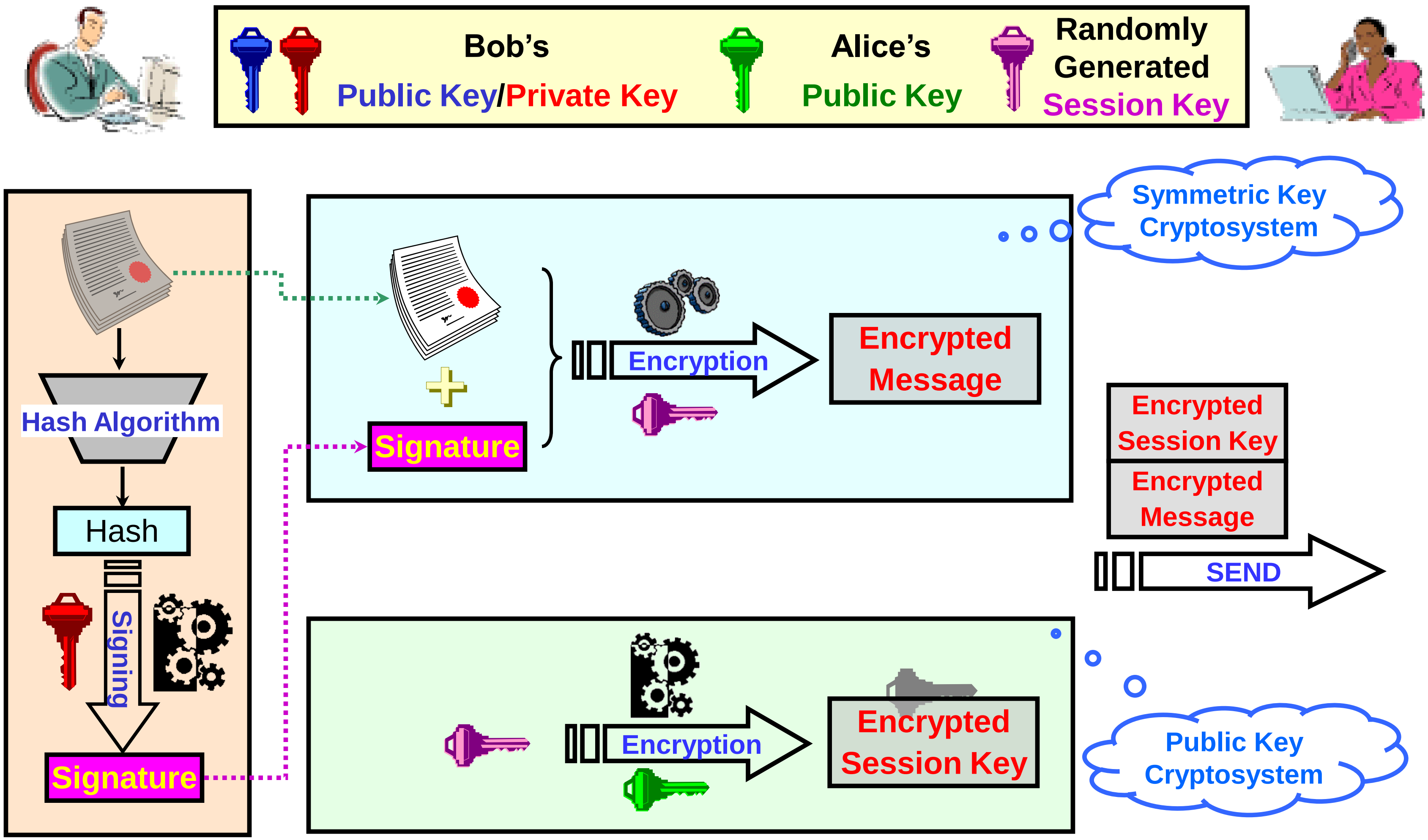
- **Key generation**
  - Choose two large (512 bits or more) primes p & q
  - Compute modulus $n = pq$, and $\phi(n) = (p-1)(q-1)$
  - Pick an integer e relatively prime to $\phi(n)$, $\gcd(e, \phi(n))=1$
  - Compute d such that $ed = 1 \bmod \phi(n)$
  - Public key (n, e) : publish
  - Private key d : keep secret (may discard p & q)

- **Signing / Verifying**
  - S: $s = m^d \bmod n$ for $0 < m < n$
  - V: $m =? s^e \bmod n$
  - S: $s = h(m)^d \bmod n$ --- hashed version
  - V: $h(m) =? s^e \bmod n$

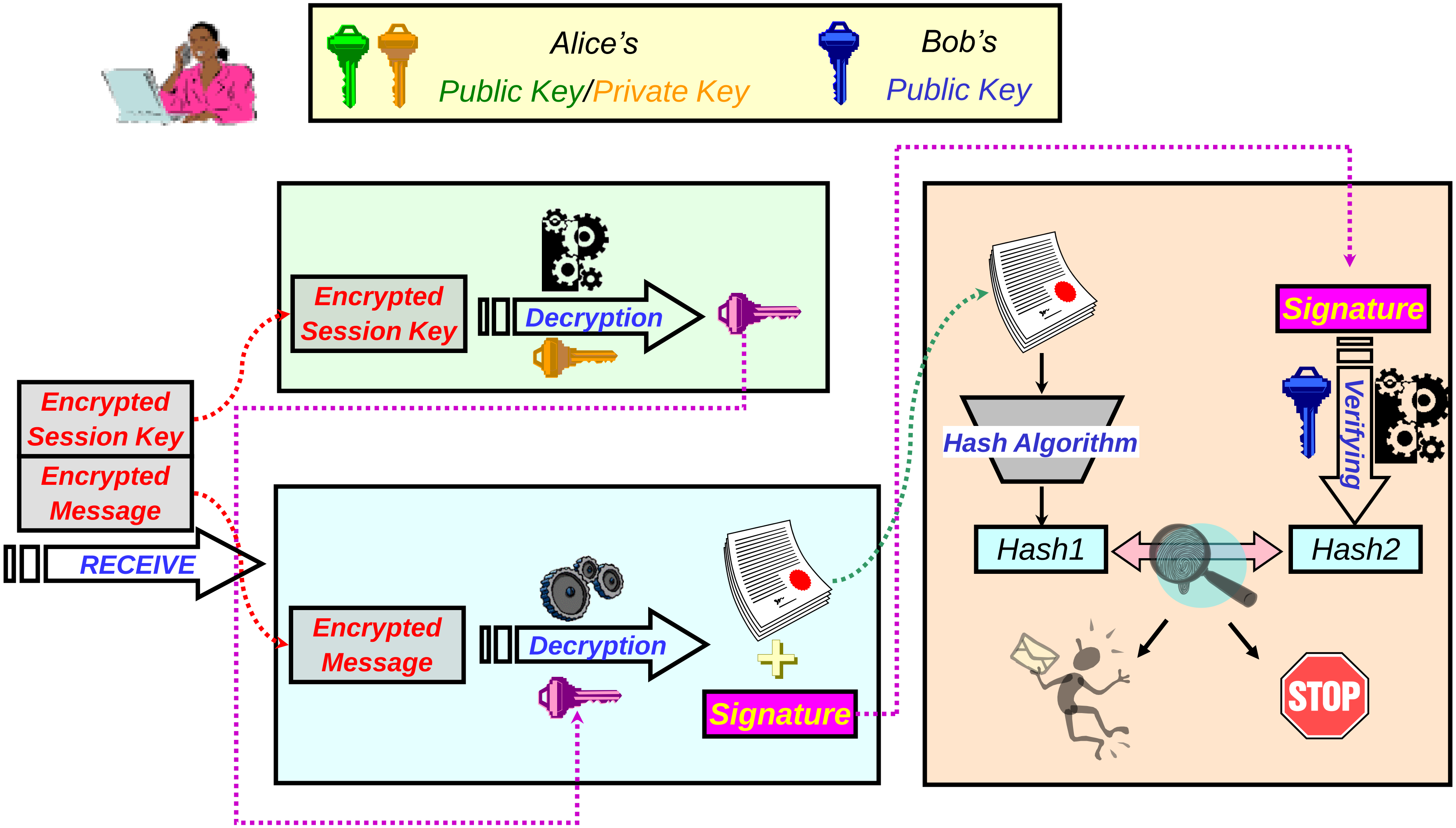
Digital Signature



Digital Enveloping : Key Transport + Encryption



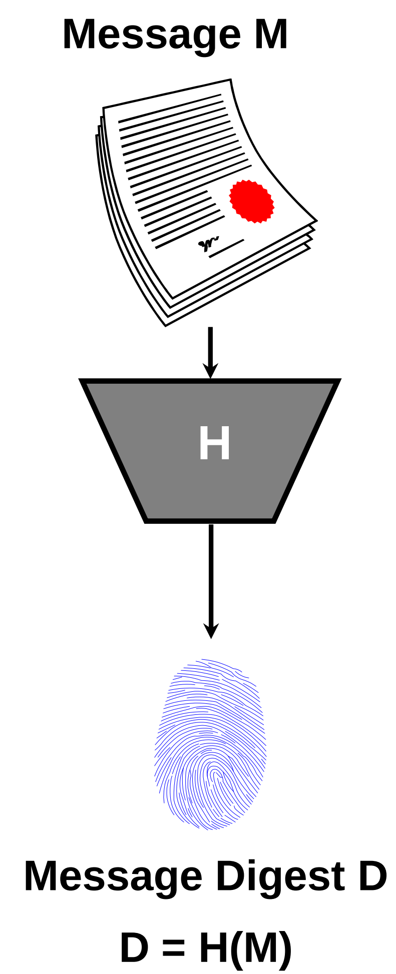
Digital Enveloping : Key Recovery + Decryption



