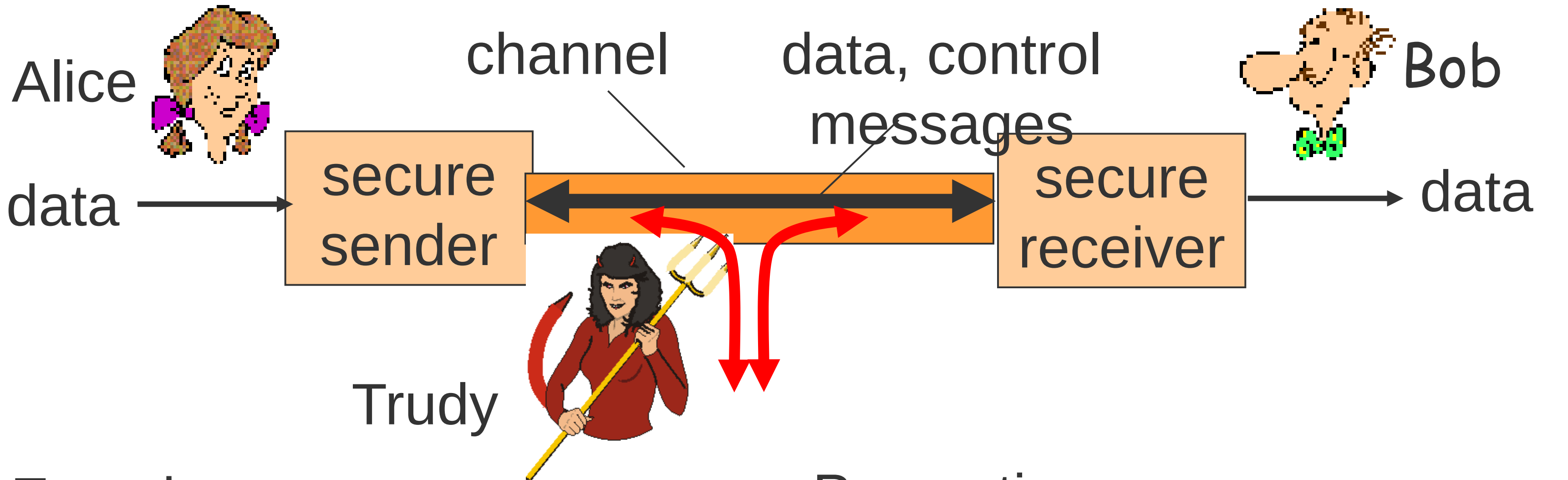
Network Security

Andy Carpenter

(Andy.Carpenter@manchester.ac.uk)

K&R: 8.1, P&D: 8

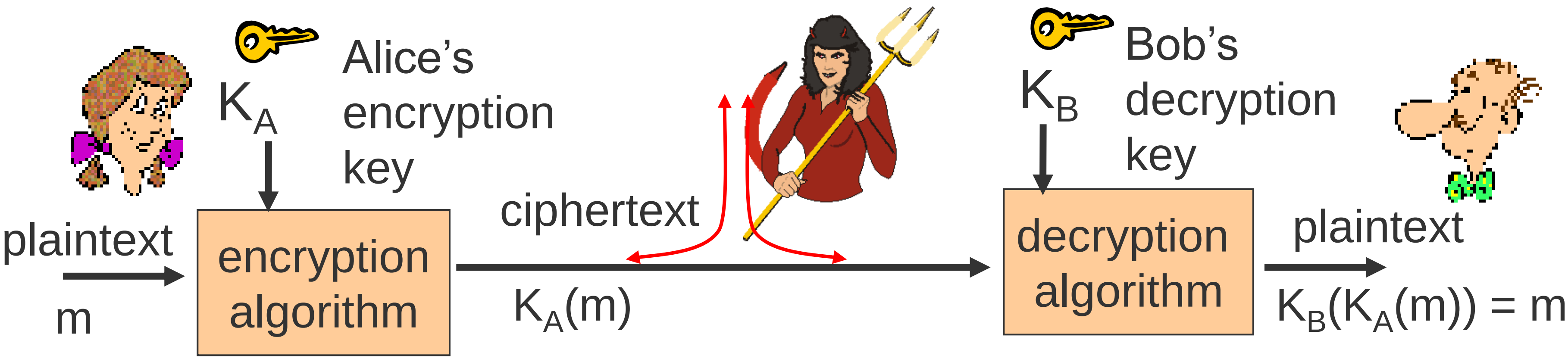
Network Security is What?



