W13
(2b)

Computer Networks

Network Security

Amitangshu Pal
Computer Science and Engineering
IIT Kanpur

What is network security?

confidentiality: only sender, intended receiver should "understand" message contents

- sender encrypts message
- receiver decrypts message

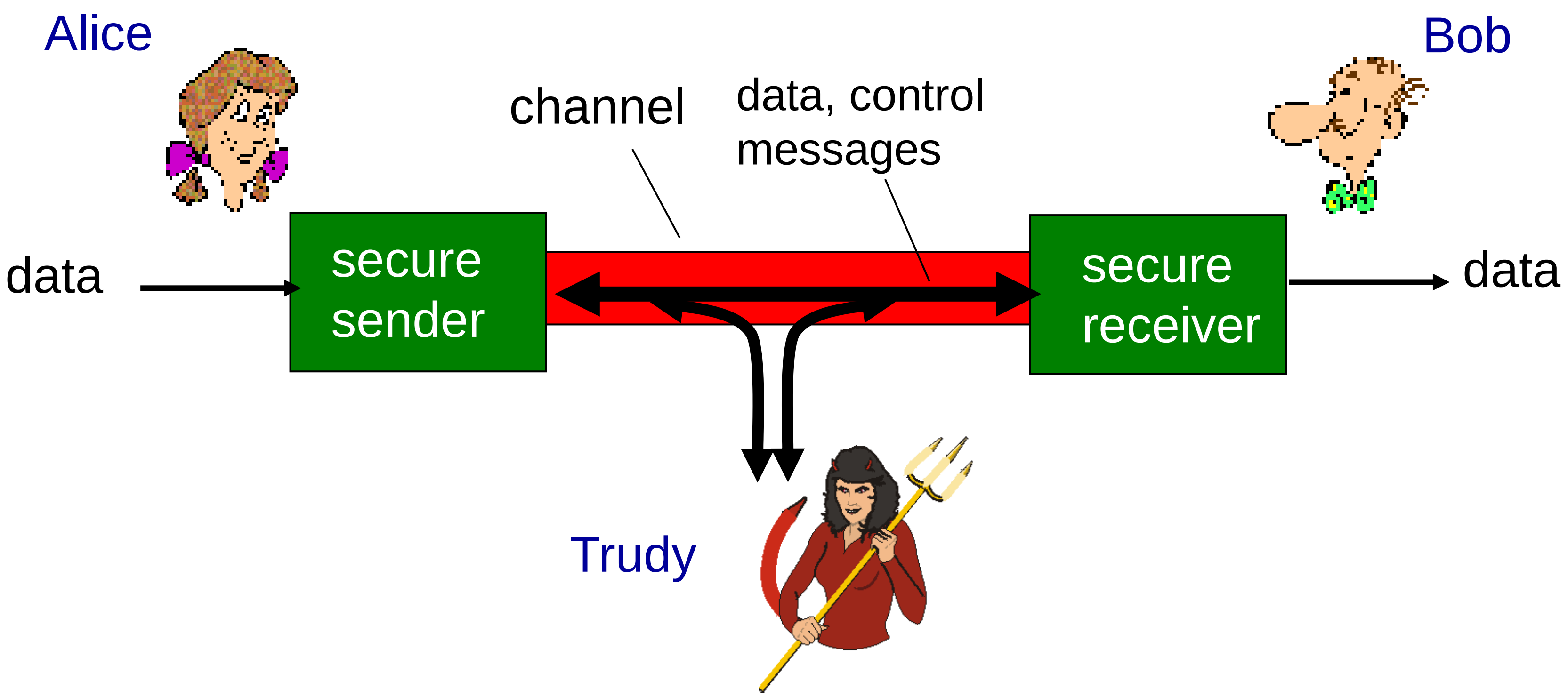
authentication: sender, receiver want to confirm identity of each other