3.5 Hash Functions

- **Hash Function**
  - ✓ **Generate a fixed length "Fingerprint" for an arbitrary Message**
  - ✓ **No Key involved**
  - ✓ **One Way Function**
  - ✓ **MD5, SHA1, SHA2, HAS160**

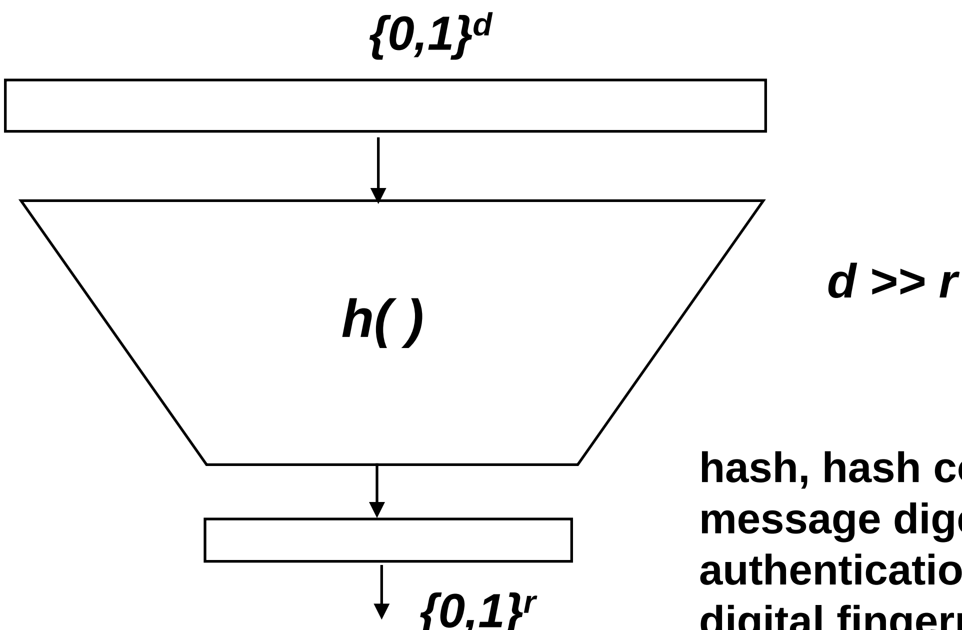
- **Applications**
  - ✓ **Keyed hash: used to generate/verify MAC(Message Authentication Code) or Integrity Check Value(ICV) → HMAC**
  - ✓ **Unkeyed hash: used to produce Digital Signature**



Hash Functions

❖ **Definition**

- **Compression: arbitrary length input to fixed length output**
- **Ease of computation**



hash, hash code/value/result
message digest, checksum, MIC,
authentication tag, seal, compression
digital fingerprint, imprint

**** Note that collision is inevitable (many-to-one function).***

Hash Functions – Requirements

- **Security Properties**
  - **Preimage resistance (One-wayness) :**
    - **Given y, it is computationally infeasible to find any input x such that y = h(x)**
    - **Hardest task, weakest requirement**
  - **2nd preimage resistance (Weak collision resistance) :**
    - **Given x, it is computationally infeasible to find another input $x' \neq x$ such that $h(x) = h(x')$**
  - **Collision resistance (Strong collision resistance) :**
    - **It is computationally infeasible to find any two distinct inputs x and x' such that $h(x) = h(x')$**
    - **Easier task, Strongest requirement**

Birthday Paradox

How many students there must be in a class for there be a greater than 50% chance that

1. **One of the students shares the teacher's birthday ? (complexity breaking one-wayness)**

   $$365/2 \approx 188$$

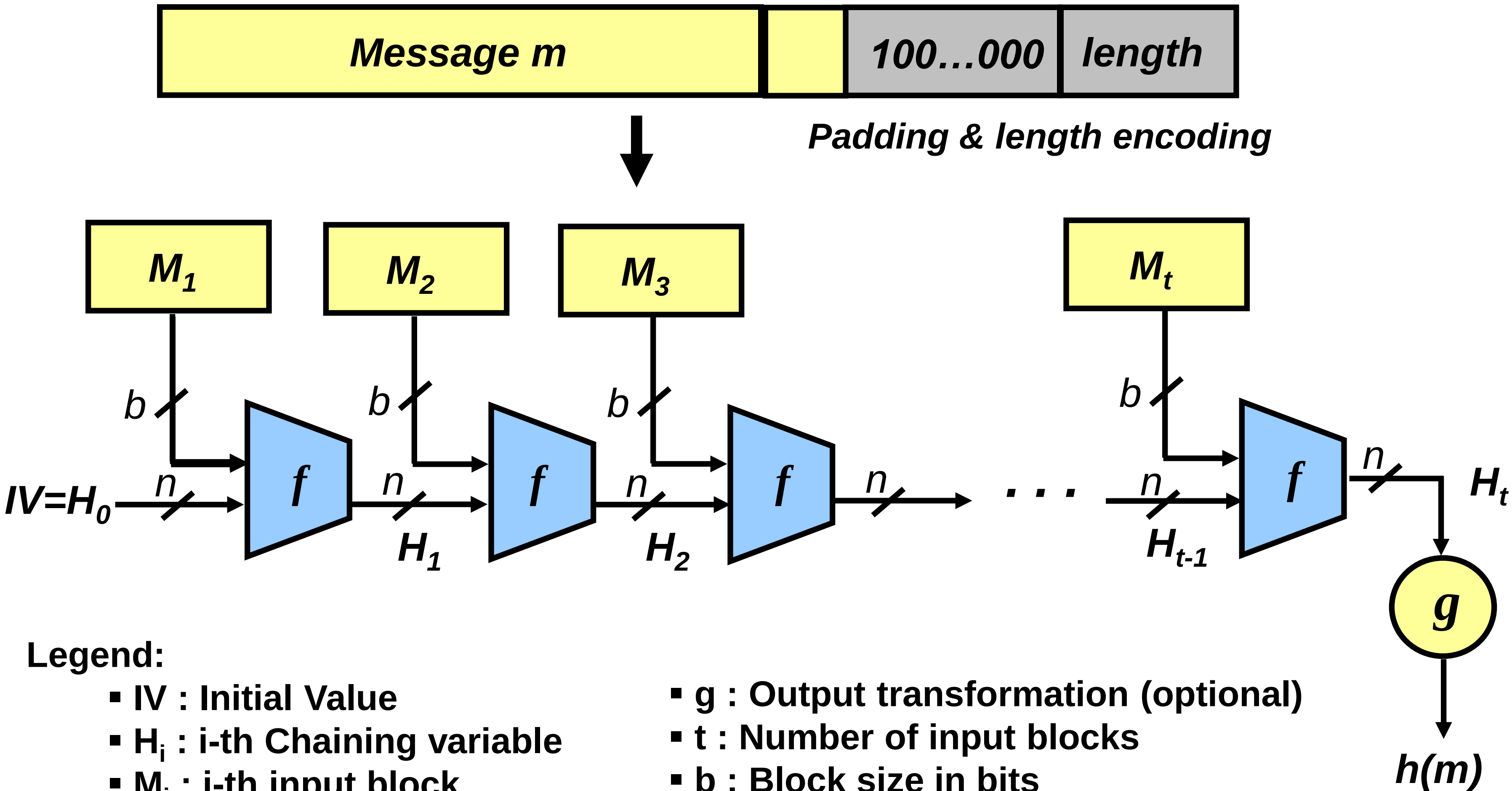
2. **Any two of the students share the same birthday ? (complexity breaking collision resistance)**

   $$1 - 365 \times 364 \times \ldots \times (365-k+1) / 365^k > 0.5 \quad \Rightarrow \quad k \approx 23$$

In general, the probability of a match being found when k samples are randomly selected between 1 and n equals