Ensuring:
- …

Preventing:
- …

Security: Implementation



- Done by cryptographic algorithms that use keys
- Algorithms are well known, keys are unique

Security comes from secrecy of secret keys

K&R: 8.2.1

Encryption: Simple Scheme

- Cryptography is substituting one thing for another
- Monoalphabetic (one letter for another) cipher:

```
 plaintext:  abcdefghijklmnopqrstuvwxyz

ciphertext:  mnbvcxzasdfghjklpoiuytrewq
```

E.g.:

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

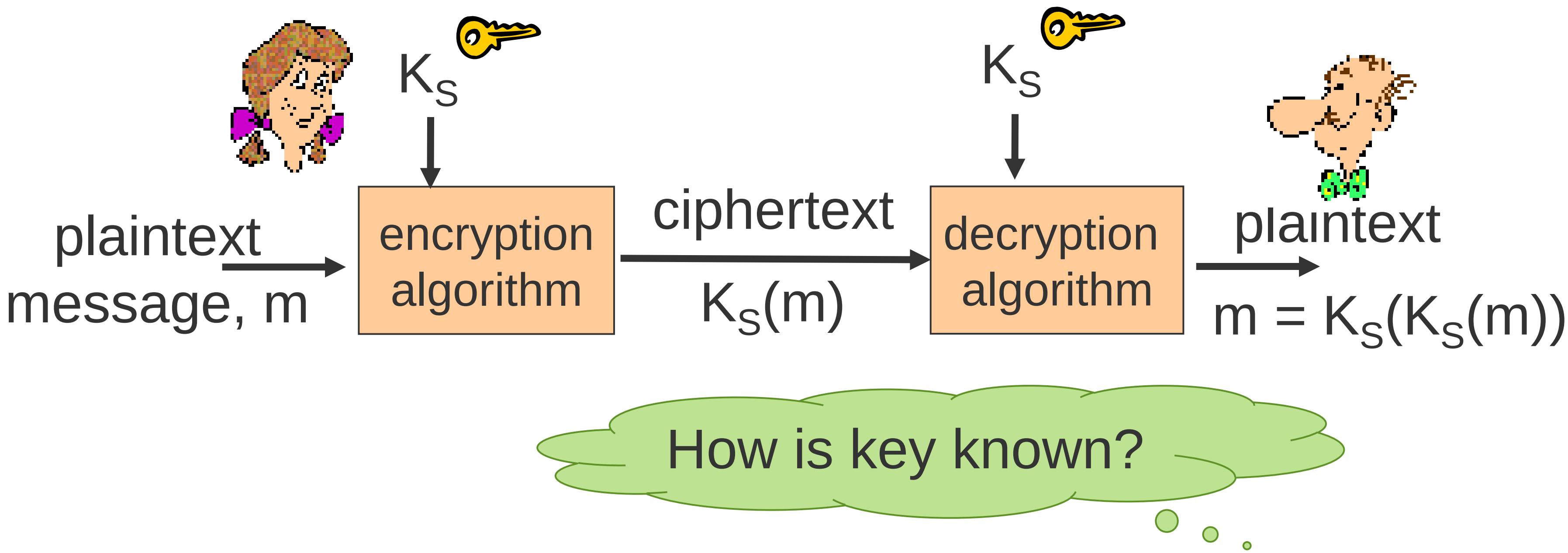
- Q: How hard to break this simple cipher?:
  - brute force (how hard?)
  - other?

Encryption: Breaking it

- **Cipher-text only attack**: two approaches:
  - Search through all keys: for each try
    - must distinguish plaintext from gibberish
  - Statistical analysis
- **Known-plaintext attack**: Trudy has some plaintext corresponding to some ciphertext
  - e.g. in monoalphabetic cipher
  - Trudy determines pairings for a,l,i,c,e,b,o,
- **Chosen-plaintext attack**:
  - get the cyphertext for some chosen plaintext