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

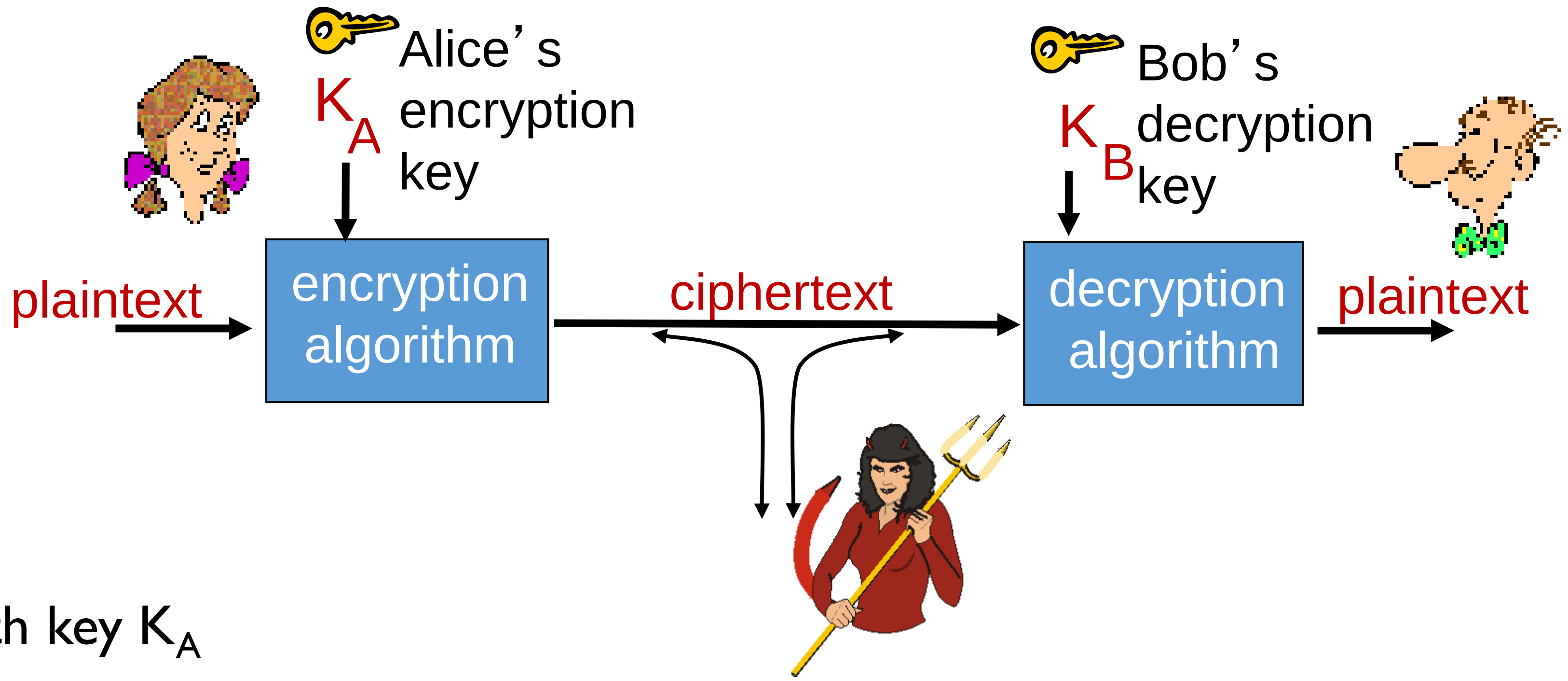
Friends and enemies: Alice, Bob, Trudy

- Bob, Alice want to communicate "securely"
- Trudy (intruder) may intercept, delete, add messages



Principles of Cryptography

The language of cryptography



m plaintext message

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

$m = K_B(K_A(m))$

Simple encryption scheme

substitution cipher: substituting one thing for another

- monoalphabetic cipher: substitute one letter for another

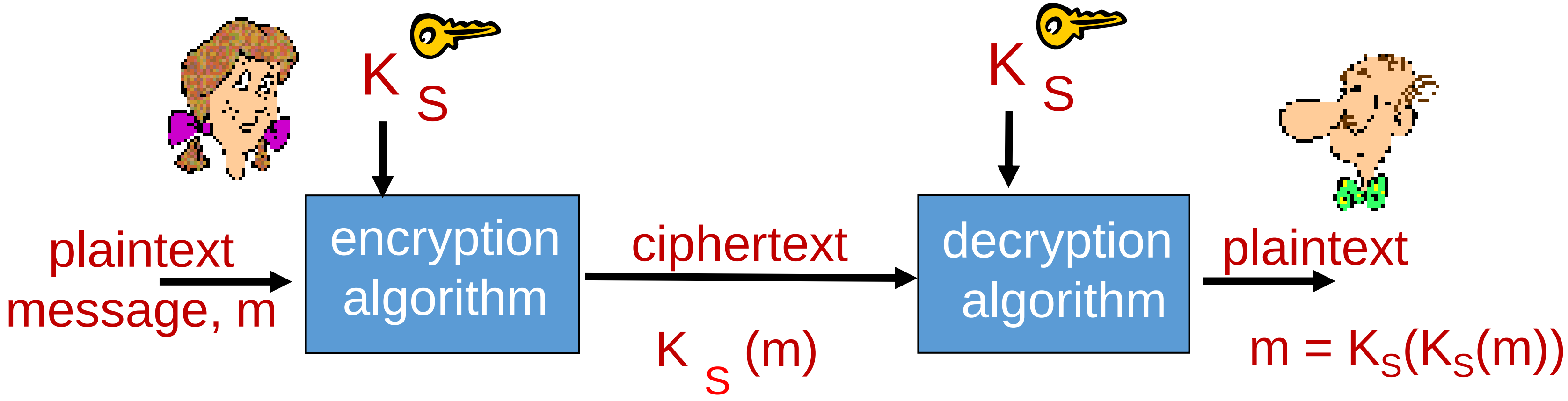
```
plaintext:   abcdefghijklmnopqrstuvwxyz

ciphertext:  mnbvcxzasdfghjklpoiuytrewq
```

e.g.:

```
Plaintext:    i             Alice
ciphertext:   s gktc wky    mgsbc
```

