Computer Networks
COL 334/672

Network Security

Slides adapted from KR

Sem 1, 2025-26

Quiz on Moodle, Password: rsa

ABR: Consider a video player that uses the following bitrates

Low	1 Mbps
Med	2 Mbps
High	4 Mbps

Maximum buffer = 5 sec
Video chunk size = 2 seconds
Initial buffered video = 3 sec
Available bandwidth is as follows:

Time(s)	B/w (Mbps)
0-5	4
5-10	1
10-15	2

CBC: Consider a 3-bit cipher block chaining scheme that uses the following cipher:

Plaintext (3-bit)	Ciphertext (3-bit)
000	101
001	011
010	000
011	110
100	010
101	111
110	001
111	100

RSA: Consider Bob is using RSA. He advertises his public key to Alice

Bob ——— Alice

K_B^- = (55, x)

K_B^+ (55, 3)

K_B^+ : Public Key

K_B^- : Private key

Recap: Cryptographic Techniques for Network Security

- Confidentiality
- Message Integrity
- End-point Authentication

How to ensure message integrity?

Use hashing functions

Hash function algorithms

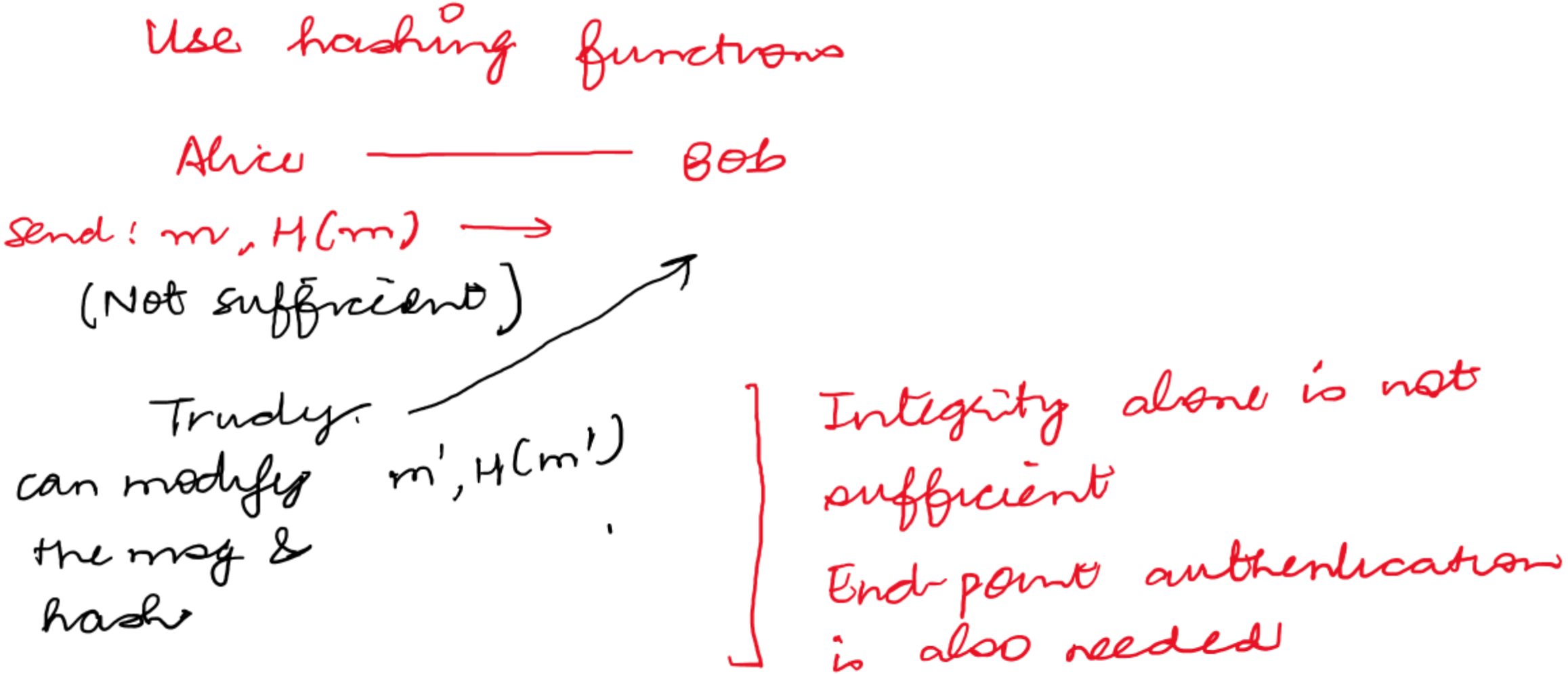
Hash function properties:

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

- 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

How to ensure message integrity?