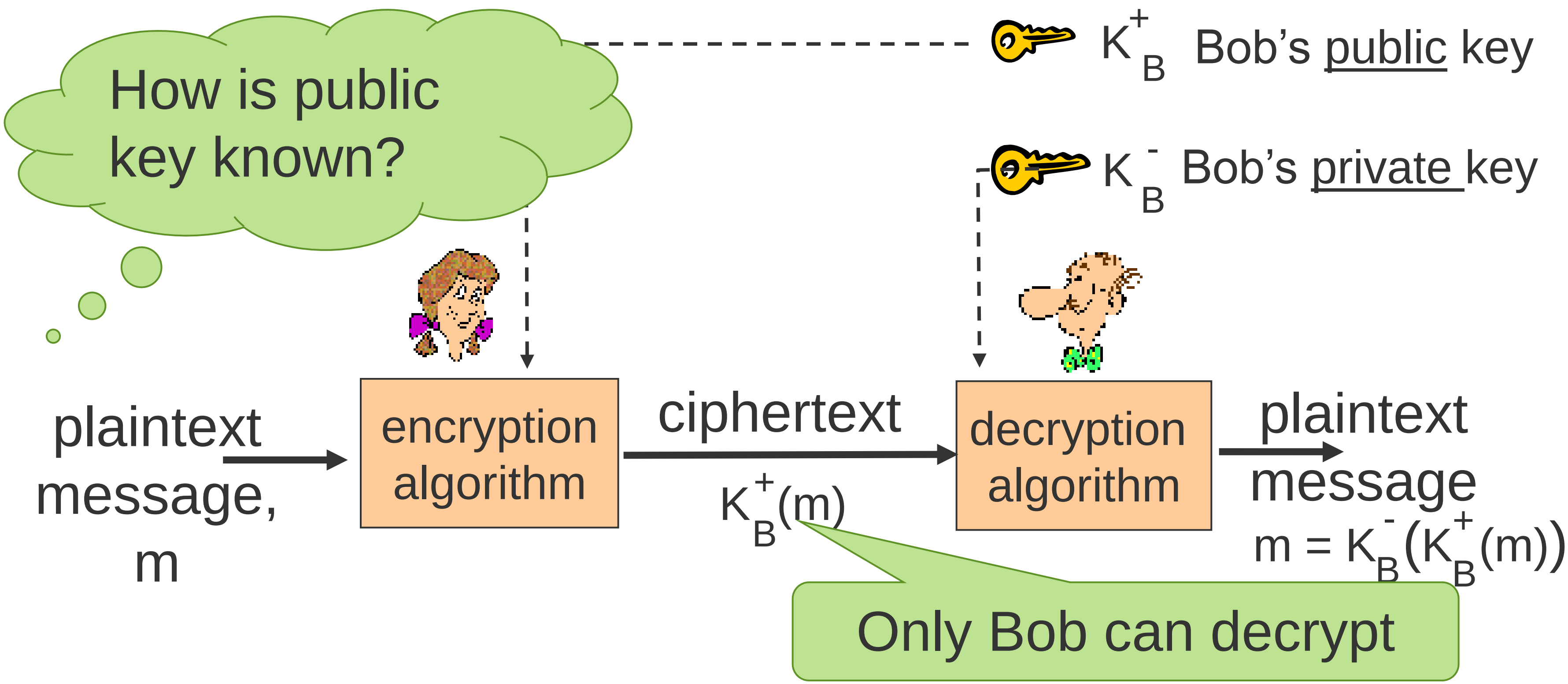
Minimise use of keys

Algorithms: Symmetric



- Both principles share a single secret key
- Examples:
  – Data Encryption Standard (DES)
  – Advanced Encryption Standard (AES)

Algorithms: Public Key



- Uses two keys called public and private (secret) keys
- Example: Rivest, Shamir and Adleman (RSA)

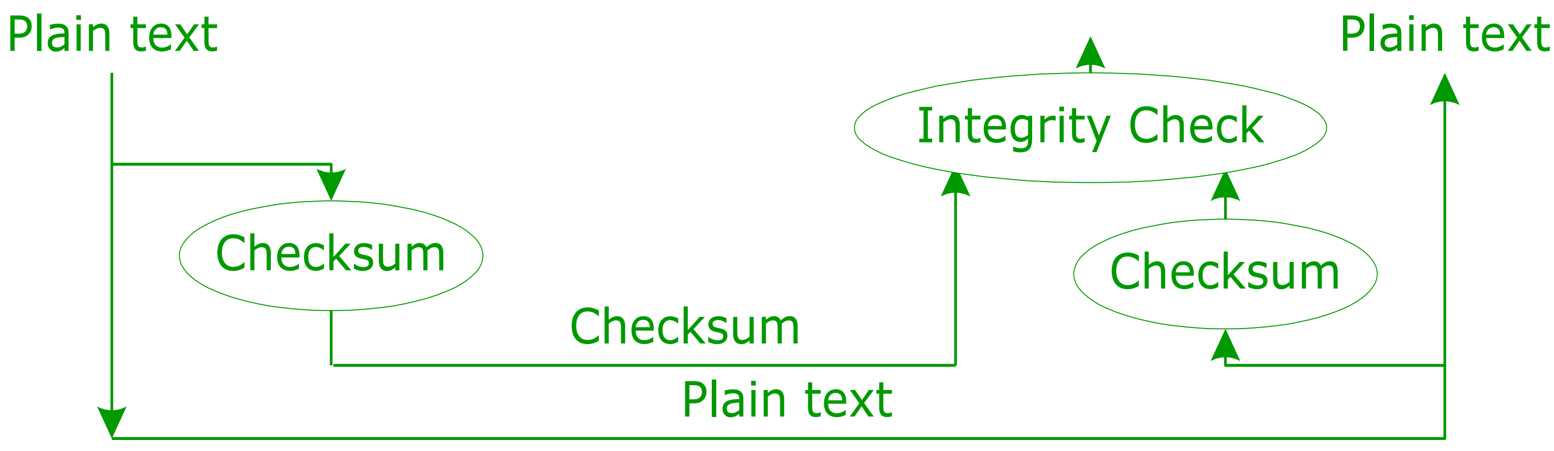
Ciphers: RSA Property

- The following property will be *very* useful later:

