

Chapter 8

Network Security 网络安全

- 安全性涉及：
 - Secrecy(保密)：保护信息不被未授权者访问
 - Authentication(鉴别)：身份确认
 - Nonrepudiation(反否认)：确认原有信息的不被修改
 - Integrity Control(完整性控制)
- 引起安全性问题的人及意图-[see fig 8-1]
- 各层次对安全性的考虑：
 - 物理层：对传输线路的保护
 - 链路层：链路加密
 - 网络层：防火墙
 - 传输层：对整个连接的加密（端到端，即过程对过程）
 - 应用层：。。。

Need for Security

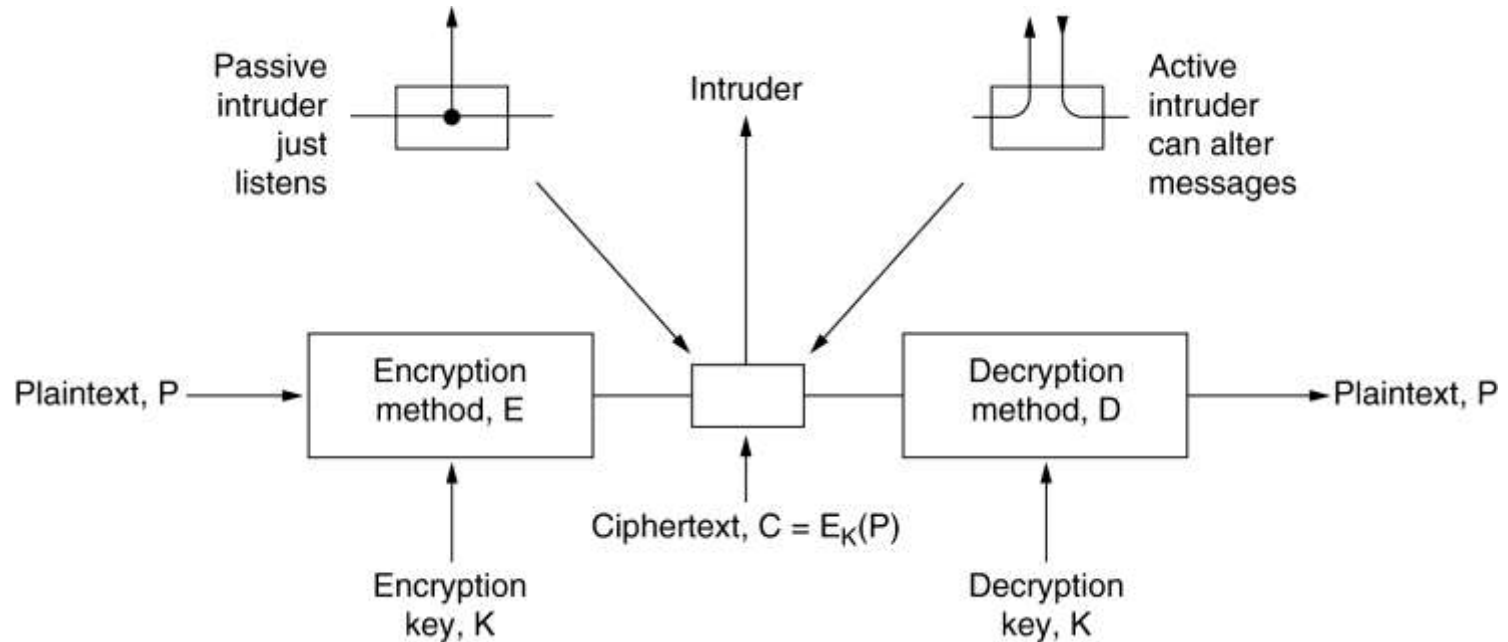
Adversary	Goal
Student	To have fun snooping on people's e-mail
Cracker	To test out someone's security system; steal data
Sales rep	To claim to represent all of Europe, not just Andorra
Businessman	To discover a competitor's strategic marketing plan
Ex-employee	To get revenge for being fired
Accountant	To embezzle money from a company
Stockbroker	To deny a promise made to a customer by e-mail
Con man	To steal credit card numbers for sale
Spy	To learn an enemy's military or industrial secrets
Terrorist	To steal germ warfare secrets

Some people who cause security problems and why.

8.1 Cryptography（传统加密技术）

- Introduction to Cryptography
- Substitution Ciphers（替换密码）
- Transposition Ciphers（变位密码）
- One-Time Pads（一次性加密）
- Two Fundamental Cryptographic Principles（两条基本加密原则）

8.1.1 An Introduction to Cryptography



The encryption model (for a symmetric-key cipher).

An Introduction to Cryptography(2)

- 加密模型： – [see fig 8-2]
 - 明文 (P)、密钥 (K)、密文 (C)、侵犯者、密码分析、密码学
 - $C = E_K(P)$
 - $D_K(E_K(P)) = P$
- 密钥长度
 - The longer the key, the higher the **work factor** the cryptanalyst has to deal with
- Kerckhof's principle: all algorithms must be public; only the keys are secret

- 密码分析问题的三个变种 (3 principal variations)
 - Ciphertext-only problem
 - Known plaintext problem
 - Chosen plaintext problem

8.1.2 Substitution Ciphers

- **替换密码：** 每个或每组字母由另一个/组伪装字母所替换
 - 凯撒密码
 - $a \rightarrow D, b \rightarrow E, c \rightarrow F, z \rightarrow C \dots \rightarrow \text{attack} \rightarrow \text{DWWDFN}$
 - 单字母表替换
 - 明文: abcdefghijklmnopqrstuvwxyz
 - 密文: QWERTYUIOPASDFGHJKLZXCVBNM
 - 破译的方法:
 - 利用自然语言的统计特点
 - 猜测可能的单词或短语

8.1.3 Transposition Ciphers

<u>M</u>	<u>E</u>	<u>G</u>	<u>A</u>	<u>B</u>	<u>U</u>	<u>C</u>	<u>K</u>
<u>7</u>	<u>4</u>	<u>5</u>	<u>1</u>	<u>2</u>	<u>8</u>	<u>3</u>	<u>6</u>
p	l	e	a	s	e	t	r
a	n	s	f	e	r	o	n
e	m	i	l	l	i	o	n
d	o	l	l	a	r	s	t
o	m	y	s	w	i	s	s
b	a	n	k	a	c	c	o
u	n	t	s	i	x	t	w
o	t	w	o	a	b	c	d

Plaintext

pleasetransferonemilliondollarsto
myswissbankaccountsixtwo

Ciphertext

AFLLSKSOSELAWAIATOOSSCTCLNMOMANT
ESILYNTWRNNTSOWDPAEDOBUEOERIRICXB

A transposition cipher.

Transposition Ciphers(2)

- 变位密码：对明文字母作重新排序，但不隐蔽它们-[see fig 8-3]
- 密码的破译
 - 确定加密方式
 - 猜测列的编号
 - 确定列的顺序

8.1.4 One-Time Pads(一次性加密)

- 用随机比特串作密钥
- 缺陷：
 - 密钥无法记忆
 - 可传送的数据总量受密钥数量限制
 - 丢失信息或信息错序将造成数据无法复原

One-Time Pads(2)

I love you

Message 1: 1001001 0100000 1101100 1101111 1110110 1100101 0100000 1111001 1101111 1110101 0101110
Pad 1: 1010010 1001011 1110010 1010101 1010010 1100011 0001011 0101010 1010111 1100110 0101011
Ciphertext: 0011011 1101011 0011110 0111010 0100100 0000110 0101011 1010011 0111000 0010011 0000101

Pad 2: 1011110 0000111 1101000 1010011 1010111 0100110 1000111 0111010 1001110 1110110 1110110
Plaintext 2: 1000101 1101100 1110110 1101001 1110011 0100000 1101100 1101001 1110110 1100101 1110011

Elvis lives

The use of a one-time pad for encryption and the possibility of getting any possible plaintext from the ciphertext by the use of some other pad.

Quantum Cryptography

Bit number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
Data	1	0	0	1	1	1	0	0	1	0	1	0	0	1	1	0	What Alice sends
(a)																	
(b)																	Bob's bases
(c)																	What Bob gets
(d)	No	Yes	No	Yes	No	No	No	Yes	Yes	No	Yes	Yes	Yes	No	Yes	No	Correct basis?
(e)		0		1				0	1		1	0	0		1		One-time pad
(f)																	Trudy's bases
(g)	x	0	x	1	x	x	x	?	1	x	?	?	0	x	?	x	Trudy's pad

An example of quantum cryptography.

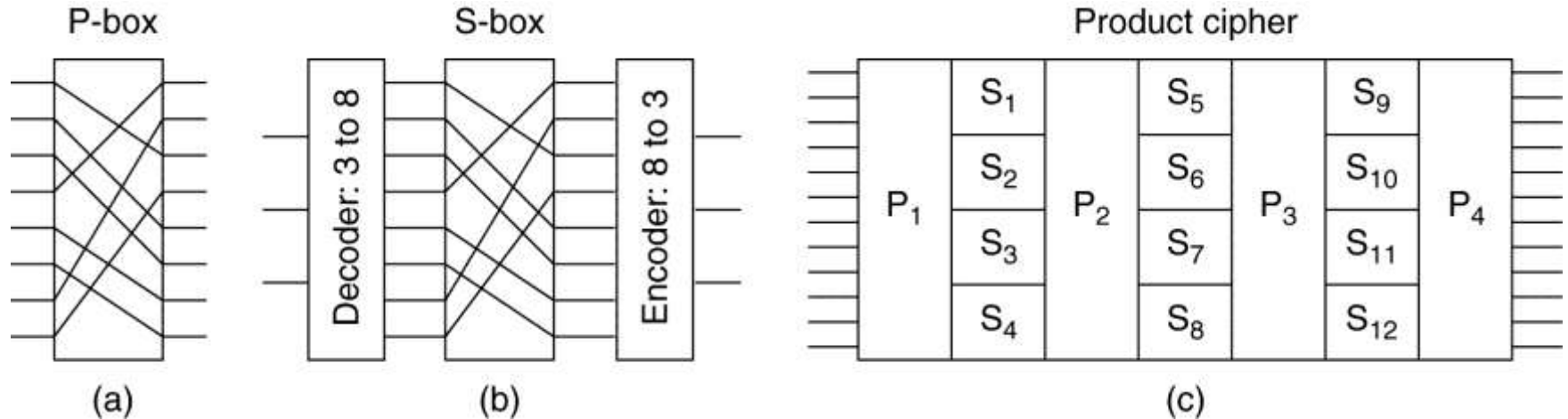
8.1.5 Two Fundamental Cryptographic Principles (两条基本加密原则)

- **Redundancy (冗余度)** : 所有的加密信息都包含有冗余信息 (Message must contain some redundancy)
 - 这些冗余信息一方面用于防止积极的入侵者欺骗接收者
 - 但同时, 又可能使消极的入侵者破坏系统更容易
- **Freshness (新鲜度)** : 必须采取措施, 防止主动入侵者发回旧的信息 (Some method is needed to foil replay attacks)
 - 措施: 加有效期很短的时间戳

8.2 Symmetric-Key Algorithms

- DES – The Data Encryption Standard
- AES – The Advanced Encryption Standard
- Cipher Modes
- Other Ciphers
- Cryptanalysis(密码分析学)

Product Ciphers

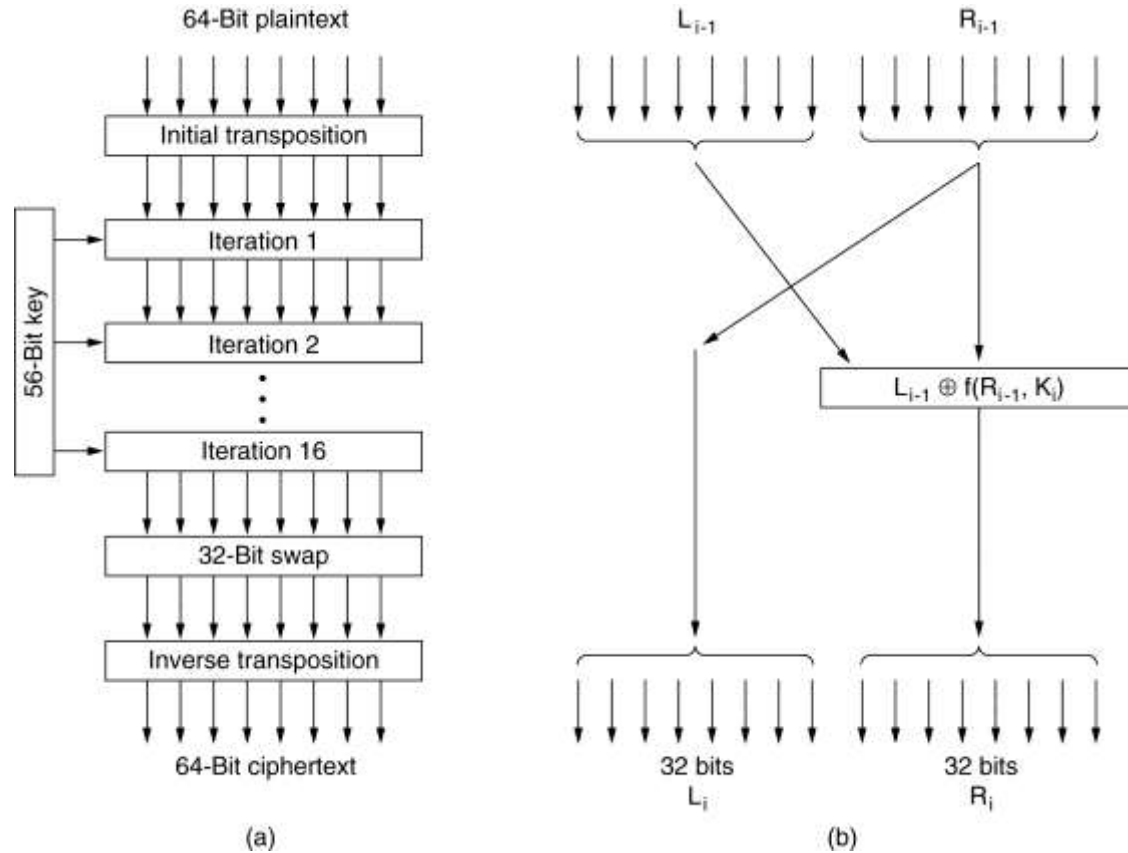


Basic elements of product ciphers.

(a) P-box(Permutation, 变位). (b) S-box(Substitution, 替换).

(c) Product(乘积).

8.2.1 DES--Data Encryption Standard

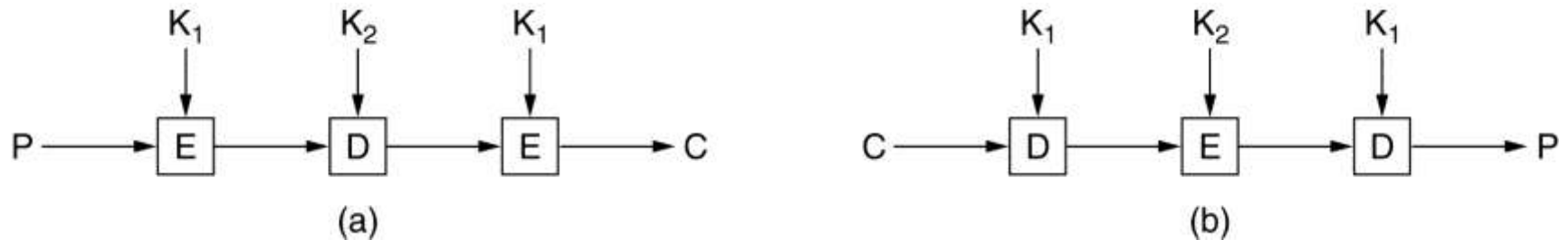


The data encryption standard. (a) General outline.
(b) Detail of one iteration. The circled + means exclusive OR.

DES--Data Encryption Standard(2)

- DES算法
 - 数据加密标准（DES） – [see fig 8-7]
 - 工作原理：
 - 64bit明文、
 - 56bit密钥、
 - 产生64bit密文
 - 由19个站组成
 - NSA: National Security Agency

Triple DES



(a) Triple encryption using DES. (b) Decryption.

- 三重加密法-[see fig 8-8]
 - 采用：加密 K_1 、解密 K_2 、再加密 K_1 （即：EDE，而非EEE，why?）
 - 考虑与单密钥DES系统的兼容性（ $k_1=k_2$ 时）
 - 尚无可以破解三重DES加密的方法

8.2.2 AES – The Advanced Encryption Standard

- NIST (National Institute of Standards and Technology)
- Rules for AES proposals:
 1. The algorithm must be a symmetric block cipher.
 2. The full design must be public.
 3. Key lengths of 128, 192, and 256 bits supported.
 4. Both software and hardware implementations required
 5. The algorithm must be public or licensed on nondiscriminatory terms(非歧视性条款).

AES(2)

- Bake-off (竞赛) in 1998
- In Oct. 2000, voted for Rijindael, number one
- Rijindael (from Joan Daemen and Vincent Rjinmen, Belgian young cryptographers, 86 votes)
 - A 128-bit block size with 128-bit key, or
 - A 128-bit block size with 256-bit key
 - More than 10^{10} years to search the key space with 1 billion parallel processors machine
- In Nov.2001 Rijindael became a U.S. Government standard (Federal Information Processing Standard, IFIPS 197)

AES(3)

- Like DES, Rijindael uses substitution(替代) and permutations(置换), and also uses multiple rounds(循环)

AES (4)

```
#define LENGTH 16 /* # bytes in data block or key */
#define NROWS 4 /* number of rows in state */
#define NCOLS 4 /* number of columns in state */
#define ROUNDS 10 /* number of iterations */
typedef unsigned char byte; /* unsigned 8-bit integer */

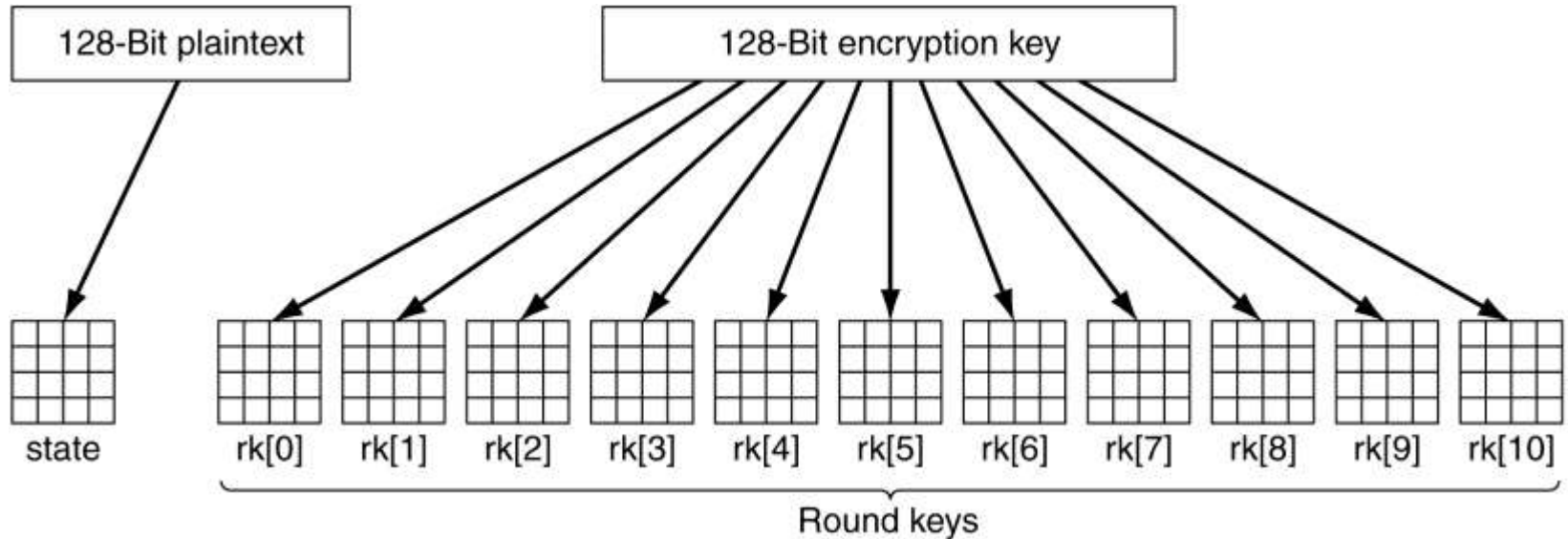
rijndael(byte plaintext[LENGTH], byte ciphertext[LENGTH], byte key[LENGTH])
{
    int r; /* loop index */
    byte state[NROWS][NCOLS]; /* current state */
    struct {byte k[NROWS][NCOLS];} rk[ROUNDS + 1]; /* round keys */

    expand_key(key, rk); /* construct the round keys */
    copy_plaintext_to_state(state, plaintext); /* init current state */
    xor_roundkey_into_state(state, rk[0]); /* XOR key into state */

    for (r = 1; r <= ROUNDS; r++) {
        substitute(state); /* apply S-box to each byte */
        rotate_rows(state); /* rotate row i by i bytes */
        if (r < ROUNDS) mix_columns(state); /* mix function */
        xor_roundkey_into_state(state, rk[r]); /* XOR key into state */
    }
    copy_state_to_ciphertext(ciphertext, state); /* return result */
}
```

- An outline of Rijndael.