Use hashing functions



How to do both endpoint authentication and message integrity?

Symmetric Cryptos

Alice — Bob

s: secret key

m, H(m+s) ⟶

H(m+s) is called message authentication code (MAC)

HMAC is the protocol that works on this principle

Issue: How to exchange s?

Public key Crypto

Idea 1: Use it for exchanging shared key

Idea 2:

Alice — Bob

K_A^+

K_A^-

Q: Can we use only Public key crypto for message authentication?

Digital Signature

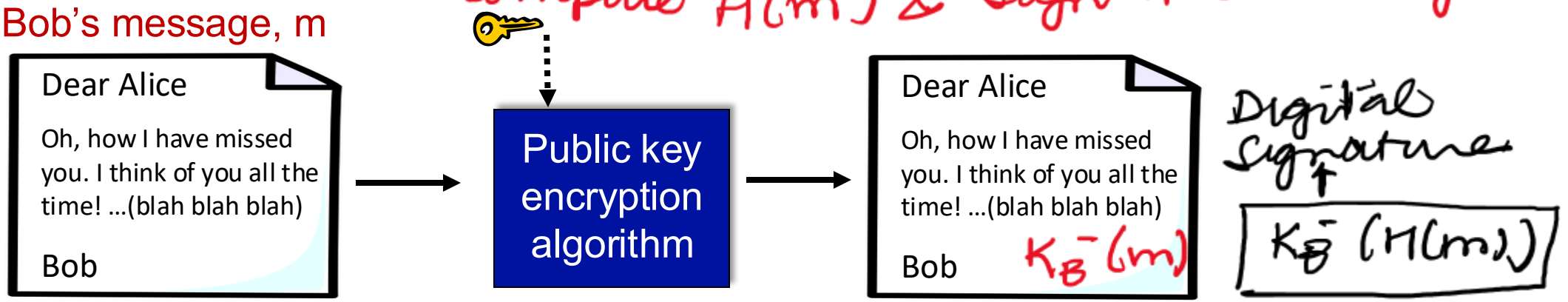
cryptographic technique analogous to hand-written signatures:

- 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

$K_B^-(m)$: signature [verifies that Bob wrote the msg]

But m can be large; computing K_B^- can be expensive

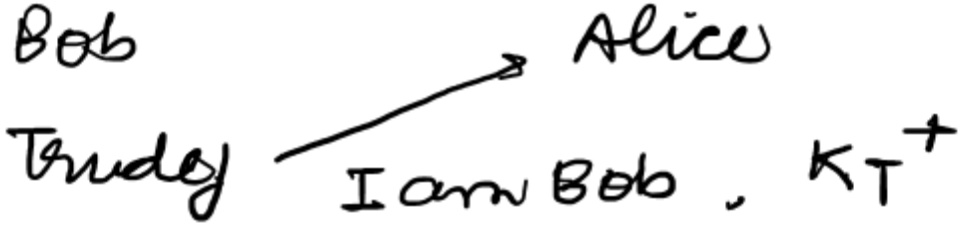
compute H(m) & sign that using K_B^-



One pending issue!

- How to really trust whether the public key I have received is really from Bob?

Impersonation Attack



Idea: what if someone whom I already trust certifies that this is Bob. certification authority

Two Questions

Who is that someone?

[centralized entity
state govt, private entity
decentralized web of trust]