$$\underbrace{K_B^-(K_B^+(m))}_{} = m = \underbrace{K_B^+(K_B^-(m))}_{}$$

K&R: 8.3.1, P&D: 8.1

Algorithms: Hashing

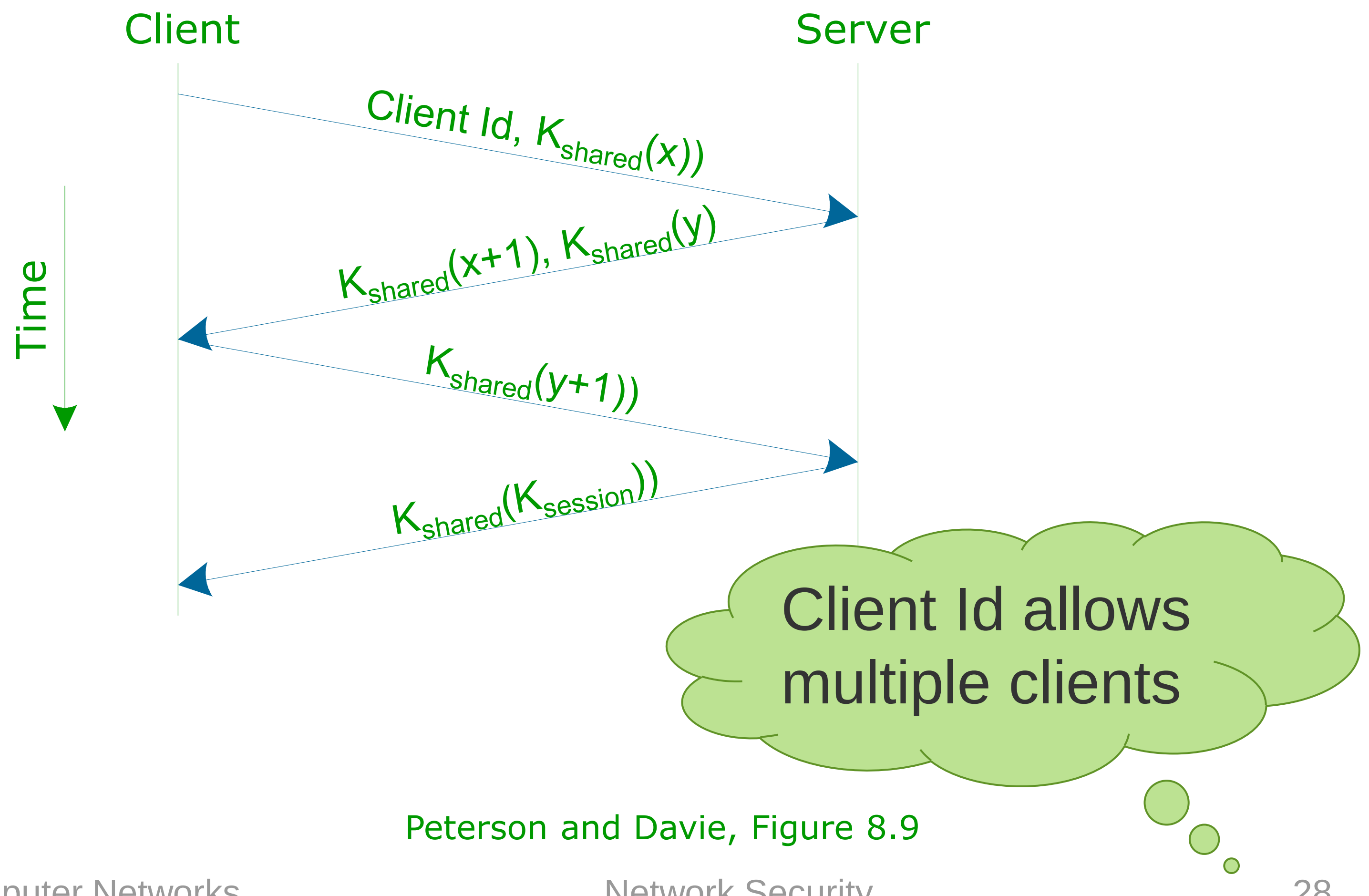


- Computes cryptographic checksum of data
- Used as fixed length message signatures
- Examples:
  - MD5: 128-bit digest [RFC 1321]
  - SHA-1: 160-bit digest [NIST, FIPS PUB 180-1]

Security Mechanisms

- Algorithms are only elements in network security
- Need mechanisms and protocols for specific tasks:
  - authentication of remote users
  - ensuring where data comes from
  - distributing keys
- Exponentiation is computationally intensive
  - DES is at least 100 times faster than RSA
- Public/private keys used to authenticate and securely exchange a shared symmetric key K_S
- Once have K_S, use symmetric key cryptography
- Good practice minimises the use of individual keys