$$1 - \frac{n!}{(n-k)!n^k} > 1 - e^{-\frac{k(k-1)}{2n}}$$

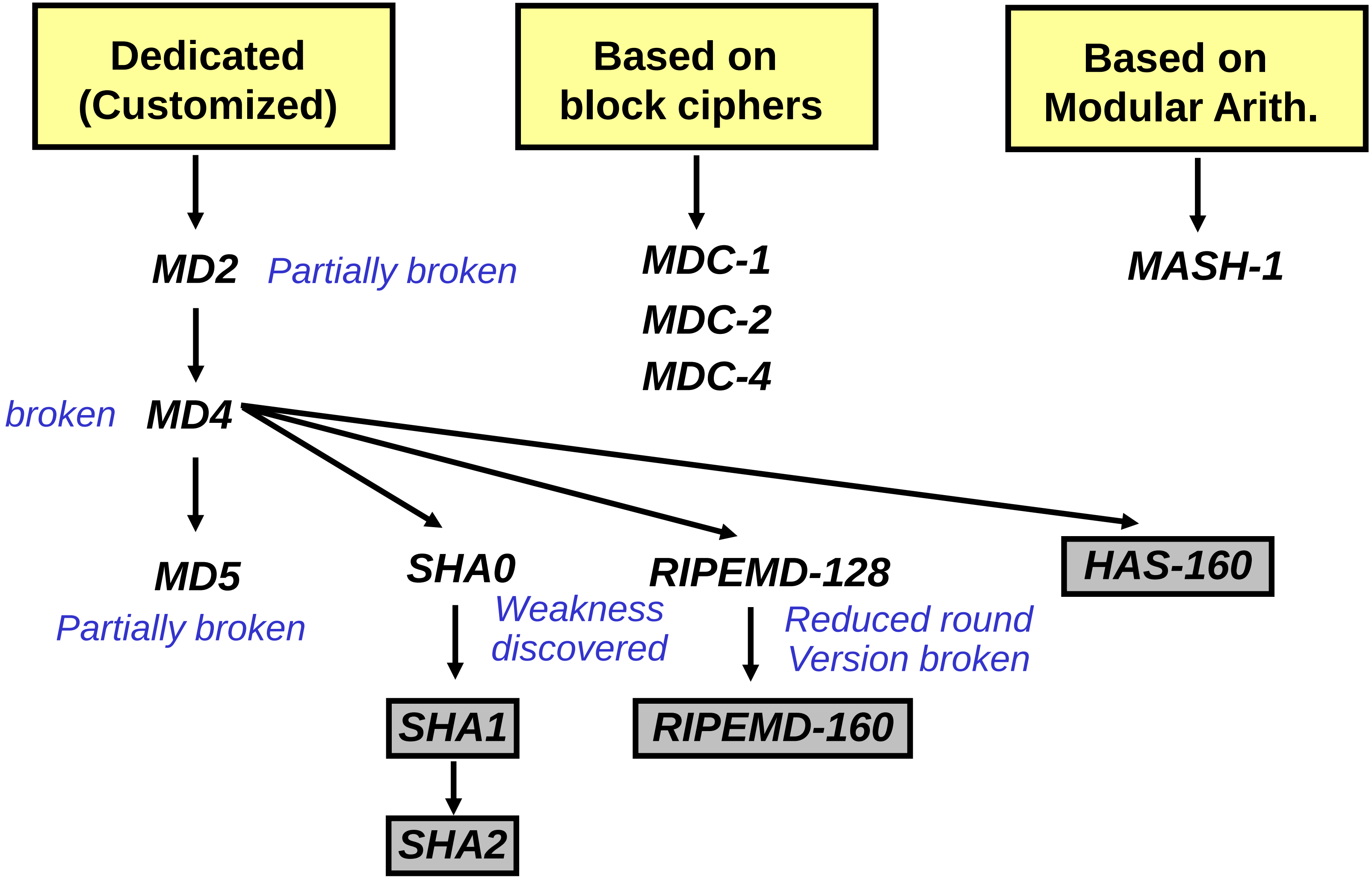
General Construction of a Secure Hash Function



Legend:

- IV : Initial Value
- H_i : i-th Chaining variable
- M_i : i-th input block
- f : Compression function
- g : Output transformation (optional)
- t : Number of input blocks
- b : Block size in bits
- n : Hash code size in bits

Classification of Hash Functions



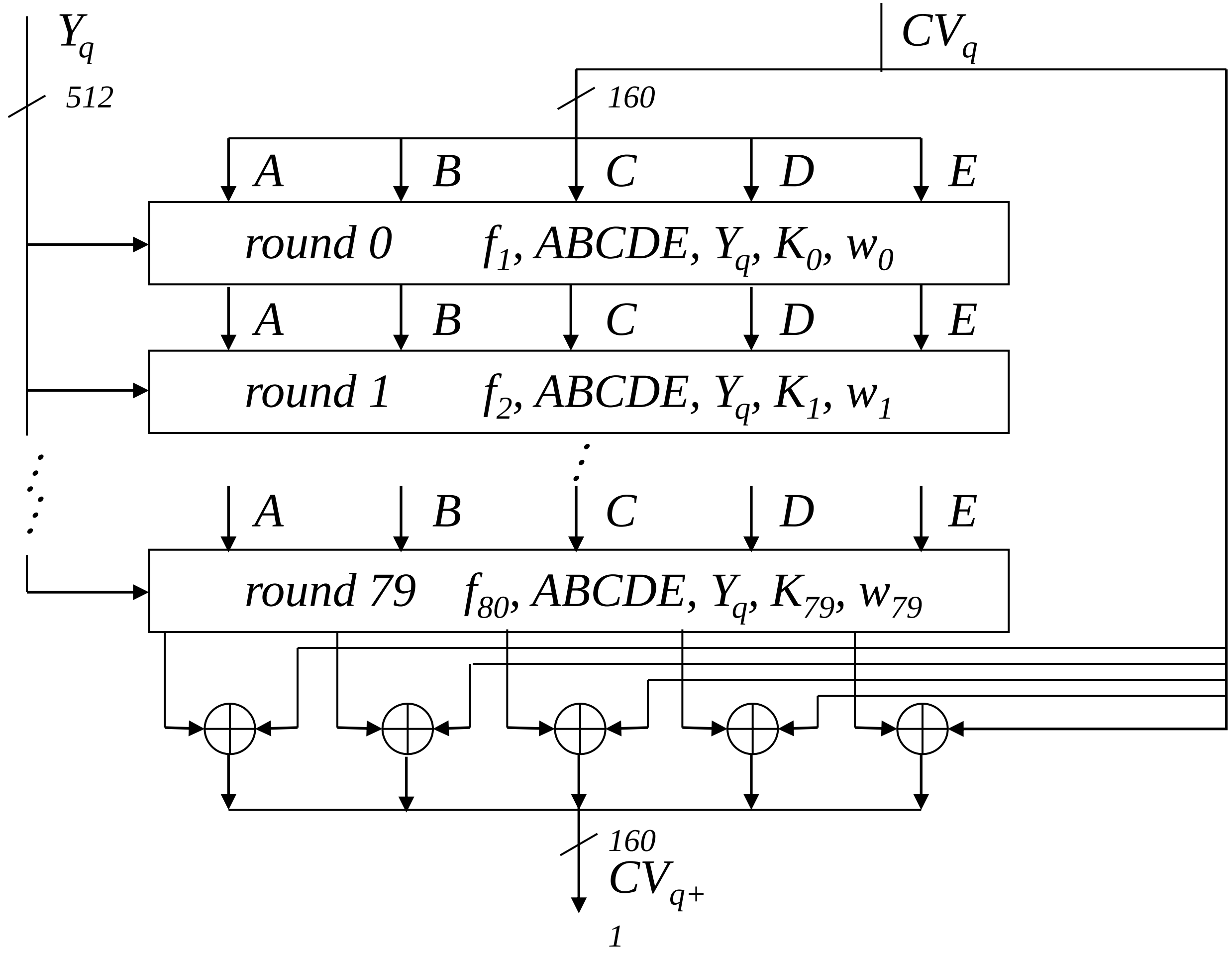
SHA (Secure Hash Algorithm)

- SHA was designed by NIST (national institute of standards and technology) & NSA (National Security Agency) in 1993, and revised as SHA-1 in 1995
  - SHA: FIPS PUB 180, 1993   ** Federal Information Processing Standard*
  - SHA-1 : FIPS Pub 180-1, 1995

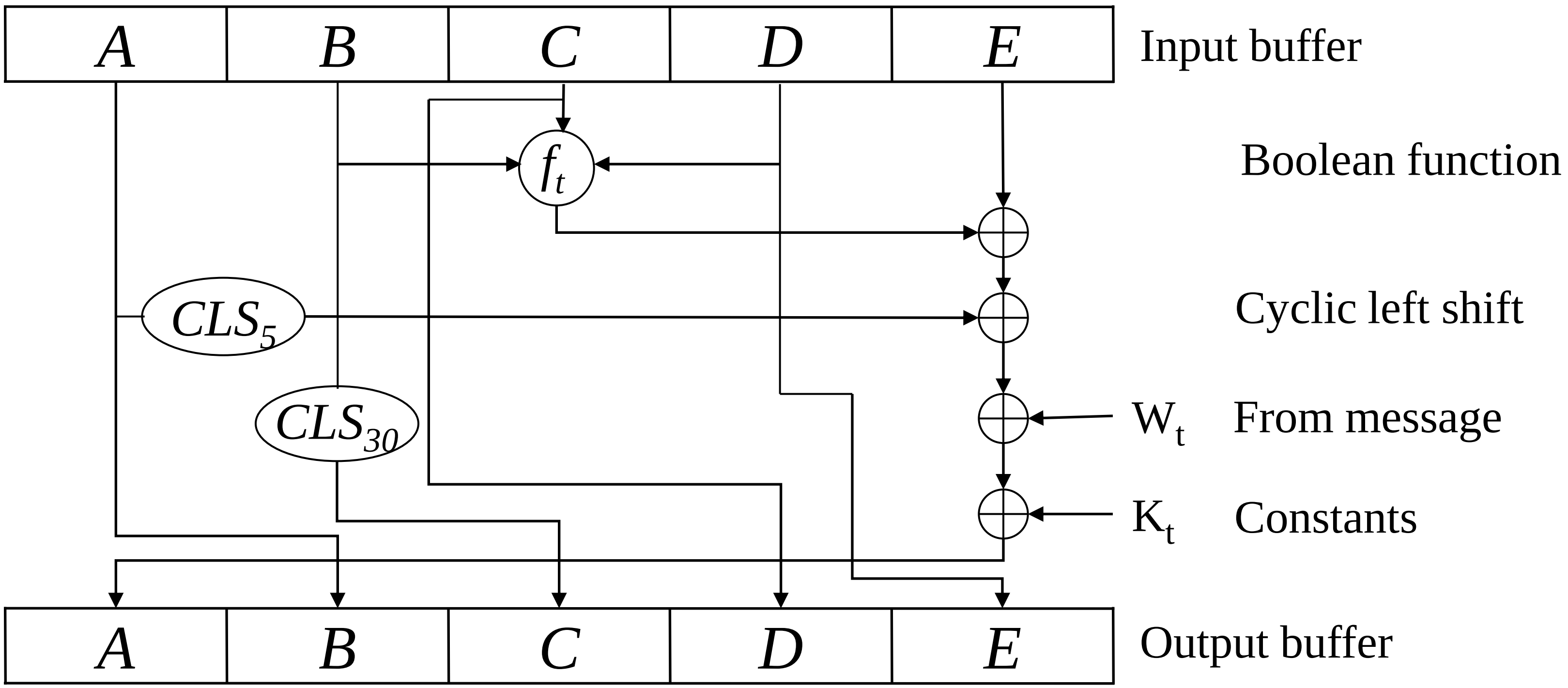
- US standard for use with DSA signature scheme
- The algorithm is SHA, the standard is SHS
- Based on the design of MD4 with key differences

