南京大學
NANJING UNIVERSITY

计算机网络中的安全

殷亚凤
智能软件与工程学院
苏州校区南雍楼东区225
yafeng@nju.edu.cn，https://yafengnju.github.io/

Outline

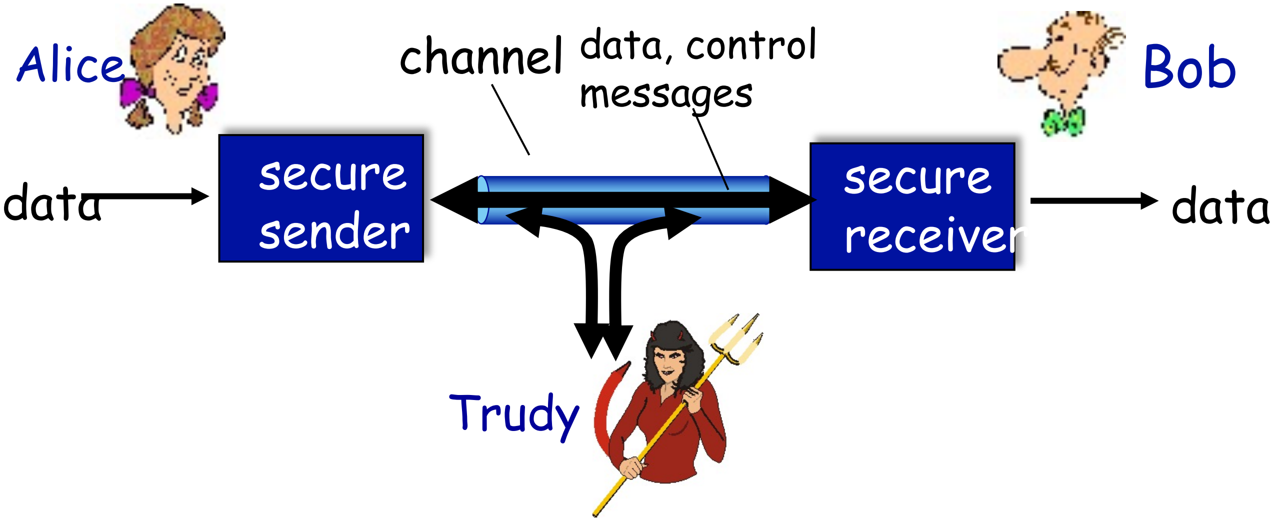
- What is network security?
- Principles of cryptography
- Authentication, message integrity,
- Securing e-mail
- Securing TCP connections: TLS
- Network layer security: IPsec
- Security in wireless and mobile networks
- Operational security: firewalls and IDS

What is network security?

- confidentiality: only sender, intended receiver should "understand" message contents
  - sender encrypts message
  - receiver decrypts message
- authentication: sender, receiver want to confirm identity of each other
- message integrity: sender, receiver want to ensure message not altered (in transit, or afterwards) without detection
- access and availability: services must be accessible and available to users

Friends and enemies: Alice, Bob, Trudy

- well-known in network security world
- Bob, Alice (lovers!) want to communicate "securely"
- Trudy (intruder) may intercept, delete, add messages



Friends and enemies: Alice, Bob, Trudy

Who might Bob and Alice be?

- … well, real-life Bobs and Alices!
- Web browser/server for electronic transactions (e.g., on-line purchases)
- on-line banking client/server
- DNS servers
- BGP routers exchanging routing table updates
- other examples?

There are bad guys (and girls) out there!

Q: What can a "bad guy" do?

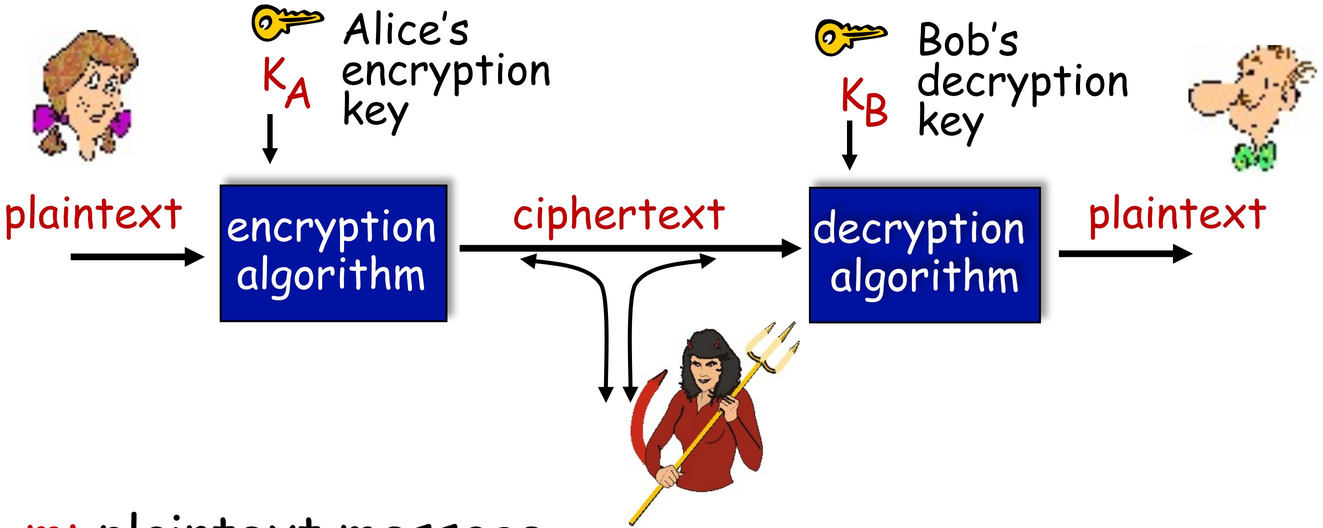
A: A lot! (recall section 1.6)

- eavesdrop: intercept messages
- actively insert messages into connection
- impersonation: can fake (spoof) source address in packet (or any field in packet)
- hijacking: "take over" ongoing connection by removing sender or receiver, inserting himself in place
- denial of service: prevent service from being used by others (e.g., by overloading resources)

Outline

- What is network security?
- Principles of cryptography
- Authentication, message integrity
- Securing e-mail
- Securing TCP connections: TLS
- Network layer security: IPsec
- Security in wireless and mobile networks
- Operational security: firewalls and IDS

The language of cryptography



m: plaintext message

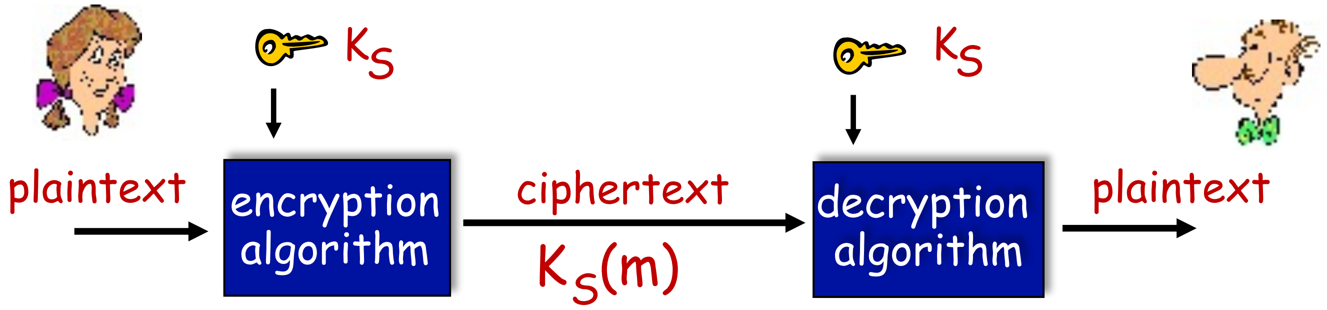
$K_A(m)$: ciphertext, encrypted with key K_A

$m = K_B(K_A(m))$

Breaking an encryption scheme

- cipher-text only attack: Trudy has ciphertext she can analyze
- two approaches:
  - brute force: search through all keys
  - statistical analysis
- known-plaintext attack: Trudy has plaintext corresponding to ciphertext
  - *e.g.,* in monoalphabetic cipher, Trudy determines pairings for a,l,i,c,e,b,o,
- chosen-plaintext attack: Trudy can get ciphertext for chosen plaintext

Symmetric key cryptography



symmetric key crypto: Bob and Alice share same (symmetric) key: K

- *e.g.*, key is knowing substitution pattern in mono alphabetic substitution cipher

Q: how do Bob and Alice agree on key value?