AES (5)



Creating of the *state* and *rk* arrays.

8.2.3 Cipher Mode (*)

- Electronic Code Book Mode
- Cipher Block Chaining Mode
- Cipher Feedback Mode
- Stream Cipher Mode
- Counter Mode

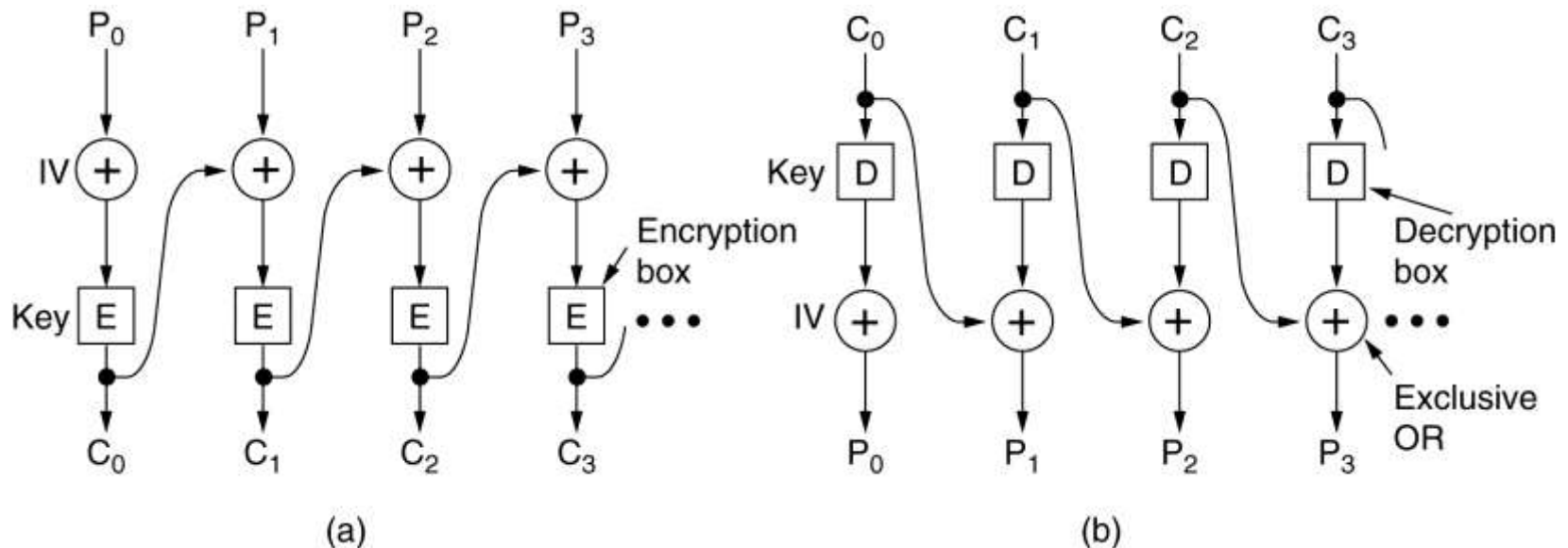
Electronic Code Book Mode (电子代码簿模式)

Name																Position								Bonus							
A	d	a	m	s	,		L	e	s	l	i	e				C	l	e	r	k				\$						1	0
B	l	a	c	k	,		R	o	b	i	n					B	o	s	s					\$	5	0	0	,	0	0	0
C	o	l	l	i	n	s	,		K	i	m					M	a	n	a	g	e	r		\$	1	0	0	,	0	0	0
D	a	v	i	s	,		B	o	b	b	i	e				J	a	n	i	t	o	r		\$							5

Bytes ← 16 → 8 → 8 →

The plaintext of a file encrypted as 16 DES blocks.

Cipher Block Chaining Mode (密码块链接)

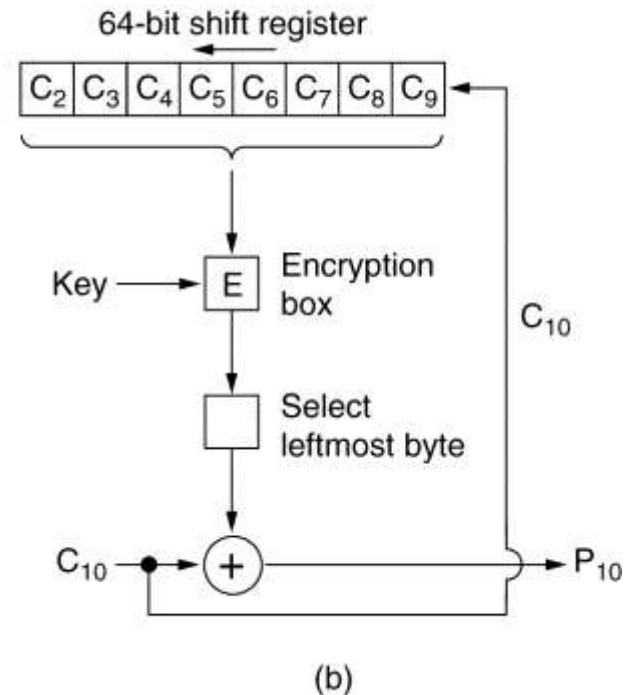
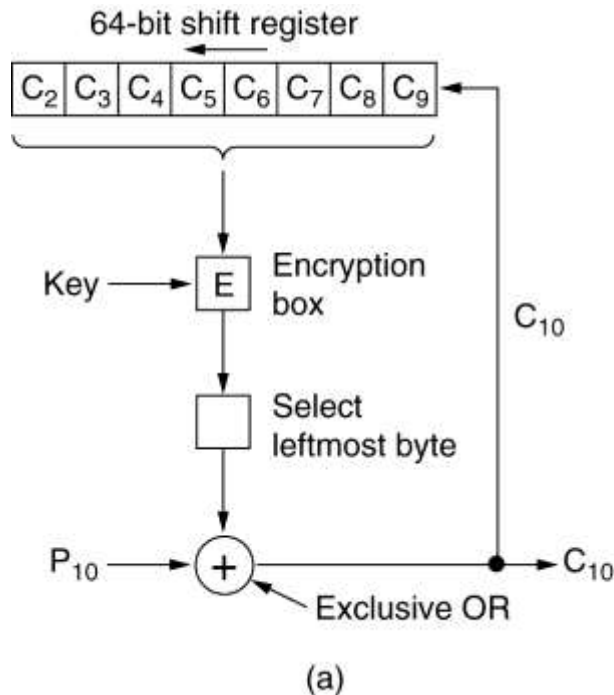


Cipher block chaining. (a) Encryption. (b) Decryption.

➤ Block-by-block encryption

Cipher Feedback Mode 方式)

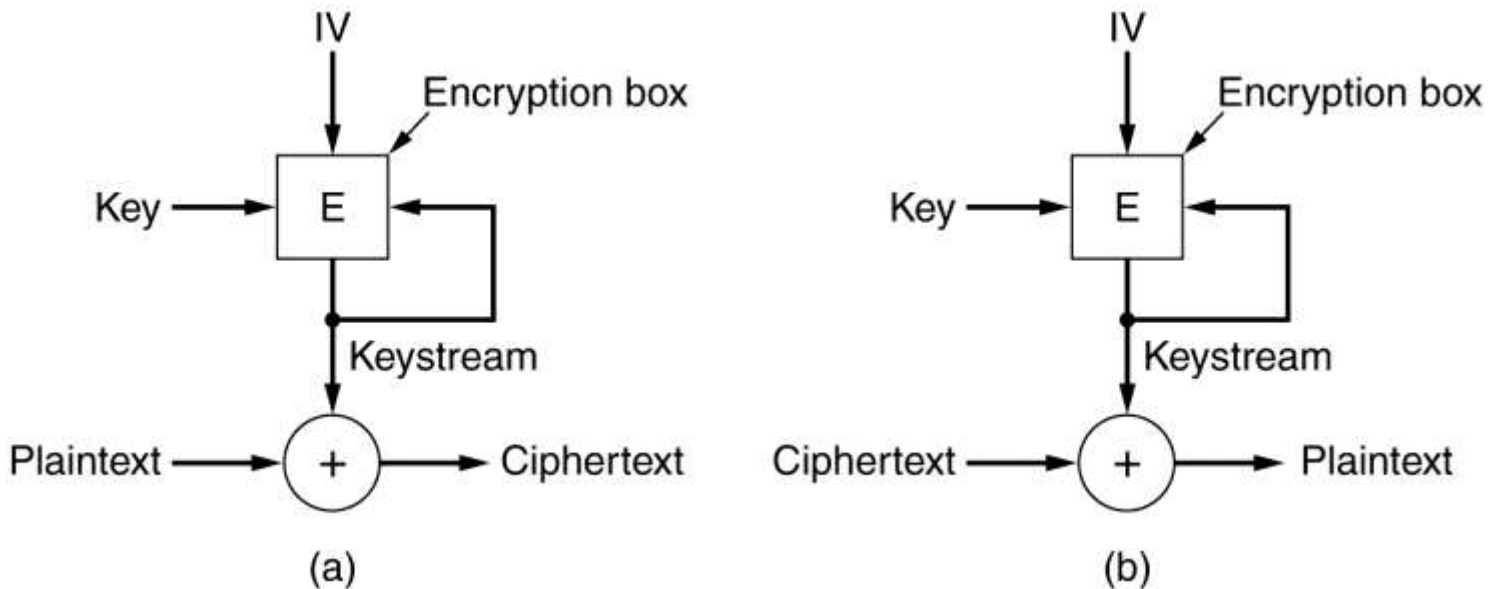
(密码反馈



(a) Encryption. (c) Decryption.

- Suitable for use with interactive terminals
- Byte-by-byte encryption

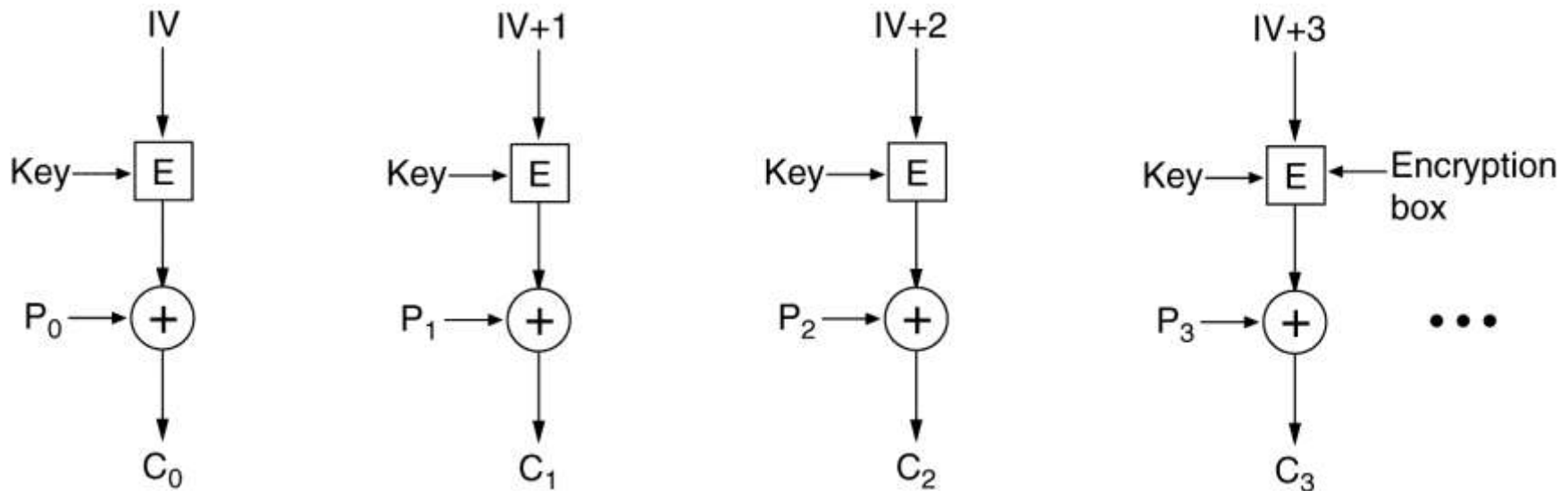
Stream Cipher Mode(流加密方式)



A stream cipher. (a) Encryption. (b) Decryption.

➤ Suitable for use with real-time streaming

Counter Mode (计数模式)



Encryption using counter mode.

- Suitable for use with disk files
- Access in non-sequential order, Counter \leftrightarrow IV

8.2.4 Other Ciphers (*)

Cipher	Author	Key length	Comments
Blowfish	Bruce Schneier	1–448 bits	Old and slow
DES	IBM	56 bits	Too weak to use now
IDEA	Massey and Xuejia	128 bits	Good, but patented
RC4	Ronald Rivest	1–2048 bits	Caution: some keys are weak
RC5	Ronald Rivest	128–256 bits	Good, but patented
Rijndael	Daemen and Rijmen	128–256 bits	Best choice
Serpent	Anderson, Biham, Knudsen	128–256 bits	Very strong
Triple DES	IBM	168 bits	Second best choice
Twofish	Bruce Schneier	128–256 bits	Very strong; widely used

Some common symmetric-key cryptographic algorithms.

8.2.5 Cryptanalysis(密码分析) (*)

- Differential Cryptanalysis(微分密码分析)
- Linear Cryptanalysis(线性密码分析)
- Using analysis of the electrical power consumption to find secret keys
- Timing analysis

8.3 Public-Key Algorithms (公开密钥算法)

➤ RSA

➤ Other Public-Key Algorithms

Public-Key Algorithms (2)

- 密钥分配问题—加密系统的脆弱点
- 使用不同的加密密钥和解密密钥，要求满足3点要求：
 1. $D(E(P)) = P$
 2. 从E导出D极其困难
 3. 由一段明文不可能破译出E
- 公开密钥加密法
 - 加密算法和密钥 E_A 是公开的 — 公开密钥(public key)
 - 解密算法和密钥 D_A 是保密的 — 秘密密钥(private key)
 - 每个使用者都有两个密钥

8.3.1 RSA

- RSA算法（基于大数因子分解）
 - 满足前面3点要求的算法
 - MIT的Rivest, Shamir和Adleman所设计
 - 基于数论原理：
 - 选择两个质数， p 和 q （典型地，应大于 10^{100} ）
 - 计算 $n=p \times q$ 和 $z=(p-1) \times (q-1)$
 - 选择一个与 z 互为质数的数 d
 - 找出 e ，使得 $e \times d \equiv 1 \pmod{z}$
 - 对信息 P 加密，计算： $C=P^e \pmod{n}$
 - 解密 C ，计算： $P=C^d \pmod{n}$

RSA(2)

Plaintext (P)		Ciphertext (C)			After decryption	
Symbolic	Numeric	P^3	$P^3 \pmod{33}$	C^7	$C^7 \pmod{33}$	Symbolic
S	19	6859	28	13492928512	19	S
U	21	9261	21	1801088541	21	U
Z	26	17576	20	1280000000	26	Z
A	01	1	1	1	01	A
N	14	2744	5	78125	14	N
N	14	2744	5	78125	14	N
E	05	125	26	8031810176	05	E
Sender's computation				Receiver's computation		

An example of the RSA algorithm.

($P=3$, $q=11 \Rightarrow n=33$, $z=20$, $d=7 \Rightarrow e=3$

$C=P^3 \pmod{33}$, $P=C^7 \pmod{33}$)

8.3.2 Other Public-Key Algorithms

- 背包算法 (Knapsack algorithm)
 - Offer a \$100 reward to anyone who can break it
 - 很快被“S”和“R” (in RSA) 破解 → \$100
 - 改进后, 又被“R”破解 → \$1000
 - 最终, 没有被投入实用
- 基于计算离散对数的困难性
- 基于椭圆曲线

8.4 Digital Signatures（数字签名）

- 保证文件的真实性和可靠性，用于代替亲笔签名
- 需要这样的一个系统：
 - 接收方能够验证发送方所宣称的身份
 - 发送方以后不能否认报文是他发送的（non-repudiation）
 - 接收方自己不能伪造该报文

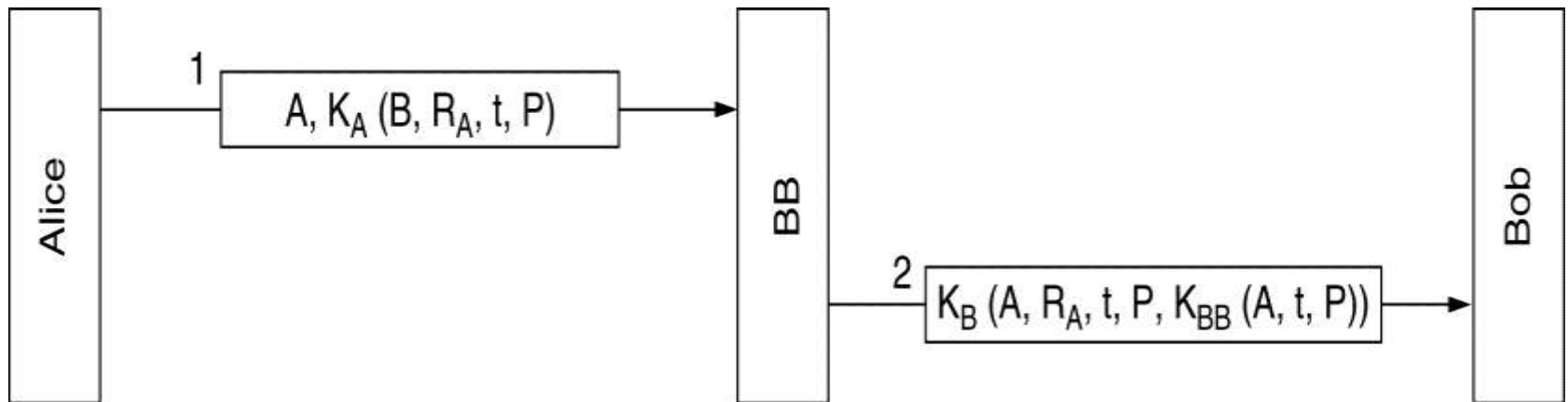
Digital Signatures (2)

- Symmetric-Key Signatures
- Public-Key Signatures
- Message Digests
- The Birthday Attack

8.4.1 Symmetric-Key Signatures (*)

- 采用密秘密钥 (Secret Key) 的数字签名
- 通过一个众人信任的中央机构，每个用户选择一个密码，并亲手交给该机构
- 使用大兄弟的数字签名-[see fig 8-18]
- 特点：通过时间戳，确保重发(伪造)信息无效

Symmetric-Key Signatures (2)



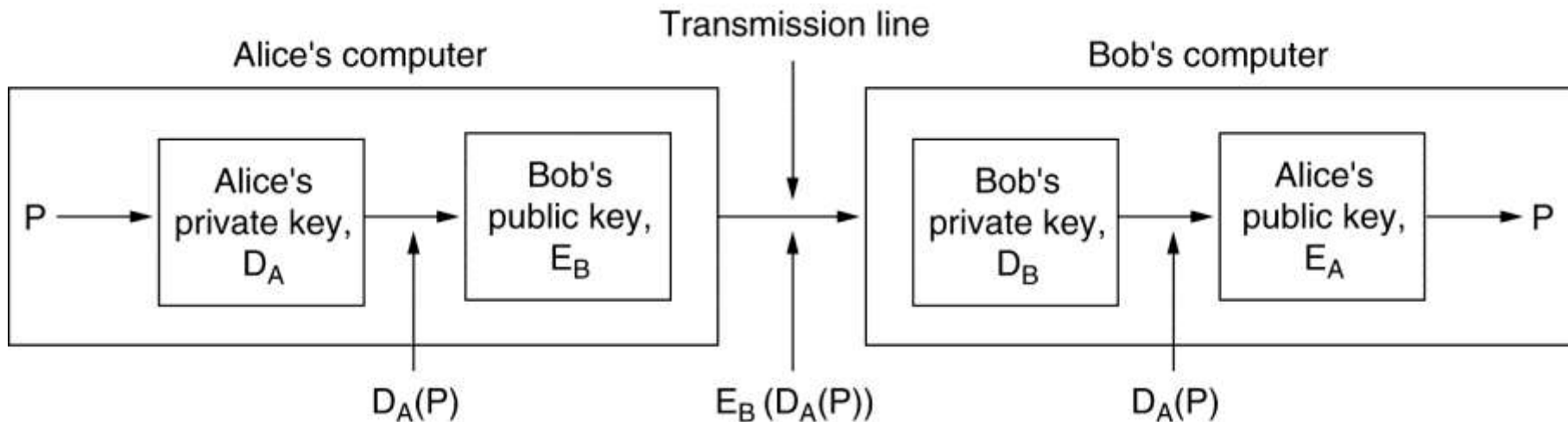
Digital signatures with Big Brother.

