

Computer Networks

COL 334/672

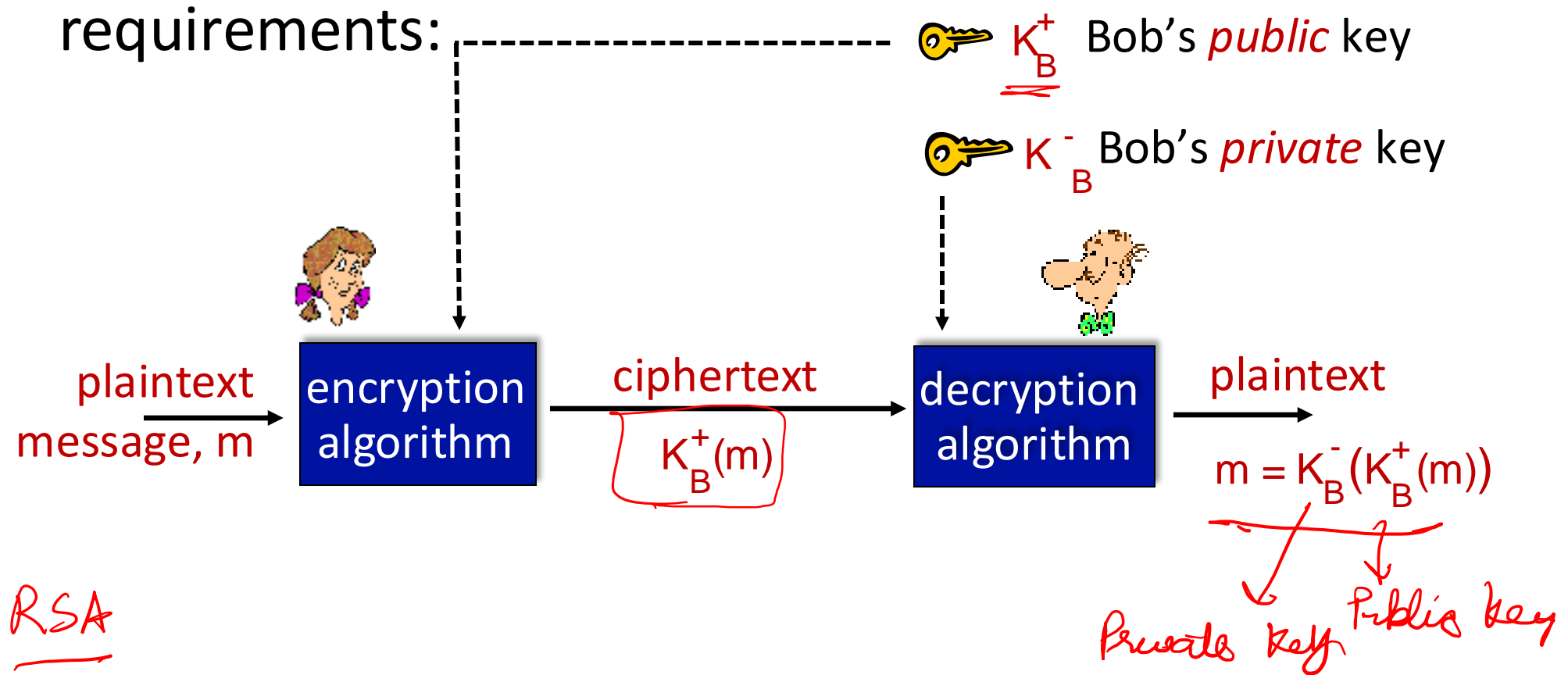
Network Security

Tarun Mangla

Slides adapted from KR

Sem 1, 2024-25

Public Key Cryptography



Prerequisite: modular arithmetic

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

$$6 \bmod 4 = 2$$

- 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: Creating public/private key pair

if a and n are co-prime

$$a^z \equiv 1 \pmod{n}$$

Euler's theorem

if p is prime and a and p are co-prime

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

$$a^p \equiv a \pmod{p} \rightarrow \text{Fermat's thm}$$

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

of co-primes of n that are less than n

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: $ed \pmod{z} = 1$).

$$ed \pmod{z} = 1 \quad \text{or} \quad ed \equiv 1 \pmod{z}$$

5. *public* key is (n, e) . *private* key is (n, d) .