南京大學 NANJING UNIVERSITY

Simple encryption scheme

substitution cipher: substituting one thing for another

- monoalphabetic cipher: substitute one letter for another

```
plaintext:   abcdefghijklmnopqrstuvwxyz

ciphertext:  mnbvcxzasdfghjklpoiuytrewq
```

e.g.:
```
 Plaintext: bob. i love you. alice
ciphertext: nkn. s gktc wky. mgsbc
```

Encryption key: mapping from set of 26 letters to set of 26 letters

A more sophisticated encryption approach

- n substitution ciphers, $M_1, M_2, \ldots, M_n$
- cycling pattern:
  - e.g., n=4: M_1, M_3, M_4, M_3, M_2; M_1, M_3, M_4, M_3, M_2; ..
- for each new plaintext symbol, use subsequent substitution pattern in cyclic pattern
  - dog: d from M_1, o from M_3, g from M_4

Encryption key: n substitution ciphers, and cyclic pattern

- key need not be just n-bit pattern

Symmetric key crypto: DES

DES: Data Encryption Standard

- US encryption standard [NIST 1993]
- 56-bit symmetric key, 64-bit plaintext input
- block cipher with cipher block chaining
- how secure is DES?
  - DES Challenge: 56-bit-key-encrypted phrase decrypted (brute force) in less than a day
  - no known good analytic attack
- making DES more secure:
  - 3DES: encrypt 3 times with 3 different keys

AES: Advanced Encryption Standard

- symmetric-key NIST standard, replaced DES (Nov 2001)
- processes data in 128 bit blocks
- 128, 192, or 256 bit keys
- brute force decryption (try each key) taking 1 sec on DES, takes 149 trillion years for AES

Public Key Cryptography

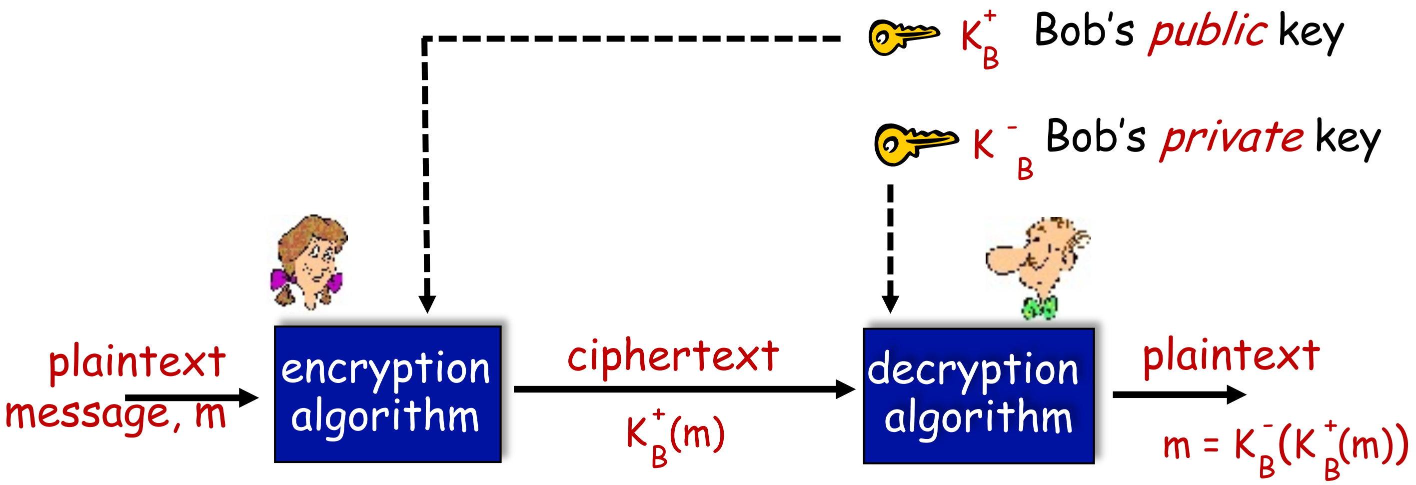
symmetric key crypto:

- requires sender, receiver know shared secret key
- Q: how to agree on key in first place (particularly if never "met")?

public key crypto

- radically different approach [Diffie-Hellman76, RSA78]
- sender, receiver do not share secret key
- public encryption key known to all
- private decryption key known only to receiver

Public Key Cryptography



Wow - public key cryptography revolutionized 2000-year-old (previously only symmetric key) cryptography!

- similar ideas emerged at roughly same time, independently in US and UK (classified)

Public key encryption algorithms

requirements:

1. need $K_B^+(\cdot)$ and $K_B^-(\cdot)$ such that

$$K_B^-(K_B^+(m)) = m$$

2. given public key K_B^+, it should be impossible to compute private key K_B^-

RSA: Rivest, Shamir, Adelson algorithm

Prerequisite: modular arithmetic

- $x \bmod n$ = remainder of x when divide by n
- facts:

  $[(a \bmod n) + (b \bmod n)] \bmod n = (a+b) \bmod n$

  $[(a \bmod n) - (b \bmod n)] \bmod n = (a-b) \bmod n$

  $[(a \bmod n) * (b \bmod n)] \bmod n = (a*b) \bmod n$
- thus

  $(a \bmod n)^d \bmod n = a^d \bmod n$
- example: $x=14$, $n=10$, $d=2$:

  $(x \bmod n)^d \bmod n = 4^2 \bmod 10 = 6$

  $x^d = 14^2 = 196 \quad x^d \bmod 10 = 6$

RSA: getting ready

- message: just a bit pattern
- bit pattern can be uniquely represented by an integer number
- thus, encrypting a message is equivalent to encrypting a number

example:

- m= 10010001. This message is uniquely represented by the decimal number 145.
- to encrypt m, we encrypt the corresponding number, which gives a new number (the ciphertext).

RSA: Creating public/private key pair

1. choose two large prime numbers $p, q.$ (e.g., 1024 bits each)

2. compute $n = pq$, $z = (p-1)(q-1)$

3. choose e (with $e<n$) that has no common factors with z (e, z are "relatively prime").

4. choose d such that $ed-1$ is exactly divisible by z. (in other words: $ed \bmod z = 1$).

5. *public* key is $\underbrace{(n,e)}_{K_B^+}$. *private* key is $\underbrace{(n,d)}_{K_B^-}$.

RSA: encryption, decryption

0. given (n,e) and (n,d) as computed above

1. to encrypt message m ($<n$), compute

   $c = m^e \bmod n$

2. to decrypt received bit pattern, c, compute

   $m = c^d \bmod n$

magic happens! $m = (\underbrace{m^e \bmod n}_{c})^d \bmod n$

RSA example:

Bob chooses *p=5, q=7*. Then *n=35, z=24*.

e=5 (so *e, z* relatively prime).

d=29 (so *ed-1* exactly divisible by z).

encrypting 8-bit messages.

encrypt: $\underbrace{\text{bit pattern}}_{00001000}$ $\underbrace{m}_{12}$ $\underbrace{m^e}_{24832}$ $\underbrace{c = m^e \bmod n}_{17}$

decrypt: $\underbrace{c}_{17}$ $\underbrace{c^d}_{481968572106750915091411825223071697}$ $\underbrace{m = c^d \bmod n}_{12}$

Why does RSA work?

- must show that $c^d \bmod n = m$, where $c = m^e \bmod n$
- fact: for any x and y: $x^y \bmod n = x^{(y \bmod z)} \bmod n$
  - where $n = pq$ and $z = (p-1)(q-1)$
- thus,

$$
\begin{aligned}
c^d \bmod n &= (m^e \bmod n)^d \bmod n \\
&= m^{ed} \bmod n \\
&= m^{(ed \bmod z)} \bmod n \\
&= m^1 \bmod n \\
&= m
\end{aligned}
$$

RSA: another important property

The following property will be *very* useful later:

$$\underbrace{K_B^-(K_B^+(m))}_{\substack{\text{use public key} \\ \text{first, followed} \\ \text{by private key}}} = m = \underbrace{K_B^+(K_B^-(m))}_{\substack{\text{use private key} \\ \text{first, followed} \\ \text{by public key}}}$$

result is the same!