Note: $K_{BB}(A, t, P)$ signed message

8.4.2 Public-Key Signatures

- 采用公开密钥的数字签名
- 文件签名不需要任何可信赖的机构将更好
- 公开密钥加密法能满足这一要求
- $E(D(P))=P$, also $D(E(P))=P$
- 数字签名过程-[see fig 8-19]

Public-Key Signatures (2)



Digital signatures using public-key cryptography.

Signature message: $D_A(P)$

Alice cannot deny that she sent P because Bob can provide with P and $D_A(P)$, and $E_A(D_A(P))=P$

Public-Key Signatures (3)

- 特点：
 - 如果某一方公开了他的私有密钥（或被盗），则系统的可靠性将不再存在
 - 私有密钥的定期修改，需要有一个机构来记录所有的密钥改变及其变化日期
 - 原则上，所有的公开密钥算法都能用于数字签名

8.4.3 Message Digests（报文摘要）

- 报文摘要
- 上述签名方法把鉴别(authentication)和保密(secretcy))两项功能放在一起,通常只需要鉴别,不需要加密整个报文
- 原理: 基于单向散列(hash)函数的思想,从一段很长的明文重计算出一个固定长度的比特串
- 报文摘要应用于公开密钥加密系统-
[see fig 8-20]

Message Digests (2)



Digital signatures using message digests.

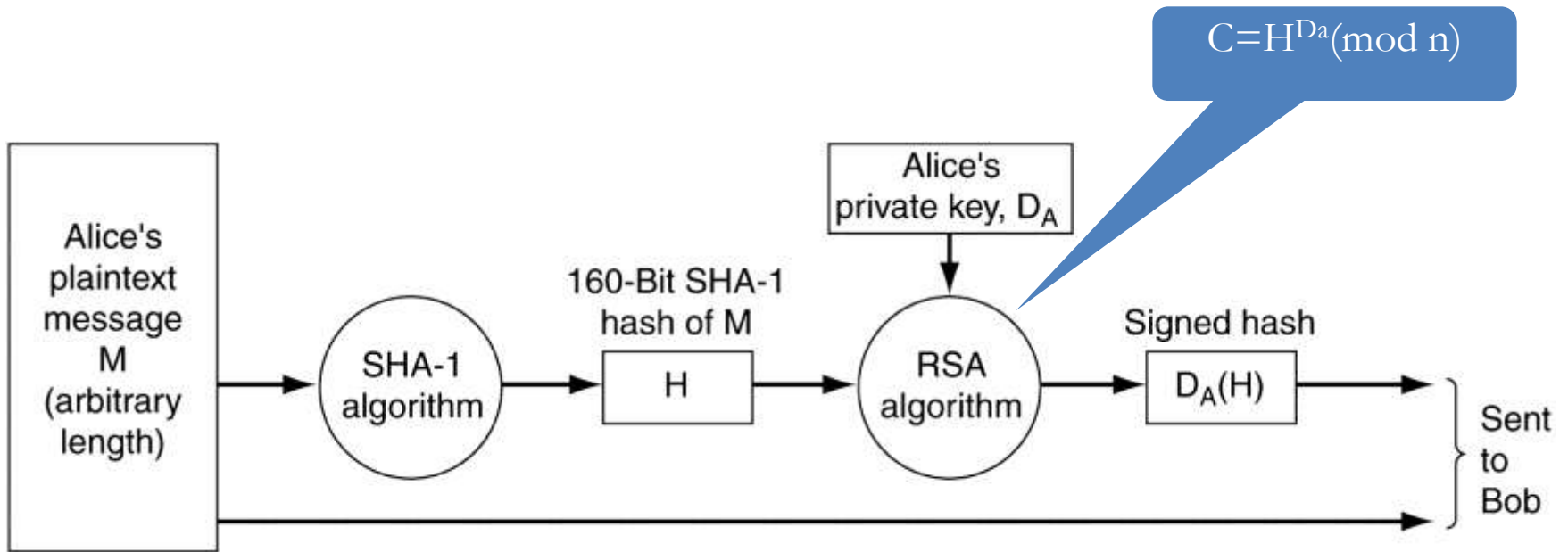
Message Digests (3)

- 四个重要属性：
 - 给出P, 就易于计算出 $MD(P)$
 - 只给出 $MD(P)$, 几乎不可能找出P
 - 不可能生成两条具有同样的报文摘要的报文, 即: $MD(P')=MD(P)$ (其中 $P \neq P'$)
 - 对输入数据的任何一点 (那怕是1bit) 将产生不同的输出结果

Message Digests (4)

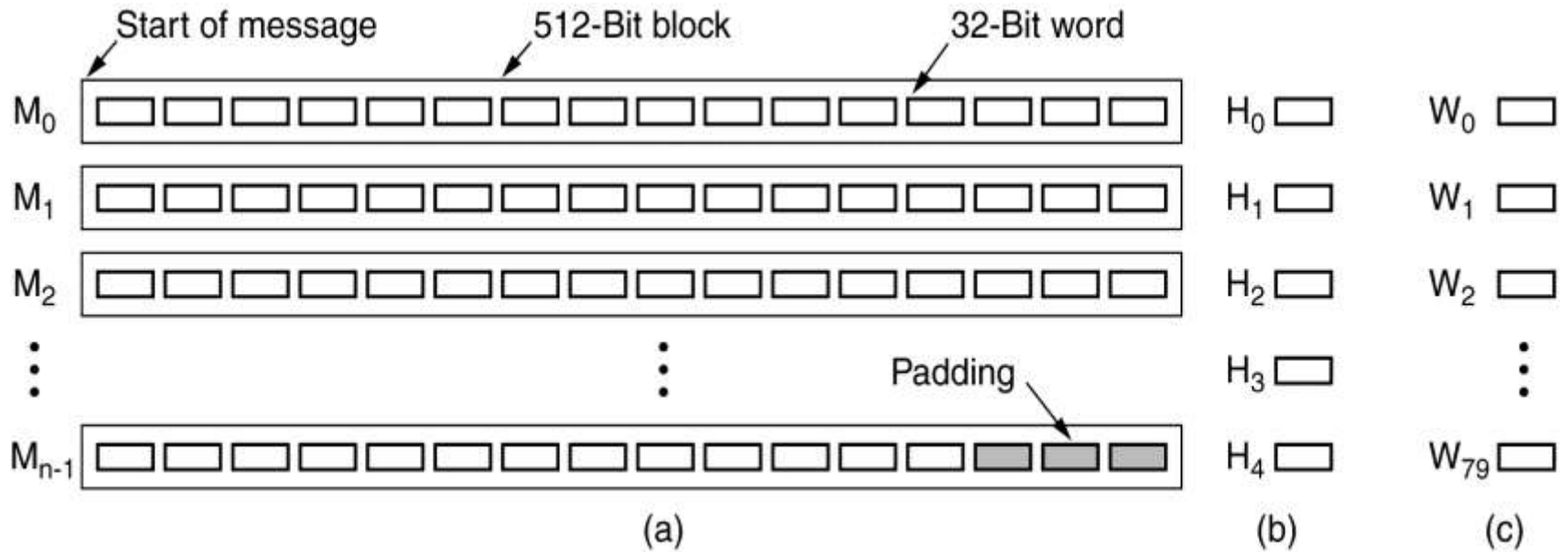
- 主要报文摘要算法
 - MD5 (Ronald Rivest, 1992)
 - 以512bit的块处理输入数据，得到128bit的报文摘要
 - One of most widely used
 - [密码学领域重大发现：山东大学王小云教授成功破解MD5](#) (2004年)
 - SHA-1 (NIST, 1993) 保密散列算法
 - Secure Hash Algorithm 1, developed by NSA,
 - 以512bit的块处理输入数据，得到160bit的报文摘要
 - [女解码高手王小云：十年破译五部顶级密码](#) (2005年)

SHA-1



Use of SHA-1 and RSA for signing nonsecret messages.

SHA-1 (2)



(a) A message padded out to a multiple of 512 bits.

(b) The output variables. (c) The word array.

SHA-1(3)

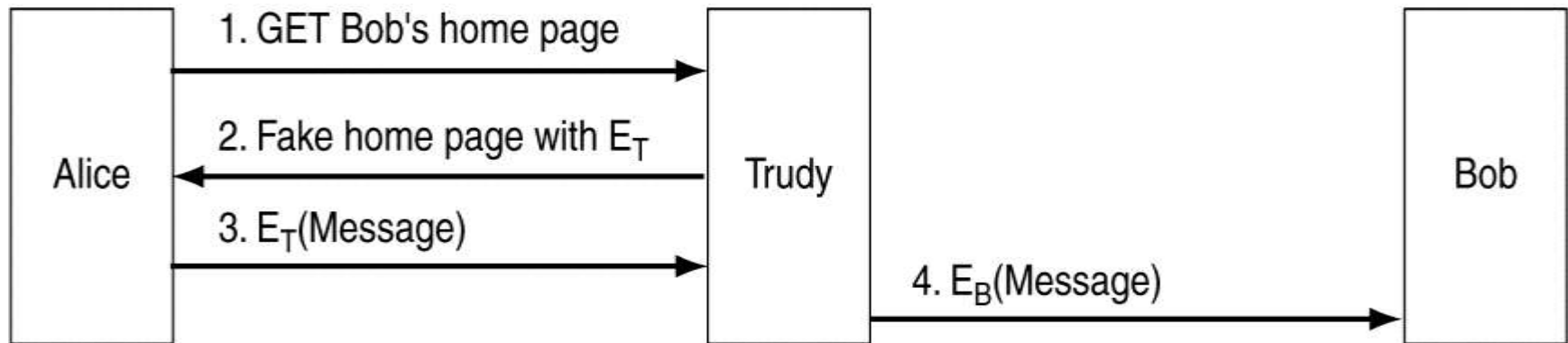
- 2005年，不幸又被王小云教授破解！
 - http://news.hexun.com/1674_1976631A.shtml
- SHA-2: a new version of SHA-1, produce hashes of 224, 256, 384 and 512 bits

8.5 Management of Public Keys (公开密钥的管理)

- Certificates
- X.509
- Public Key Infrastructures

- Problem: if Alice and Bob do not know each other, how do they get each other's public keys to start the communication process?
- Maybe Answer: put your public key on your Web site.
 - But it does not work.
 - Why? Trudy's intrude

Problems with Public-Key Encryption



A way for Trudy to subvert public-key encryption.

8.5.1 Certificates

I hereby certify that the public key

19836A8B03030CF83737E3837837FC3s87092827262643FFA82710382828282A

belongs to

Robert John Smith

12345 University Avenue

Berkeley, CA 94702

Birthday: July 4, 1958

Email: bob@superdupernet.com

SHA-1 hash of the above certificate signed with the CA's private key

A possible certificate and its signed hash.

A return result after clicking:

1. the **certificate** (public key, algorithm, key distribution center, identity, digest) and
2. the signature block (the signed SHA-1 hash of the certificate)

CA(数字证书认证中心) – Certification Authority: an organization that certifies public keys

8.5.2 X.509 (*)

Field	Meaning
Version	Which version of X.509
Serial number	This number plus the CA's name uniquely identifies the certificate
Signature algorithm	The algorithm used to sign the certificate
Issuer	X.500 name of the CA
Validity period	The starting and ending times of the validity period
Subject name	The entity whose key is being certified
Public key	The subject's public key and the ID of the algorithm using it
Issuer ID	An optional ID uniquely identifying the certificate's issuer
Subject ID	An optional ID uniquely identifying the certificate's subject
Extensions	Many extensions have been defined
Signature	The certificate's signature (signed by the CA's private key)

The basic fields of an X.509 certificate.

- **Approved by ITU (IETF version of X.509 is RFC 3280)**
- **Certificates are encoded using the OSI ASN.1**
- **Version 3**

证书的组成

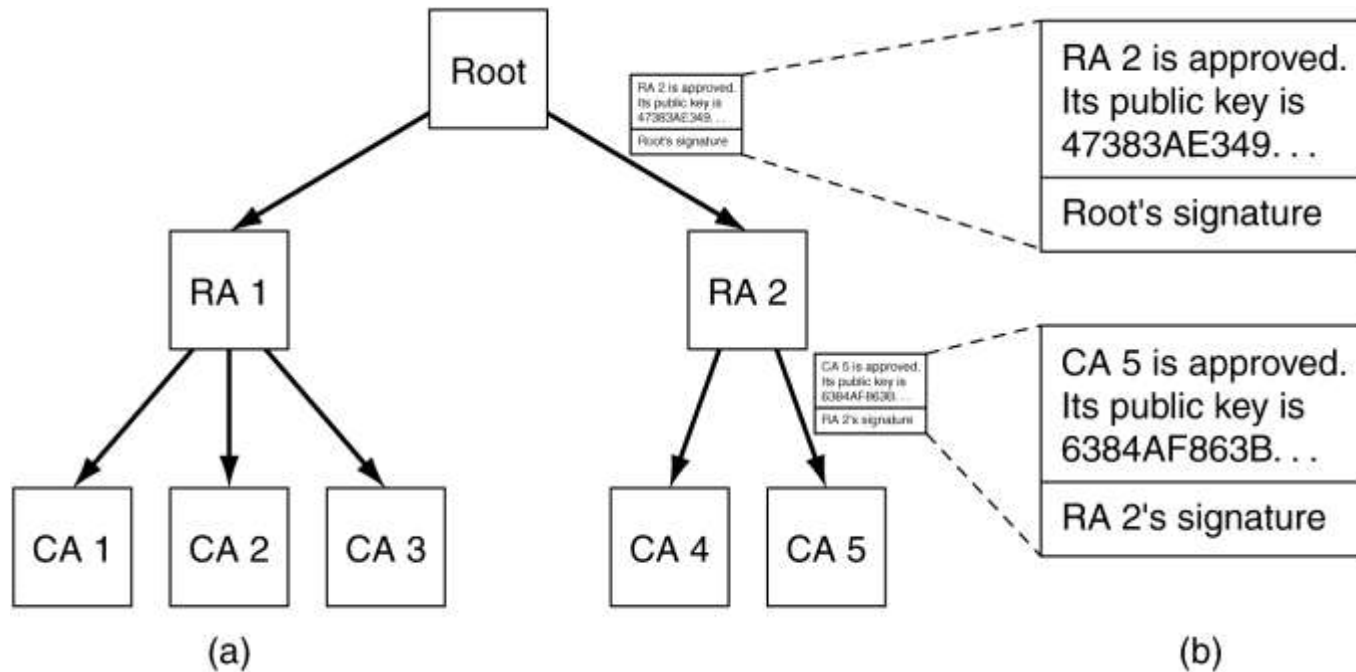
- 证书由以下两部分组成：
 - (1) 证书数据
 - 版本信息，用来与X.509的将来版本兼容；
 - 证书序列号，每一个由CA发行的证书必须有一个唯一的序列号；
 - CA所使用的签名算法；
 - 发行证书CA的名称；
 - 证书的有效期限；
 - 证书主题名称；
 - 被证明的公钥信息，包括公钥算法、公钥的位字符串表示；
 - 包含额外信息的特别扩展。
 - (2) 发行证书的CA签名
 - 证书第二部分包括发行证书的CA签名和用来生成数字签名的签名算法。任何人收到证书后都能使用签名算法来验证证书是由CA的签名密钥签发的

8.5.3 Public-Key Infrastructures

公开密钥基础设施

- Problem
 - Where to locate the CAs? Single or multiple?
 - Which organization to operate the CA?
 - Government or non government?
- Solution:
 - Goes under the PKI (Public Key Infrastructure, 公开密钥基础设施)
 - A PKI has multiple components:
 - Users, CAs, certificates, and directories
 - A hierarchical PKI
 - CA, RA (Regional Authorities, 区域权威机构), Root

Public-Key Infrastructures(2)



(a) A hierarchical PKI. (b) A chain of certificates.

Public-Key Infrastructures(3)

- A **chain of trust** or a **certification path**
 - A chain of certificates going back to the root
- Directories
 - Where to store the certificates?
 - Each user store his or her own certificates
 - Use DNS as a certificate directory
 - Dedicated directory servers
- Certificate Revocation(撤销/回)

Public-Key Infrastructures(4)

- **PKI: Public Key Infrastructure**
- **PKI的基本服务**
 - 认证身份
 - 完整性: 数字签名, MAC和HMAC
 - 保密性: 用公钥分发随机密钥, 用随机密钥加密数据
 - 不可否认: 发送方和接收方的不可否认
- **PKI中的证书**
 - 证书格式X.509
 - CA层次结构
 - 证书发放机制
 - 证书验证机制
 - 证书注销机制

8.6 Communication Security (通信的安全性)

- **IPsec (IP security, IP安全)**
- **Firewalls**
- **Virtual Private Networks(VPN)**
- **Wireless Security**

8.6.1 IPsec

➤ IPsec design

- Multiple services, algorithms and granularities(颗粒度)
- IPsec is in IP layer, but it's connection oriented, “connection” – SA (Security Association, 安全关联)

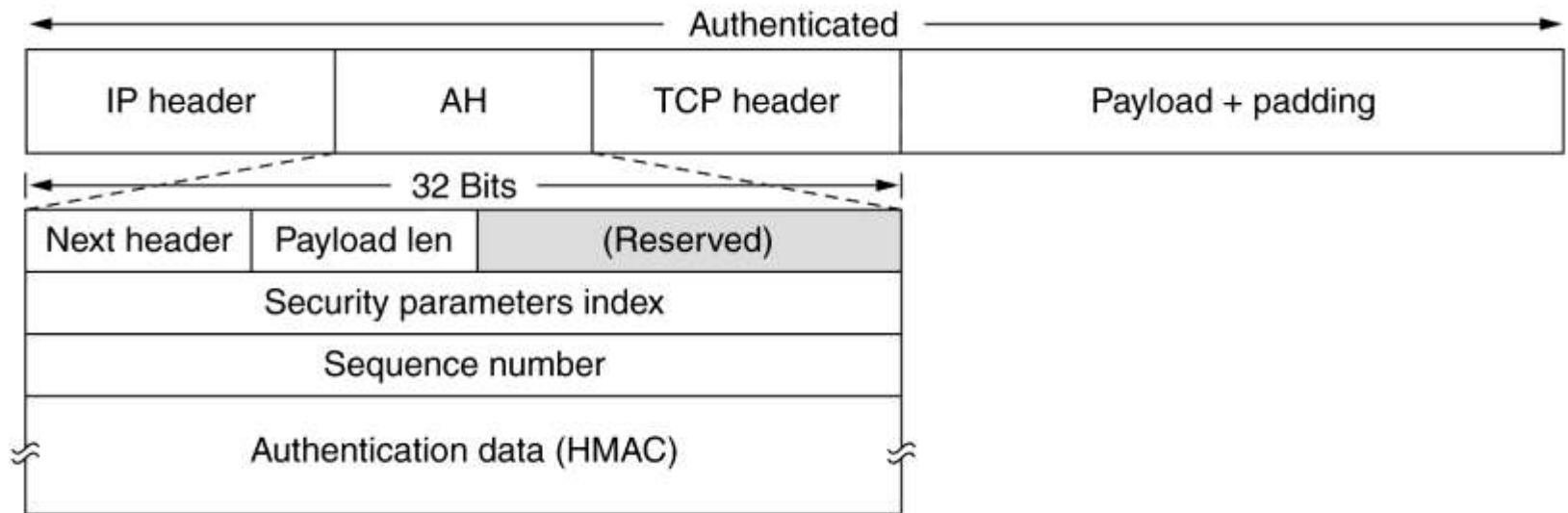
➤ IPsec has two main parts:

- New header (two kinds)
 - **AH** (Authentication Header, 认证头), or
 - **ESP** (Encapsulating Security Payload, 封装的安全净荷)
- ISAKMP (Internet Security Association and Key Management Protocol, 安全关联和密钥管理协议)
 - Deal with establishing keys
 - (extremely complex)

➤ IPsec has two use modes:

- **Transport mode** (传输模式) : IPsec header is inserted after IP header
- **Tunnel mode** (隧道模式) (have a new IP header): IP packet被放在一个全新的IP packet里面.