Authentication: Three-Way Handshake



Peterson and Davie, Figure 8.9

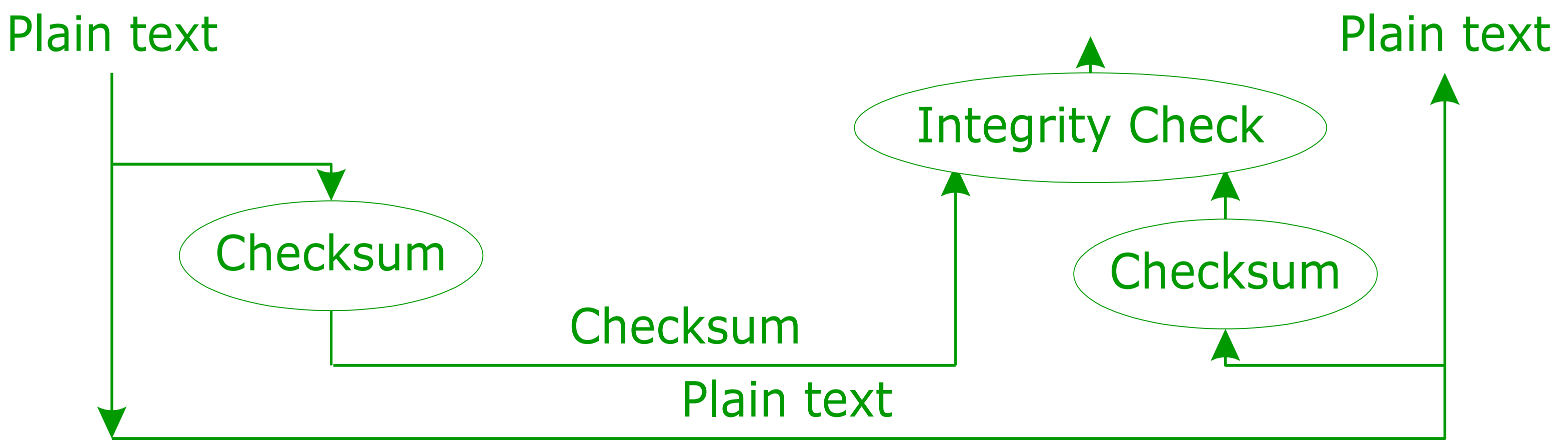
Ciphers: RSA Property

- The following property will be *very* useful later:

$$\underbrace{K_B^-(K_B^+(m))} = m = \underbrace{K_B^+(K_B^-(m))}$$

K&R: 8.3.1, P&D: 8.1

Algorithms: Hashing



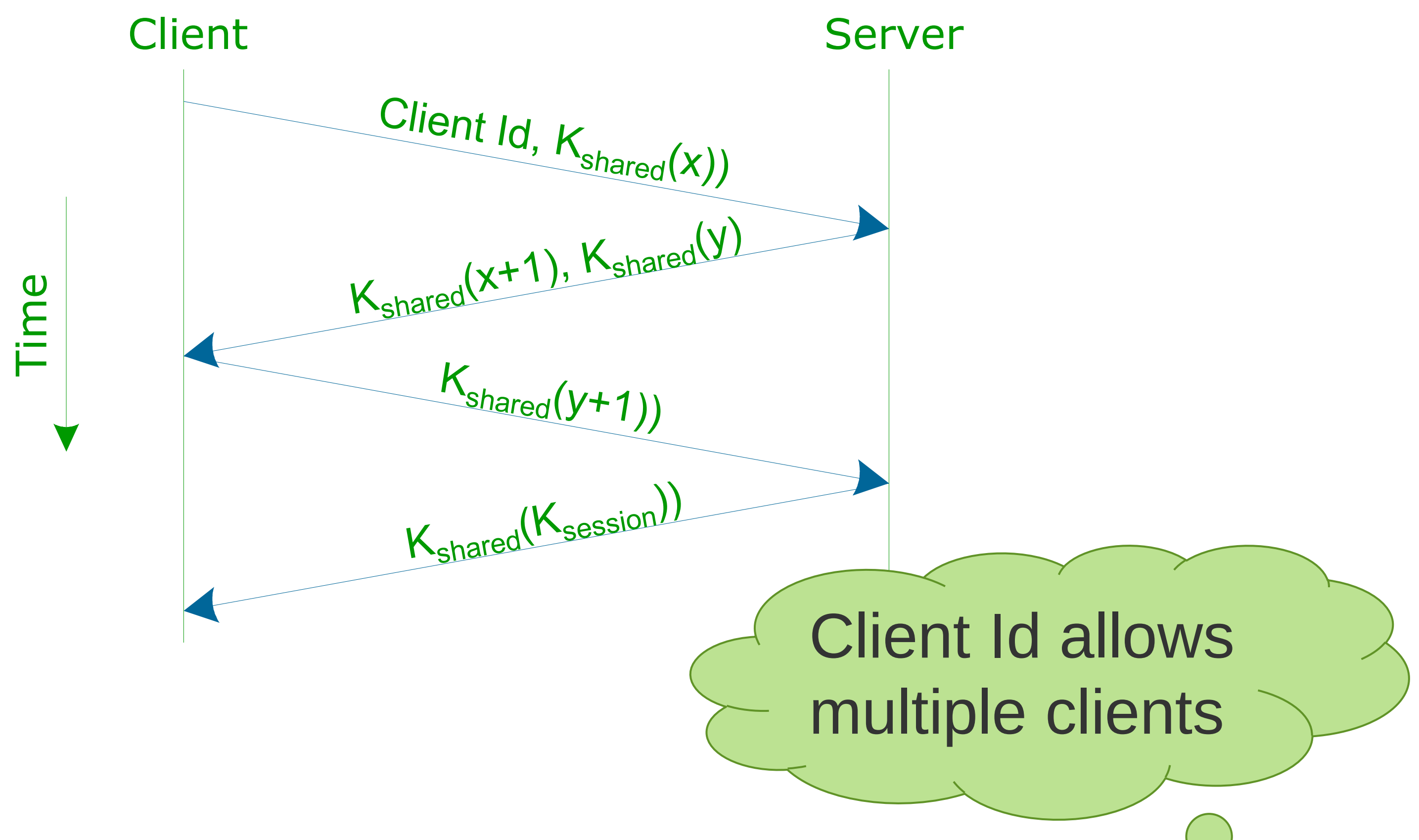
- Computes cryptographic checksum of data
- Used as fixed length message signatures
- Examples:
  - MD5: 128-bit digest [RFC 1321]
  - SHA-1: 160-bit digest [NIST, FIPS PUB 180-1]