RSA: another important property

Why $K_B^-(K_B^+(m)) = m = K_B^+(K_B^-(m))$?

follows directly from modular arithmetic:

$$(m^e \bmod n)^d \bmod n = m^{ed} \bmod n$$
$$= m^{de} \bmod n$$
$$= (m^d \bmod n)^e \bmod n$$

Why is RSA secure?

- suppose you know Bob's public key (n,e). How hard is it to determine d?
- essentially need to find factors of n without knowing the two factors p and q
  - fact: factoring a big number is hard

RSA in practice: session keys

- exponentiation in RSA is computationally intensive
- DES is at least 100 times faster than RSA
- use public key crypto to establish secure connection, then establish second key – symmetric session key – for encrypting data

session key, K_S

- Bob and Alice use RSA to exchange a symmetric session key K_S
- once both have K_S, they use symmetric key cryptography

Outline

- What is network security?
- Principles of cryptography
- Authentication, message integrity
- Securing e-mail
- Securing TCP connections: TLS
- Network layer security: IPsec
- Security in wireless and mobile networks
- Operational security: firewalls and IDS

Authentication

Goal: Bob wants Alice to “prove” her identity to him

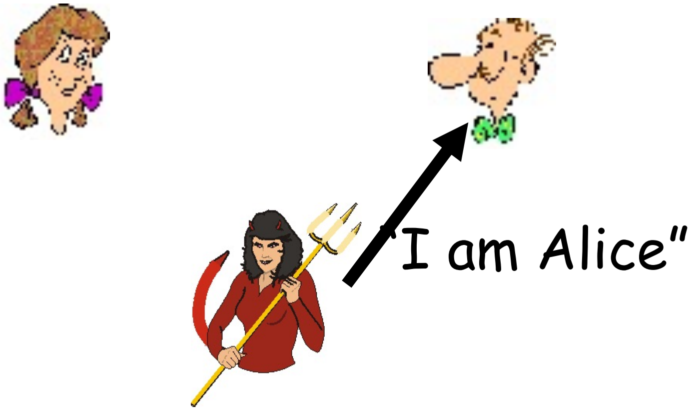
Protocol ap1.0: Alice says “I am Alice”

failure scenario??

Authentication

Goal: Bob wants Alice to "prove" her identity to him

Protocol ap1.0: Alice says "I am Alice"



in a network, Bob can not "see" Alice, so Trudy simply declares herself to be Alice

Authentication: another try

Goal: Bob wants Alice to "prove" her identity to him

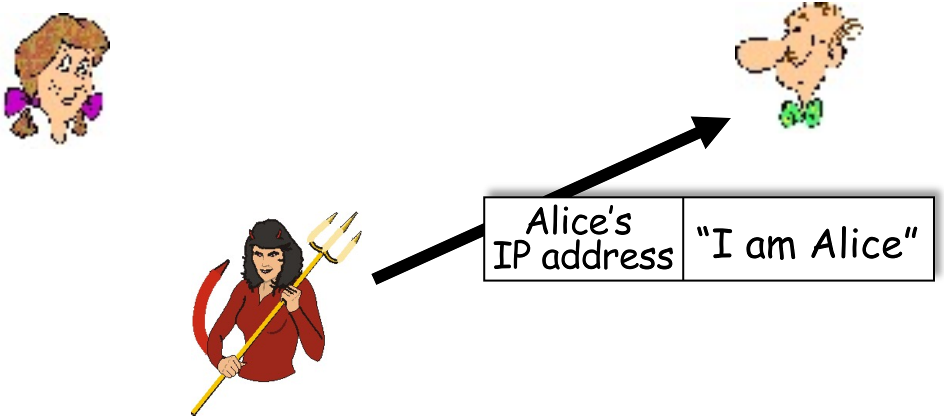
Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address

failure scenario??

Authentication: another try

Goal: Bob wants Alice to "prove" her identity to him

Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address

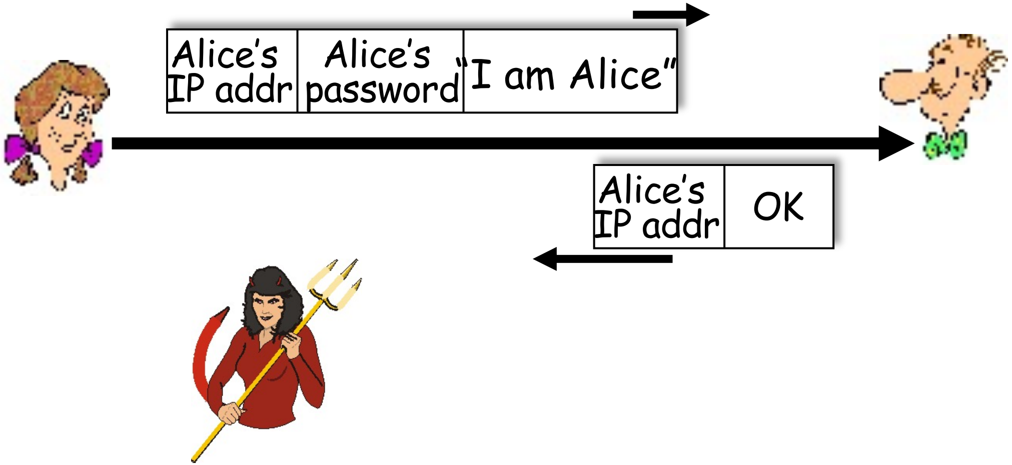


Trudy can create a packet "spoofing" Alice's address

Authentication: a third try

Goal: Bob wants Alice to "prove" her identity to him

Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.

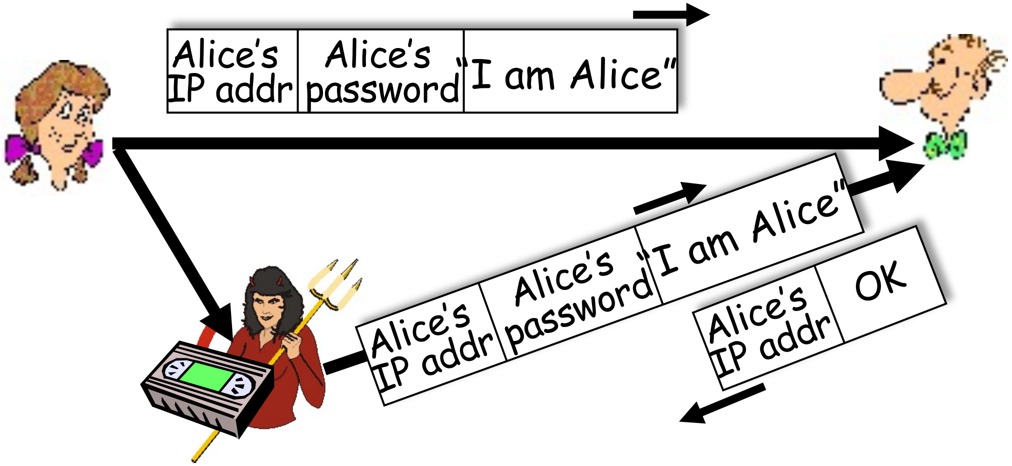


failure scenario??

Authentication: a third try

Goal: Bob wants Alice to "prove" her identity to him

Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.

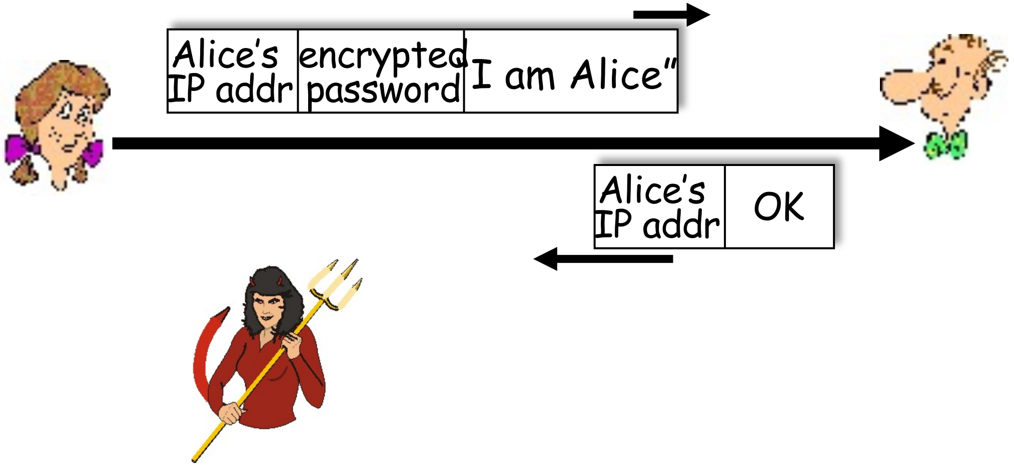


playback attack: Trudy records Alice's packet and later plays it back to Bob

Authentication: a modified third try

Goal: Bob wants Alice to "prove" her identity to him

Protocol ap3.0: Alice says "I am Alice" and sends her encrypted secret password to "prove" it.