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

Symmetric key cryptography



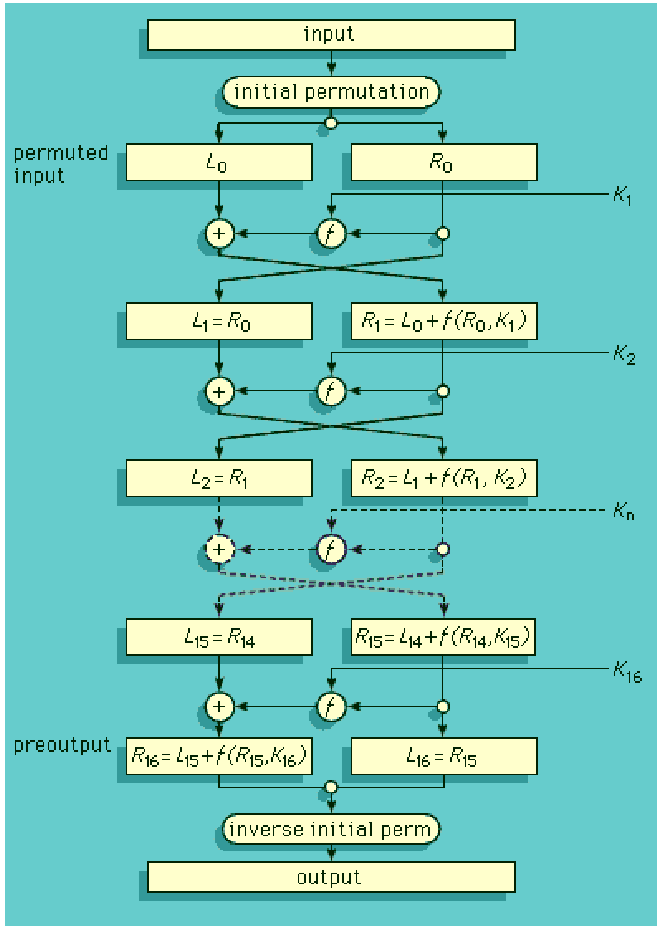
symmetric key crypto: Bob and Alice share same (symmetric) key: K_S

Examples: DES (Data Encryption Standard), AES (Advanced Encryption Standard)

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
- making DES more secure:
  - 3DES: encrypt 3 times with 3 different keys



Src: https://www.britannica.com/topic/Data-Encryption-Standard

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

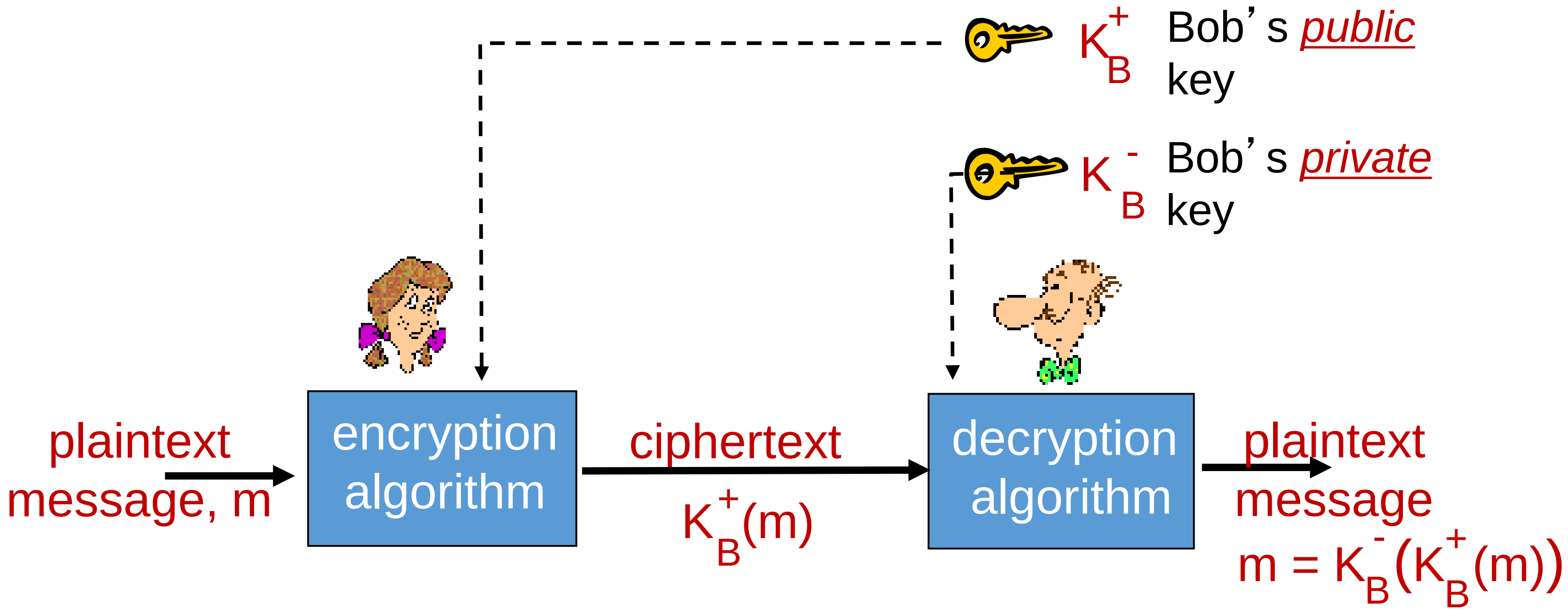
symmetric key crypto

- 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



Public key encryption algorithms

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

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

$$\textit{Why?} \quad m = \underbrace{(m^e \bmod n)}_{c}{}^{d} \bmod n$$

- Useful number theory result: If p, q are prime and n = pq, then $x^y \bmod n = x^{y \bmod (p-1)(q-1)} \bmod n$
- $(m^e \bmod n)^d \bmod n$