Security Mechanisms

- Algorithms are only elements in network security
- Need mechanisms and protocols for specific tasks:
  - authentication of remote users
  - ensuring where data comes from
  - distributing keys
- Exponentiation is computationally intensive
  - DES is at least 100 times faster than RSA
- Public/private keys used to authenticate and securely exchange a shared symmetric key K_S
- Once have K_S, use symmetric key cryptography
- Good practice minimises the use of individual keys

P&D: 8.2.1

Authentication: Three-Way Handshake

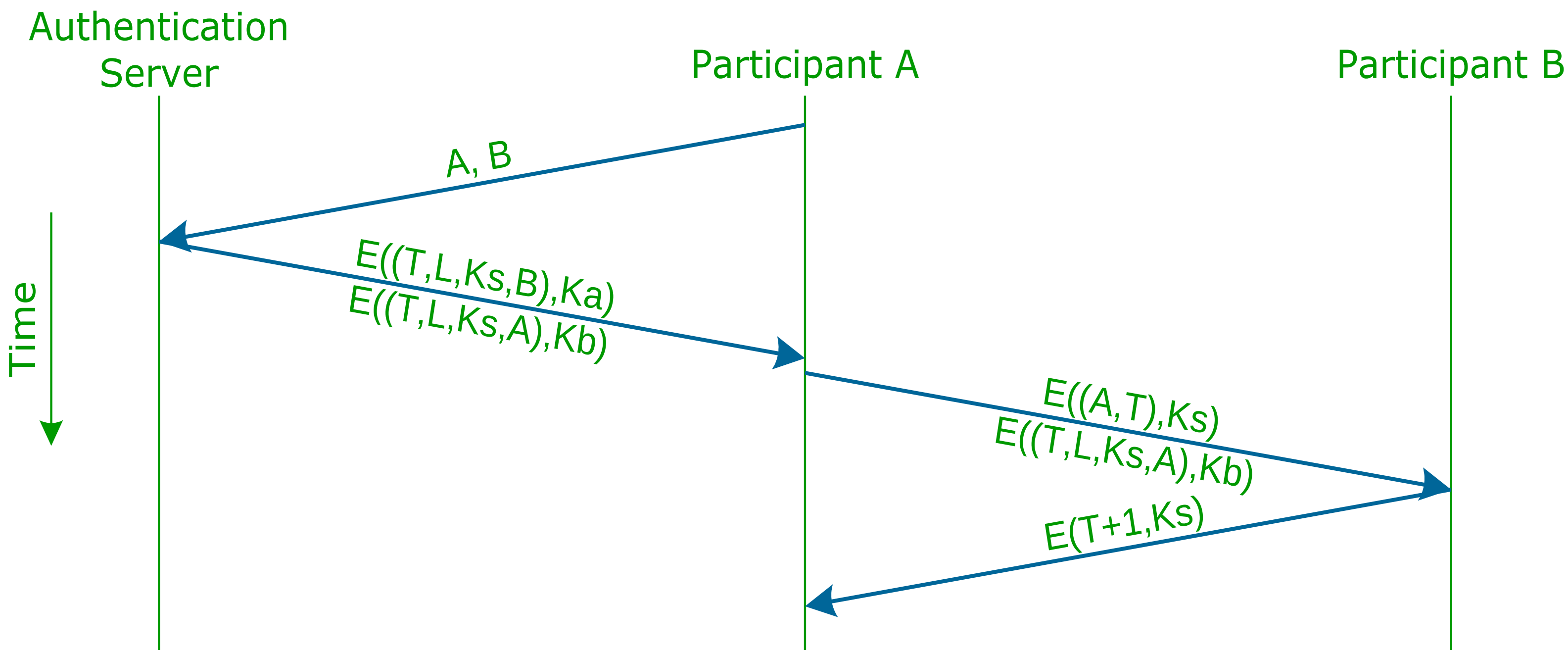