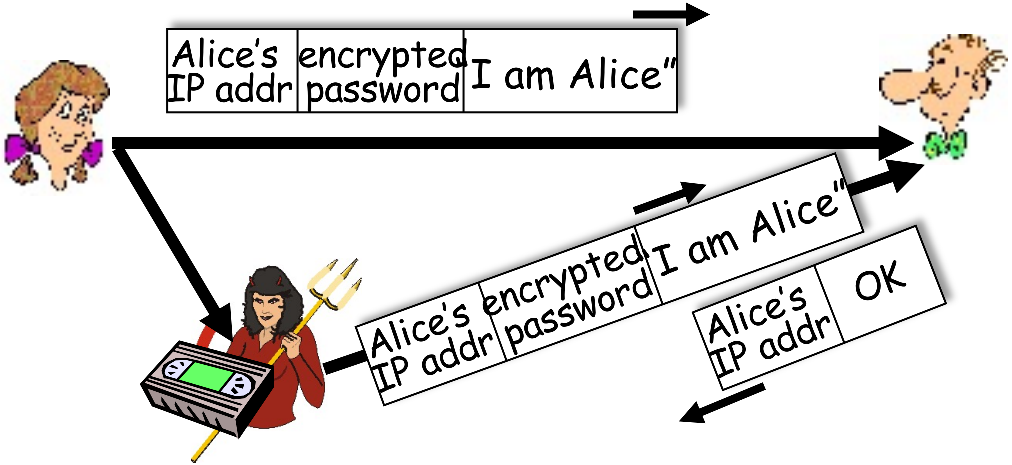


failure scenario??

Authentication: a modified third try

Goal: Bob wants Alice to "prove" her identity to him

Protocol ap3.0: Alice says "I am Alice" and sends her encrypted secret password to "prove" it.



playback attack still works: Trudy records Alice's packet and later plays it back to Bob

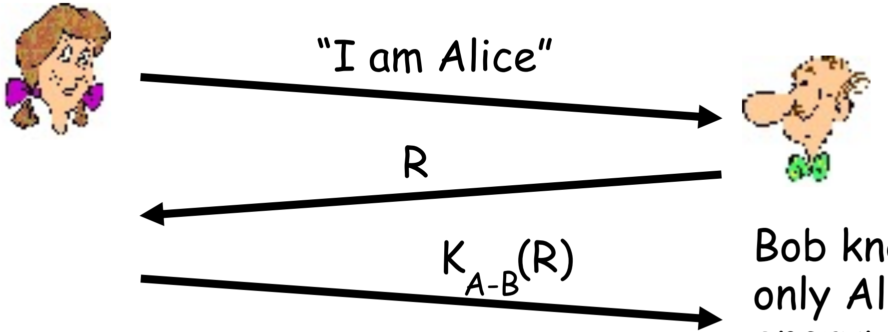
Authentication: a fourth try

Goal: avoid playback attack

nonce: number (R) used only once-in-a-lifetime

protocol ap4.0: to prove Alice "live", Bob sends Alice nonce, R

- Alice must return R, encrypted with shared secret key



Bob know Alice is live, and only Alice knows key to encrypt nonce, so it must be Alice!

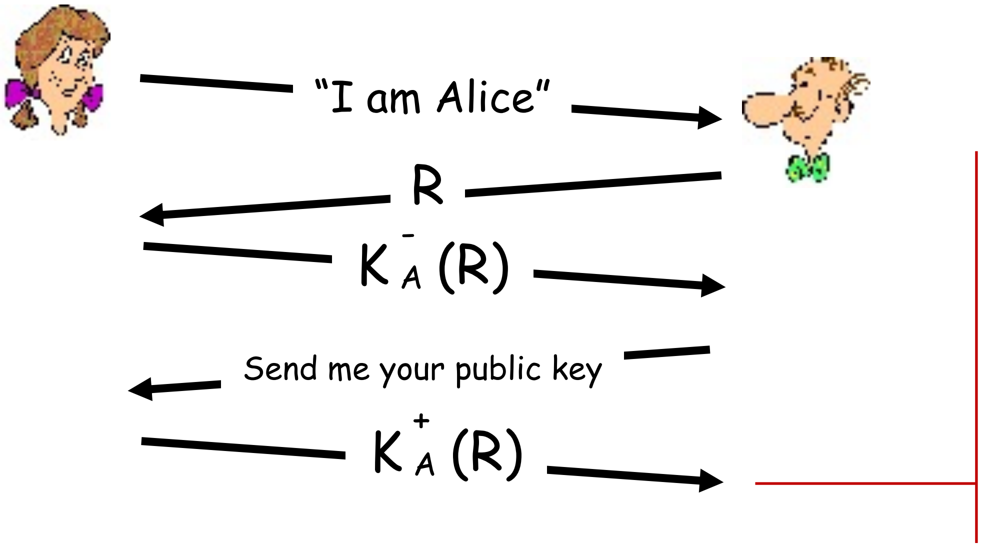
Failures, drawbacks?

NANJING UNIVERSITY

Authentication: ap5.0

ap4.0 requires shared symmetric key - can we authenticate using public key techniques?

ap5.0: use nonce, public key cryptography



Bob computes

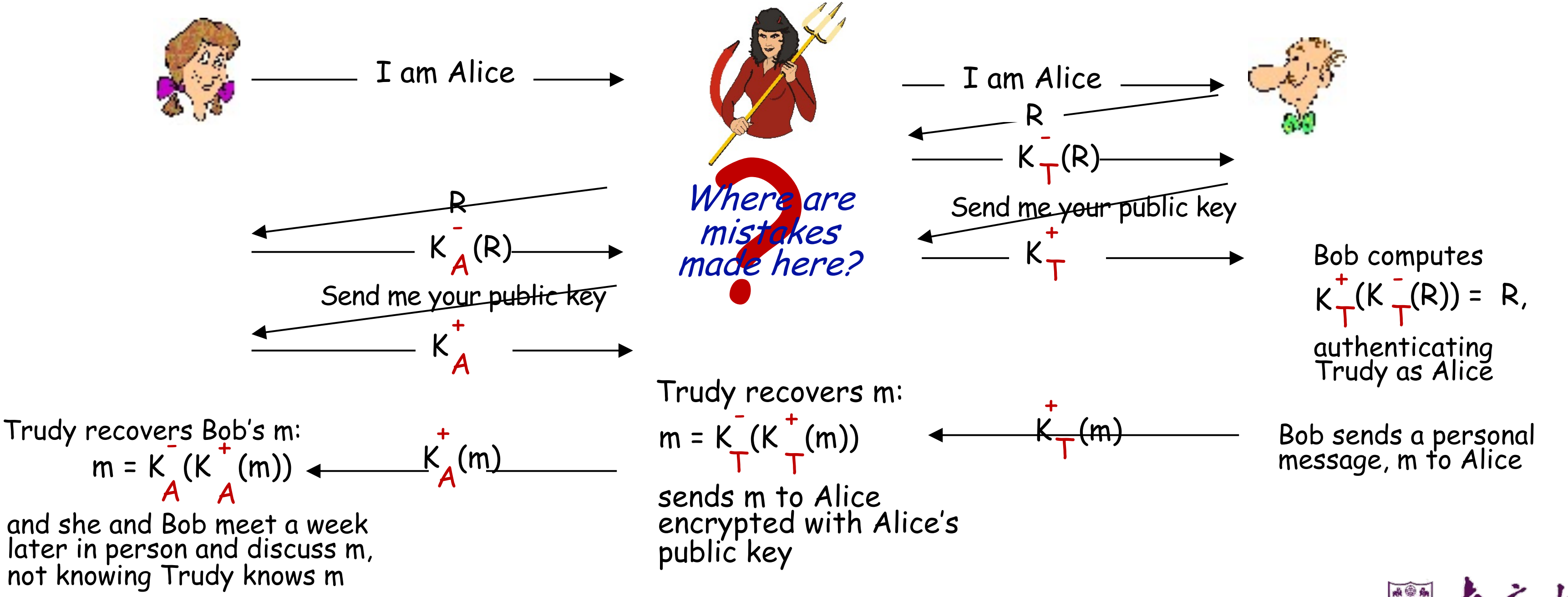
$K_A^+(K_A^-(R)) = R$

and knows only Alice could have the private key, that encrypted R such that

$K_A^+(K_A^-(R)) = R$

Authentication: ap5.0 – there's still a flaw!