K_B^+

K_B^-

RSA: encryption, decryption

public key
or

private key

$$\begin{array}{r} 1001 \\ = 9 \\ \hline \end{array}$$

$$n = 35$$

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

$$m = c^d \bmod n$$

$$\text{magic happens!} \quad m = \underbrace{(m^e \bmod n)}_c^d \bmod n$$

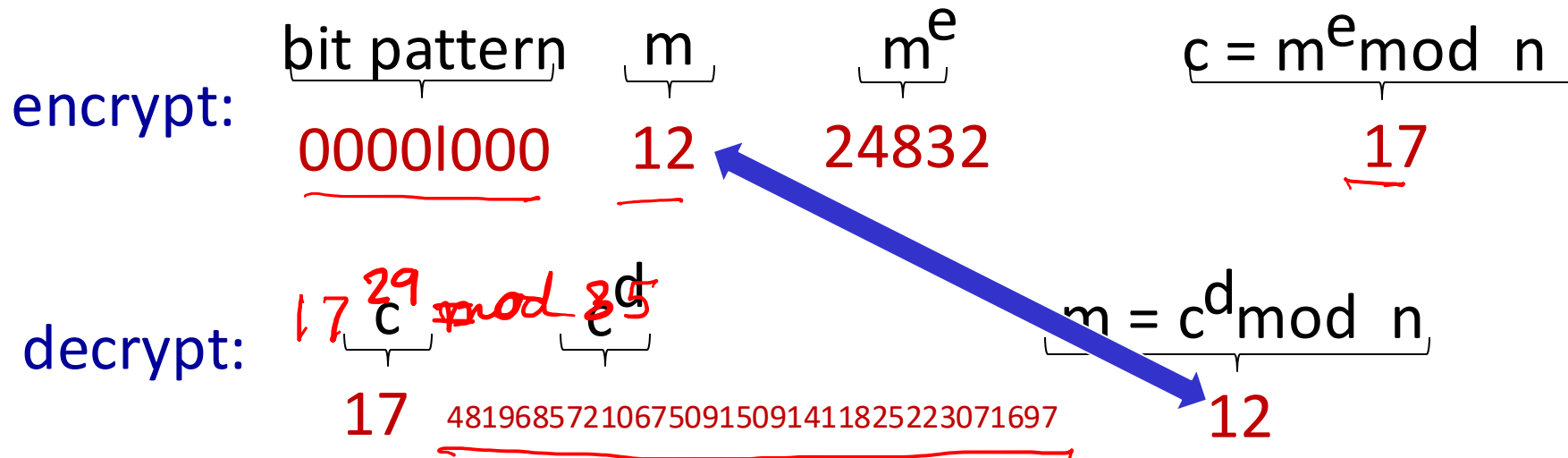
RSA example:

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.



$$ed = 2k + 1$$

$$c = m^e \bmod n$$

$$(m^e \bmod n)^d \bmod n$$

Why does RSA work?

$$ed \equiv 1 \bmod z$$

Need to show this
 $c^d \bmod n = m$

$$\downarrow$$

$$\underline{m^{ed} \bmod n}$$

- 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,

$$c^d \bmod n = (m^e \bmod n)^d \bmod n$$

$$= m^{ed} \bmod n$$

$$= m^{(ed \bmod z)} \bmod n$$

$$= m^1 \bmod n$$

$$= m$$

$$m^{2k+1} \bmod n$$

$$= 1$$

$$m \cdot m^{2k} \bmod n$$

$$m \cdot (m^2 \bmod n)^k \bmod n$$

$$= m \bmod n$$

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

\sqrt{n}

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

What is network security?

confidentiality: only sender, intended receiver should “understand” message contents

- sender encrypts message
- receiver decrypts message

message integrity: sender, receiver want to ensure message not altered (in transit, or afterwards) without detection

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

access and availability: services must be accessible and available to users

What about integrity and authenticity?