  $= m^{ed} \bmod n$

  $= m^{ed \bmod (p-1)(q-1)} \bmod n$ [using the theorem]

  $= m^1 \bmod n$ [as ed-1 is divisible by (p-1)(q-1)]

  $= m$

RSA: another important property

The following property will be *very* useful later:

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

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

follows directly from modular arithmetic:

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

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.

	bit pattern	m	m^e	$c = m^e \bmod n$
encrypt:	0000l000	12	24832	17

	c	c^d	$m = c^d \bmod n$
decrypt:	17	481968572106750915091411825223071697	12

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

Authentication

Authentication

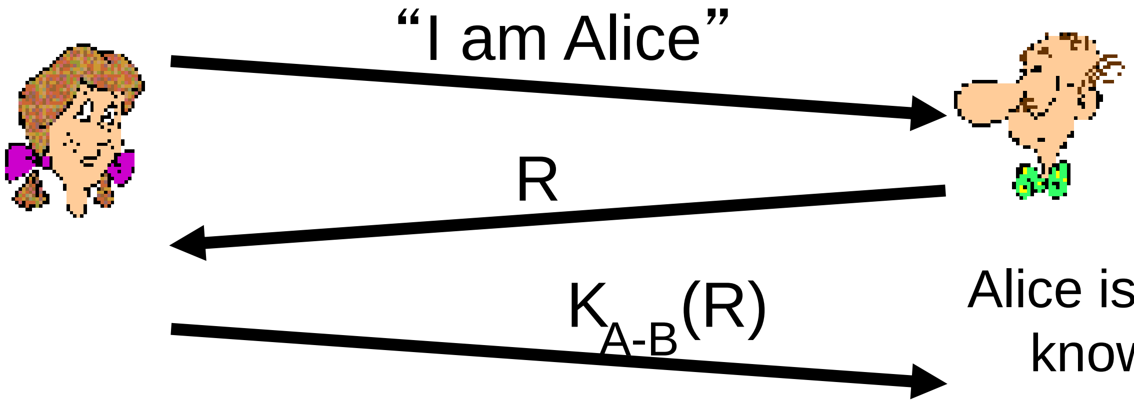
Goal: Bob wants Alice to "prove" her identity to him

<u>*Approach:*</u> Alice says "I am Alice"

Authentication

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

Approach: to prove Alice "live", Bob sends Alice *nonce*, R. Alice must return R, encrypted with shared secret key

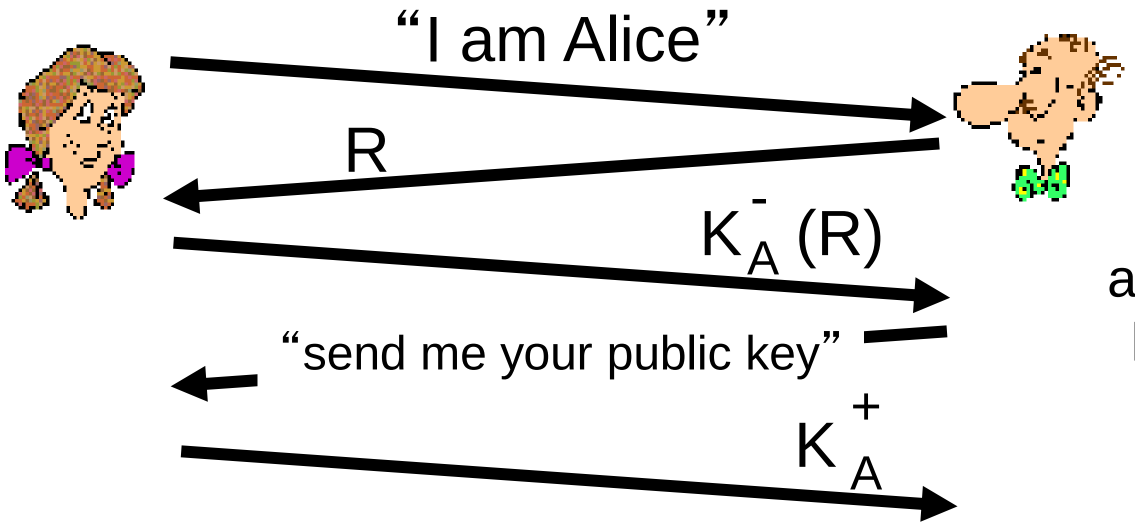


Alice is live, and only Alice knows key to encrypt nonce, so it must be Alice!

Authentication

- Can we authenticate using public key techniques?