man (or woman) in the middle attack: Trudy poses as Alice (to Bob) and as Bob (to Alice)



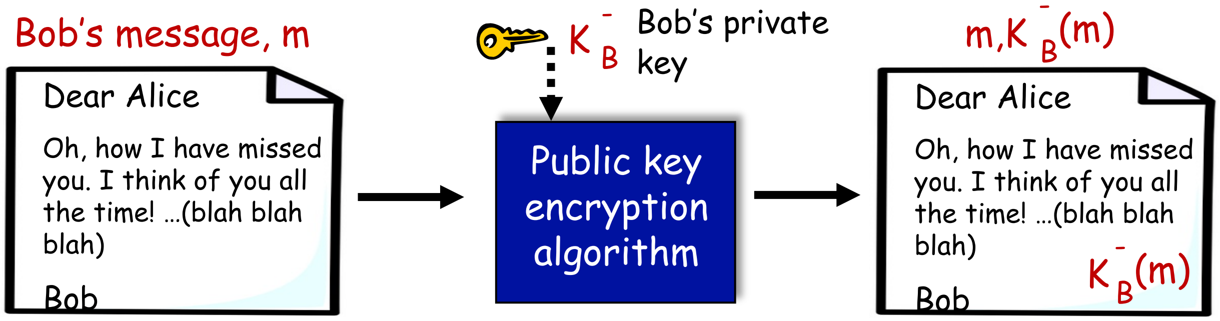
Outline

- What is network security?
- Principles of cryptography
- Authentication, message integrity
- Securing e-mail
- Securing TCP connections: TLS
- Network layer security: IPsec
- Security in wireless and mobile networks
- Operational security: firewalls and IDS

Digital signatures

cryptographic technique analogous to hand-written signatures:

- sender (Bob) digitally signs document: he is document owner/creator.
- *verifiable, nonforgeable:* recipient (Alice) can prove to someone that Bob, and no one else (including Alice), must have signed document
- simple digital signature for message m:
  - Bob signs m by encrypting with his private key K_B, creating "signed" message, $K_B^-(m)$



Digital signatures

- suppose Alice receives msg m, with signature: m, $K_B^-(m)$
- Alice verifies m signed by Bob by applying Bob's public key K_B^+ to $K_B^-(m)$ then checks $K_B^+(K_B^-(m)) = m$.
- If $K_B^+(K_B^-(m)) = m$, whoever signed m must have used Bob's private key

Alice thus verifies that:

- Bob signed m
- no one else signed m
- Bob signed m and not m'

non-repudiation:

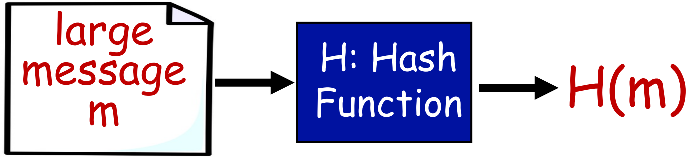
✓ Alice can take m, and signature $K_B^-(m)$ to court and prove that Bob signed m

Message digests

computationally expensive to public-key-encrypt long messages

goal: fixed-length, easy- to-compute digital "fingerprint"

- apply hash function H to *m*, get fixed size message digest, *H(m)*



Hash function properties:

- many-to-1
- produces fixed-size msg digest (fingerprint)
- given message digest *x*, computationally infeasible to find *m* such that *x = H(m)*

Internet checksum: poor crypto hash function

Internet checksum has some properties of hash function:

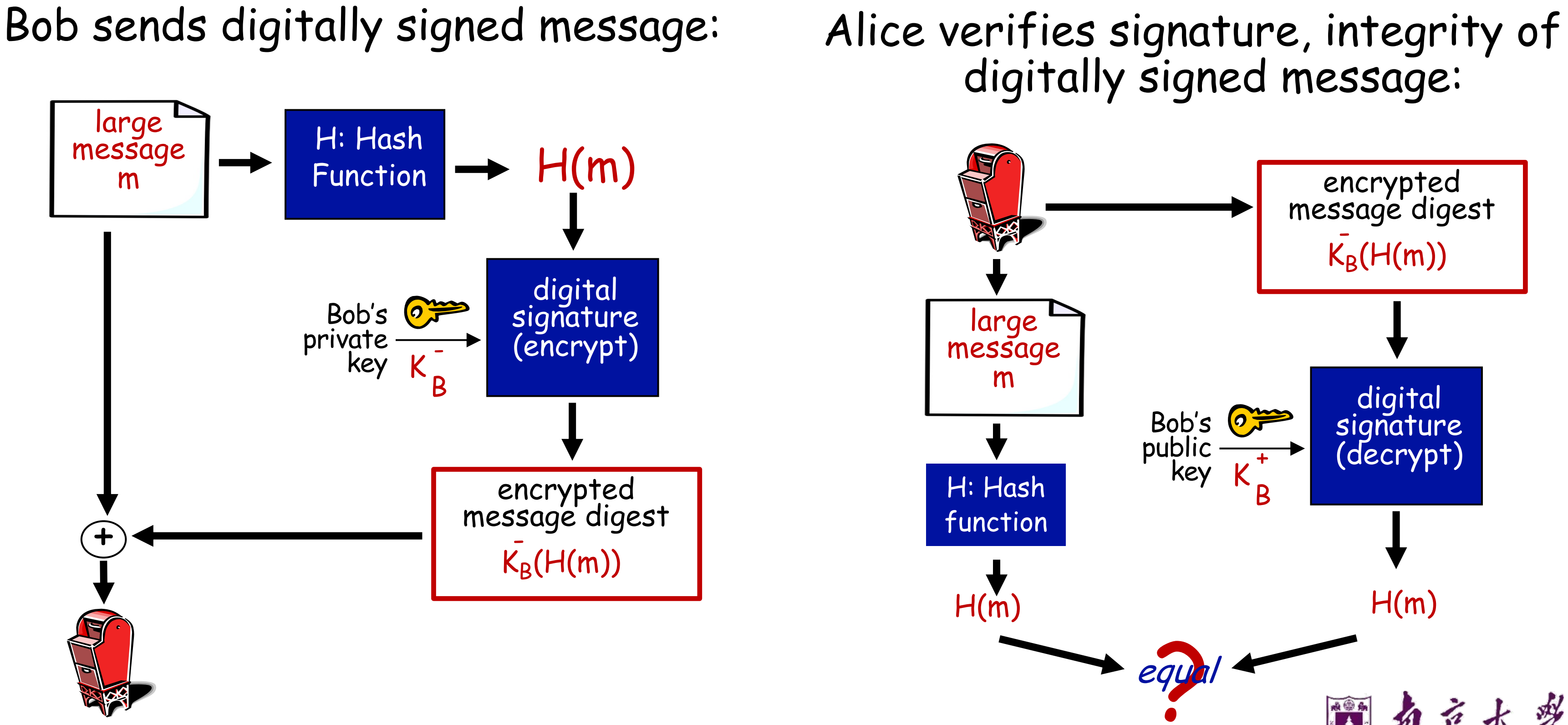
- produces fixed length digest (16-bit sum) of message
- is many-to-one

but given message with given hash value, it is easy to find another message with same hash value:

message	ASCII format	message	ASCII format
I O U 1	49 4F 55 31	I O U 9	49 4F 55 39
0 0 . 9	30 30 2E 39	0 0 . 1	30 30 2E 31
9 B O B	39 42 D2 42	9 B O B	39 42 D2 42
	B2 C1 D2 AC		B2 C1 D2 AC

different messages but identical checksums!