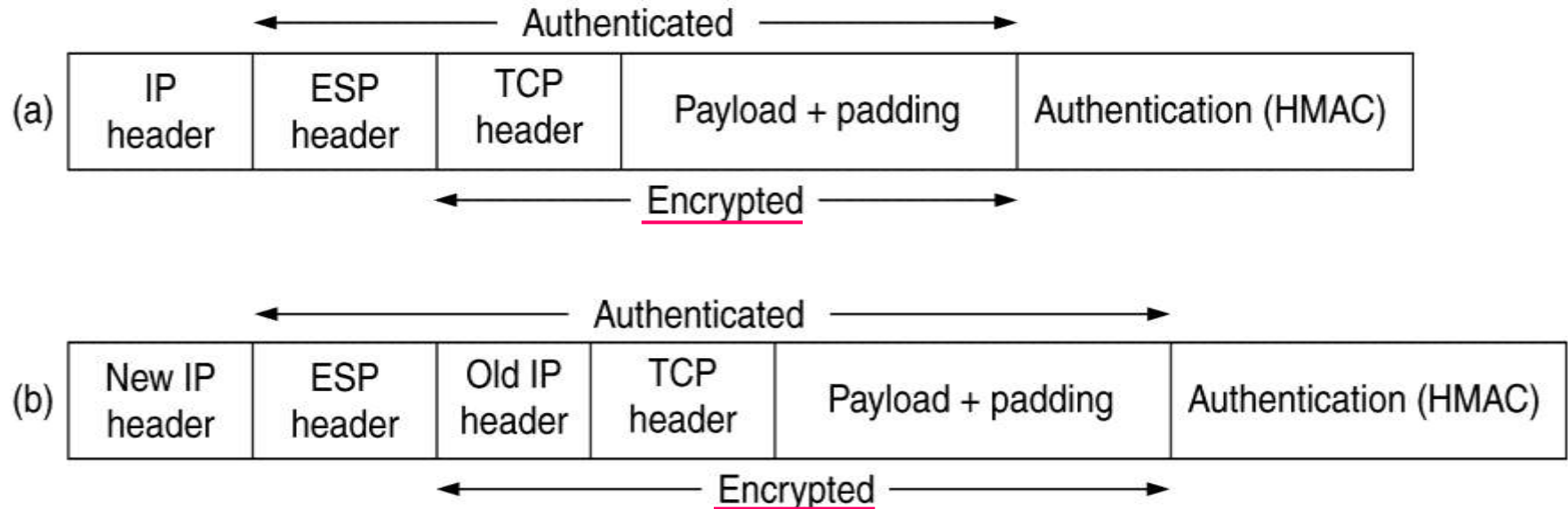
IPsec(2)--AH认证头



The IPsec authentication header (AH, 认证头) in **transport mode** for IPv4.

- **No data encryption**
- **Integrity checking**
- **Antireplay security**
- **HMAC: Hashed Message Authentication Code, 散列的消息认证码**

IPsec (3) -- ESP封装的安全净荷



(a) ESP in transport mode. (b) ESP in tunnel mode.

Protocol field in the IP header is changed

IPsec (4) — 总结

➤ IPsec: 网络层安全性

- 需求: 真实性(认证)、完整性和保密性
- IPsec的组成部分: 协议部分, 密钥管理

➤ 协议部分

- AH(Authentication Header): 完整性
- ESP(Encapsulating Security Payload): 保密性、完整性

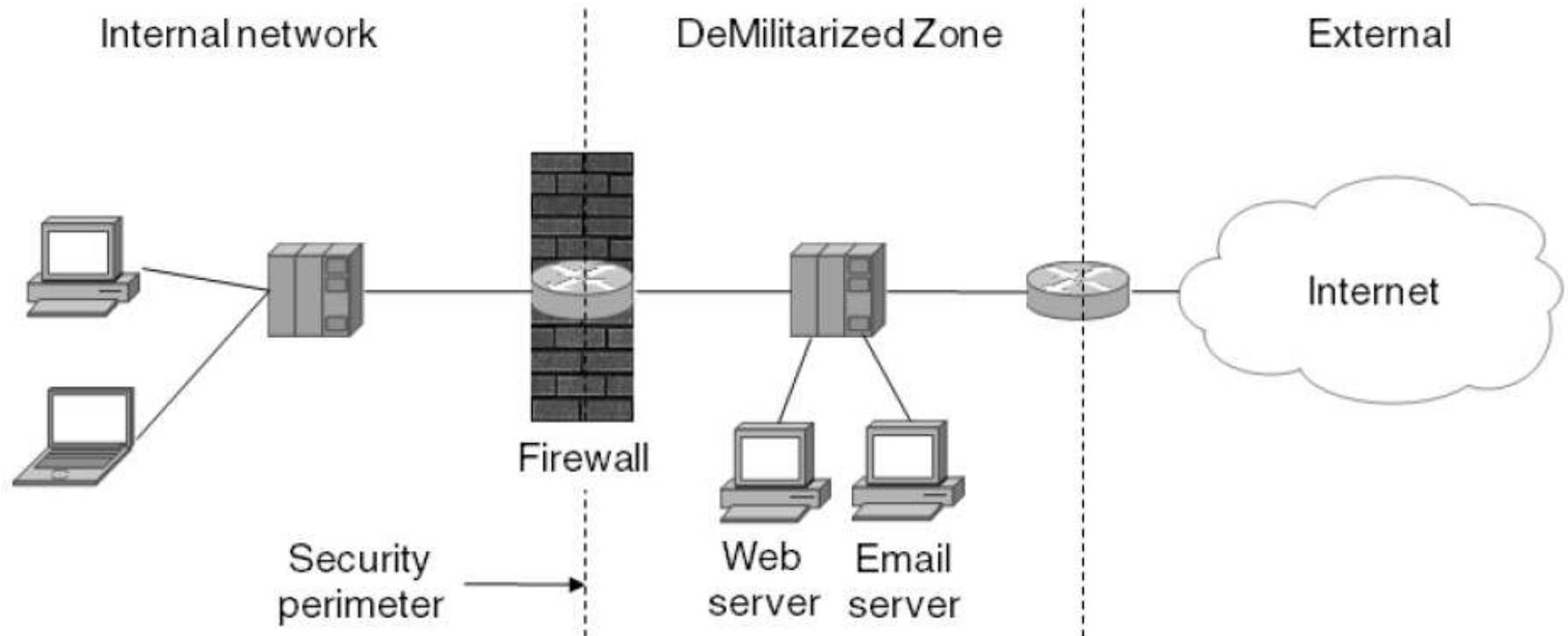
➤ SA(Security Association)

- 一个简单的单向逻辑连接, 安全信息参数集合
- SA的标识:SPI、目标IP地址、安全协议标识

➤ IPsec密钥管理

- ISAKMP: 是一个针对认证和密钥交换的框架
 - 两阶段协商
- IKE: The Internet Key Exchange
 - 基于ISAKMP框架, 结合了Oakley和SKEME的部分密钥交换技术
- Windows 2000中的IPsec

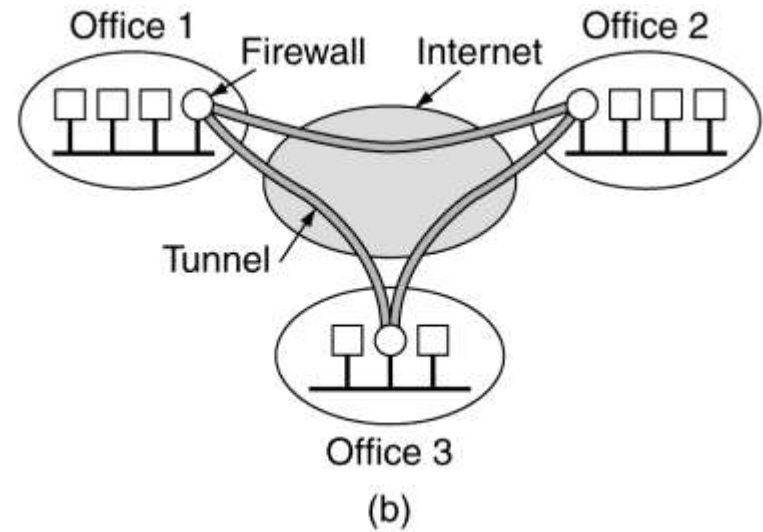
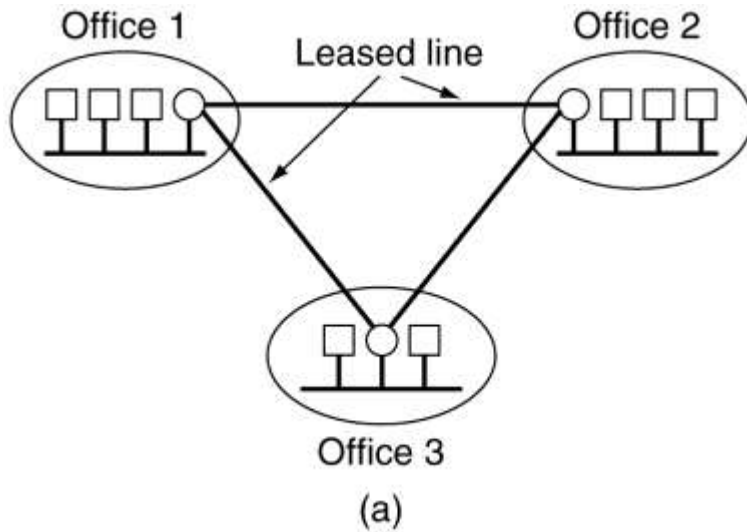
8.6.2 Firewalls



A firewall consisting of **two packet filters** and **an application gateway**.

- DoS (Denial of Service) attacks
- DDoS (Distributed Denial of Service) attacks (difficult to defend)

8.6.3 Virtual Private Networks



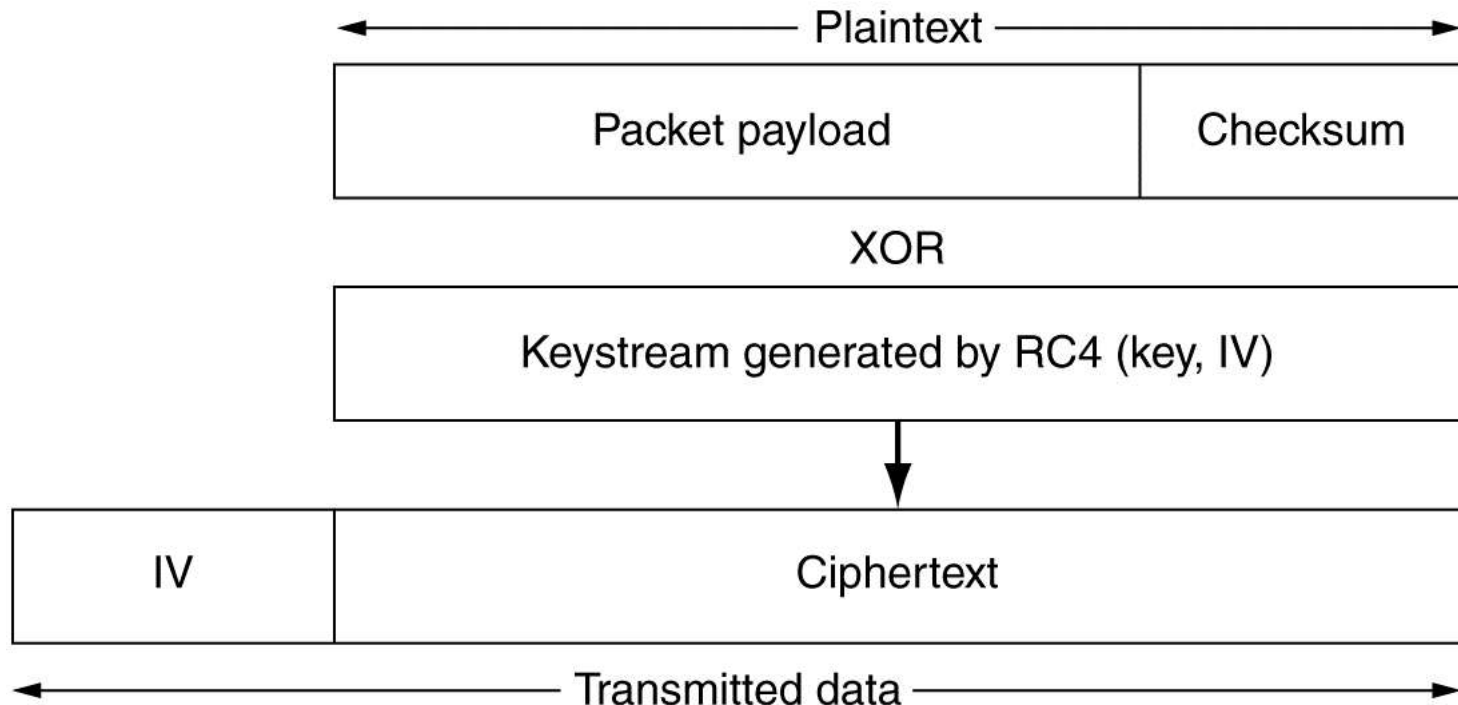
(a) A leased-line private network. (b) A virtual private network.

- A natural combination: firewalls, VPNs, and IPsec with ESP in tunnel mode

8.6.4 Wireless Security

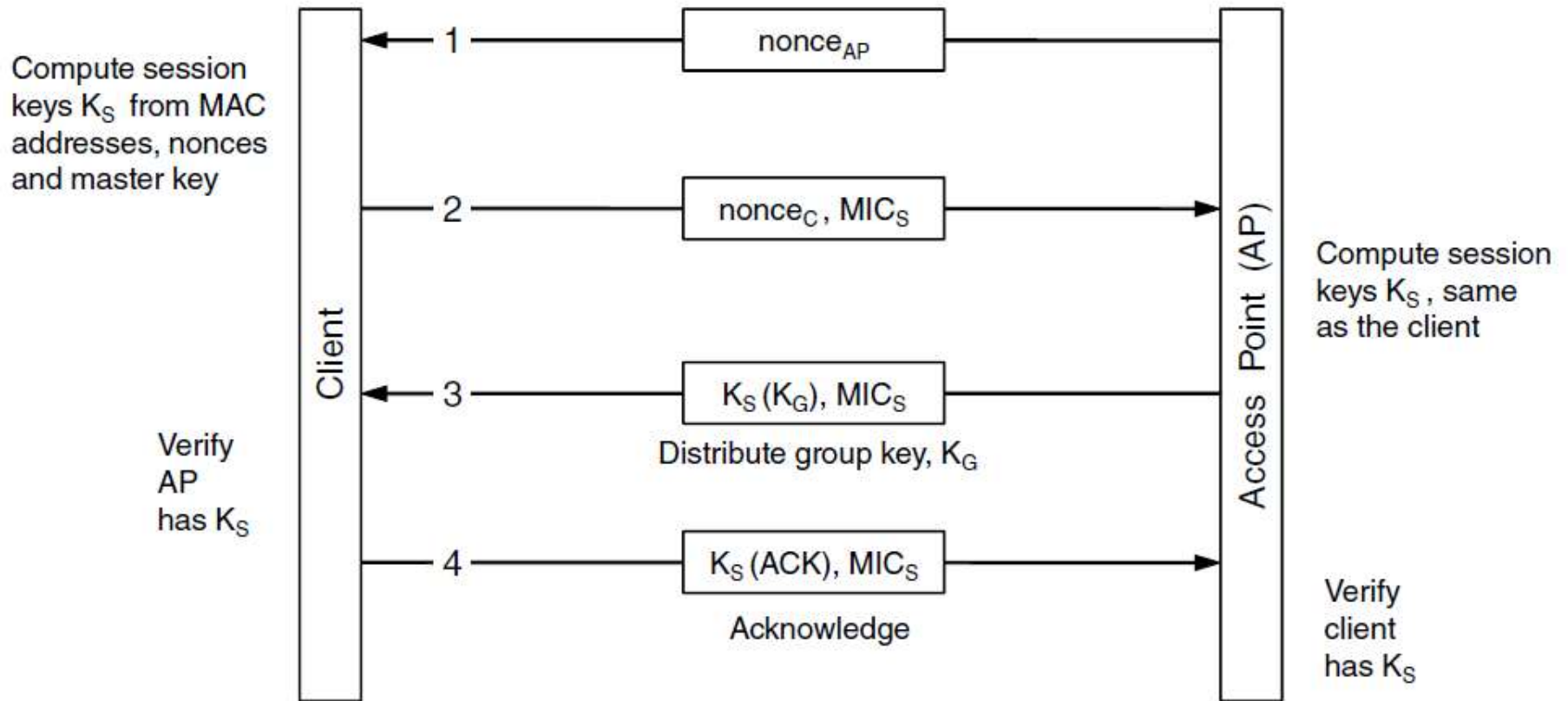
- 802.11 Security
 - WEP (Wired Equivalent Privacy, 相当于有线网的保密性),
hopeless
 - Next version 802.11i
- Bluetooth Security
 - Provide security in multiple layers
- WAP 2.0 Security
 - Based on well-known standards
 - Security services may better than 802.11 and bluetooth

802.11 Security



Packet encryption using WEP.
WEP: Wireless Equivalent Privacy

802.11i Security



The 802.11i key setup handshake

8.7 Authentication Protocols

(鉴别协议)

- **鉴别(Authentication)**: 验证通信对象是原定的那位而不是冒名顶替者的技术 (the technique by which a process verifies that its communication partner is who it's supposed to be and not an imposer)
 - 关心是否和一个特定的进程进行通信 (deals with the question of whether you are actually communicating with a specific process)
- **授权(Authorization)**: 允许进程或某实体作什么
 - is concerned with what that process or entity is permitted to do.
- 鉴别协议的基本模型
 - 主体、密钥分发中心KDC、会话密钥
- Session Key
 - Often use symmetric-key cryptography (AES or 3DES)

Authentication Protocols (2)

- Example:
 - Client process contacts a file server and say " I am Alice and I want to delete the file *cookbook.old*"
 - Server, two questions:
 1. Is this actually Alice's process (**authentication**)?
 2. Is Alice allowed to delete *cookbook.old* (**authorization**)?

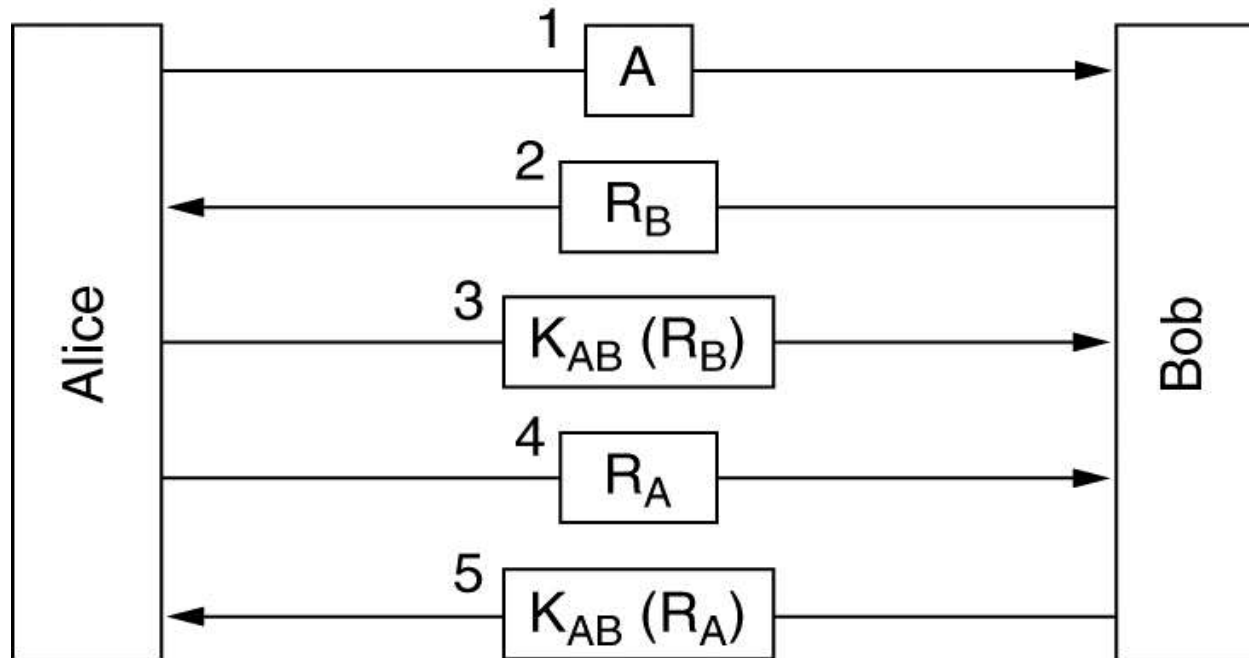
Authentication Protocols(3)

- Authentication Based on a Shared Secret Key
- Establishing a Shared Key: Diffie-Hellman
- Authentication Using a Key Distribution Center
- Authentication Using Kerberos
- Authentication Using Public-Key Cryptography