- SHA-1 : Secure Hash Standard (SHS), FIPS Pub 180-1, 1995
  - 160-bit hash value (5 words Big Endian)
  - 512-bit block size
  - 4 round hash, each round has 20 steps, total 80 steps

SHA-1 Overview



SHA-1 round function



SHA-1

Initial values

$A = 6\ 7\ 4\ 5\ 2\ 3\ 0\ 1$

$B = E\ F\ C\ D\ A\ B\ 8\ 9$

$C = 9\ 8\ B\ A\ D\ C\ F\ E$

$D = 1\ 0\ 3\ 2\ 5\ 4\ 7\ 6$

$E = C\ 3\ D\ 2\ E\ 1\ F\ 0$

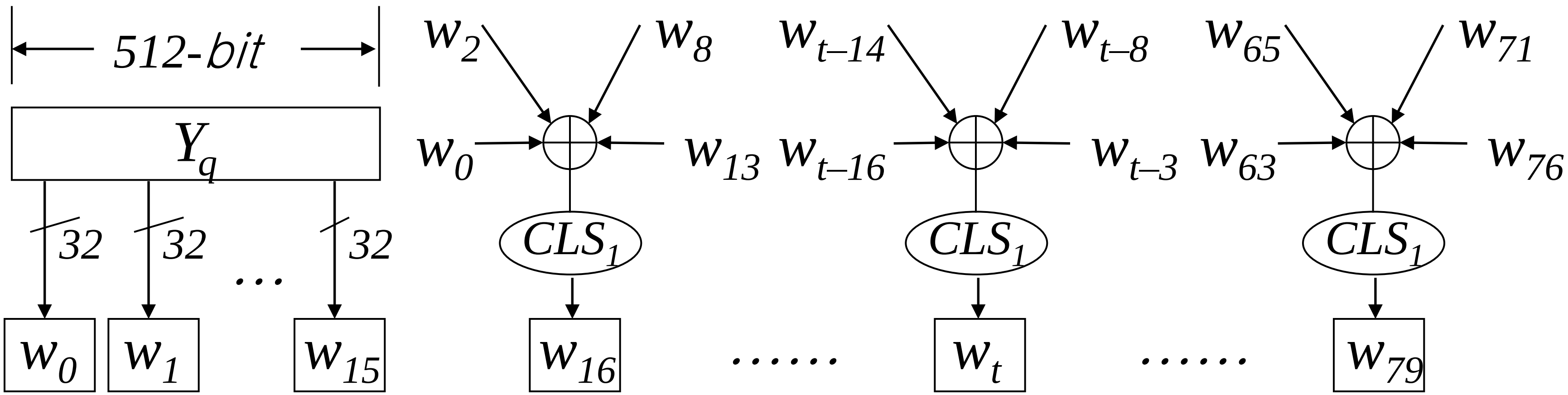
Constants K_t

$t = 0 \sim 19$	$K_t = 5\ A\ 8\ 2\ 7\ 9\ 9\ 9$
$t = 20 \sim 39$	$K_t = 6\ E\ D\ 9\ E\ B\ A\ 1$
$t = 40 \sim 59$	$K_t = 8\ F\ 1\ B\ B\ C\ D\ C$
$t = 60 \sim 79$	$K_t = C\ A\ 6\ 2\ C\ 1\ D\ 6$

Boolean function f_t

$t = 0 \sim 19$	$f_t\ (B, C, D) = B \cdot C + \overline{B} \cdot D$
$t = 20 \sim 39$	$f_t\ (B, C, D) = B \oplus C \oplus D$
$t = 40 \sim 59$	$f_t\ (B, C, D) = B \cdot C + B \cdot D + C \cdot D$
$t = 60 \sim 79$	$f_t\ (B, C, D) = B \oplus C \oplus D$

SHA-1 message inputs

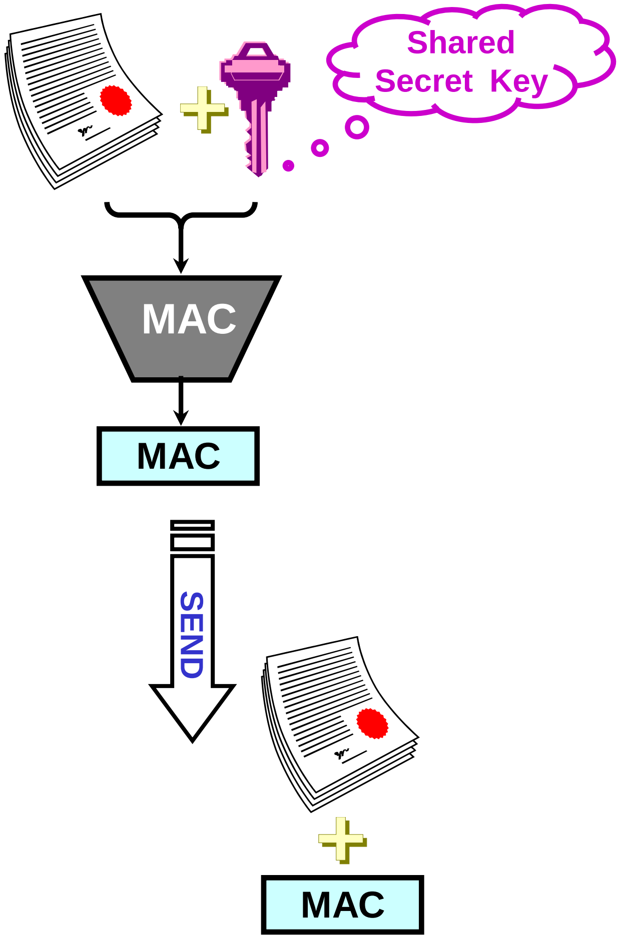


Comparison of Popular Hash Functions

Hash Func.	MD5	SHA1	RMD160	HAS160
Digest size(bits)	128	160	160	160
Block size(bits)	512	512	512	512
No of steps	64(4x16)	80(4x20)	160(5x2x16)	80(4x20)
Boolean func.	4	4(3)	5	4(3)
Constants	64	4	9	4
Endianness	Little	Big	Little	Little
Speed ratio	1.0	0.57	0.5	0.94

Message Authentication Code (MAC)

- **Purposes**
  - ✓ Secure tag for authentication
  - ✓ Message origin authentication
  - ✓ User authentication
  - ✓ Message integrity
- **Schemes**
  - ✓ Keyed hash: HMAC
  - ✓ Block cipher: CBC-MAC, XCBC-MAC
  - ✓ Dedicated MAC: UMAC



4. Certification and PKI

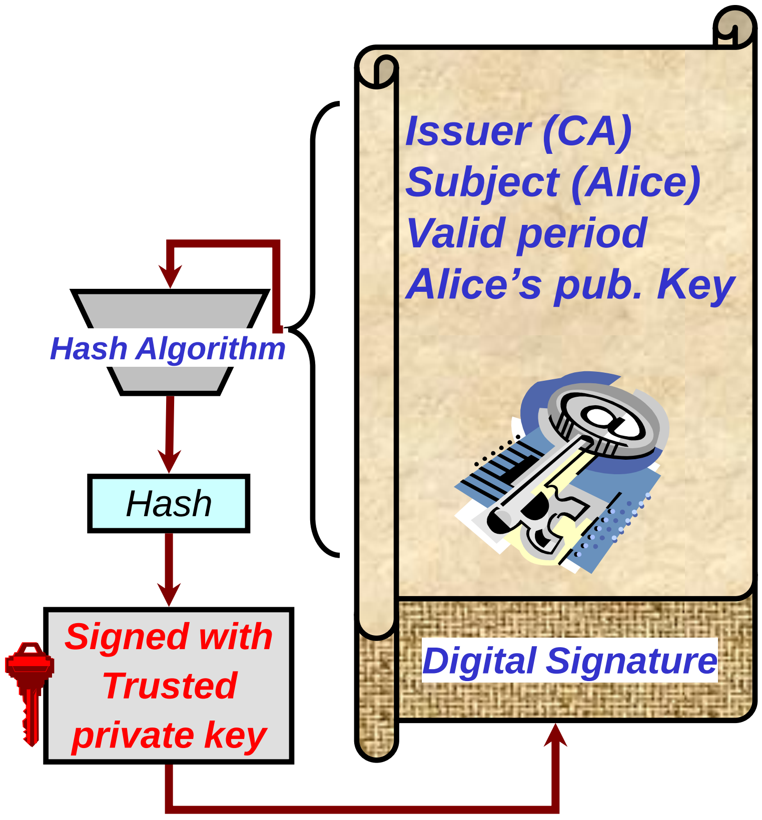
How to Guarantee the Authenticity of Peer Public Key?