Digital signature = signed message digest

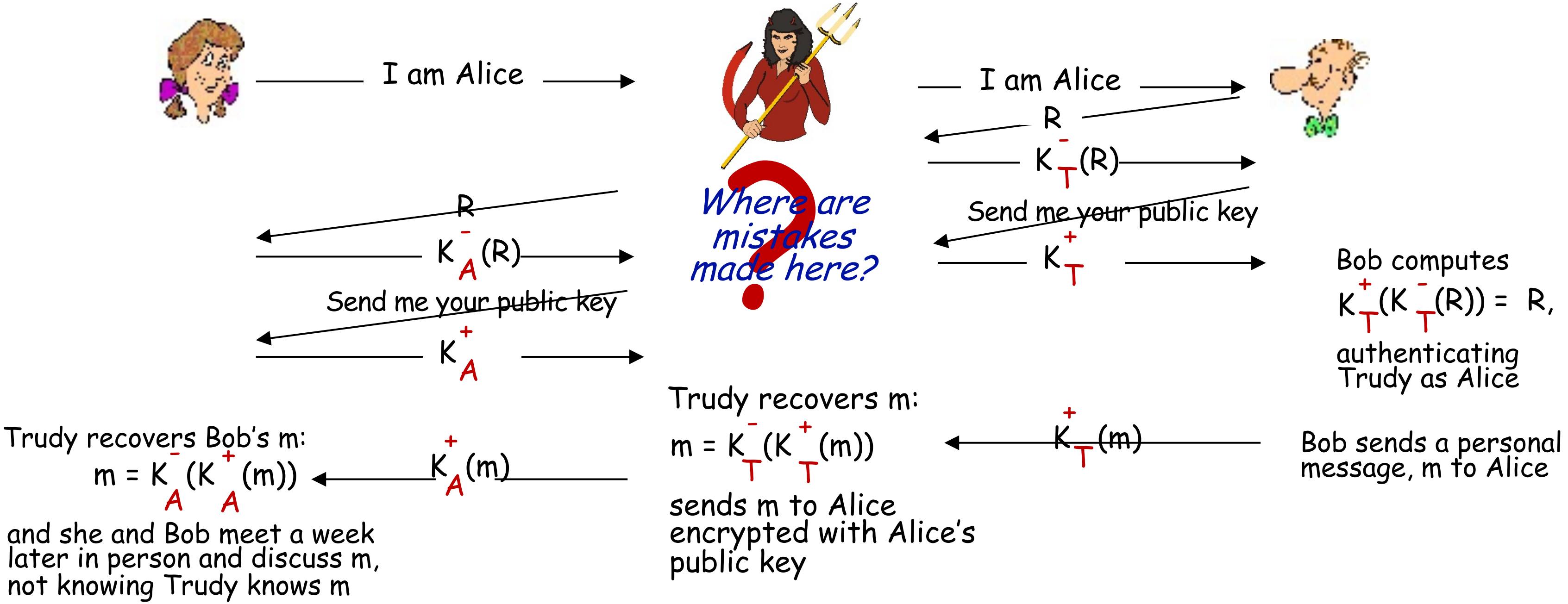


Hash function algorithms

- MD5 hash function widely used (RFC 1321)
  - computes 128-bit message digest in 4-step process.
  - arbitrary 128-bit string x, appears difficult to construct msg m whose MD5 hash is equal to x
- SHA-1 is also used
  - US standard [NIST, FIPS PUB 180-1]
  - 160-bit message digest

Authentication: ap5.0 – let's fix it!!

Recall the problem: Trudy poses as Alice (to Bob) and as Bob (to Alice)

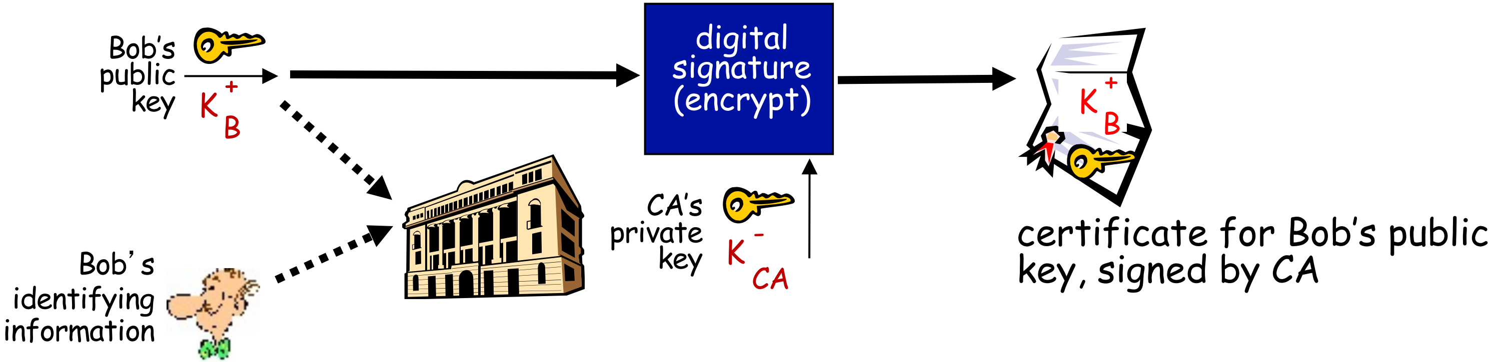


Need for certified public keys

- motivation: Trudy plays pizza prank on Bob
  - Trudy creates e-mail order:
    Dear Pizza Store, Please deliver to me four pepperoni pizzas. Thank you, Bob
  - Trudy signs order with her private key
  - Trudy sends order to Pizza Store
  - Trudy sends to Pizza Store her public key, but says it's Bob's public key
  - Pizza Store verifies signature; then delivers four pepperoni pizzas to Bob
  - Bob doesn't even like pepperoni

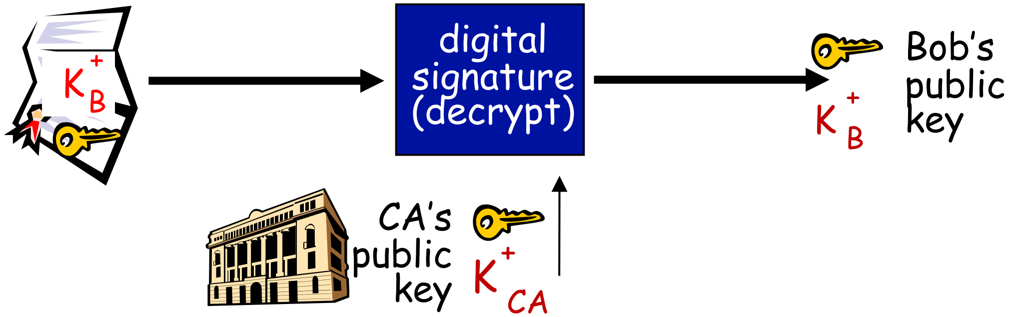
Public key Certification Authorities (CA)

- certification authority (CA): binds public key to particular entity, E
- entity (person, website, router) registers its public key with CE provides "proof of identity" to CA
  - CA creates certificate binding identity E to E's public key
  - certificate containing E's public key digitally signed by CA: CA says "this is E's public key"



Public key Certification Authorities (CA)

- when Alice wants Bob's public key:
  - gets Bob's certificate (Bob or elsewhere)
  - apply CA's public key to Bob's certificate, get Bob's public key

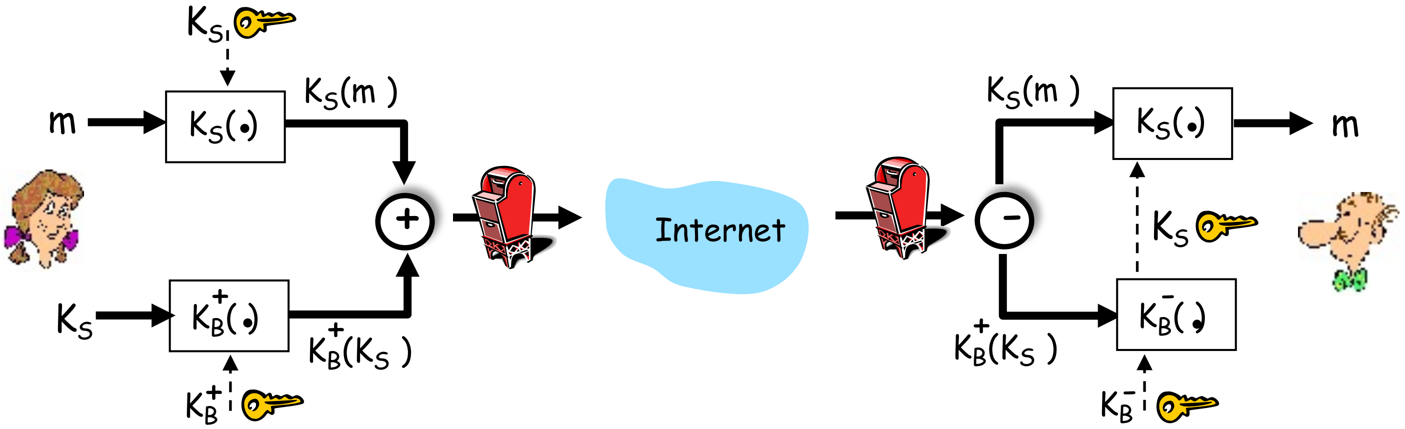


Outline

- What is network security?
- Principles of cryptography
- Authentication, message integrity
- Securing e-mail
- Securing TCP connections: TLS
- Network layer security: IPsec
- Security in wireless and mobile networks
- Operational security: firewalls and IDS

Secure e-mail: confidentiality

Alice wants to send confidential e-mail, m, to Bob.