8.7.1 Authentication Based on a Shared Secret Key

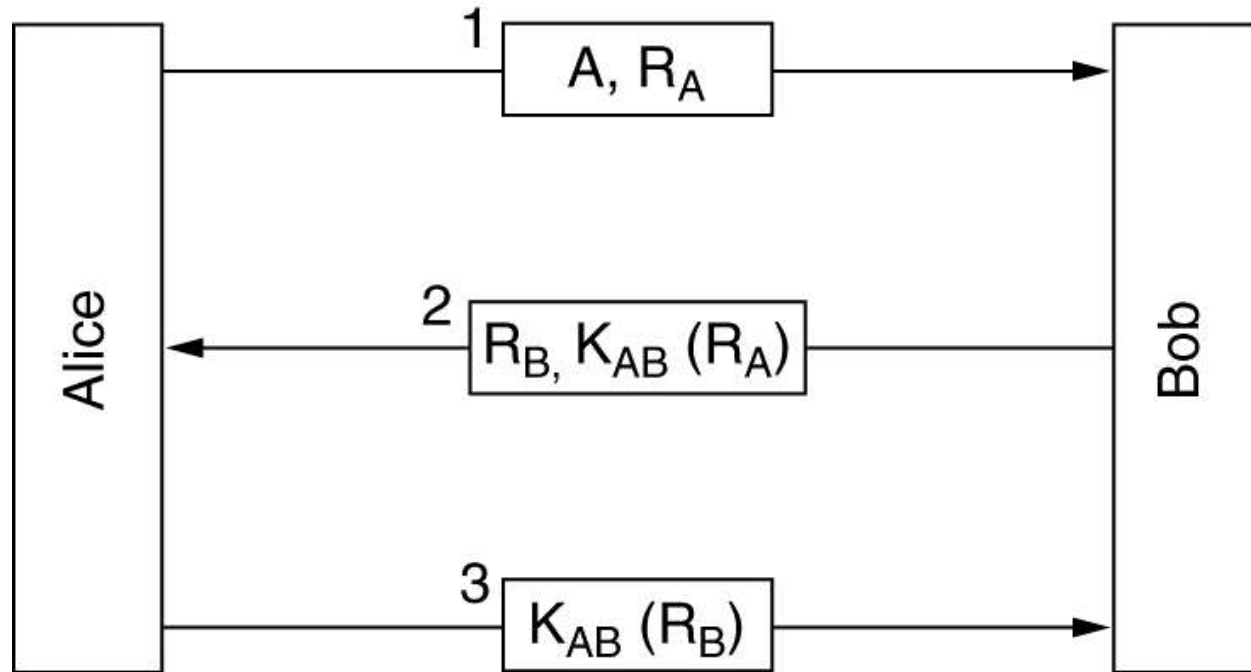
•基于共享秘密密钥的鉴别

- 原则：一方发送一个随机数给另一方，后者以一种特殊形式转换它并传回结果（查询-应答协议）
- 使用查询-应答协议的双向鉴别



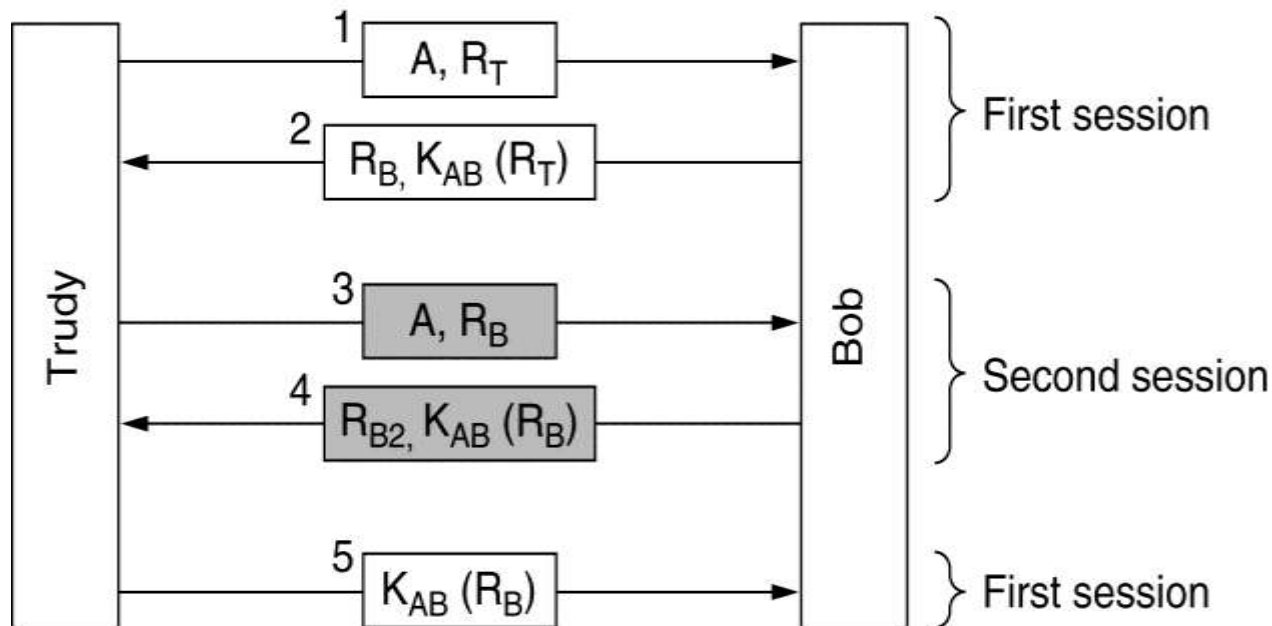
Two-way authentication using a challenge-response(查询—应答) protocol.

Authentication Based on a Shared Secret Key (2)



A shortened two-way authentication protocol.

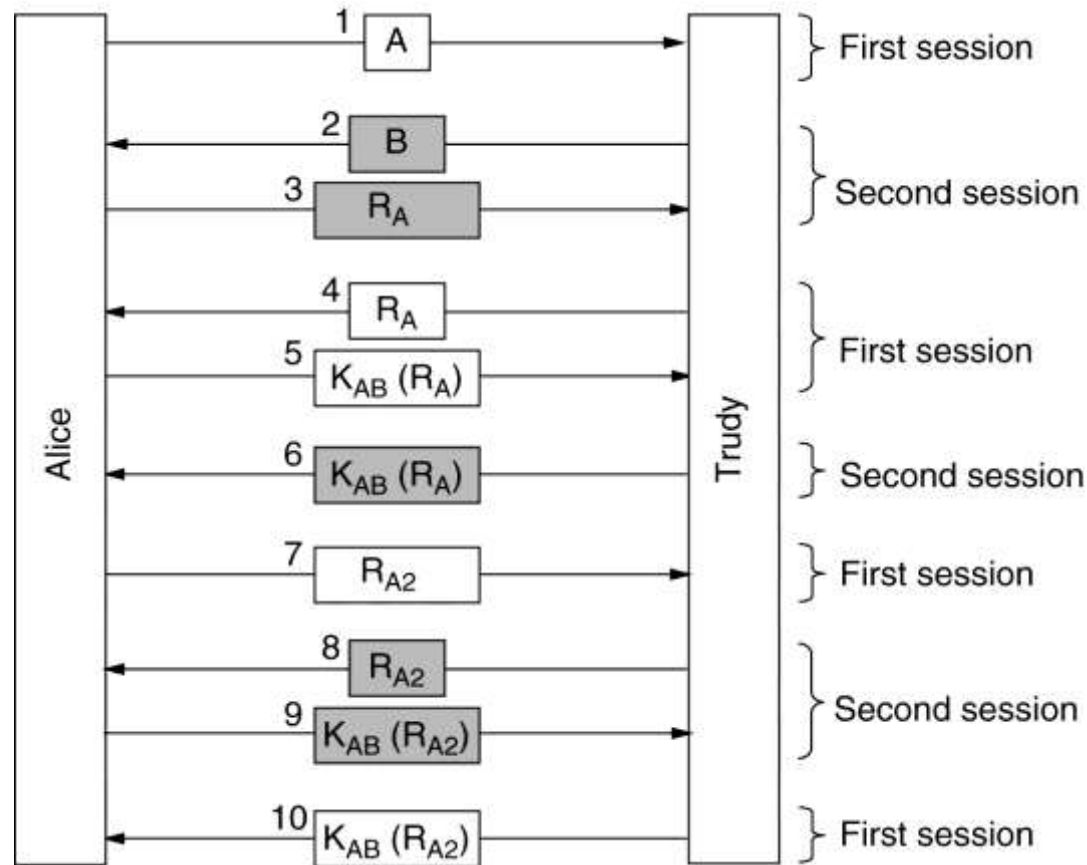
Authentication Based on a Shared Secret Key (3)



The reflection attack.

- 可通过“反射攻击”来挫败协议
- 教训：设计一个完善的鉴别协议比看上去要复杂

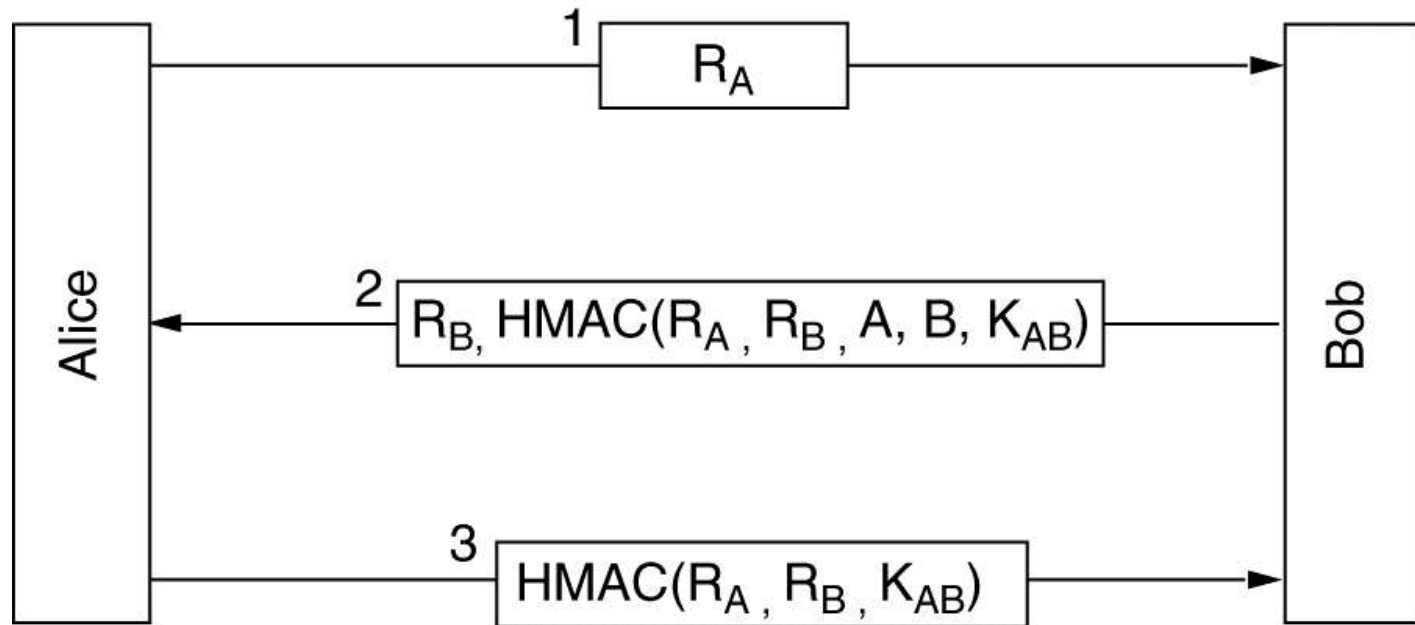
Authentication Based on a Shared Secret Key (4)



A reflection attack on the protocol of Fig. 8-32.

Finally, Trudy has two authenticated connections with Alice

Authentication Based on a Shared Secret Key (5)

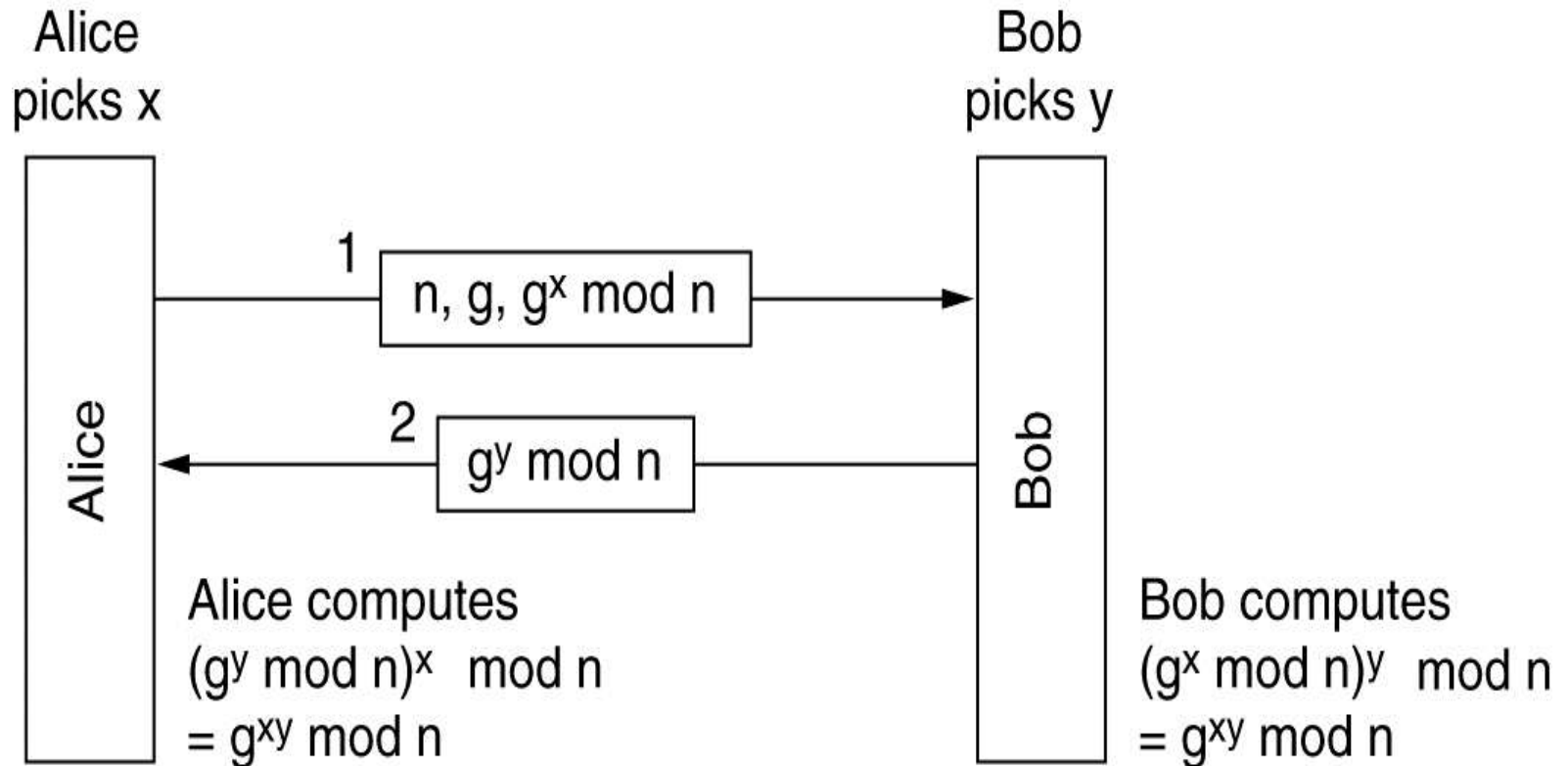


Authentication using HMACs.

8.7.2 Establishing a Shared Key

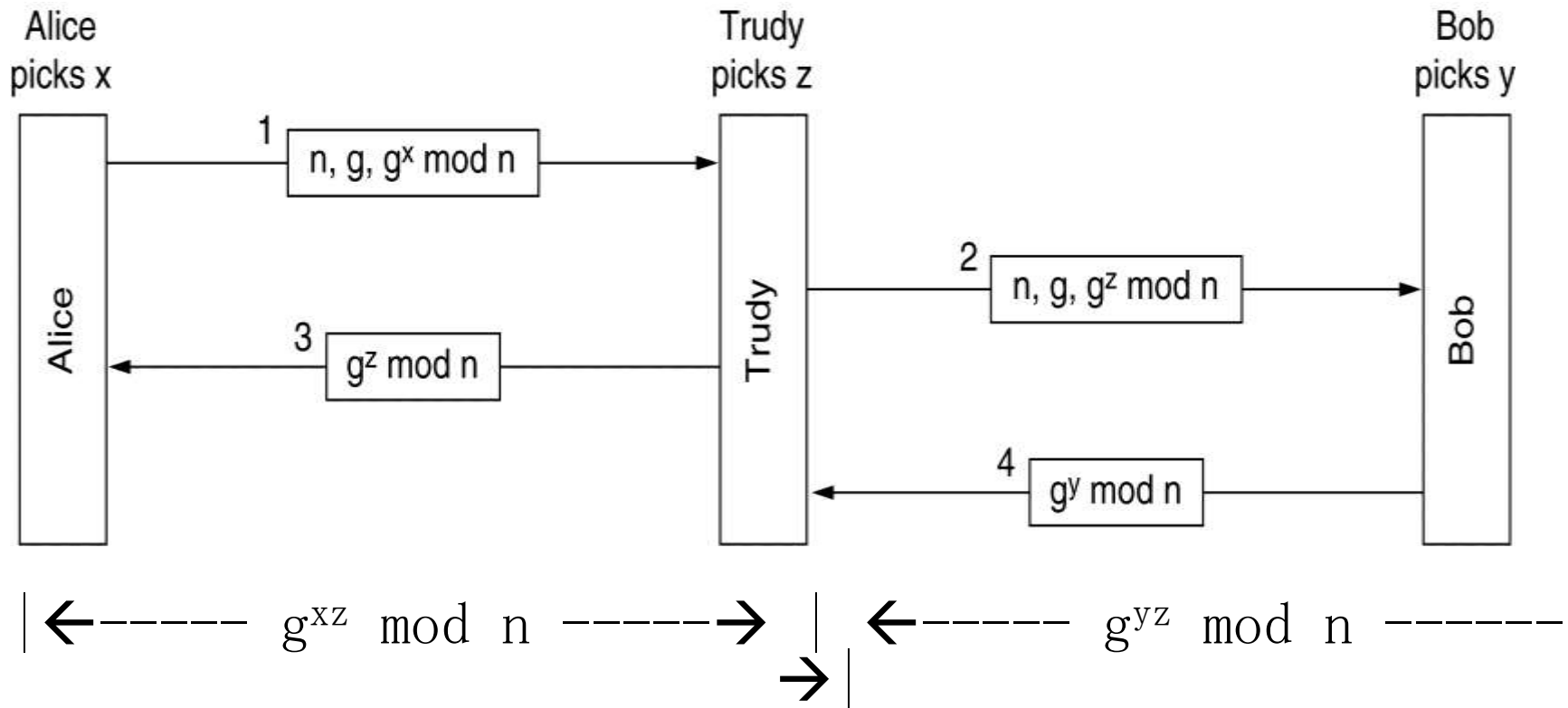
- Diffie-Hellman key exchange: the protocol that allows strangers to establish a shared secret key
- 建立一个共享密钥：Diffie-Hellman密钥交换协议
 - 原理：
 - 公开选择大质数 n , g , $(n-1)/2$, $(g-1)/2$
 - 一方秘密选择 x , 另一方选择 y
 - 然后开始交换密钥

Establishing a Shared Key: The Diffie-Hellman Key Exchange(2)



The Diffie-Hellman key exchange.

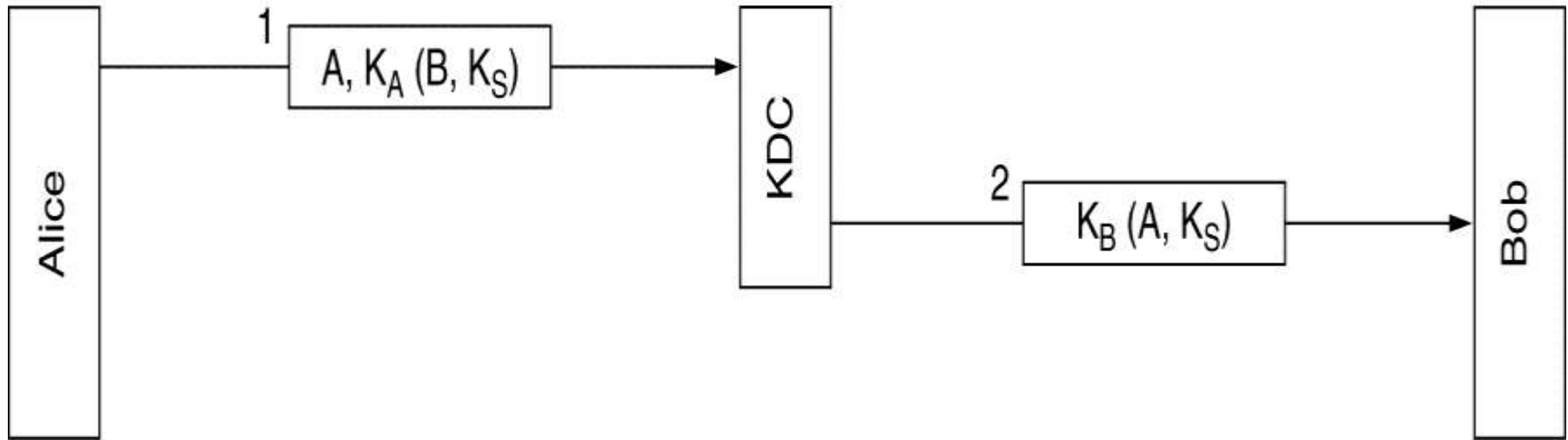
Establishing a Shared Key: The Diffie-Hellman Key Exchange



The **bucket brigade** or **man-in-the-middle attack**.