- ❖ For secure use of public key systems,
  - ➢ Everyone should be able to obtain the public key of any communication peer that he wants to communicate with, in such a way that he can be sure that the obtained public key is the correct and right public key of the peer
  - ➢ How to guarantee that the public key obtained is the right one ?
  - ➢ How to guarantee that the public key obtained is authentic ?
- ❖ Using Certificate !

What is a Digital Certificate?

❖ **Digital Certificate**

✓ A file containing **Identification information (**CA's name (Issuer), Alice's name (Subject), valid period, Alice's public key, etc) and **digital signature** signed by trusted third (CA) to guarantee its authenticity & integrity



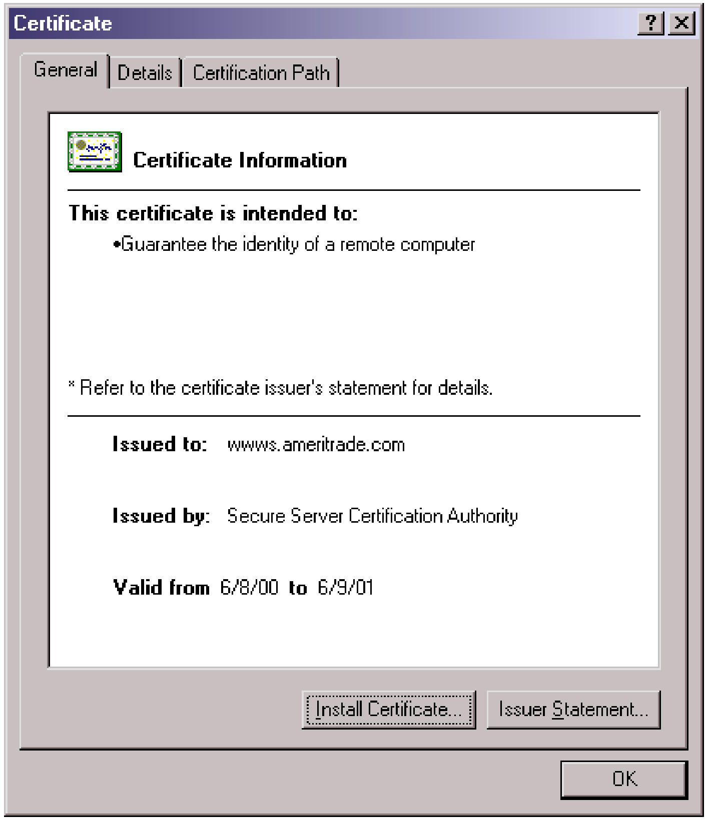
❖ **Certificate Authority (CA)**

✓ Trusted third party like a government for passports

✓ CA authenticates that the public key belongs to Alice

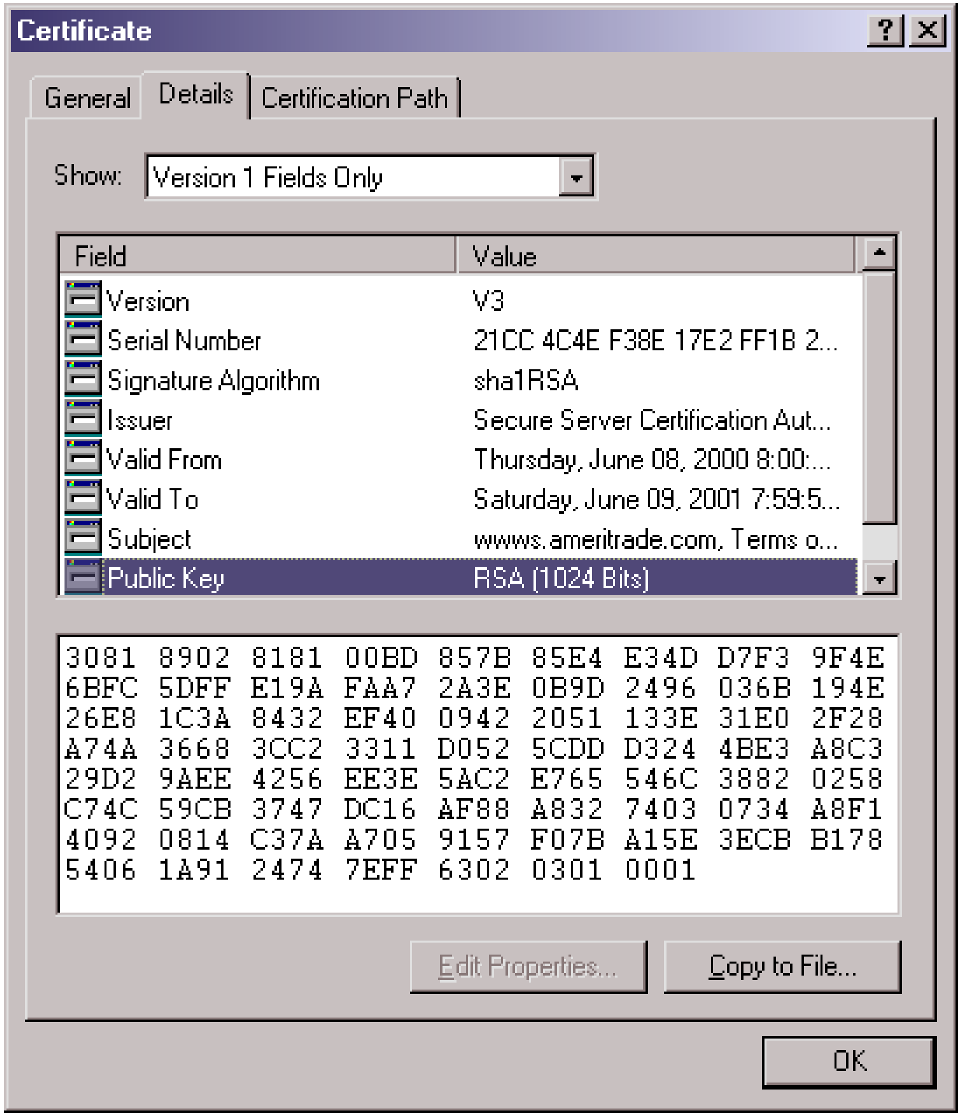
✓ CA creates Alice's Digital Certificate

Certificate



Data encrypted using secret key exchanged using some public key associated with some certificate.

Certificate



X.509 V3 Certificate Format

```
Certificate  ::=  SEQUENCE  {
     tbsCertificate            TBSCertificate,
     signatureAlgorithm        AlgorithmIdentifier,
     signatureValue            BIT STRING  }

  TBSCertificate  ::=  SEQUENCE  {
     version                   [0]  EXPLICIT Version DEFAULT v1,
     serialNumber               CertificateSerialNumber,
     signature                 AlgorithmIdentifier,
     issuer                    Name,
     validity                  Validity,
     subject                   Name,
     subjectPublicKeyInfo      SubjectPublicKeyInfo,
     issuerUniqueID            [1]  IMPLICIT UniqueIdentifier OPTIONAL,
                               -- If present, version shall be v2 or v3
     subjectUniqueID           [2]  IMPLICIT UniqueIdentifier OPTIONAL,
                               -- If present, version shall be v2 or v3
     extensions                [3]  EXPLICIT Extensions OPTIONAL
                               -- If present, version shall be v3
     }
```

Sample Certificate

```
Certificate:
   Data:
      Version: v3 (0x2)
      Serial Number: 3 (0x3)
      Signature Algorithm: PKCS #1 MD5 With RSA Encryption
      Issuer: OU=Ace Certificate Authority, O=Ace Industry, C=US
      Validity:
         Not Before: Fri Oct 17 18:36:25 1997
         Not  After: Sun Oct 17 18:36:25 1999
      Subject: CN=Jane Doe, OU=Finance, O=Ace Industry, C=US
      Subject Public Key Info:
         Algorithm: PKCS #1 RSA Encryption
         Public Key:
            Modulus:
               00:ca:fa:79:98:8f:19:f8:d7:de:e4:49:80:48:e6:2a:2a:86:
               ed:27:40:4d:86:b3:05:c0:01:bb:50:15:c9:de:dc:85:19:22:
               43:7d:45:6d:71:4e:17:3d:f0:36:4b:5b:7f:a8:51:a3:a1:00:
               98:ce:7f:47:50:2c:93:36:7c:01:6e:cb:89:06:41:72:b5:e9:
               73:49:38:76:ef:b6:8f:ac:49:bb:63:0f:9b:ff:16:2a:e3:0e:
               9d:3b:af:ce:9a:3e:48:65:de:96:61:d5:0a:11:2a:a2:80:b0:
               7d:d8:99:cb:0c:99:34:c9:ab:25:06:a8:31:ad:8c:4b:aa:54:
               91:f4:15
            Public Exponent: 65537 (0x10001)
    Extensions:
         Identifier: Certificate Type
            Critical: no
            Certified Usage:
               SSL Client
         Identifier: Authority Key Identifier
            Critical: no
            Key Identifier:
               f2:f2:06:59:90:18:47:51:f5:89:33:5a:31:7a:e6:5c:fb:36:
               26:c9
```