Recall

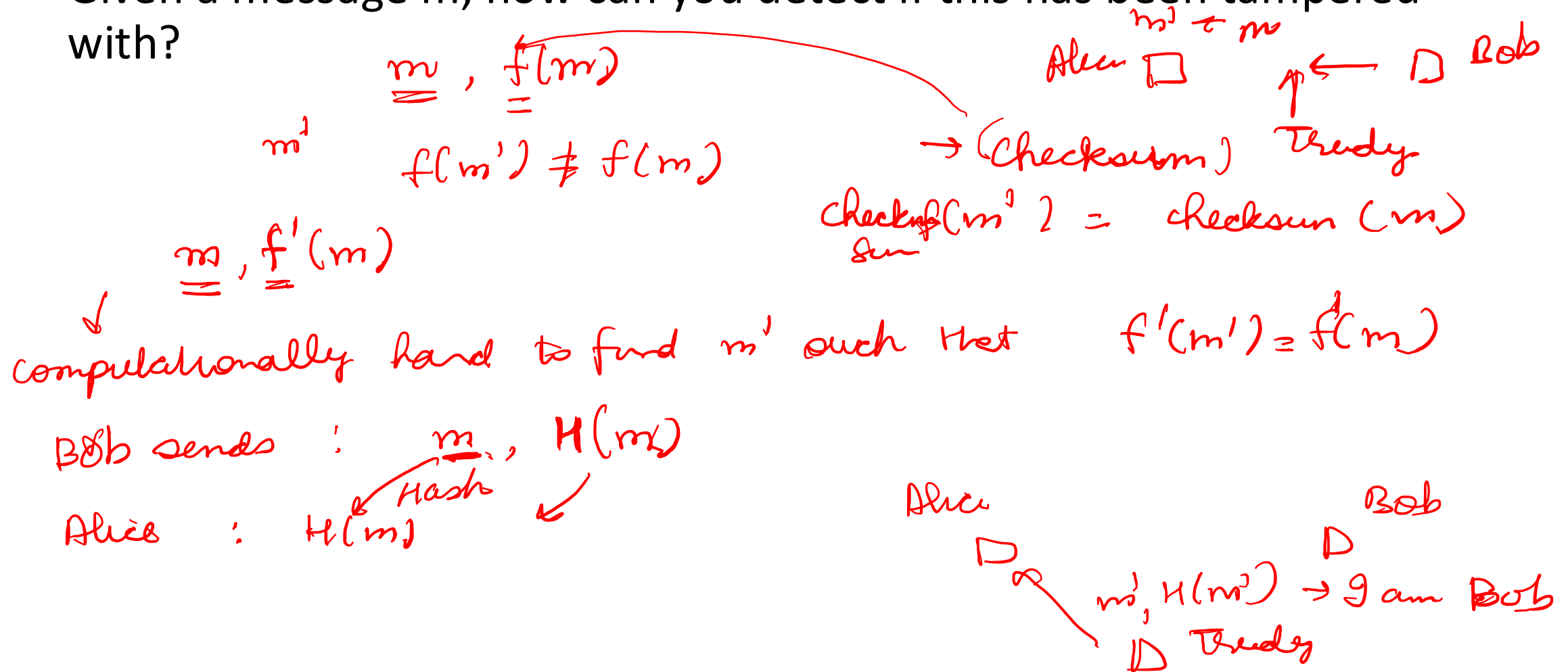
- **message integrity**: sender, receiver want to ensure message not altered (in transit, or afterwards) without detection
- **authentication**: sender, receiver want to confirm identity of each other

Encryption alone can not provide integrity and authentication.

How to provide integrity and authenticity?

Providing integrity

- Given a message m , how can you detect if this has been tampered with?