(传递队列攻击, 也称 (水桶队列攻击) 或 (中间人攻击))

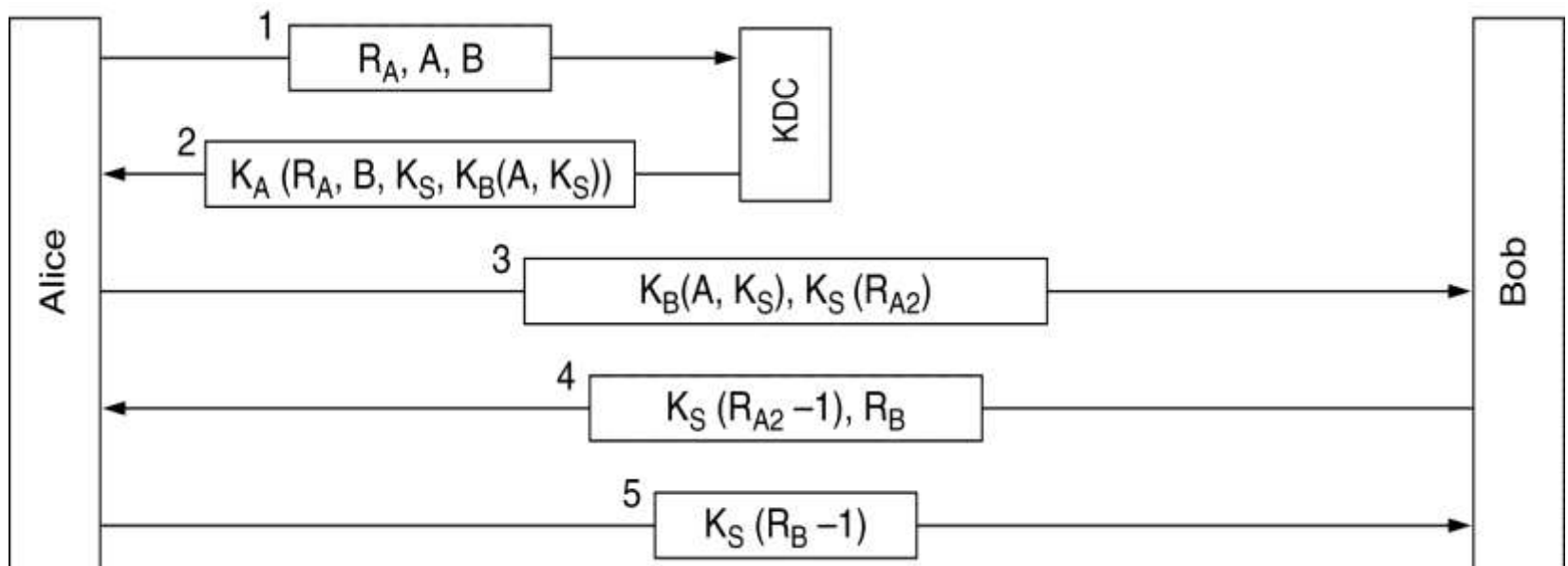
8.7.3 Authentication Using a Key Distribution Center



A first attempt at an authentication protocol using a KDC.

- 最简单的KDC鉴别协议：大嘴蛙协议
- 存在严重缺陷 – 重发攻击(即将KDC给Bob的信息重复发送)
- 解决办法：
 - 可通过加时间戳来解决
 - 在每条信息中放置一个一次的、唯一的信息号，“暂时号”
 - 或将上述两种方法组合起来

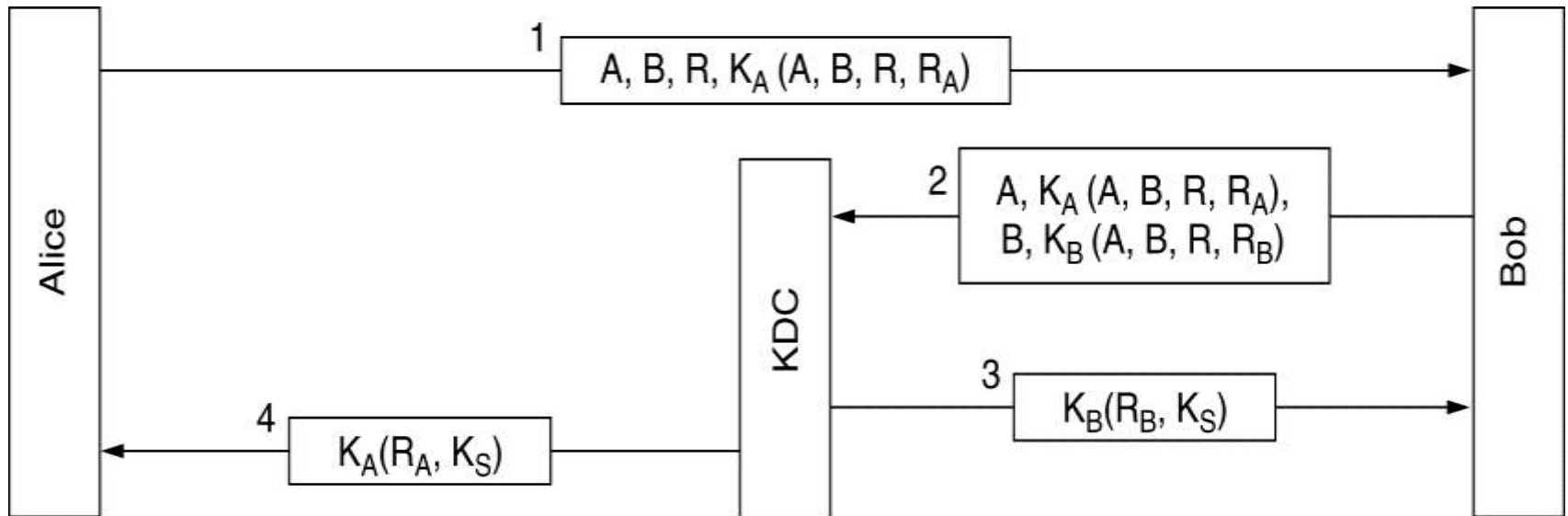
Authentication Using a Key Distribution Center (2)



The Needham-Schroeder authentication protocol.

多路查问-应答协议 - 一个更成熟的方法

Authentication Using a Key Distribution Center (3)



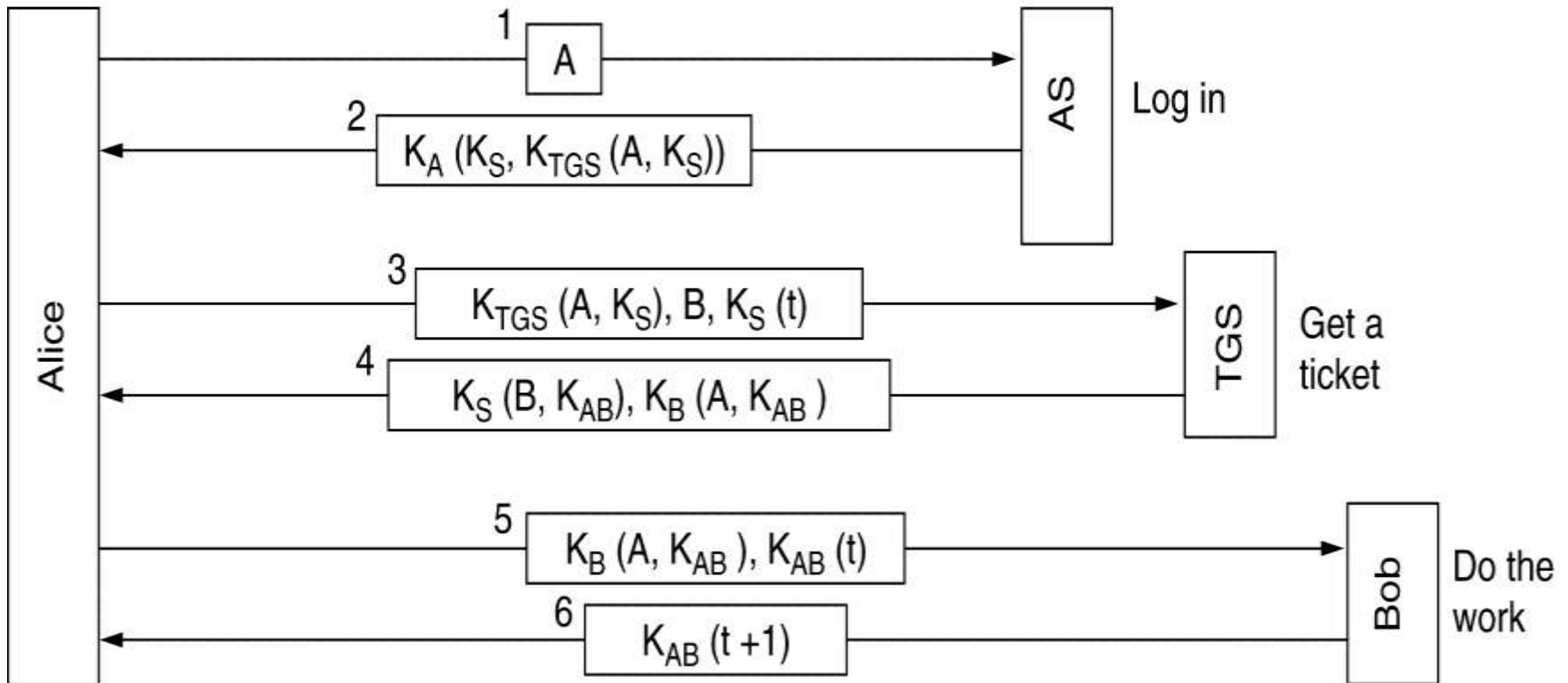
The Otway-Rees authentication protocol (slightly simplified).

多路查问-应答协议 – 一个更成熟的方法

8.7.4 Authentication Using Kerberos

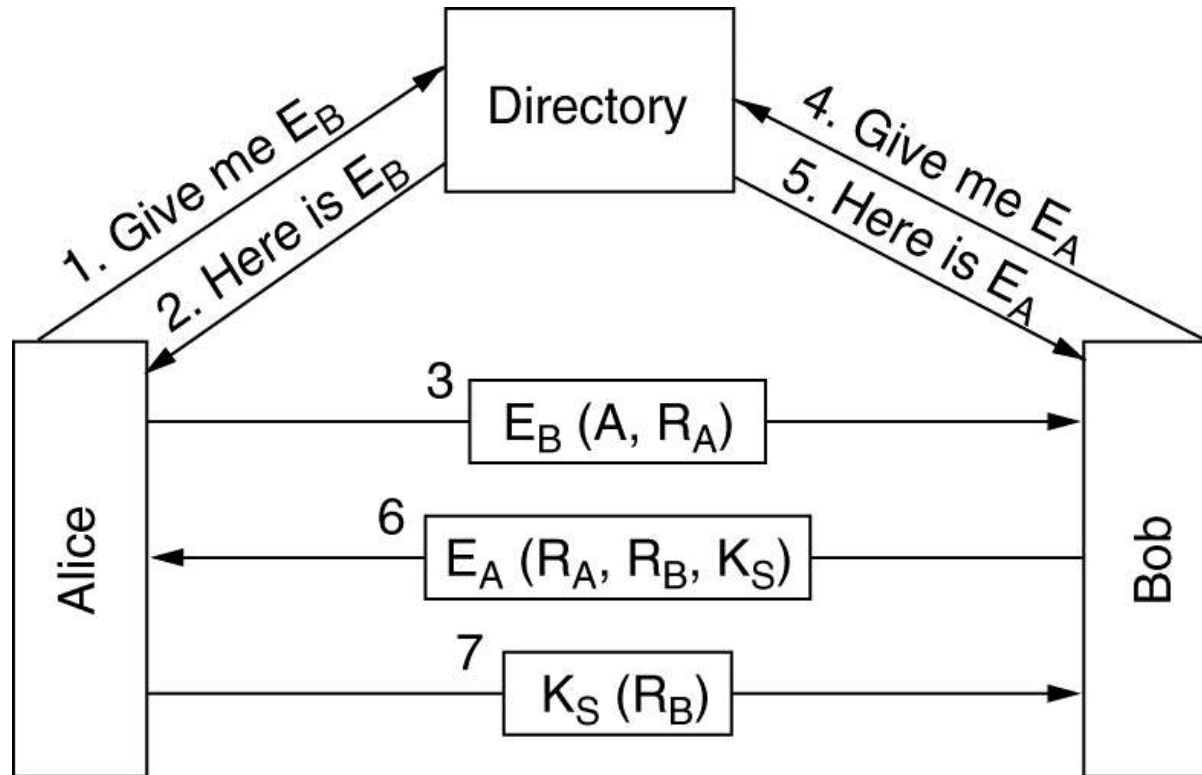
- 基于Needham-Schroeder鉴别协议的变形
- 区别：假定所有的时钟都是完全同步的
- 工作原理-[see fig 8-42]
- 除客户工作站外，使用三个服务器：
 - 鉴别服务器(AS)：在登录时验证用户身份
 - 授予许可证服务器(TGS)：发放“身份证明许可证”
 - 工作服务器：工作的实际执行者
- 特点：攻击困难（因为加了时间戳）、口令简单

Authentication Using Kerberos



The operation of Kerberos V4.

8.7.5 Authentication Using Public-Key Cryptography



Mutual authentication using public-key cryptography.

- 重要条件(缺点): 假定双方都知道对方的公开密钥 或者 通过目录(Directory)服务器去获得对方的公开密钥
- 特点: 攻击困难

8.8 E-Mail Security

- PGP – Pretty Good Privacy
- PEM – Privacy Enhanced Mail (*)
- S/MIME (*)

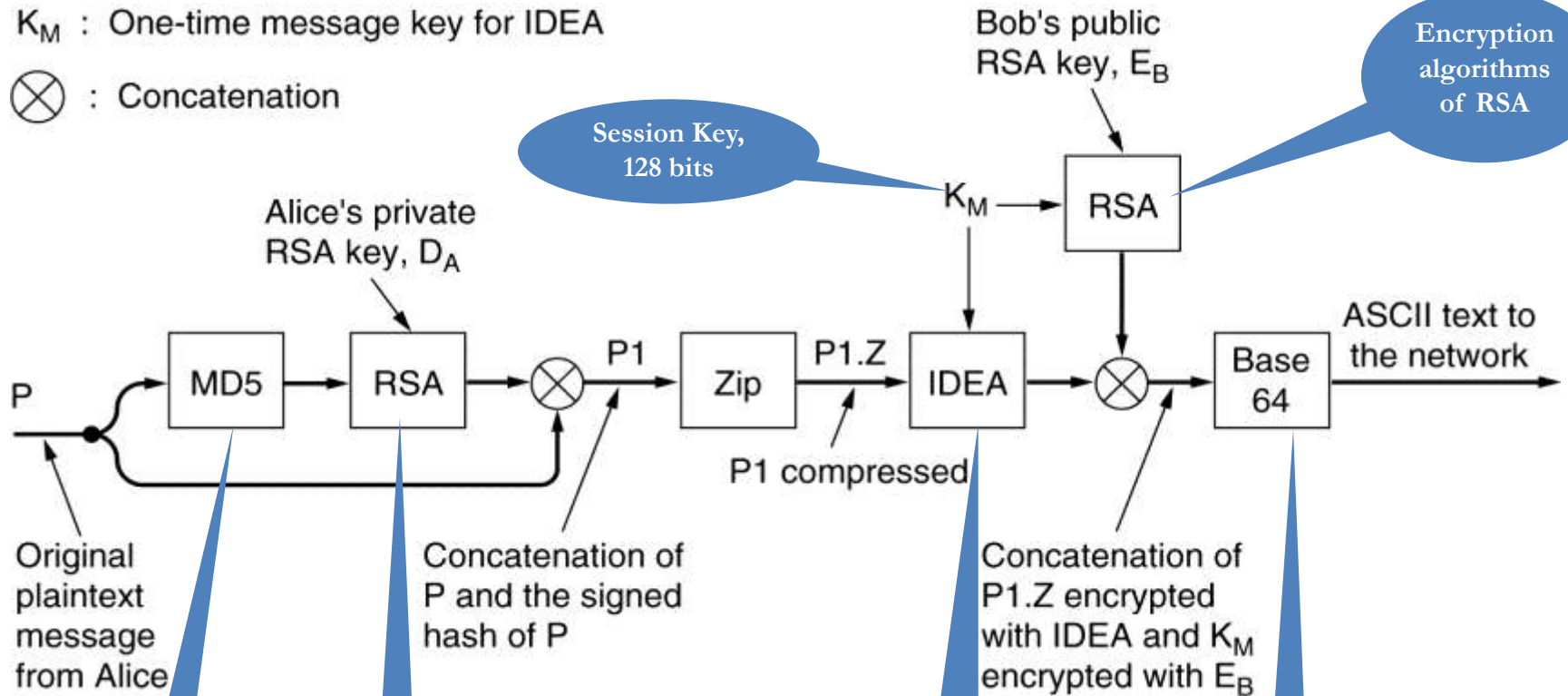
8.8.1 PGP – Pretty Good Privacy

- A complete e-mail security package
- Released in 1991, all in an easy-to-use form
- Provide:
 - Privacy
 - Authentication
 - Digital signatures
 - Compression
- Encrypt data → **IDEA** (International Data Encryption Algorithm)
 - Similar to **DES** and **AES**
- Key management → **RSA**
- Data integrity → **MD5**

8.8.1 PGP – Pretty Good Privacy

K_M : One-time message key for IDEA

\otimes : Concatenation



PGP in operation for sending a message.