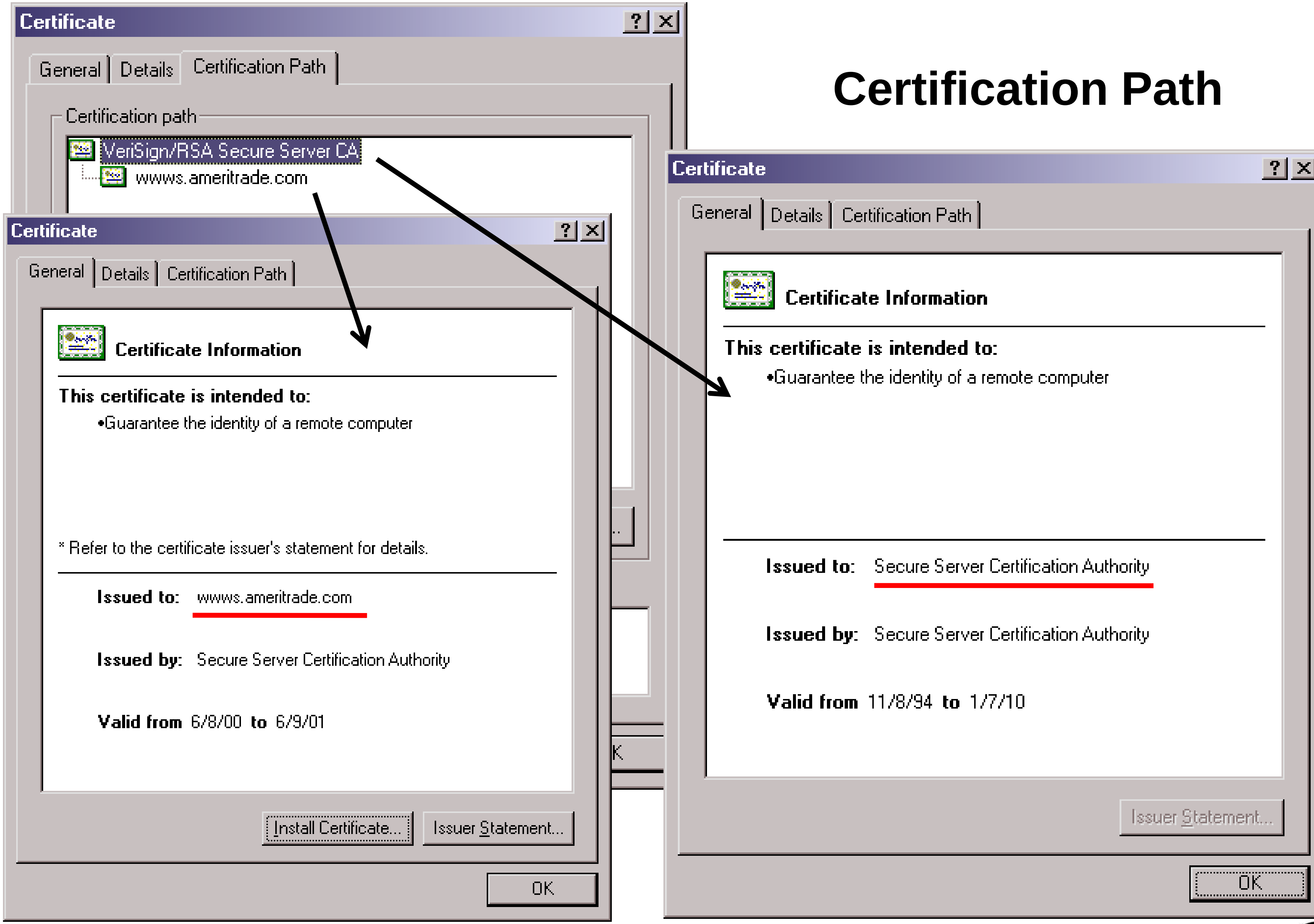
```
   Signature:
      Algorithm: PKCS #1 MD5 With RSA Encryption
      Signature:
         6d:23:af:f3:d3:b6:7a:df:90:df:cd:7e:18:6c:01:69:8e:54:65:fc:06:
         30:43:34:d1:63:1f:06:7d:c3:40:a8:2a:82:c1:a4:83:2a:fb:2e:8f:fb:
         f0:6d:ff:75:a3:78:f7:52:47:46:62:97:1d:d9:c6:11:0a:02:a2:e0:cc:
         2a:75:6c:8b:b6:9b:87:00:7d:7c:84:76:79:ba:f8:b4:d2:62:58:c3:c5:
         b6:c1:43:ac:63:44:42:fd:af:c8:0f:2f:38:85:6d:d6:59:e8:41:42:a5:
         4a:e5:26:38:ff:32:78:a1:38:f1:ed:dc:0d:31:d1:b0:6d:67:e9:46:a8:
          dd:c4
```


-----BEGIN CERTIFICATE-----
MIICKzCCAZSgAwIBAgIBAzANBgkqhkiG9w0BAQQFADA3MQswCQYD
VQQGEwJVUzERMA8GA1UEChMITmV0c2NhcGUxFTATBgNVBAsTDF
N1cHJpeWEncyBDQTAeFw05NzEwMTgwMTM2MjVaFw05OTEwMTgw
MTM2MjVaMEgxCzAJBgNVBAYTAlVTMREwDwYDVQQKEwhOZXRzY
2FwZTENMAsGA1UECxMEUHViczEXMBUGA1UEAxMOU3Vwcml5YSB
TaGV0dHkwgZ8wDQYJKoZIhvcNAQEFBQADgY0AMIGJAoGBAMr6eZiP
GfjX3uRJgEjmKiqG7SdATYazBcABu1AVyd7chRkiQ31FbXFOGD3wNktb
f6hRo6EAmM5/R1AskzZ8AW7LiQZBcrXpc0k4du+2Q6xJu2MPm/8WKuM
OnTuvzpo+SGXelmHVChEqooCwfdiZywyZNMmrJgaoMa2MS6pUkfQVAg
MBAAGjNjA0MBEGCWCGSAGG+EIBAQQEAwIAgDAfBgNVHSMEGDAW
gBTy8gZZkBhHUfWJM1oxeuZc+zYmyTANBgkqhkiG9w0BAQQFAAOBgQ
BtI6/z07Z635DfzX4XbAFpjlRl/AYwQzTSYx8GfcNAqCqCwaSDKvsuj/vwbf
91o3j3UkdGYpcd2cYRCgKi4MwqdWyLtpuHAH18hHZ5uvi00mJYw8W2w
UOsY0RC/a/IDy84hW3WWehBUqVK5SY4/zJ4oTjx7dwNMdGwbWfpRqjd
1A==
-----END CERTIFICATE-----