Peterson and Davie, Figure 8.9

P&D: 8.2.1

Authentication: Trusted 3rd Party

- Initiate by sending identifiers to authentication server

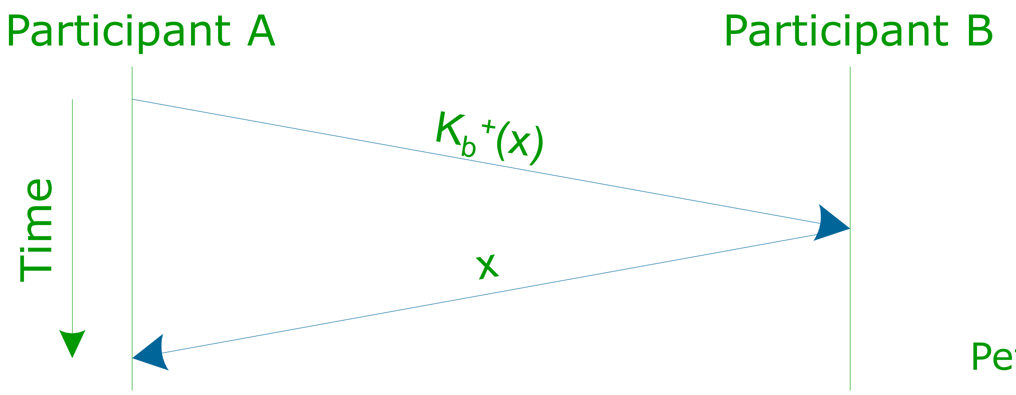


Key: E(a,b) is data a encrypted with key b; T is timestamp; L is lifetime

Peterson and Davie, Figure 8.10

P&D: 8.2.1

Authentication: Public Key

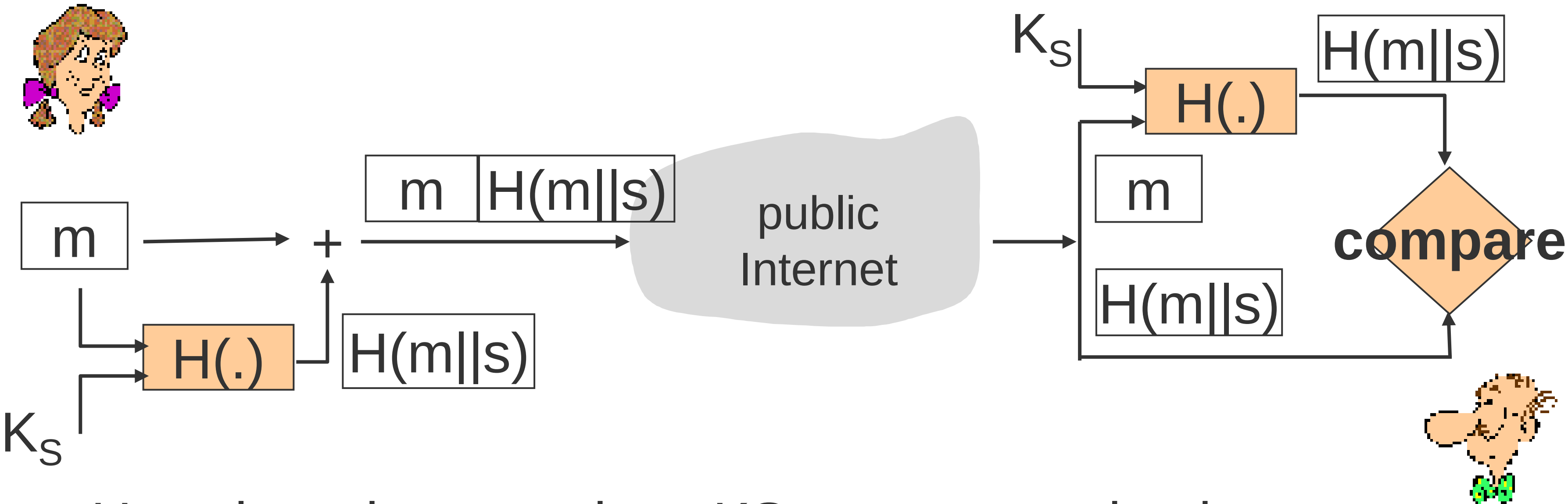


Peterson and Davie, Figure 8.11

- A encrypts random number, *x*, using B's public key
- B proves knows corresponding private key by:
  - decrypting *x* and returning it to A
- Only authenticates B to A, reverse process for A to B