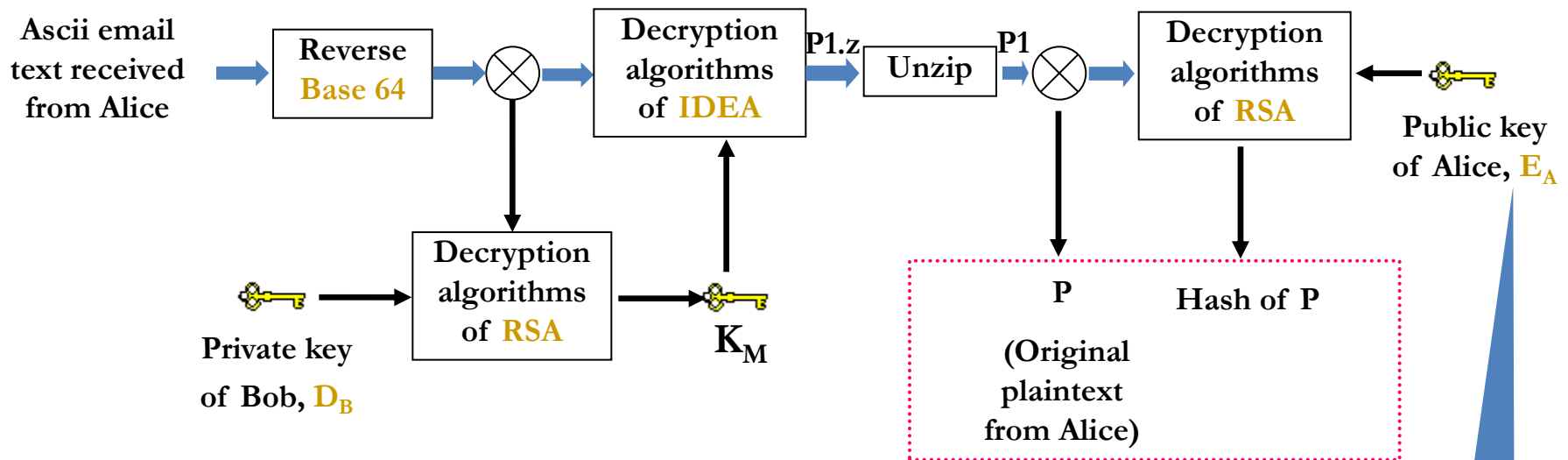
Hash
algorithms
of MD5

Encryption
algorithms
of RSA

Encryption
algorithms of
IDEA, DES
or AES

Encode
algorithms
of Base64

PGP – Pretty Good Privacy(2)

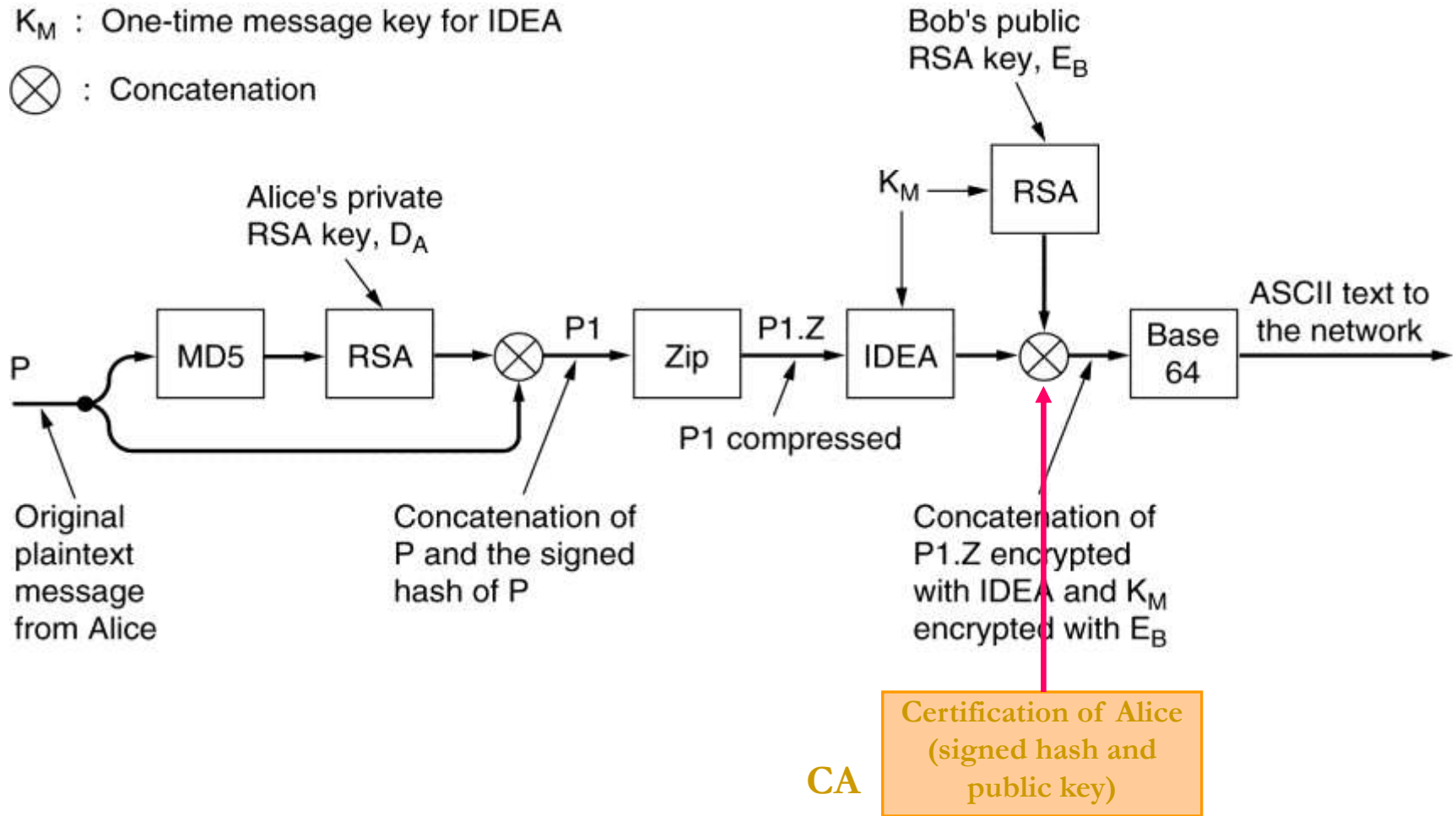


PGP in operation for receiving a message.

PGP – Pretty Good Privacy(3)

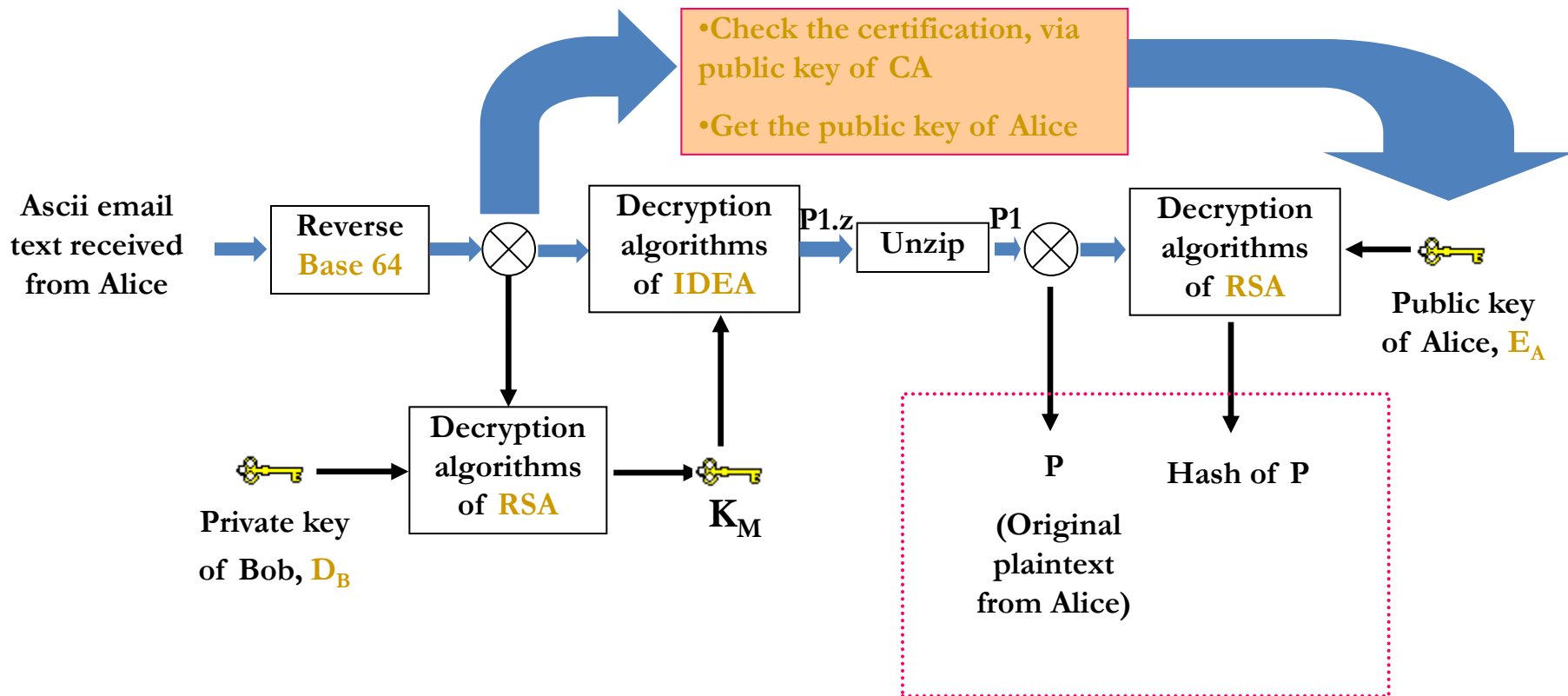
K_M : One-time message key for IDEA

\otimes : Concatenation



PGP with CA in operation for sending a message.

PGP – Pretty Good Privacy(4)

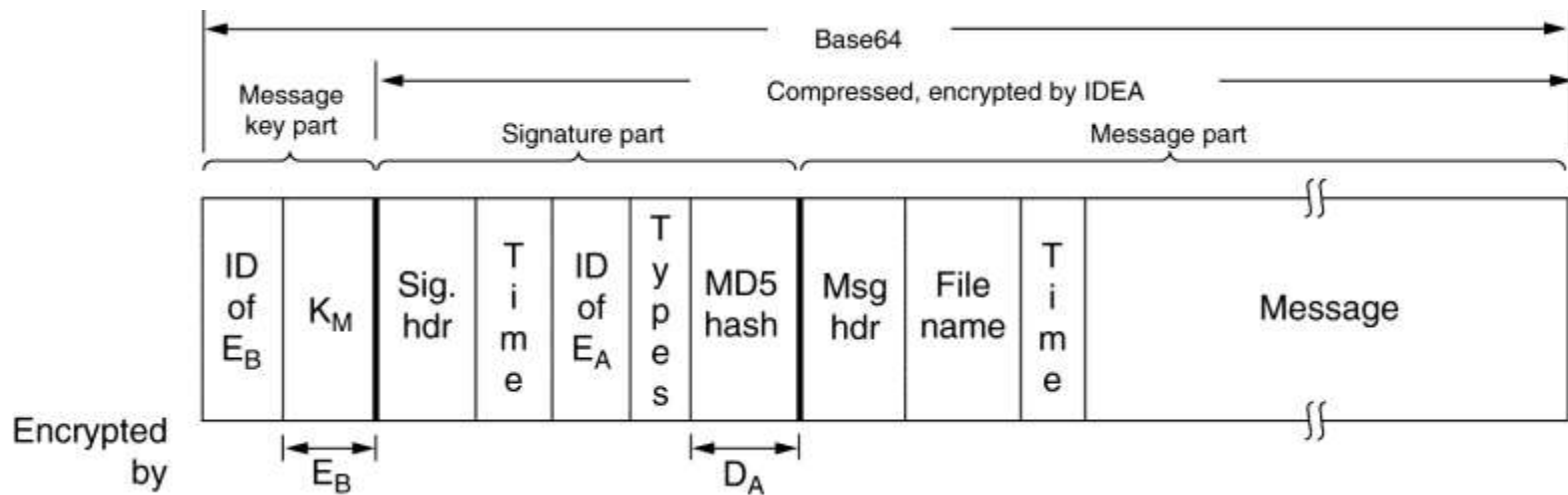


PGP with CA in operation for receiving a message.

PGP – Pretty Good Privacy (5)

- PGP supports four RSA key lengths:
 - Casual (384 bits): can be broken easily today
 - Commercial (512 bits): breakable
 - Military (1024 bits): not breakable by anyone on earth
 - Alien (2048 bits): not breakable by anyone on other planets, either

PGP – Pretty Good Privacy (6)



Format of a classic PGP message.

8.9 Web Security

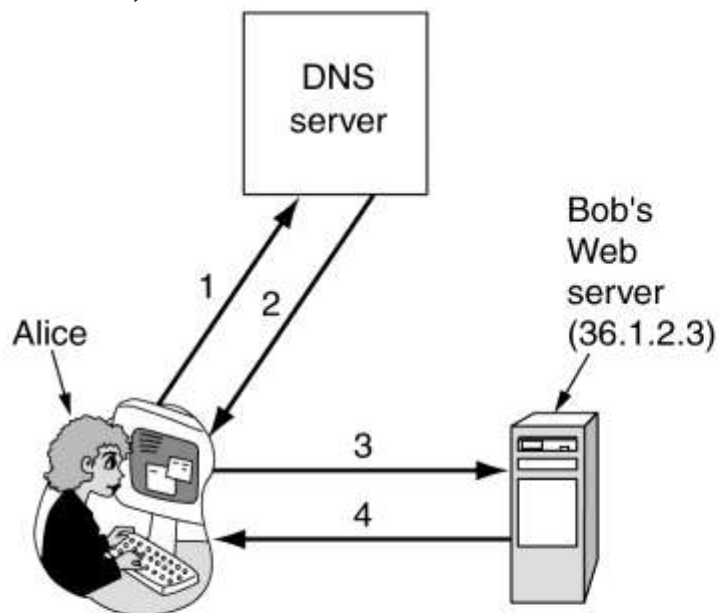
- Threats
- Secure Naming
- **SSL** – The Secure Sockets Layer
- Mobile Code Security

8.9.1 Threats

- Hackers, great programmers, crackers
- DoS
- DDoS
- Examples
 - 1999, a Swedish cracker broke into Microsoft's Hotmail web and created a mirror
 - A 19-year-old Russian cracker broke into an e-commerce web site and stole 300,000 credit card numbers,...
blackmail \$100,000
 - A 23-year-old California student emailed false news about the Emulex Corporation and caused the company's stock dropped by 60% and stockholders to lose over \$2 billion
 - ...

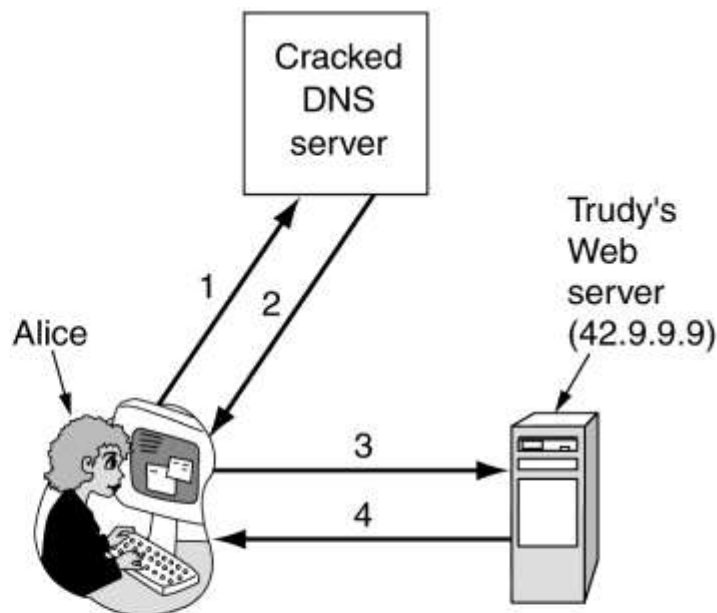
8.9.2 Secure Naming (*)

- DNS Spoofing(欺骗) a **poisoned cache**中毒的缓存 (with a false IP address)



1. Give me Bob's IP address
2. 36.1.2.3 (Bob's IP address)
3. GET index.html
4. Bob's home page

(a)



1. Give me Bob's IP address
2. 42.9.9.9 (Trudy's IP address)
3. GET index.html
4. Trudy's fake of Bob's home page

(b)

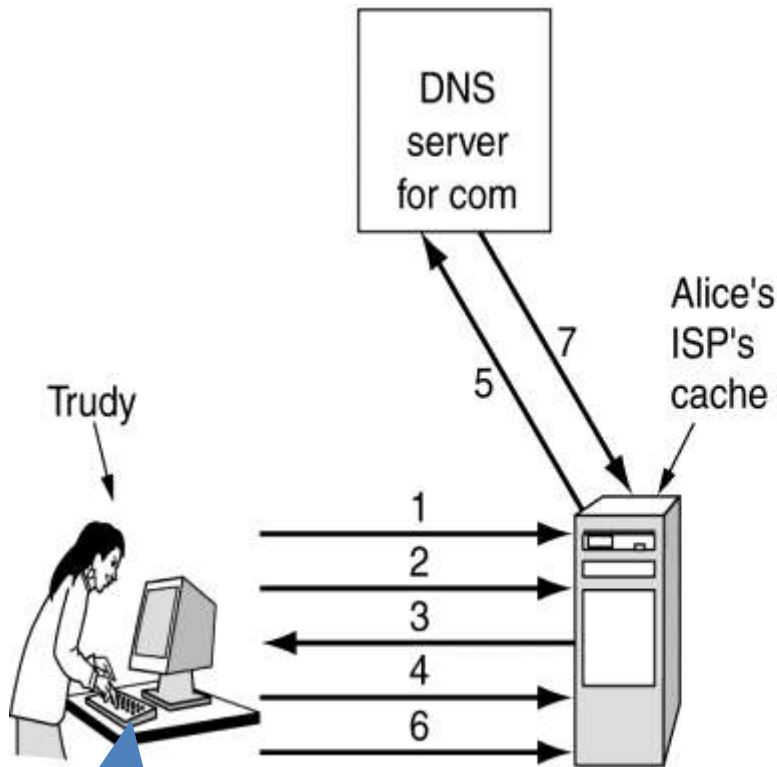
(a) Normal situation. (b) An attack based on breaking into DNS and modifying Bob's record.

Secure Naming (2)

挫败这种欺骗的方式:

用随机的而非递增的ID (or sequence number)

•DNS Spoofing(欺骗)



1. Look up foobar.trudy-the-intruder.com (to force it into the ISP's cache)
2. Look up www.trudy-the-intruder.com (to get the ISP's next sequence number)
3. Request for www.trudy-the-intruder.com (Carrying the ISP's next sequence number, n)
4. Quick like a bunny, look up bob.com (to force the ISP to query the com server in step 5)
5. Legitimate query for bob.com with $\text{seq} = n+1$
6. Trudy's forged answer: Bob is 42.9.9.9, $\text{seq} = n+1$
7. Real answer (rejected, too late)

How Trudy spoofs(欺骗) Alice's ISP.

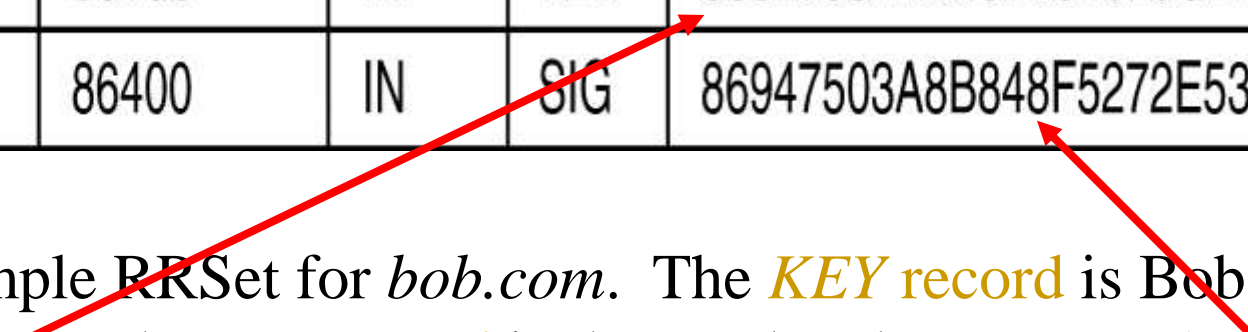
Trudy is also [dns.trudy-the-intruder.com](#) for step 3

Secure DNS

- 一种保护名字的方法
- **DNSsec (DNS security), RFC 2535---1994**
 - Has not been fully deployed yet
 - All information sent by a DNS server is signed with the originating zone's private key (发起域的私钥), so the receiver can verify its authenticity(真实性)
- **DNSsec offer three services:**
 - Proof of where the data originated
 - Public key distribution
 - Transaction and request authentication
- **RRSets (Resource Record Sets资源记录集): DNS records are grouped into sets**
 - **KEY record**: holds the public key of a zone, user, host or other principal
 - **SIG record**: holds the signed hash according the algorithm specified in the KEY record

Secure DNS(2)

Domain name	Time to live	Class	Type	Value
bob.com.	86400	IN	A	36.1.2.3
bob.com.	86400	IN	KEY	3682793A7B73F731029CE2737D...
bob.com.	86400	IN	SIG	86947503A8B848F5272E53930C...



An example RRSet for *bob.com.* The **KEY** record is Bob's public key. The **SIG** record is the top-level *com* server's signed hash of the A and KEY records to verify their authenticity.

Self-Certifying Names (自证明的名字)

- 另一种保护名字的方法:

- self-certifying name (or self-certifying URL)

Server SHA-1 (Server, Server's Public key) File name

⏟ ⏟ ⏟

http://www.bob.com:2g5hd8bfjkc7mf6hg8dgany23xds4pe6/photos/bob.jpg

A self-certifying URL containing a SHA-1 hash of server's name and public key,
160bits(=32chars[0-9, a-z但去掉了0, 1, o, l]x5bits/char).