How do they certify? → using digital certificates

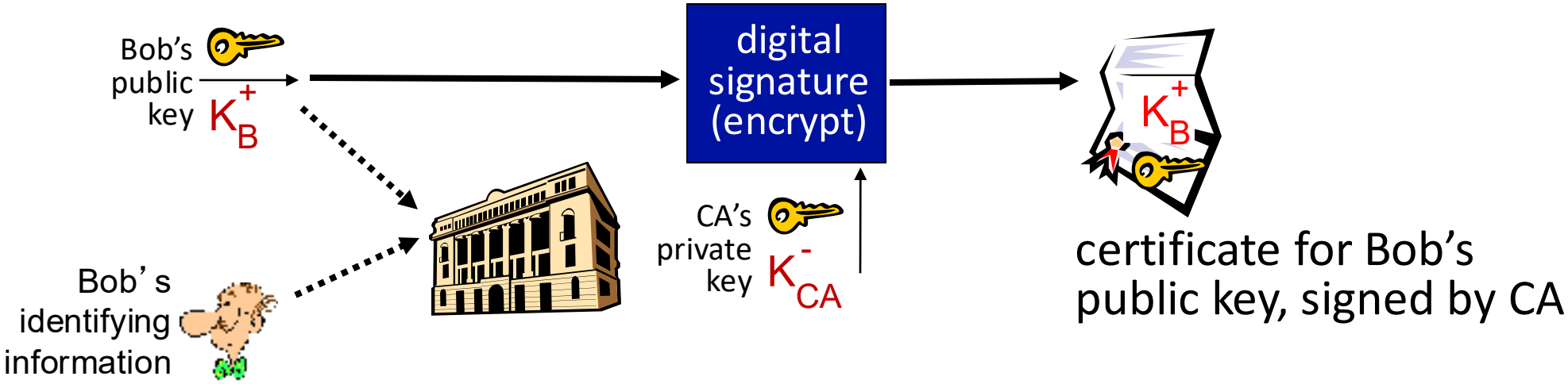
e.g. Verisign Bob
K_V^-, K_V^+ K_B^-, K_B^+

certificate

K_B^+ } m
K_V^- (H(m))

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

Alice ⟶ Bob

m: send 1000 Rs to john

Sends: m, H(m+S) to Bob

Trudy can simply replay the message

Bob will send money to john twice

REPLAY ATTACK → How to solve this?

[Use Random # for each session]