P&D: 8.2.2

Message Integrity – Keyed Hash



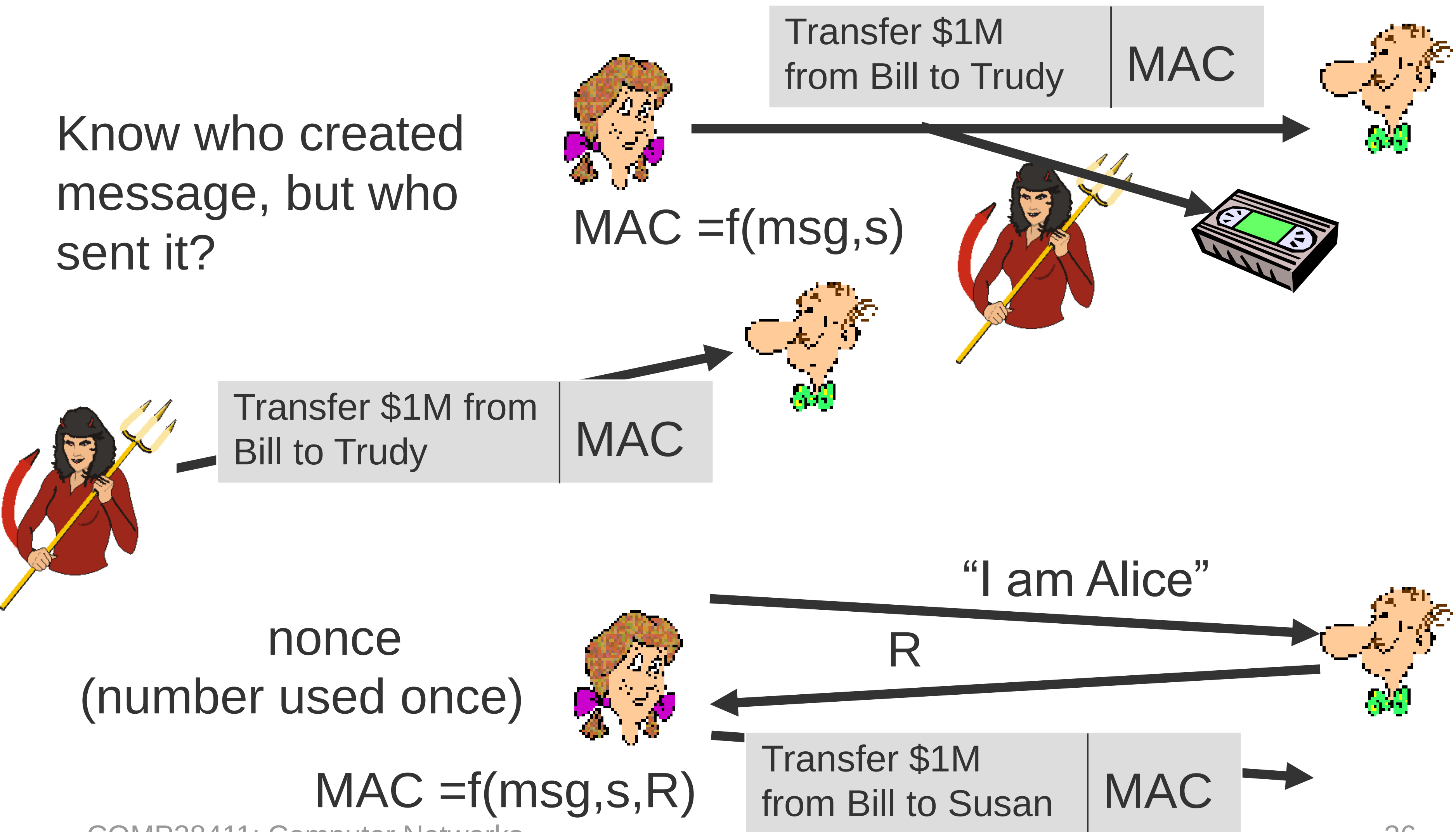
- Use shared secrete key, KS, to encrypt checksum
- Checksum = Message Authentication Code (MAC)
- Example: HMAC

As key only known to Alice and Bob, only Alice or Bob can have sent message (end-point authentication)

Playback Attack and Defence

Know who created message, but who sent it?

nonce
(number used once)



Message Integrity: Signature

- Message (encrypted) with Alice's private key
  - only Alice can have sent (non-repudiation)

Alice's message, m

Dear Bob

Oh, how I have missed you. I think of you all the time! …(blah blah blah)

Alice

K_A^- Alice's private key

public key encryption algorithm

Costly

$K_B^-(m)$

Alice's message, m, encrypted with Her private key

- Anyone can decrypt/verify sender

Note: $m = K^-_B(K^+_B(m)) = K^+_B(K^-_B(m))$

Message Integrity – Digital Signatures