Providing integrity

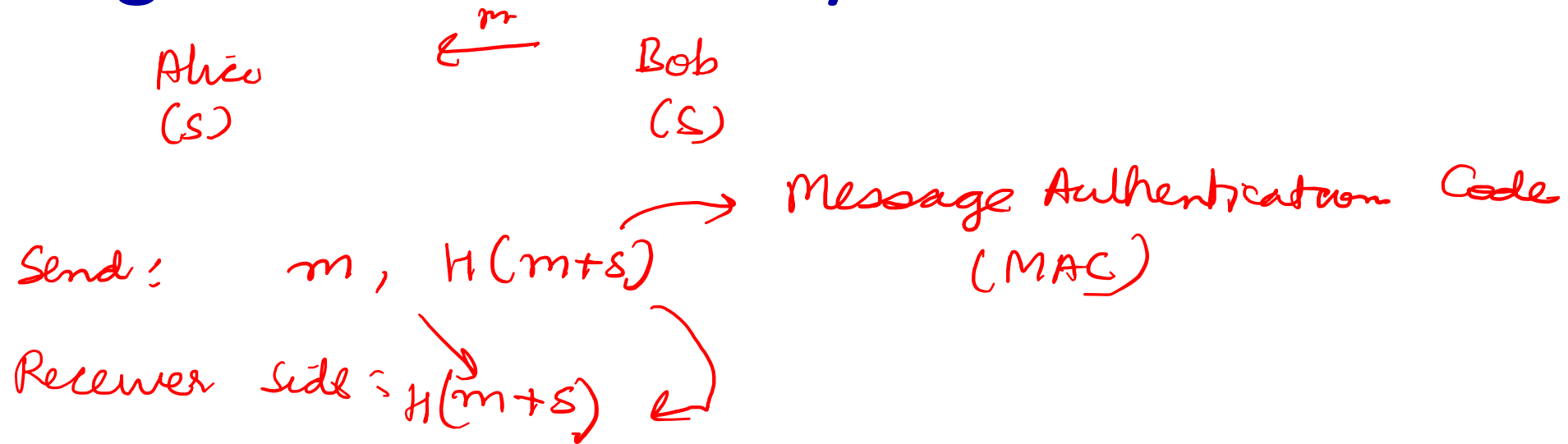
- Given a message m , how can you detect if this has been tampered with?

Hash function properties:

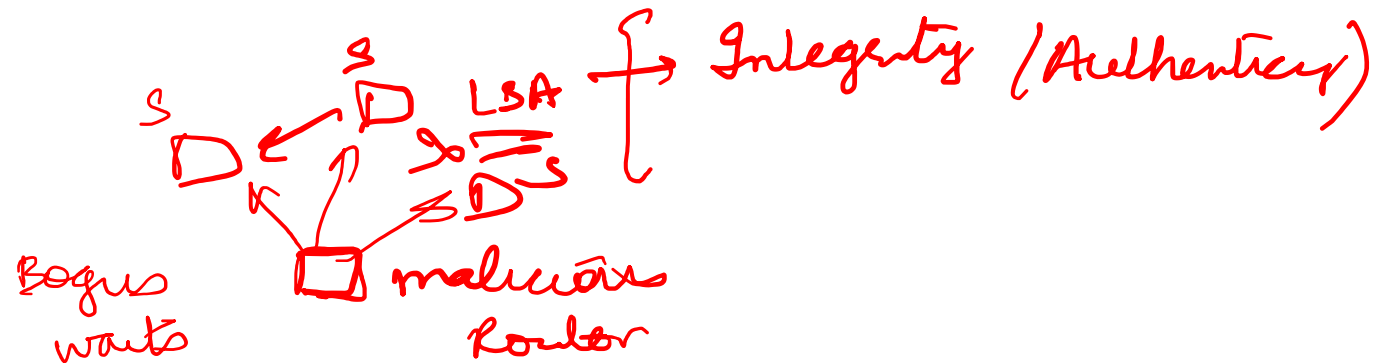
- many-to-1
- produces fixed-size msg digest
- given message digest x , computationally infeasible to find m such that $x = H(m)$

How to provide authenticity?