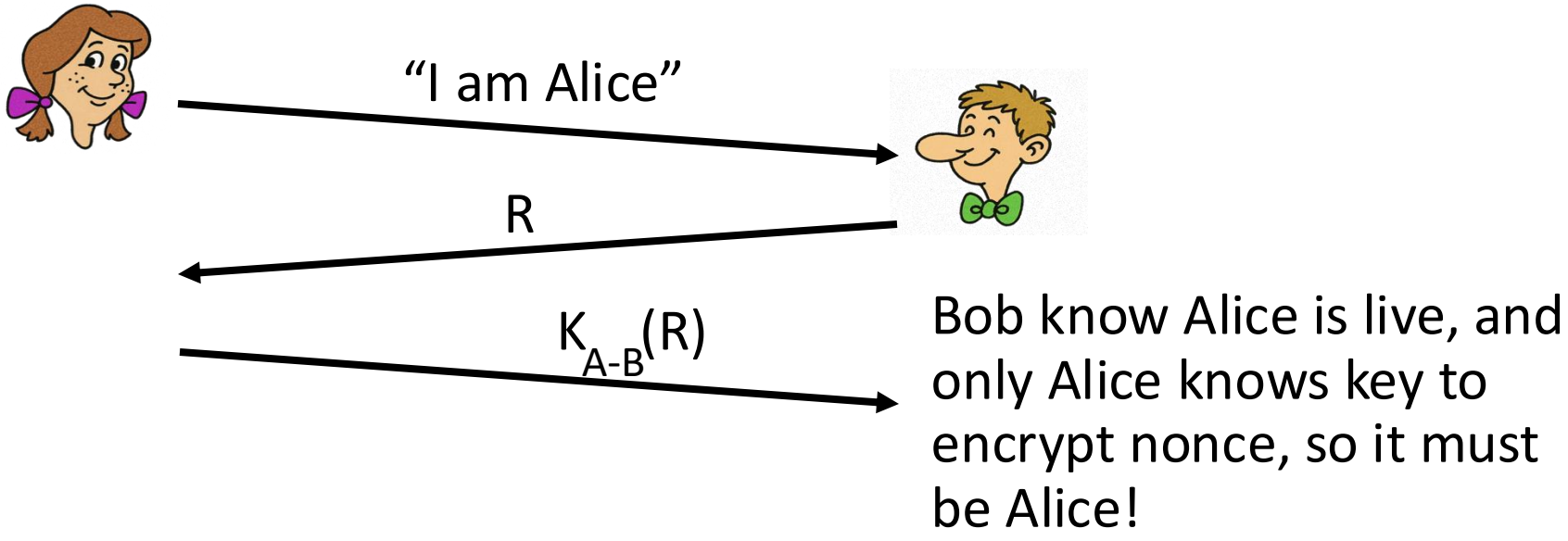
Authentication

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



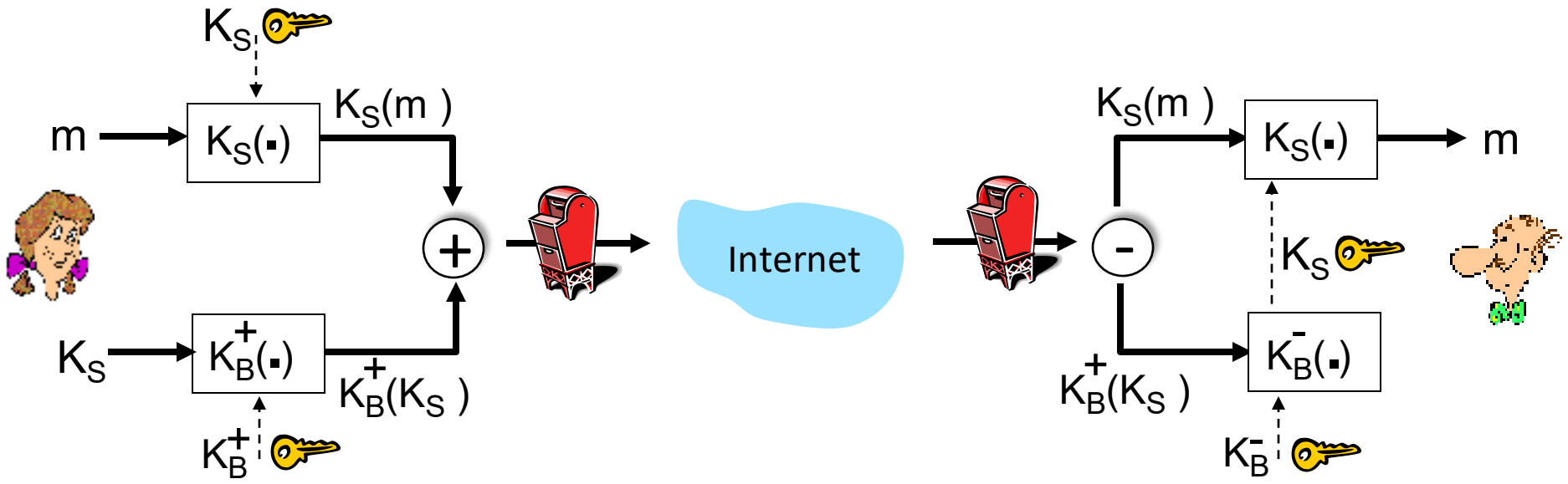
Implementing Security

- Security for:
  - Email
  - TCP
  - Network-layer

Operational security: firewall and IDS

Secure e-mail: confidentiality

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



Alice:

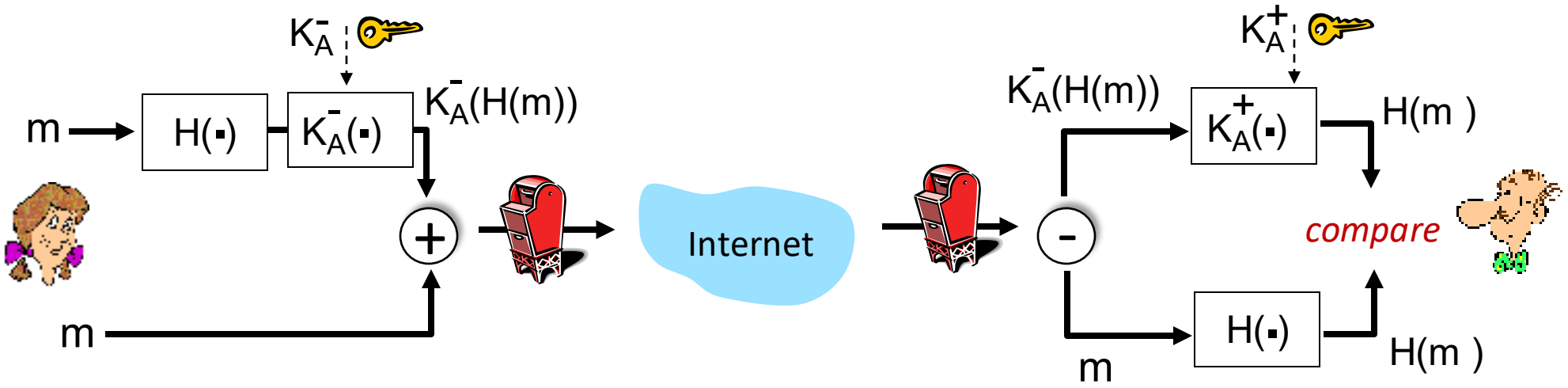
- generates random *symmetric* private key, K_S
- encrypts message with K_S (for efficiency)
- also encrypts K_S with Bob's public key
- sends both $K_S(m)$ and $K^+_B(K_S)$ to Bob