Approach: use nonce, public key cryptography



Bob computes

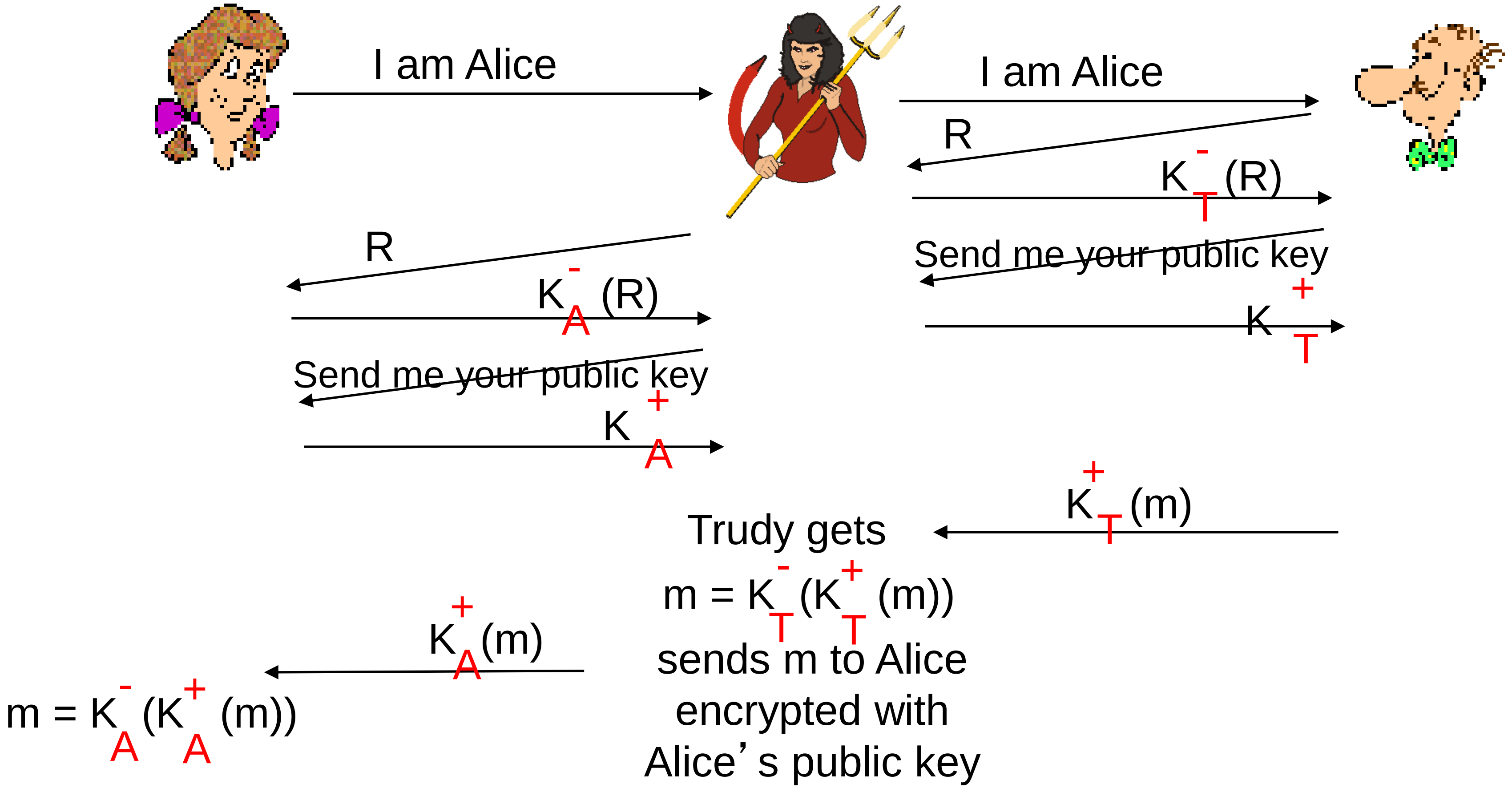
$$K_A^+(K_A^-(R)) = R$$

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

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

Authentication

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



Message Integrity

Digital signatures

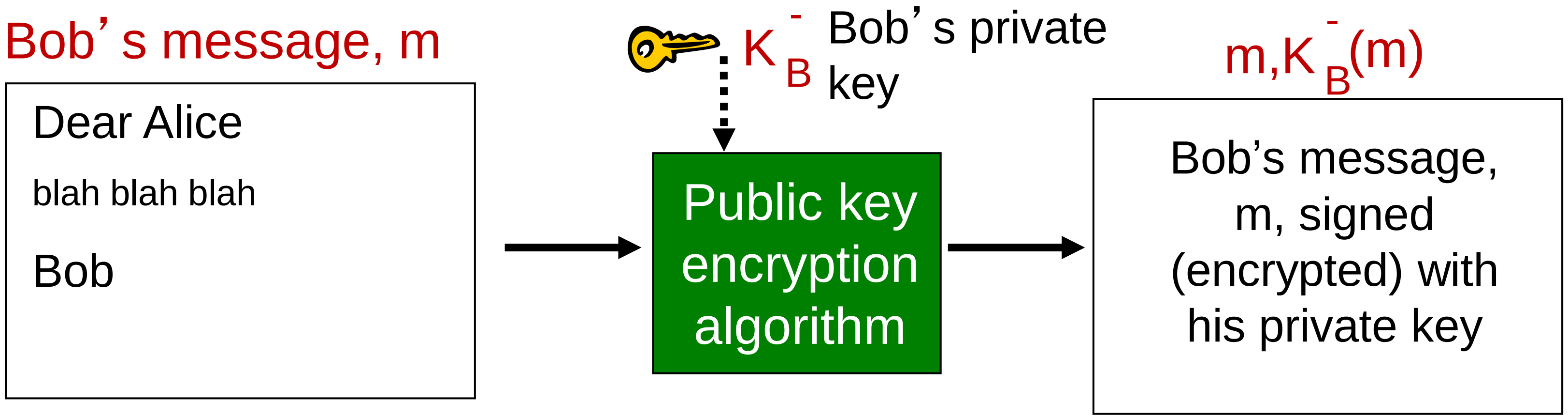
cryptographic technique analogous to hand-written signatures:

- sender (Bob) digitally signs document, establishing 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

Digital signatures

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

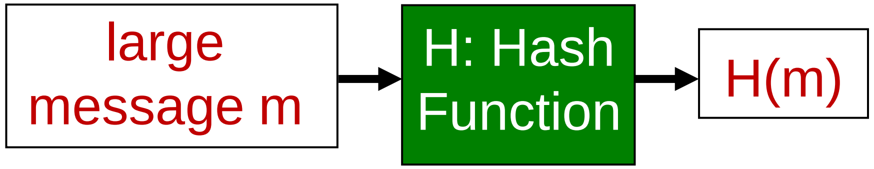
-

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)*.

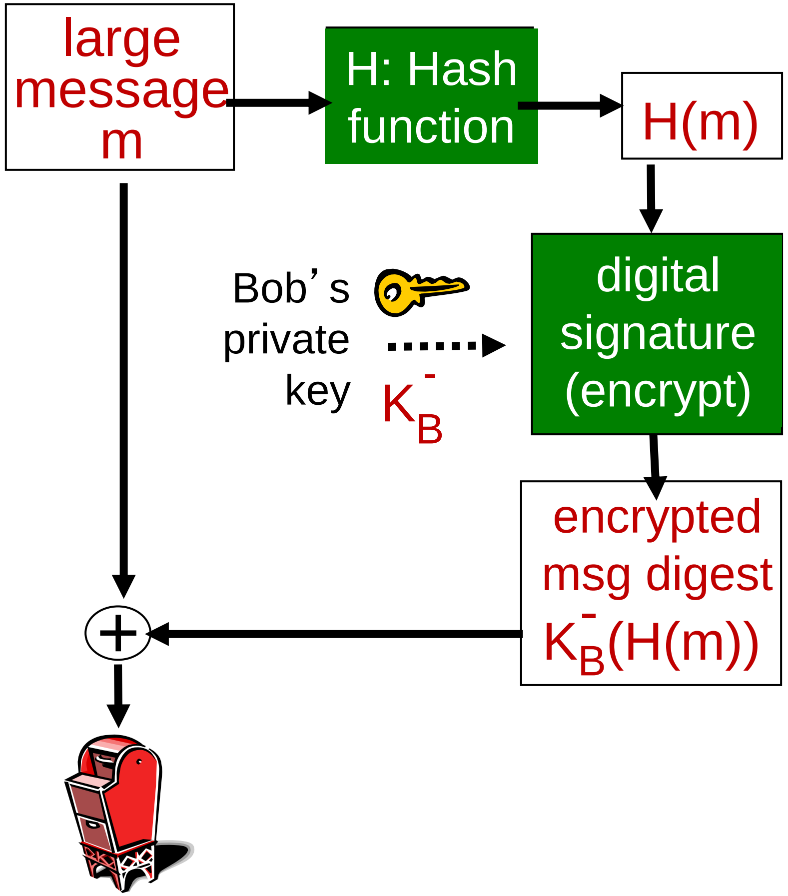


Hash function properties:

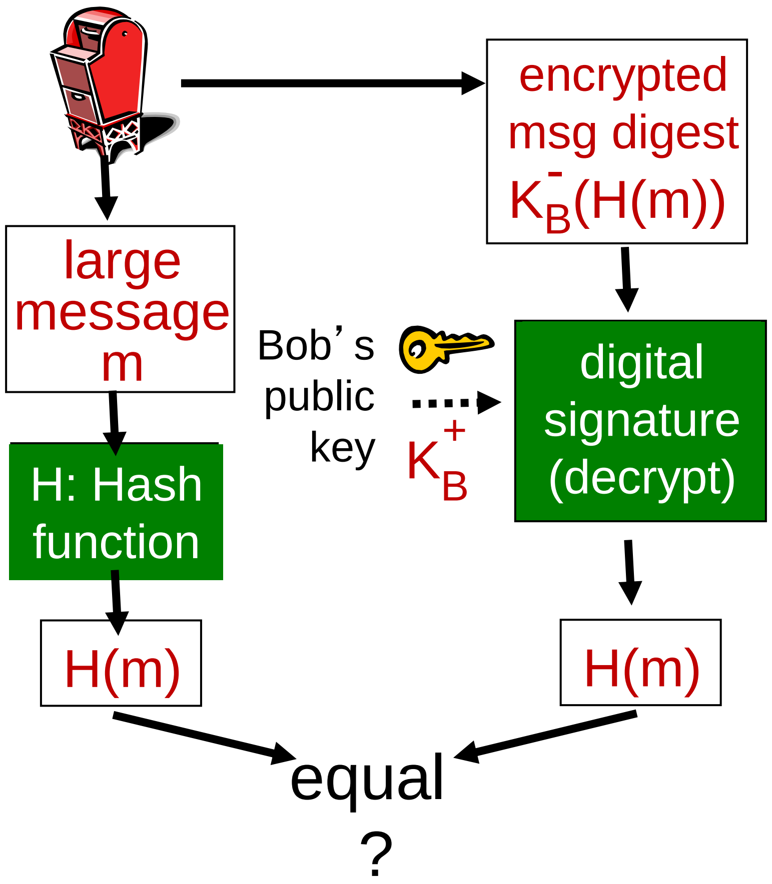
- many-to-1
- produces fixed-size msg digest (fingerprint)
- given message digest x, computationally infeasible to find m such that x = H(m)