KAIST ICC

KIPO

Certification Path

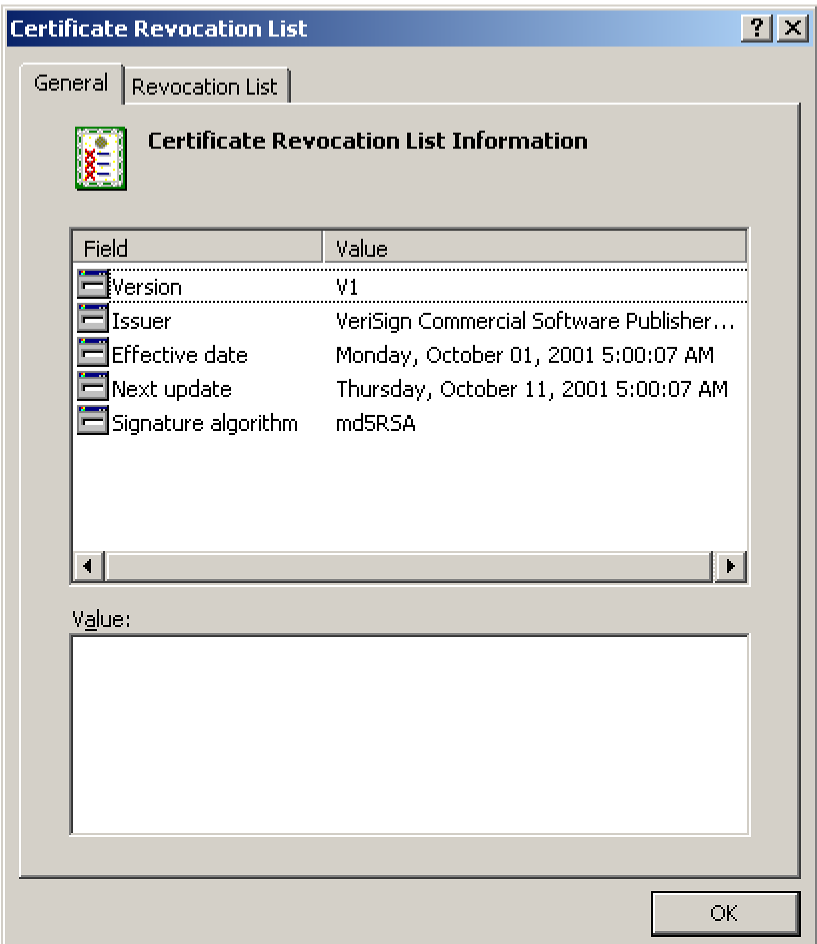


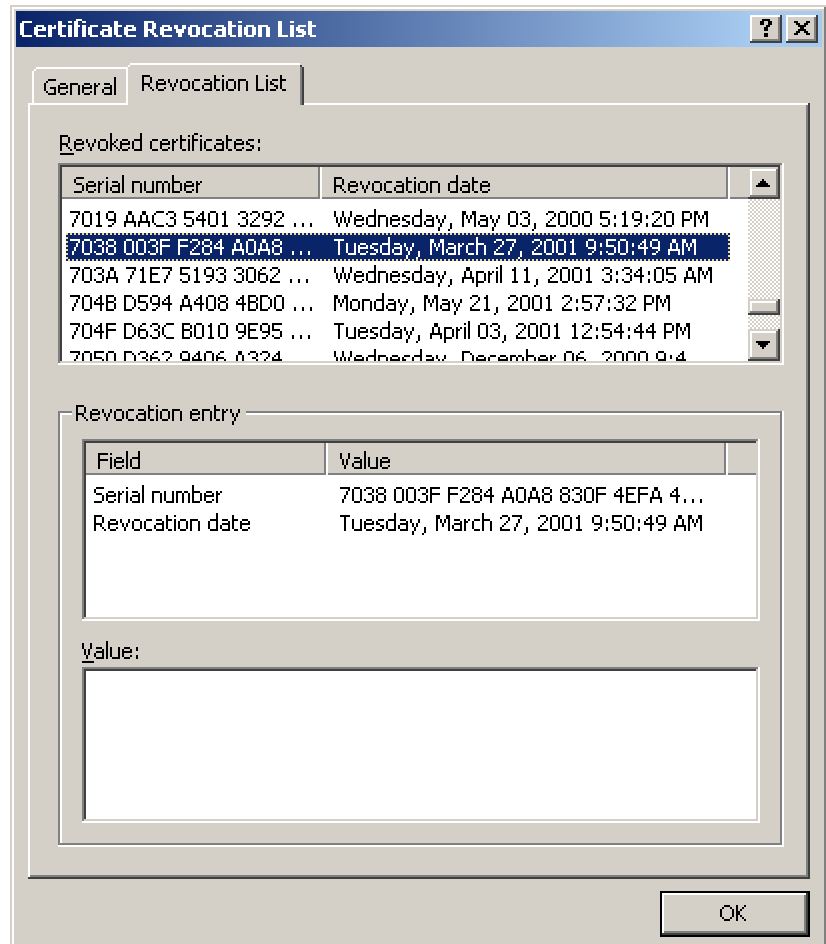
How to Revoke a Certificate?

- ❖ Why revoke a certificate?
  - ❖When the user leave (retire from) the organization
  - ❖Lost the private key, need to use a new key

- ❖ Certificate Revocation List (CRL)
  - ❖A digital document which has a list of revoked certificates
  - ❖Signed by CA
  - ❖Defined in X.509 v2

Certificate Revocation List





X.509 V2 Certificate Revocation List (CRL) Format

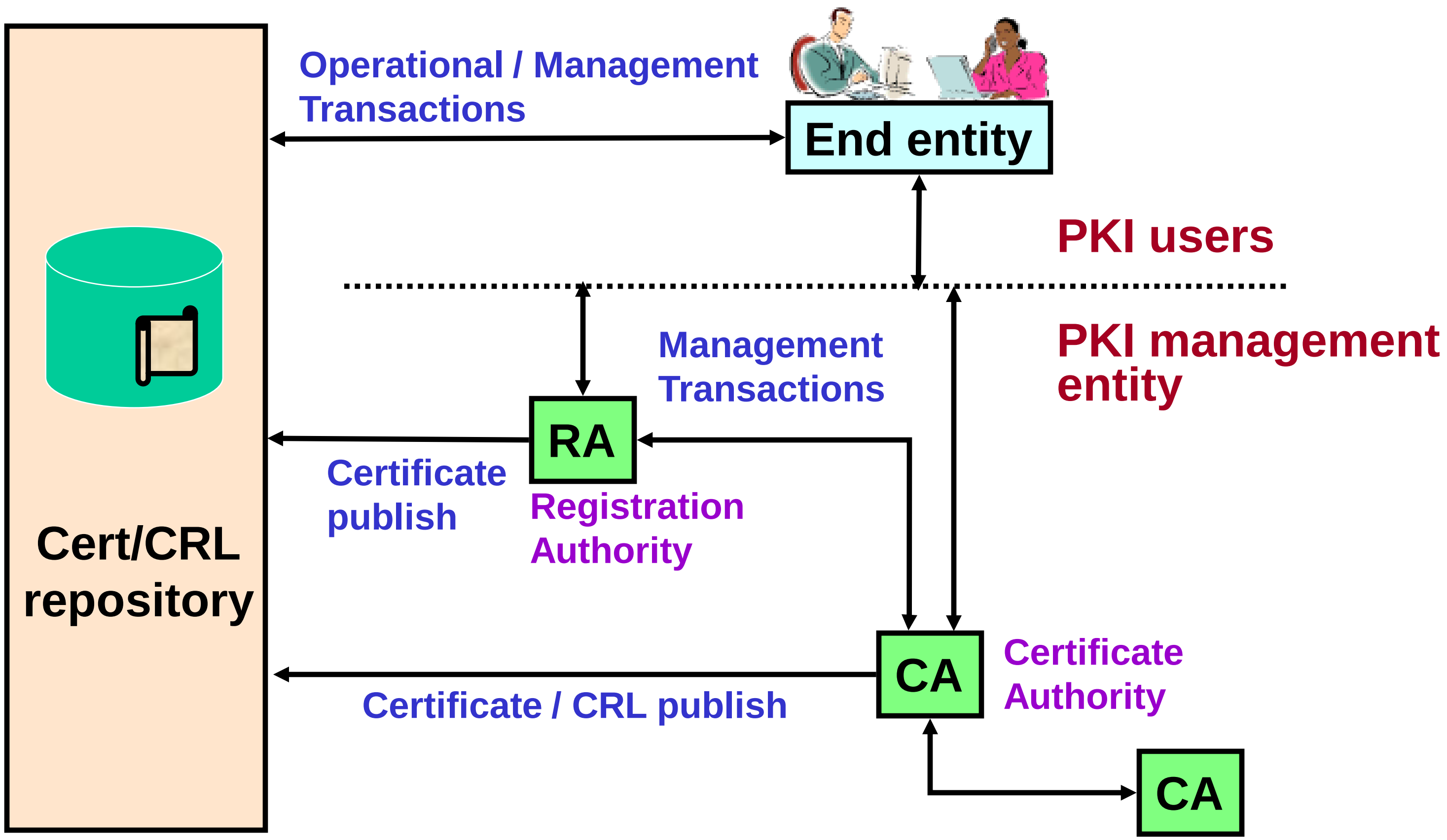
```
CertificateList  ::=  SEQUENCE  {
     tbsCertList                 TBSCertList,
     signatureAlgorithm          AlgorithmIdentifier,
     signatureValue              BIT STRING  }

 TBSCertList  ::=  SEQUENCE  {
     version                     Version OPTIONAL,
                                  -- if present, shall be v2
     signature                   AlgorithmIdentifier,
     issuer                      Name,
     thisUpdate                  Time,
     nextUpdate                  Time OPTIONAL,
     revokedCertificates         SEQUENCE OF SEQUENCE  {
              userCertificate            CertificateSerialNumber,
              revocationDate             Time,
              crlEntryExtensions         Extensions OPTIONAL
                                         -- if present, shall be v2
              }  OPTIONAL,
     crlExtensions                [0]  EXPLICIT Extensions OPTIONAL
                                  -- if present, shall be v2
     }
```