Using Shared Secret Key

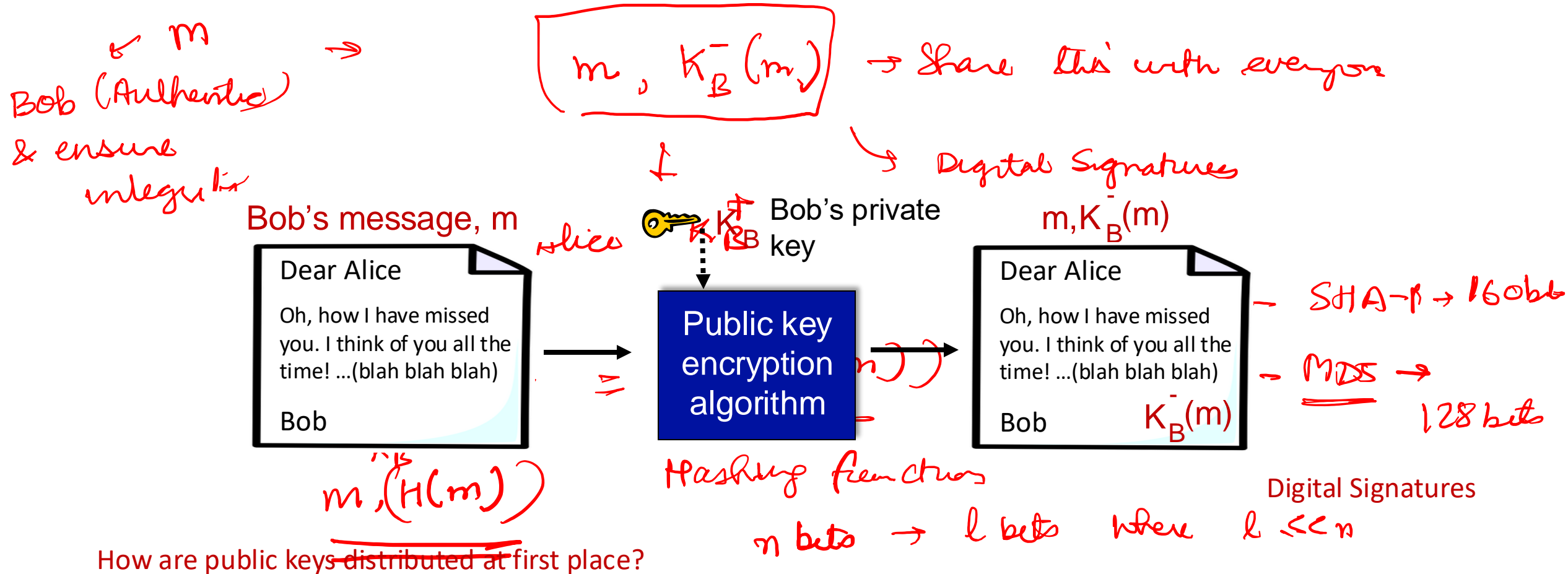


How do you have a shared secret key at first place?



Using Public Key Cryptography

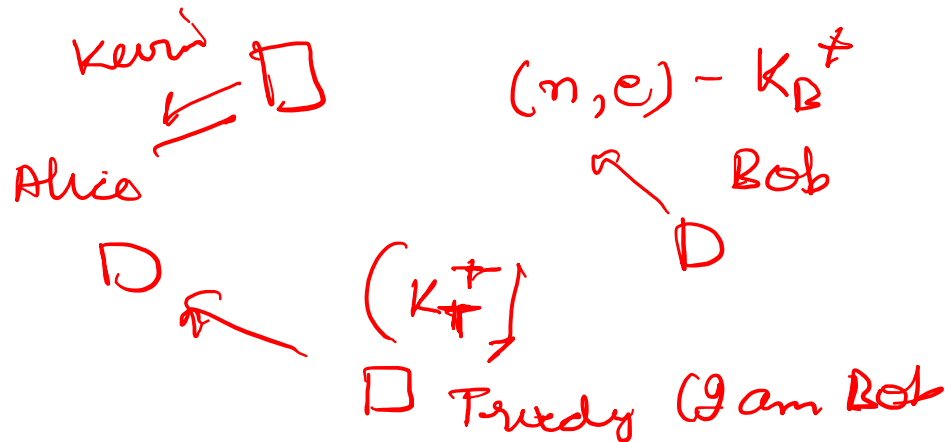
Public key cryptography



How to share secret key?