Digital signature = signed message digest

Bob sends digitally signed message:



Alice verifies signature, integrity of digitally signed message:



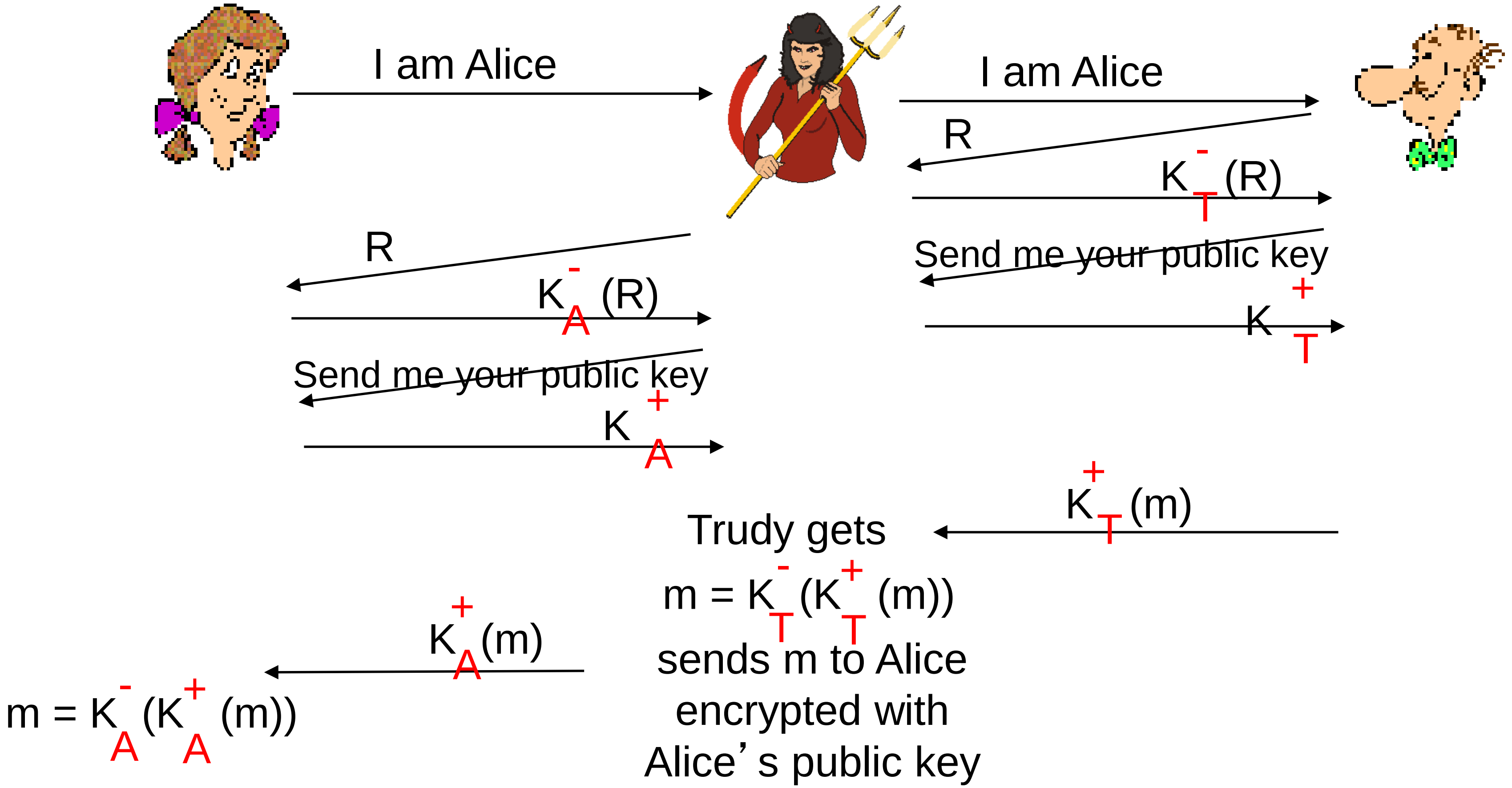
Hash function algorithms

- MD5 hash function widely used (RFC 1321)
  - computes 128-bit message digest in 4-step process.
- SHA-1 is also used
  - US standard [NIST, FIPS PUB 180-1]
  - 160-bit message digest

- Other SHA standards: https://en.wikipedia.org/wiki/SHA-1

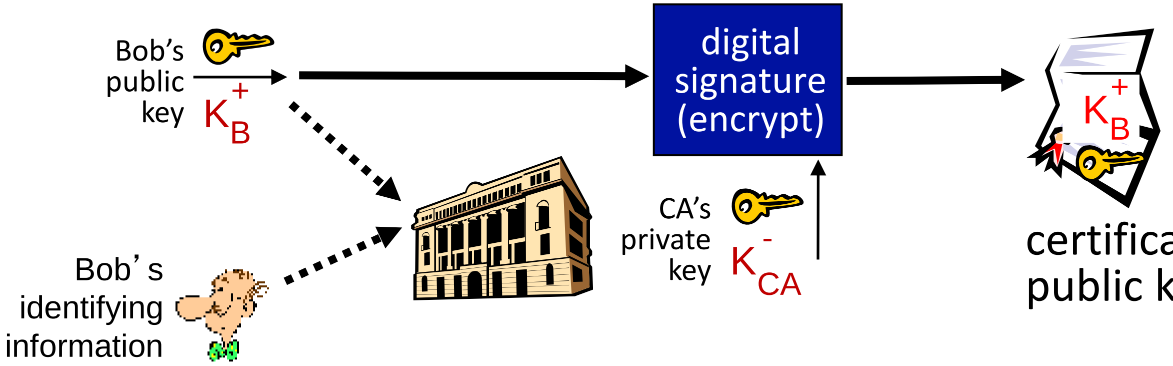
Can we fix this?