- 在Alice使用时，先向bob Server获取Public Key，然后通过SHA-1计算出散列值；
- 如果与链接中包含的散列值相同则所访问的Server是正确的。

8.9.3 SSL—The Secure Sockets Layer

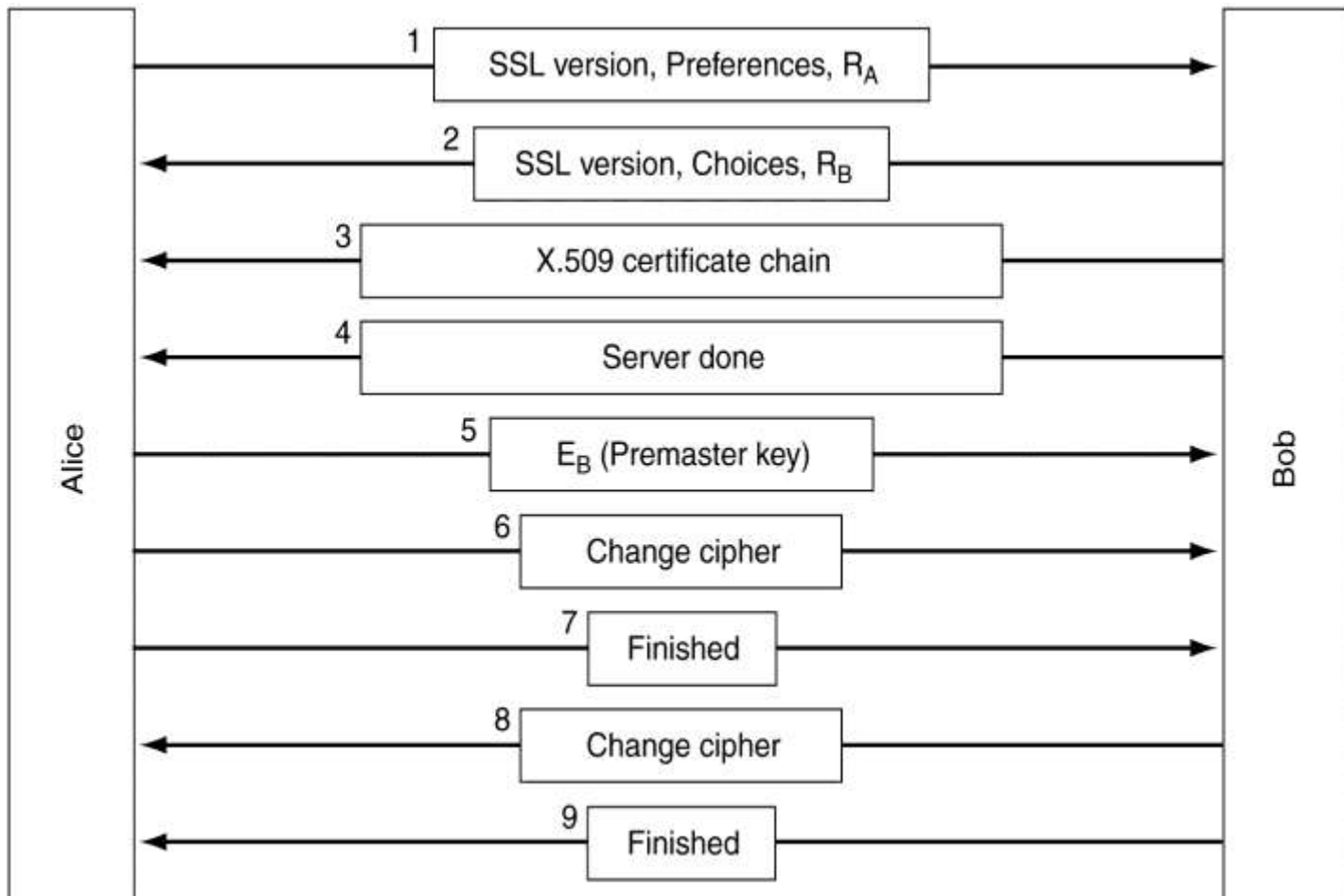
- In 1995, Netscape responded by introducing the security package, called SSL, to meet the security requirement
- **SSL builds a secure connection between two sockets**, including:
 - Parameters negotiation between client and server
 - Mutual authentication of client and server
 - Secret communication
 - Data integrity protection
- **HTTPS (Secure HTTP):** HTTP used over SSL
 - Use port number **443**, instead of the standard port 80 (**有时候**)
 - Version 3 we discussed here
 - Support different algorithms(3DES/SHA-1—for banking, RC4/MD5—for e-commerce) and options (compression, cryptographic algorithms)
- **SSL consist of two sub-protocols (2个子协议) :**
 - One for establishing a secure connection— (**建立安全的连接**)
 - One for using it -- (**使用安全的连接**)

SSL(2)

Application (HTTP)
Security (SSL)
Transport (TCP)
Network (IP)
Data link (PPP)
Physical (modem, ADSL, cable TV)

Layers (and protocols) for a home user
browsing with SSL.

SSL(3)

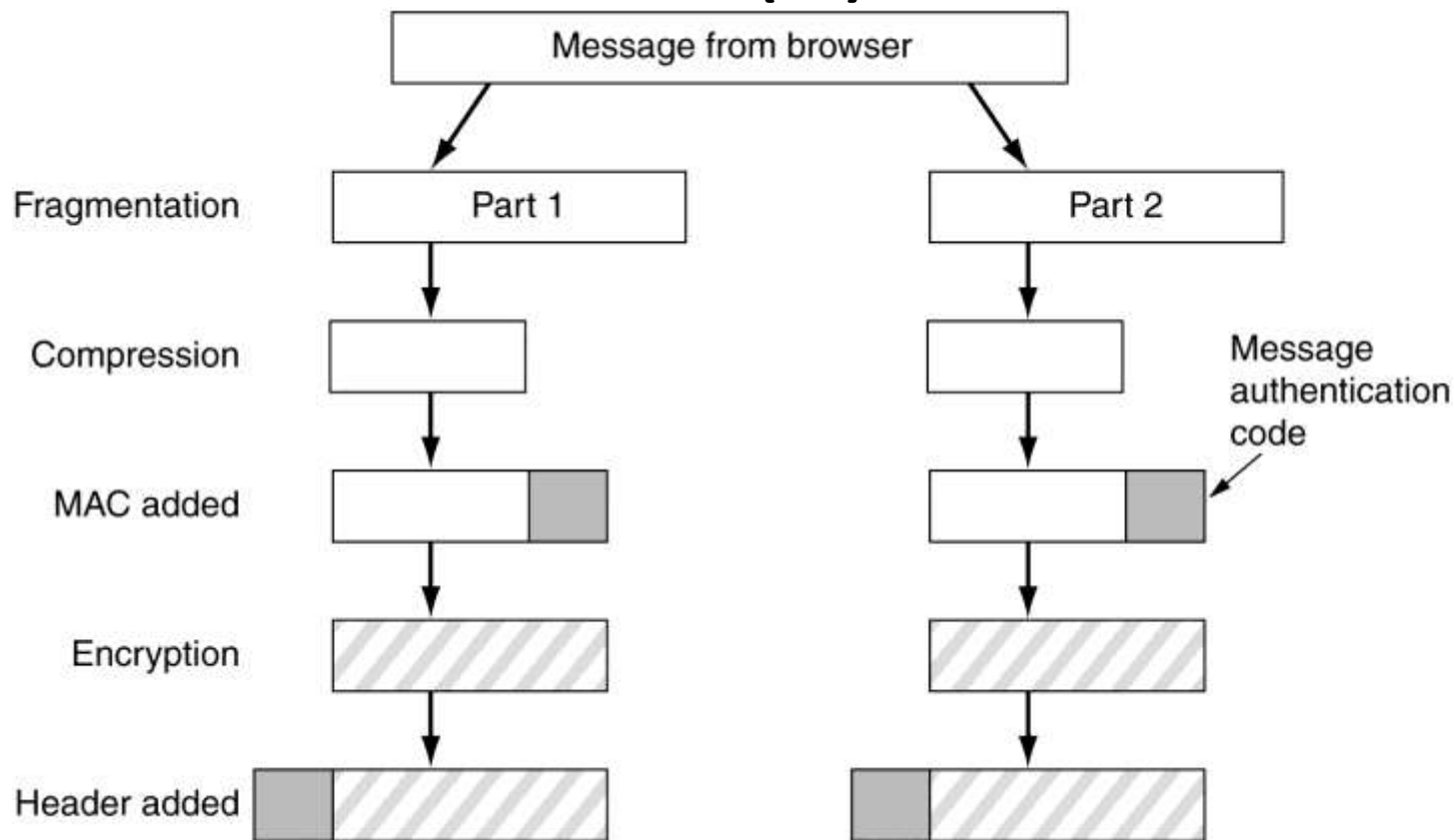


A simplified version of the **SSL connection establishment sub-protocol**.
(SSL连接建立子协议的简化版本)

SSL(4)

- **A SSL connection establishment sub-protocol**
 - Step 1: Alice send a request to Bob establish a connection with her preferences(compression and cryptographic algorithms) and a nonce R_A
 - Step 2: Bob makes a choice and a R_B
 - Step 3: Bob sends a certificate containing his public key
 - Step 4: Bob tell Alice that he is done and it's her turn
 - Step 5: Alice send Bob a random 384-bit premaster key (预设主秘钥) which encrypted with Bob's public key
 - --- Now both Alice and Bob can compute the session key
 - Step 6/7: Alice tells Bob to switch to the new cipher and finished establishment sub-protocol
 - Step 8/9: Bob acknowledges her and finish the sub-protocol
- **How does Bob know who Alice is ?**
 - Bob will ask for Alice to log in using a previously established login name and password, which is outside the scope of SSL

SSL(5)



Data transmission using SSL. (数据传输子协议)

- 加密密码由双方各自在建立连接时通过 R_A 、 R_B 和 **PreMasterKey** 计算得到
- 每个 **Fragmentation** 最大 **16KB**
- **MAC**—消息认证码 (**Message Authentication Code**), **MD5 Hash**
- 用 **RC4** 加密 [**Default**] (**128位** = (**88位公开** + **40位**) 不安全), or **3DES**, **AES**

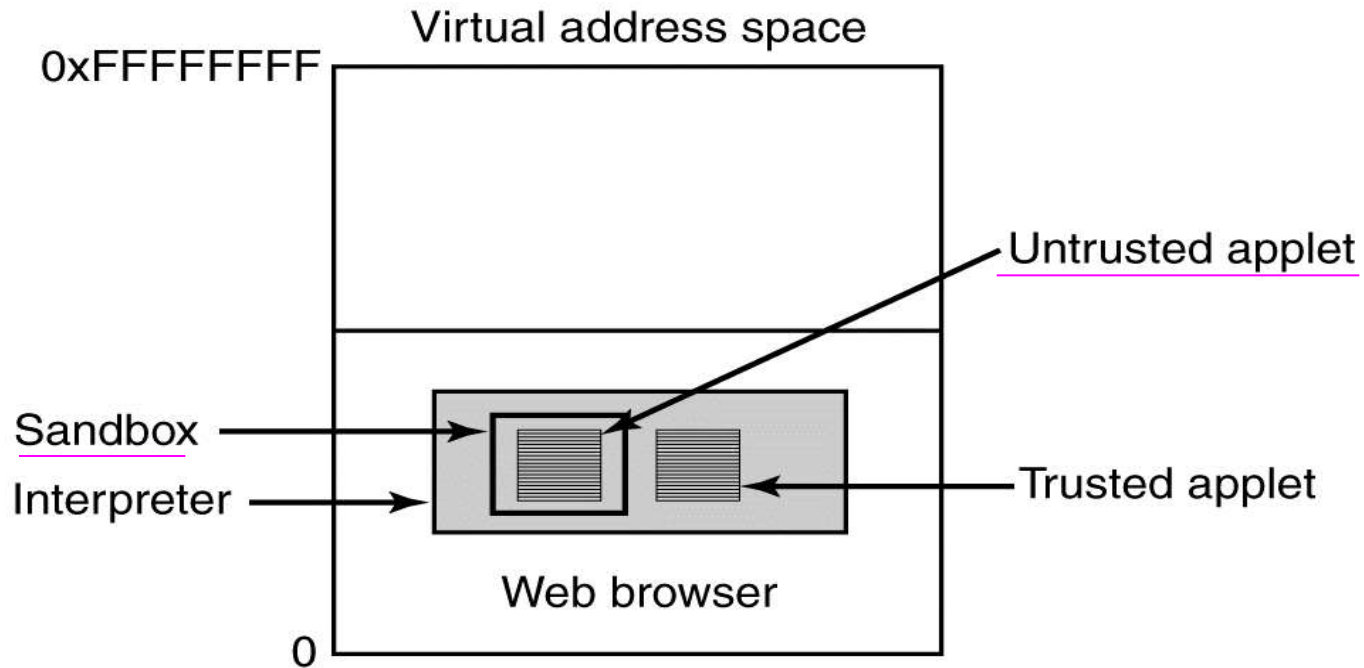
SSL(6)

- In 1996, Netscape turned SSL over to IETF for standardization . The result is TLS (Transport Layer Security)
- TLS version is also know as SSL Version 3.1
- SSL3.0 和TLS无法互操作

8.9.4 Mobile Code Security

- Mobile Code （移动代码—从服务器下载到客户端来运行的代码），形式：
 - Java applet
 - 信任的、不可信任的applet
 - 解释器，安全模型：沙箱
 - ActiveX controls
 - 二进制程序，直接执行->不安全->代码签名，安全模型：认证码(Authenticode,用来验证ActiveX控件的系统)
 - 使用有安全风险
 - Javascript
 - 没有正式的安全模型。Netscape使用 代码签名
 - Viruses
 - 不是被邀请进来的
 - 具有繁殖能力
- Download mobile code is a massive security risk

Java Applet Security



Applets inserted into a Java Virtual Machine interpreter inside the browser.

- 沙箱（**Sandbox**）检查不可信的**applet**，限制其使用系统资源

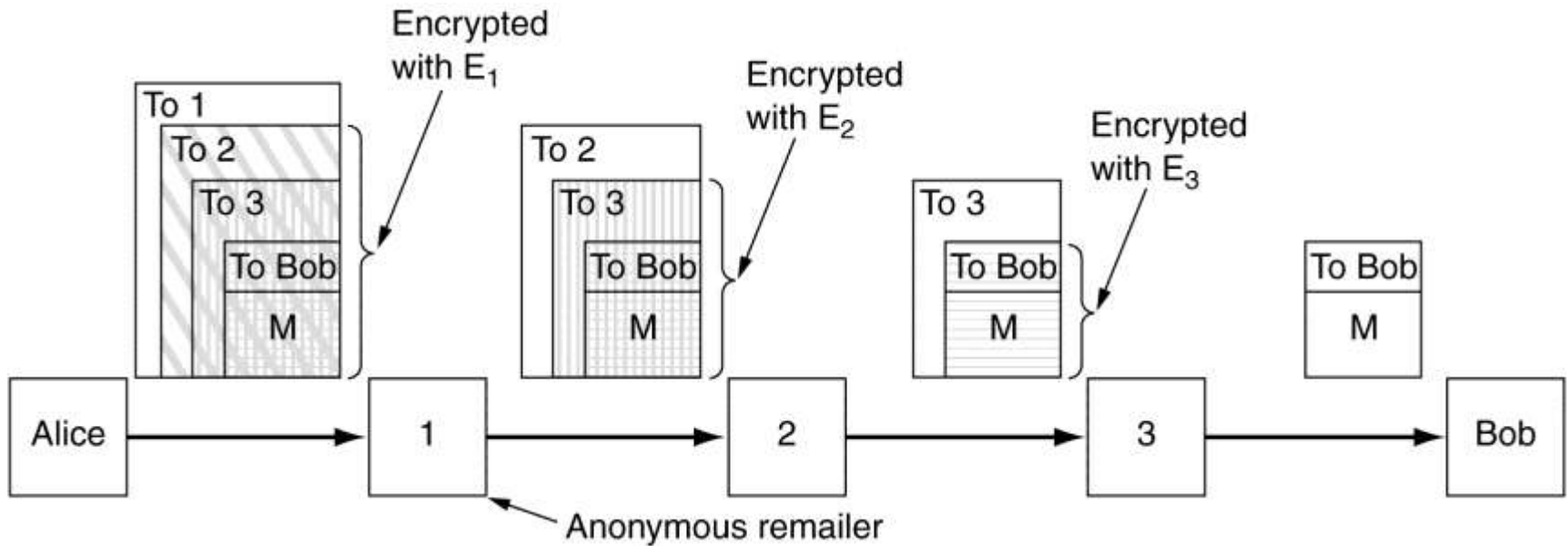
Java 和ActiveX的比较

- **Java:** 用户无法确定applet是谁写的，通过解释器限制
- **ActiveX:** 使用代码签名的方法，浏览器不监视其运行时的行为。风险很大！
 - 信任一个不熟悉的软件公司是一件可怕的事情！

8.10 Social Issues (*)

- Privacy
- Freedom of Speech
- Copyright

8.10.1 Privacy

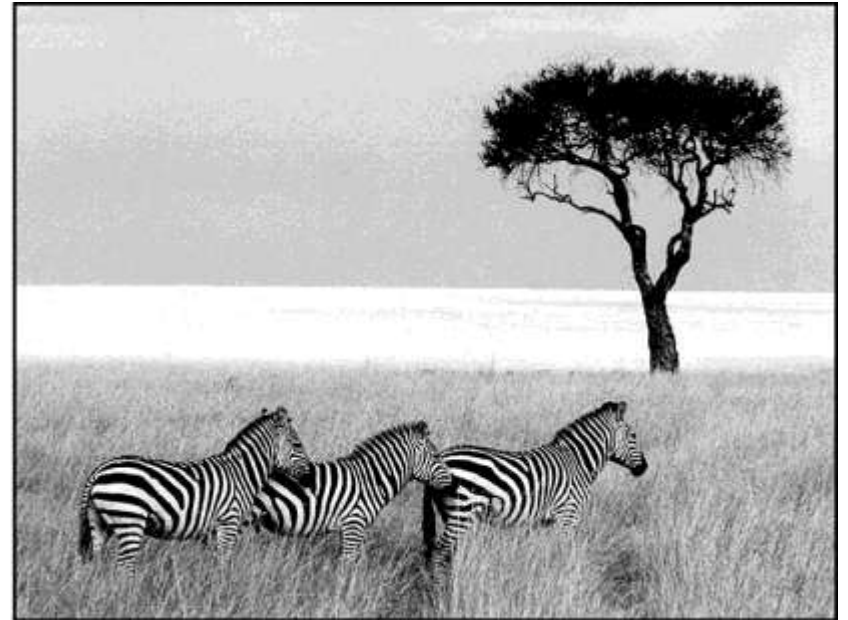
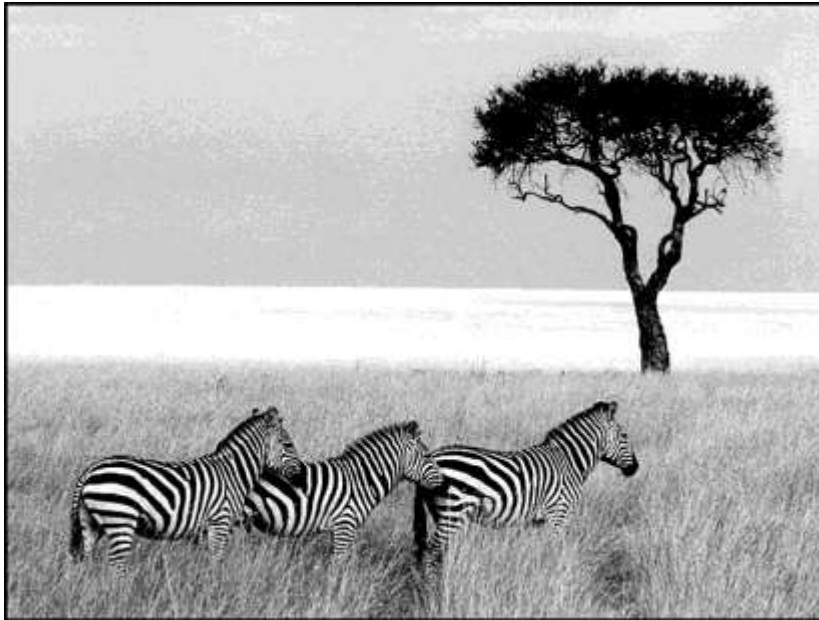


Users who wish anonymity chain requests through multiple
anonymous remailers.

8.10.2 Freedom of Speech

- Possibly banned material:
 1. Material inappropriate for children or teenagers.
 2. Hate aimed at various ethnic, religious, sexual, or other groups.
 3. Information about democracy and democratic values.
 4. Accounts of historical events contradicting the government's version.
 5. Manuals for picking locks, building weapons, encrypting messages, etc.

Steganography



(a) Three zebras and a tree. (b) Three zebras, a tree, and the complete text of five plays(contain700KB) by William Shakespeare.

8.10.3 Copyright

8.11 Summary

- **Cryptography**
 - Principal: a public-known algorithm and a secret key
 - substitution ciphers/transposition ciphers
- **Cryptographic algorithms**
 - Symmetric-key algorithms: Triple DES and Rijndael (AES)
 - Public-key algorithms: RSA
- **Digital signatures using symmetric-key and public-key algorithms**
 - Hash algorithms: MD5 or SHA-1
- **Public-key management using certificates, CA**
- **IPsec, Firewall, VPN, and WEP**
- **Authentication protocol**
 - A trusted third party, diffie-Hellman, Kerberos, and public-key cryptography
- **E-mail security: PGP**
- **Web security**
 - Secure naming, DNSsec, self-certifying names
 - Using SSL to establish secure, authenticated sessions between client and server
 - HTTPS

Recommended Exercises

In 4th Edition:

- 7、 11、 14、 25

In 5th Edition:

- 9、 10、 15、 20、 27、 28