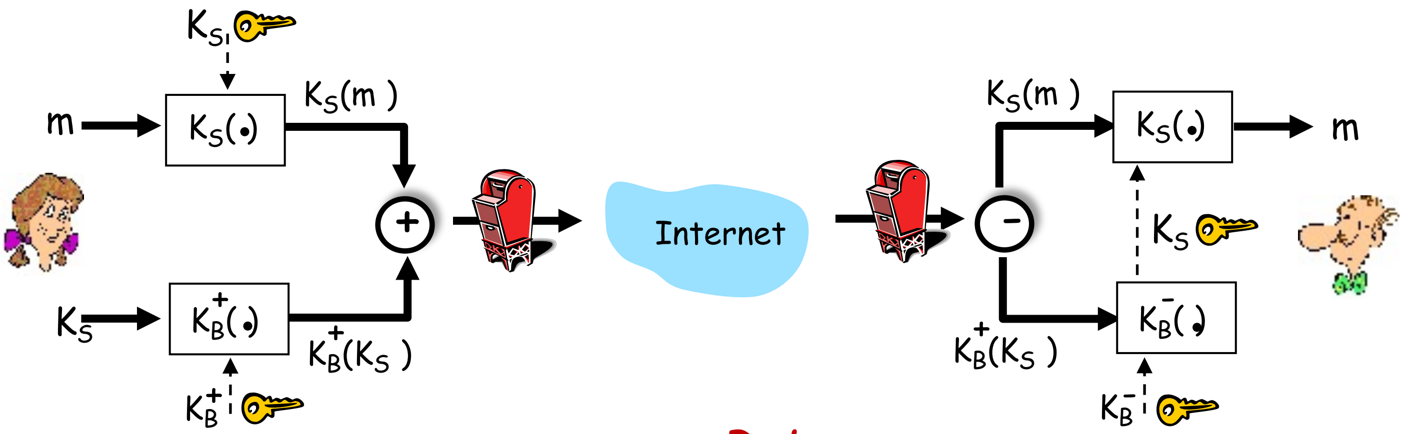
Alice:

- generates random *symmetric* private key, K_S
- encrypts message with K_S (for efficiency)
- also encrypts K_S with Bob's public key
- sends both $K_S(m)$ and $K^+_B(K_S)$ to Bob

南京大學 NANJING UNIVERSITY

Secure e-mail: confidentiality (more)

Alice wants to send confidential e-mail, m, to Bob.



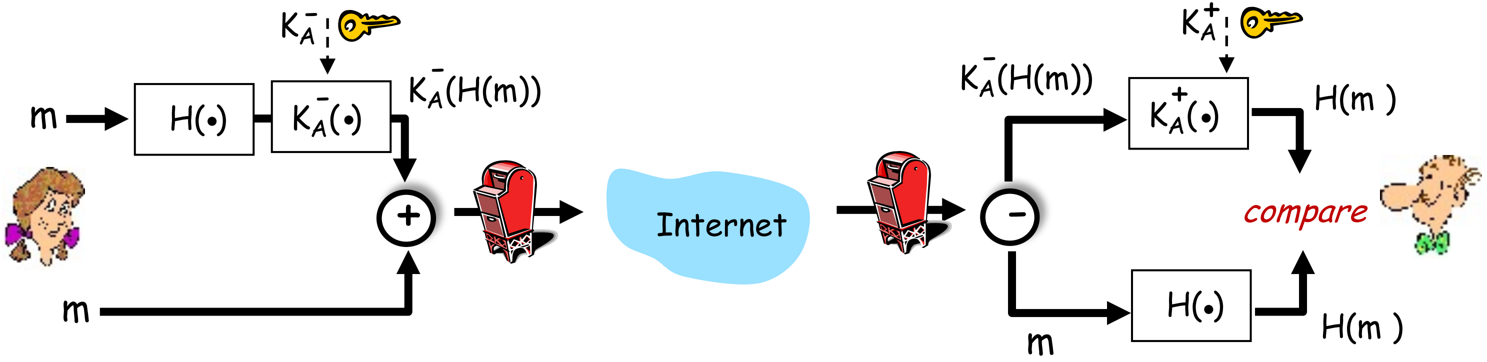
Bob:

- uses his private key to decrypt and recover K_S
- uses K_S to decrypt $K_S(m)$ to recover m

南京大學
NANJING UNIVERSITY

Secure e-mail: integrity, authentication

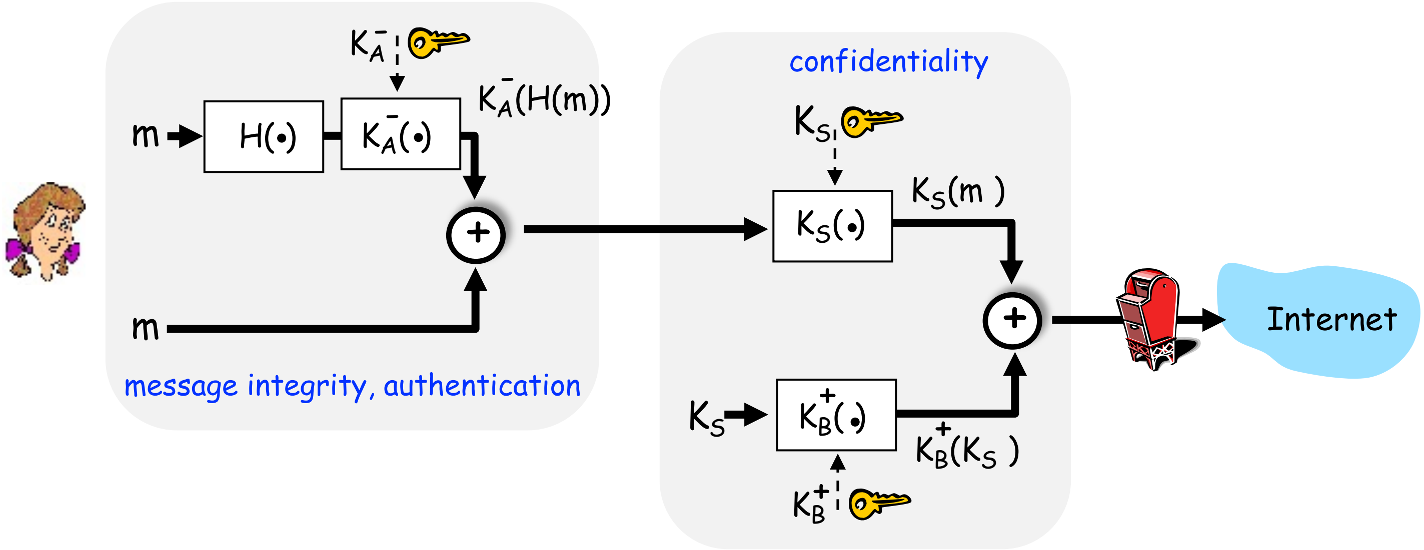
Alice wants to send m to Bob, with message integrity, authentication



- Alice digitally signs hash of her message with her private key, providing integrity and authentication
- sends both message (in the clear) and digital signature

Secure e-mail: integrity, authentication

Alice sends m to Bob, with confidentiality, message integrity, authentication

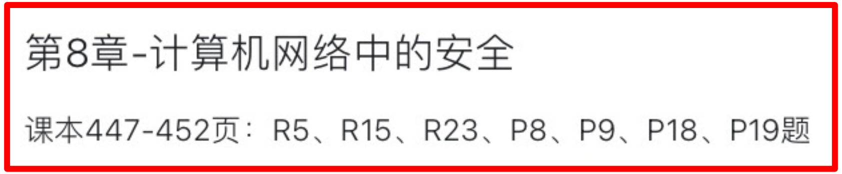


Alice uses three keys: her private key, Bob's public key, new symmetric key

What are Bob's complementary actions?