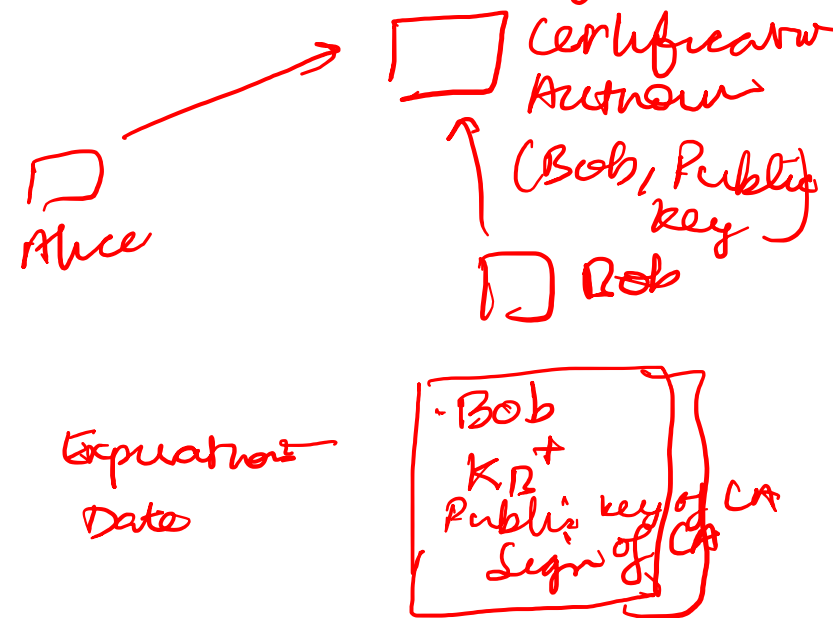
How to securely share public key?

- ① Conf
- ② Auth
- ③ Integrity



→ May be Alice knows IP addr of Bob (spoofing)

Public Key Infrastructure

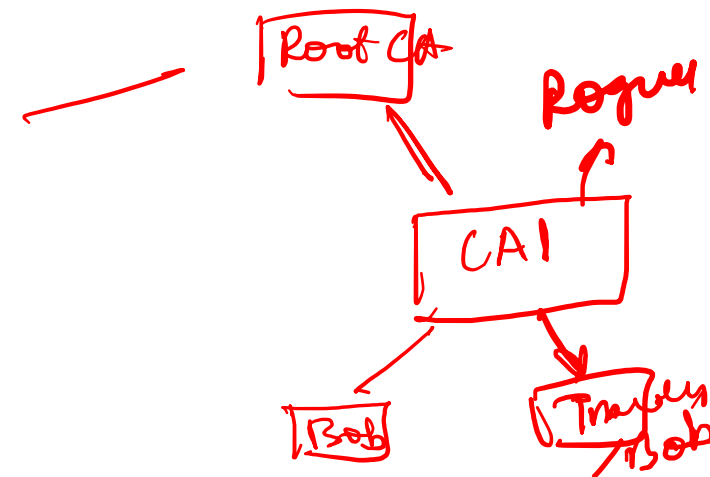
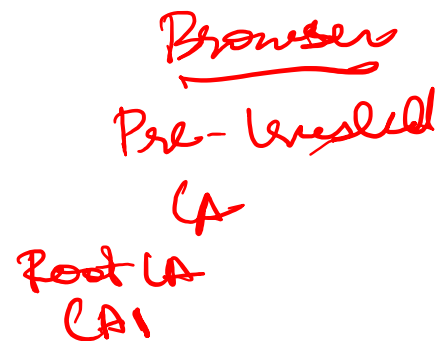