KAIST ICC

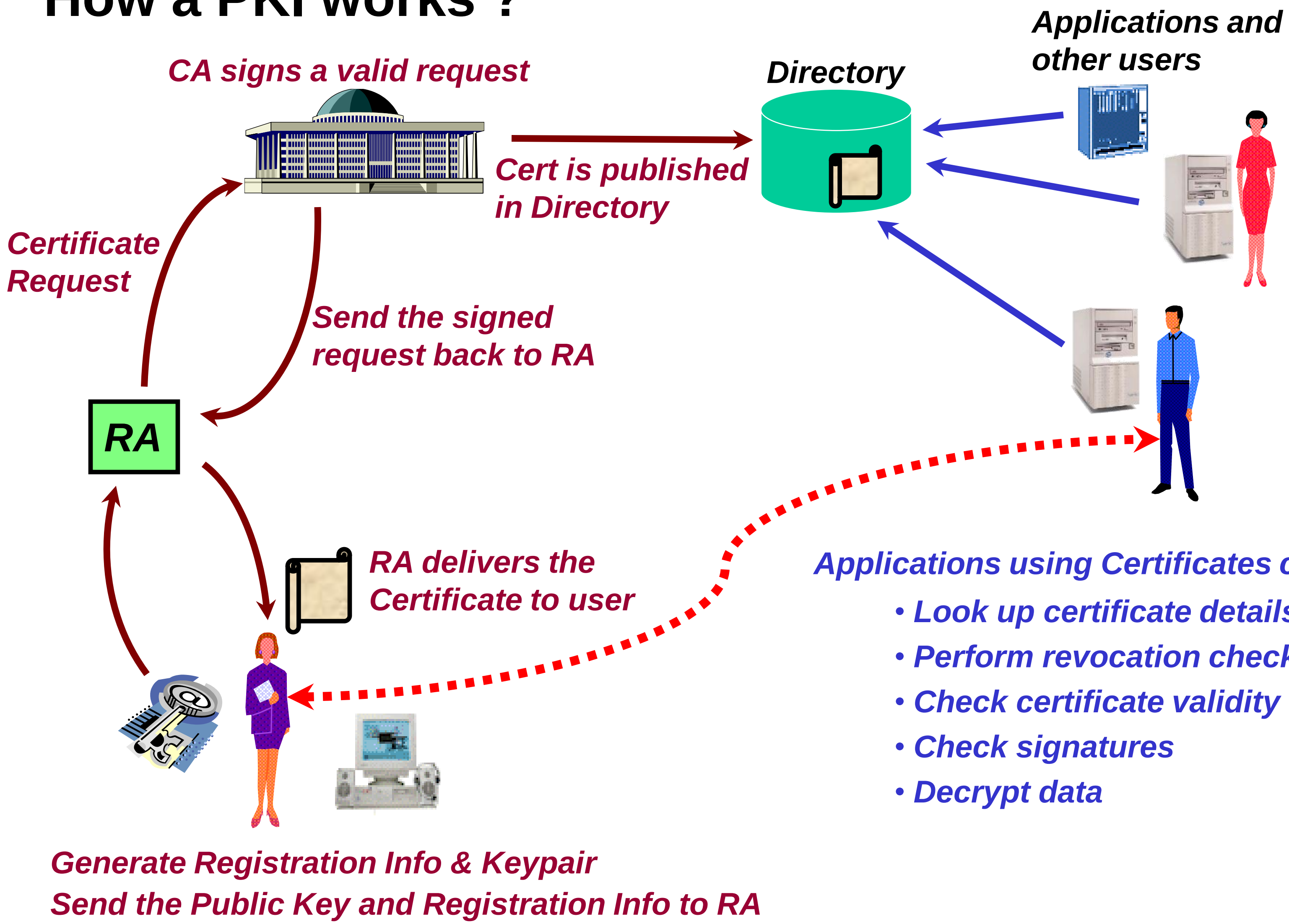
KIPO

Public Key Infrastructure (PKI) Architecture

PKI is the hardware, software, people, policies, & procedures needed to create, manage, store, distribute, & revoke certificates



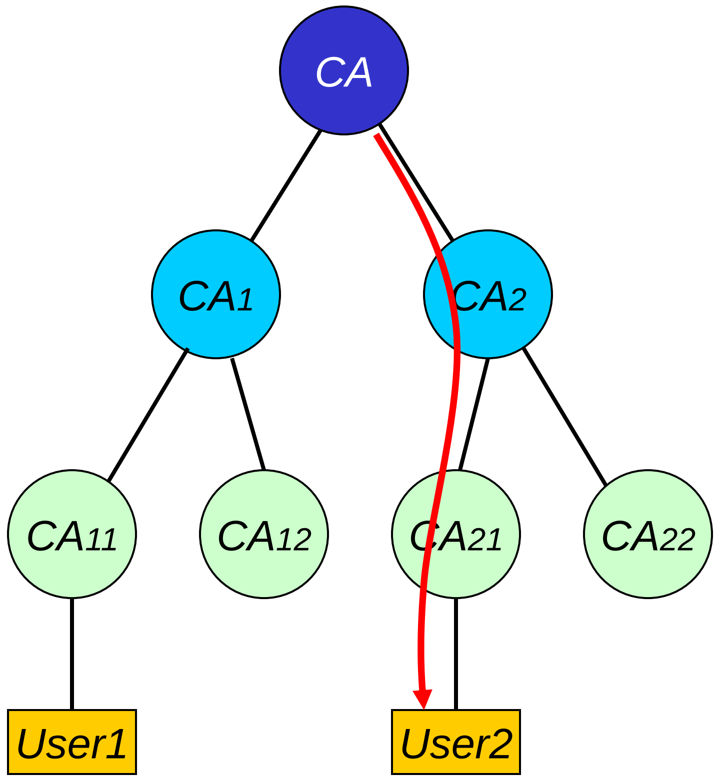
How a PKI works ?

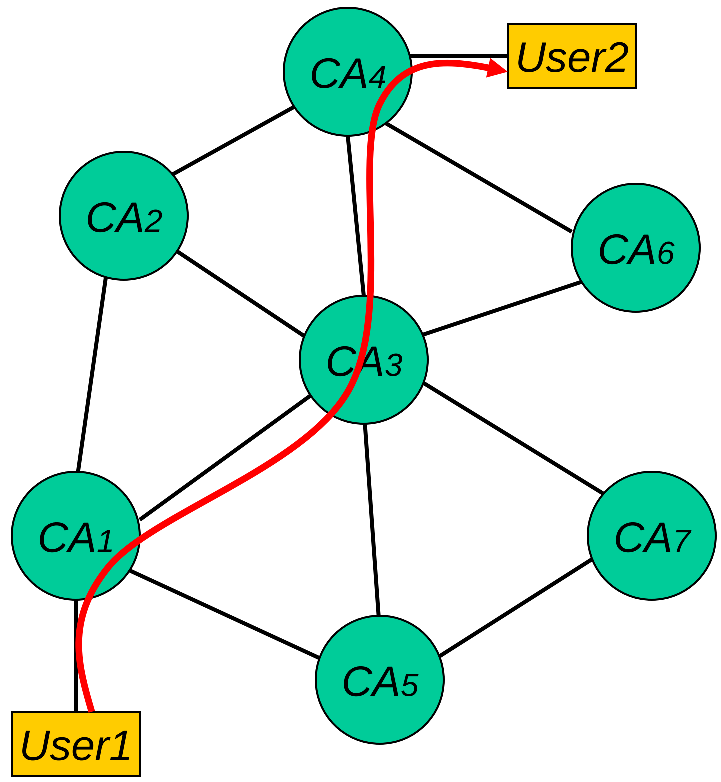


Applications using Certificates can :

- *Look up certificate details*
- *Perform revocation checks*
- *Check certificate validity*
- *Check signatures*
- *Decrypt data*

PKI Trust Relationship





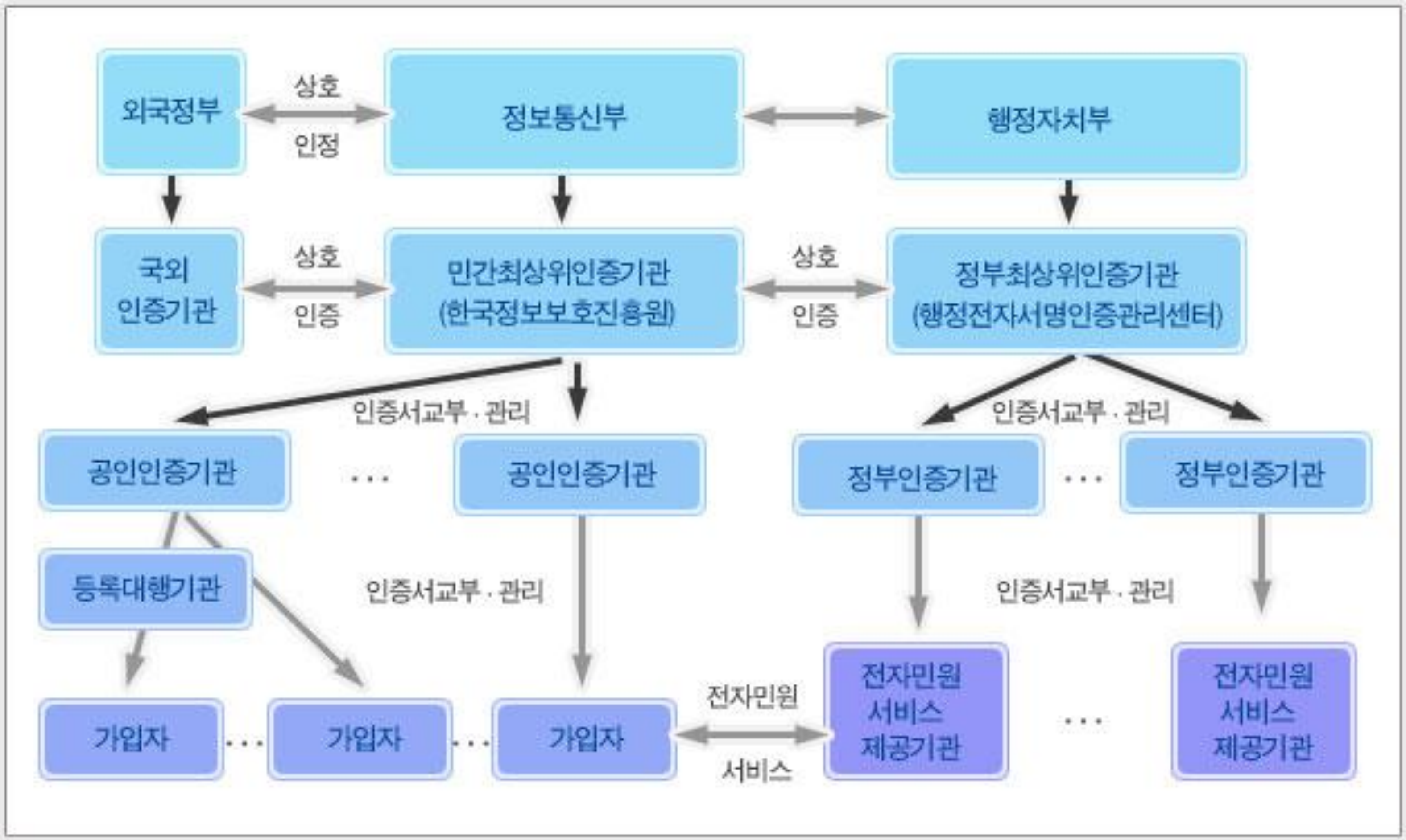
Hierarchical Structure

Network Structure

Korean PKI Structure

전자서명 인증관리센터

http://www.kisa.or.kr/kisa/kcac/jsp/kcac.jsp




THANK YOU