课程习题（作业）——截止日期：6月3日晚23:59

- **课本447-452页**：R5、R15、R23、P8、P9、P18、P19题
- 提交方式：https://selearning.nju.edu.cn/（教学支持系统）



- 命名：学号+姓名+第*章。
- 若提交遇到问题请及时发邮件或在下一次上课时反馈。

课程习题（作业）——截止日期：6月3日晚23:59

R5. 考虑一个 8 块密码。这个密码有多少种可能的输入块？有多少种可能的映射？如果我们将每种映射视为一个密钥，则该密码具有多少种可能的密钥？

R15. 假设 Alice 有一个准备发送给任何请求者的报文。数以千计的人要获得 Alice 的报文，但每个人都要确保该报文的完整性。在这种场景下，你认为是基于 MAC 还是基于数字签名的完整性方案更为适合？为什么？

R23. 假设 Bob 向 Trudy 发起一条 TCP 连接，而 Trudy 正在伪装她是 Alice。在握手期间，Trudy 向 Bob 发送 Alice 的证书。在 SSL 握手算法的哪一步，Bob 将发现他没有与 Alice 通信？

P8. 考虑具有 $p=5$ 和 $q=11$ 的 RSA。

a. n 和 z 是什么？

b. 令 e 为 3。为什么这是一个对 e 的可接受的选择？

c. 求 d 使得 $de=1 \pmod z$ 和 $d<160$。

d. 使用密钥 (n, e) 加密报文 $m=8$。令 c 表示对应的密文。显示所有工作。提示：为了简化计算，使用如下事实。

$$[(a \bmod n) \cdot (b \bmod n)] \bmod n = (a \cdot b) \bmod n$$

课程习题（作业）——截止日期：6月3日晚23:59

P9. 在这个习题中，我们探讨 Diffie-Hellman(DH) 公钥加密算法，该算法允许两个实体协商一个共享的密钥。该 DH 算法利用一个大素数 p 和另一个小于 p 的大数 g。p 和 g 都是公开的（因此攻击者将知道它们）。在 DH 中，Alice 和 Bob 每人分别独立地选择秘密密钥 S_A 和 S_B。Alice 则通过将 g 提高到 S_A 并以 p 为模来计算她的公钥 T_A。类似地，Bob 则通过将 g 提高到 S_B 并以 p 为模来计算他的公钥 T_B。此后 Alice 和 Bob 经过因特网交换他们的公钥。Alice 则通过将 T_B 提高到 S_A 并以 p 为模来计算出共享密钥 S。类似地，Bob 则通过将 T_A 提高到 S_B 并以 p 为模来计算出共享密钥 S'。

a. 证明在一般情况下，Alice 和 Bob 得到相同的对称密钥，即证明 $S = S'$。

b. 对于 $p = 11$ 和 $g = 2$，假定 Alice 和 Bob 分别选择私钥 $S_A = 5$ 和 $S_B = 12$，计算 Alice 和 Bob 的公钥 T_A 和 T_B。显示所有计算过程。

c. 接着（b），现在计算共享对称密钥 S。显示所有计算过程。

d. 提供一个时序图，显示 Diffie-Hellman 是如何能够受到中间人攻击的。该时序图应当具有 3 条垂直线，分别对应 Alice、Bob 和攻击者 Trudy。

课程习题（作业）——截止日期：6月3日晚23:59

P18. 假定 Alice 要向 Bob 发送电子邮件。Bob 具有一个公共－私有密钥对（K_B^+，K_B^-），并且 Alice 具有 Bob 的证书。但 Alice 不具有公钥私钥对。Alice 和 Bob（以及全世界）共享相同的散列函数 $H(\cdot)$。

a. 在这种情况下，能设计一种方案使得 Bob 能够验证 Alice 创建的报文吗？如果能，用方框图显示 Alice 和 Bob 是如何做的。

b. 能设计一个对从 Alice 向 Bob 发送的报文提供机密性的方案吗？如果能，用方块图显示 Alice 和 Bob 是如何做的。

P19. 考虑下面对于某 SSL 会话的一部分的 Wireshark 输出。

a. Wireshark 分组 112 是由客户还是由服务器发送的？

b. 服务器的 IP 地址和端口号是什么？

c. 假定没有丢包和重传，由客户发送的下一个 TCP 报文段的序号将是什么？

d. Wireshark 分组 112 包含了多少个 SSL 记录？

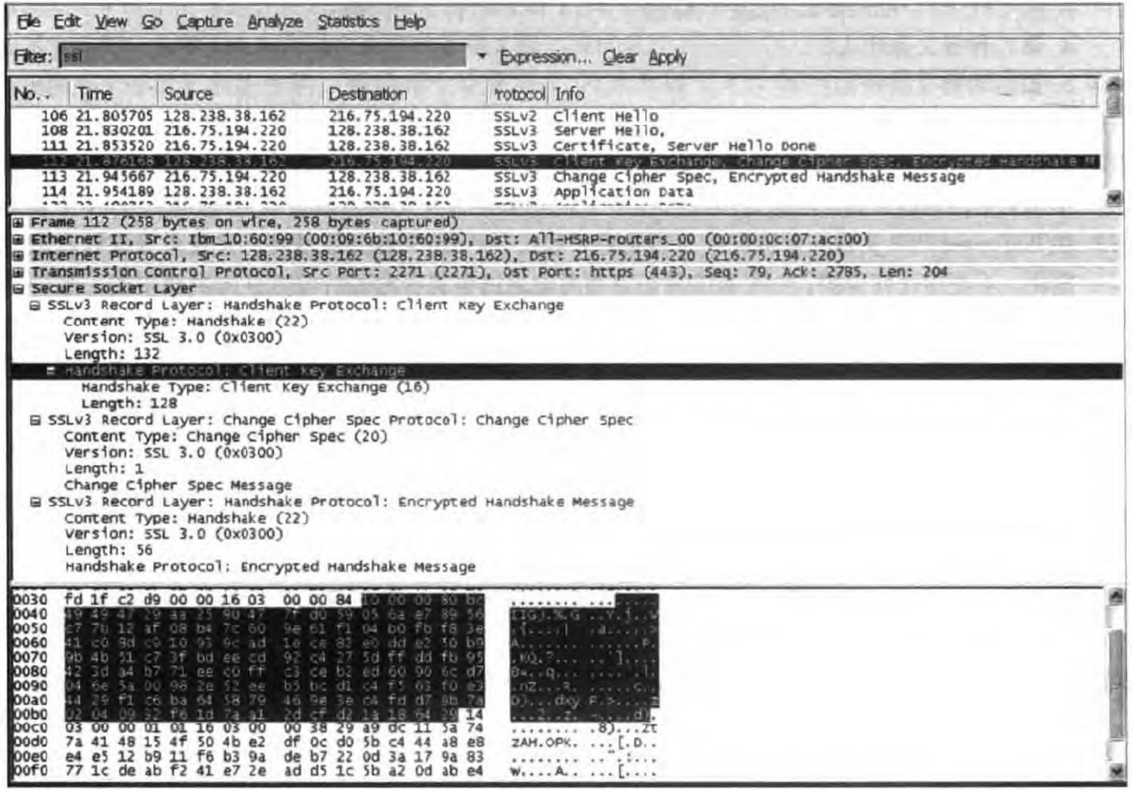
e. 分组 112 包含了一个主密钥或者一个加密的主密钥吗？或者两者都不是？

f. 假定握手类型字段是 1 字节并且每个长度字段是 3 字节，主密钥（或加密的主密钥）的第一个和最后一个字节的值是什么？

g. 客户加密的握手报文考虑了多少 SSL 记录？

h. 服务器加密的握手报文考虑了多少 SSL 记录？

课程习题（作业）——截止日期：6月3日晚23:59



（Wireshark 屏幕截图的重印获得 Wireshark 基金会的许可）

南京大學
NANJING UNIVERSITY

Q & A

殷亚凤
智能软件与工程学院
苏州校区南雍楼东区225
yafeng@nju.edu.cn，https://yafengnju.github.io/