→ Some trust b/w Bob on Alice

Certificates → Trusted entity

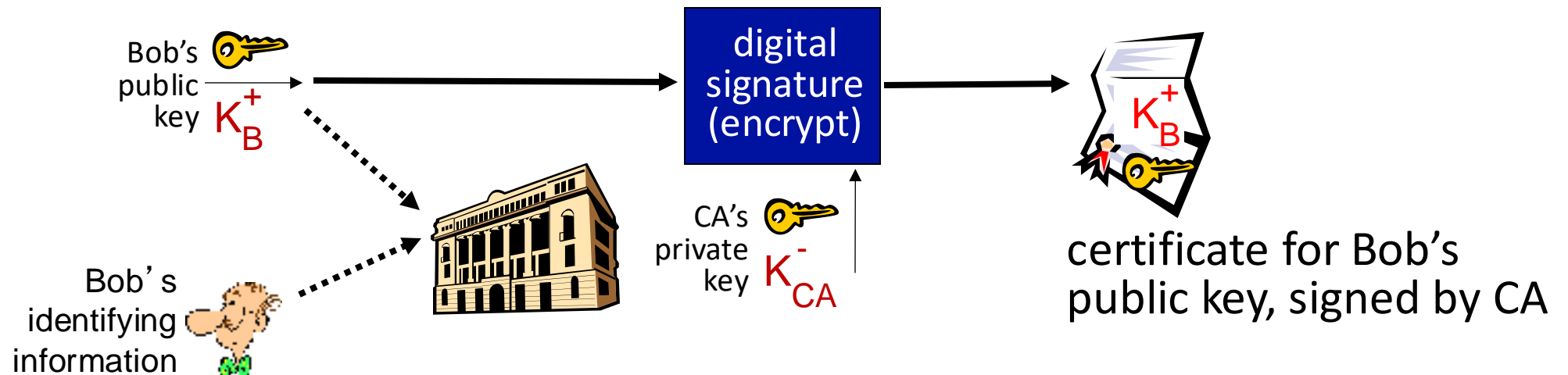
$X \rightarrow Y \rightarrow Z$

$X \rightarrow Z$



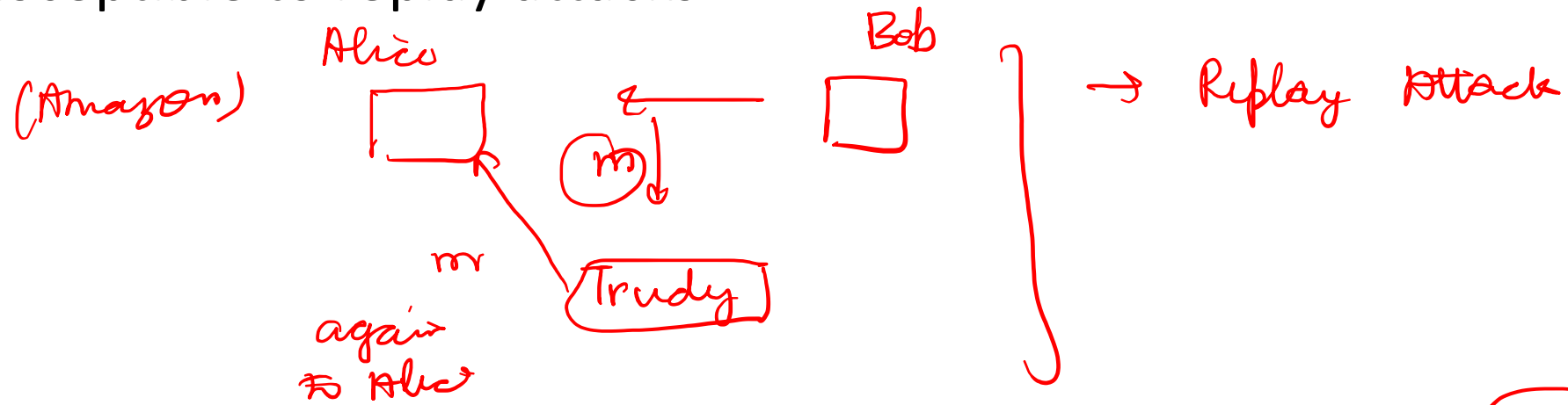
Public key Certification Authorities (CA)

- **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”



Revisiting Authentication

- Susceptible to replay attacks



$m / \underline{(\quad)}$] \rightarrow (timeliness of the message)