Bob:

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

Secure e-mail: integrity, authentication

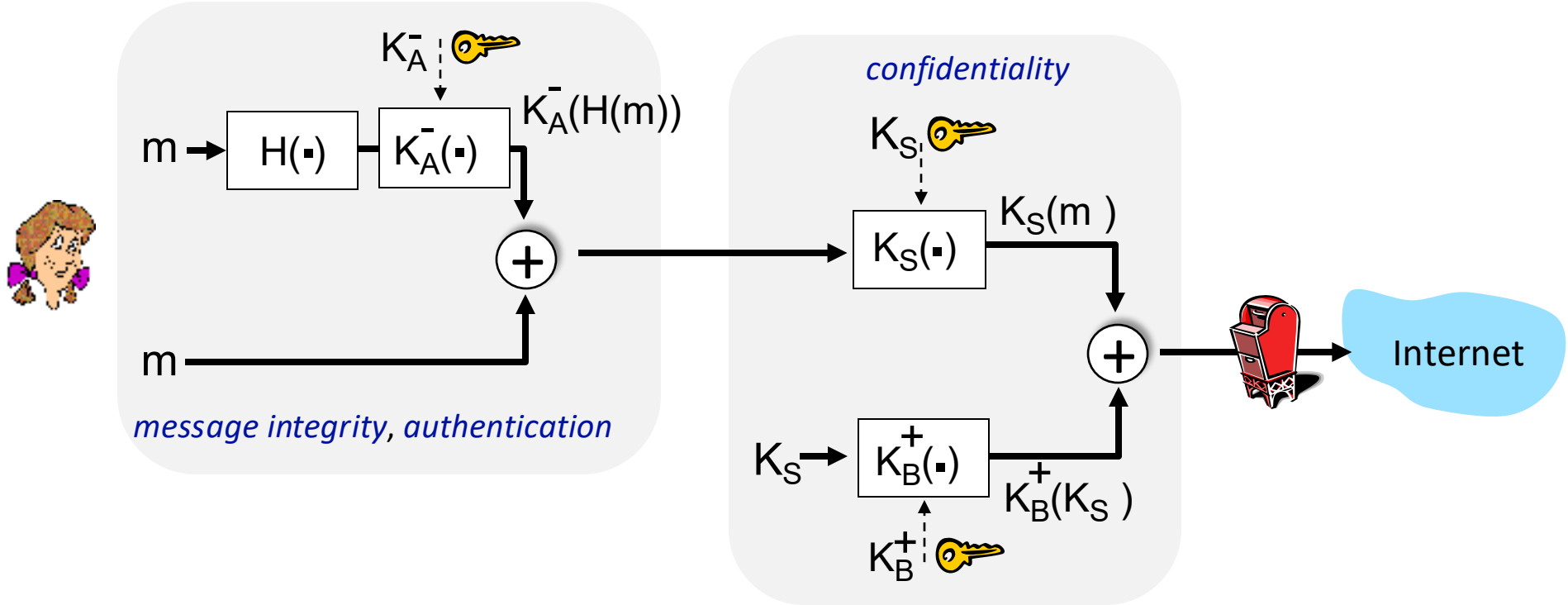
Alice wants to send m to Bob, with *message integrity*, *authentication*



- 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*



How do Alice and Bob obtain each other's public keys?

Next Class

- Security for:
  - Email
  - **TCP**
  - Network-layer

Operational security: firewall and IDS