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



Public key Certification Authorities (CA)

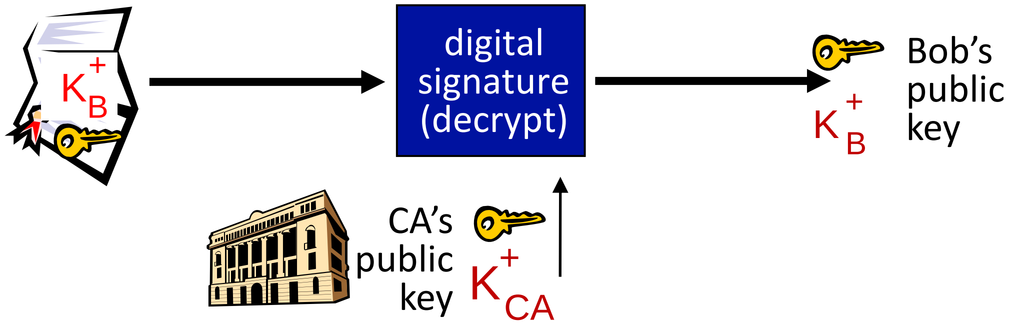
- Certification authority (CA): binds public key to particular entity, E
- Entity (person, website, router) registers its public key with CA
  - E provides "proof of identity" to CA
  - CA creates certificate binding identity E to E's public key
  - certificate containing E's public key digitally signed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



certificate for Bob's public key, signed by CA

Public key Certification Authorities (CA)

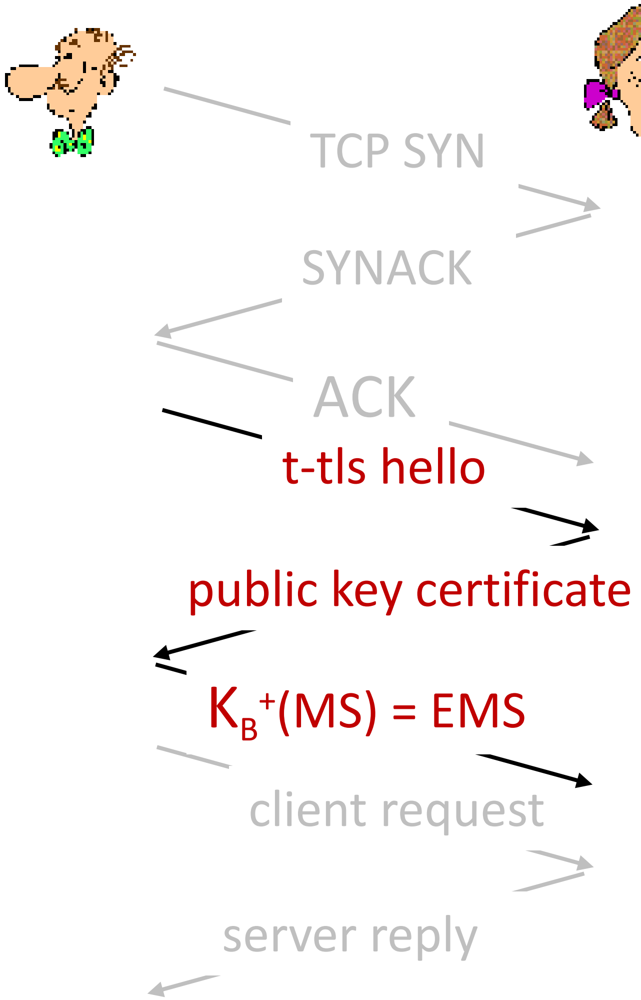
- When Alice wants Bob's public key:
  - Gets Bob's certificate (Bob or elsewhere)
  - Apply CA's public key to Bob's certificate, get Bob's public key



Transport-layer security (TLS)

- widely deployed security protocol above the transport layer
  - supported by almost all browsers, web servers: https (port 443)
- provides:
  - confidentiality: via *symmetric encryption*
  - integrity: via *cryptographic hashing*
  - authentication: via *public key cryptography*

all techniques we have studied!



t-tls handshake phase:

- Bob establishes TCP connection with Alice
- Bob verifies that Alice is really Alice
- Bob sends Alice a master secret key (MS), used to generate all other keys for TLS session
- potential issues:
  - 3 RTT before client can start receiving data (including TCP handshake)

Useful link: https://comodosslstore.com/blog/what-is-ssl-tls-client-authentication-how-does-it-work.html#:~:text=SSL%2FTLS%20client%20authentication%2C%20as,ahead%20and%20establishes%20a%20connection.