timestamp

- ① Com
- ② Auth
- ③ Int
- ④ Timeliness

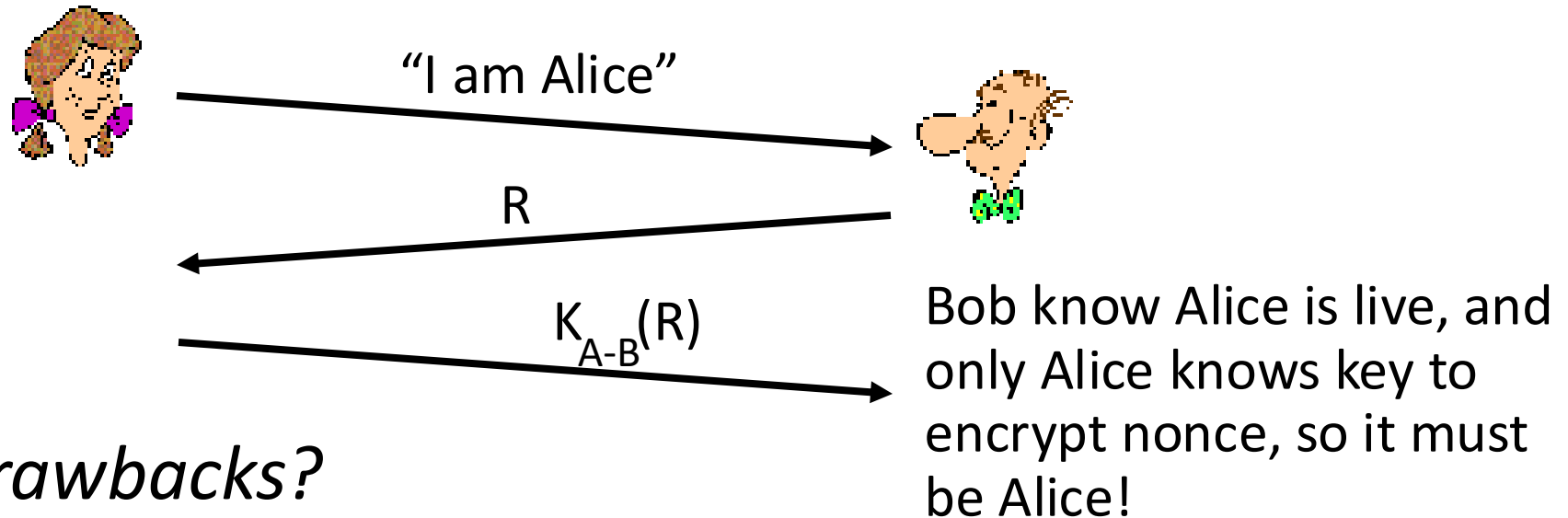
Authentication

Goal: avoid playback attack

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

protocol: to prove Alice “live”, Bob sends Alice nonce, R

- Alice must return R, encrypted with shared secret key

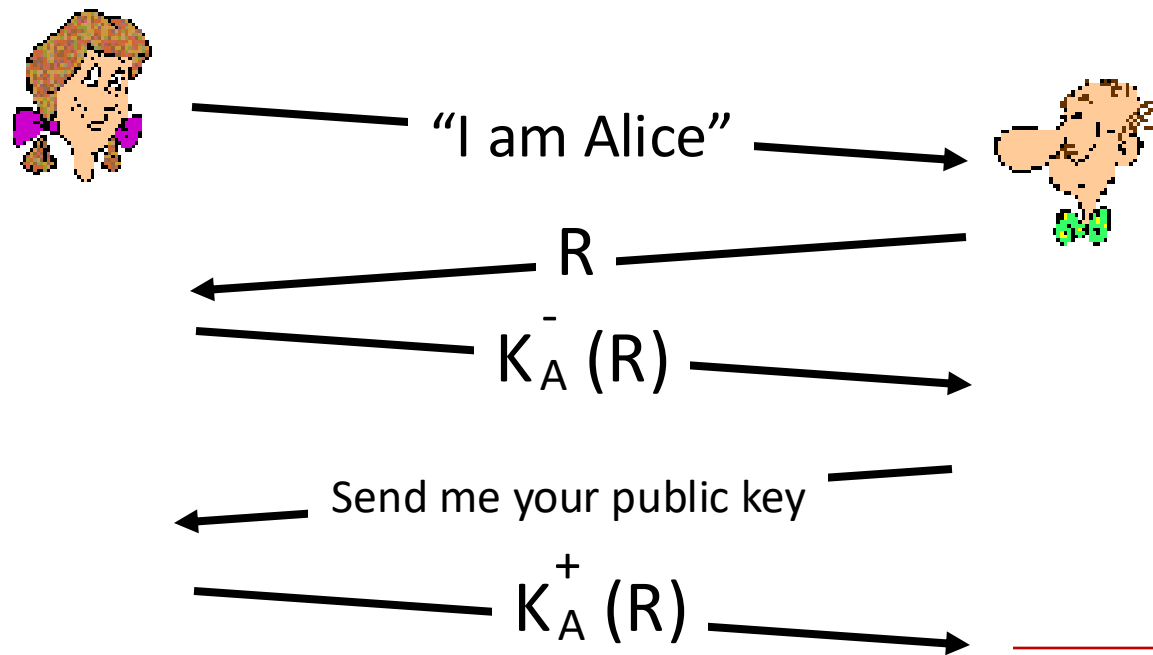


Failures, drawbacks?

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

Live Authentication using Public Keys

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

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