CSC4200/5200 – COMPUTER NETWORKING
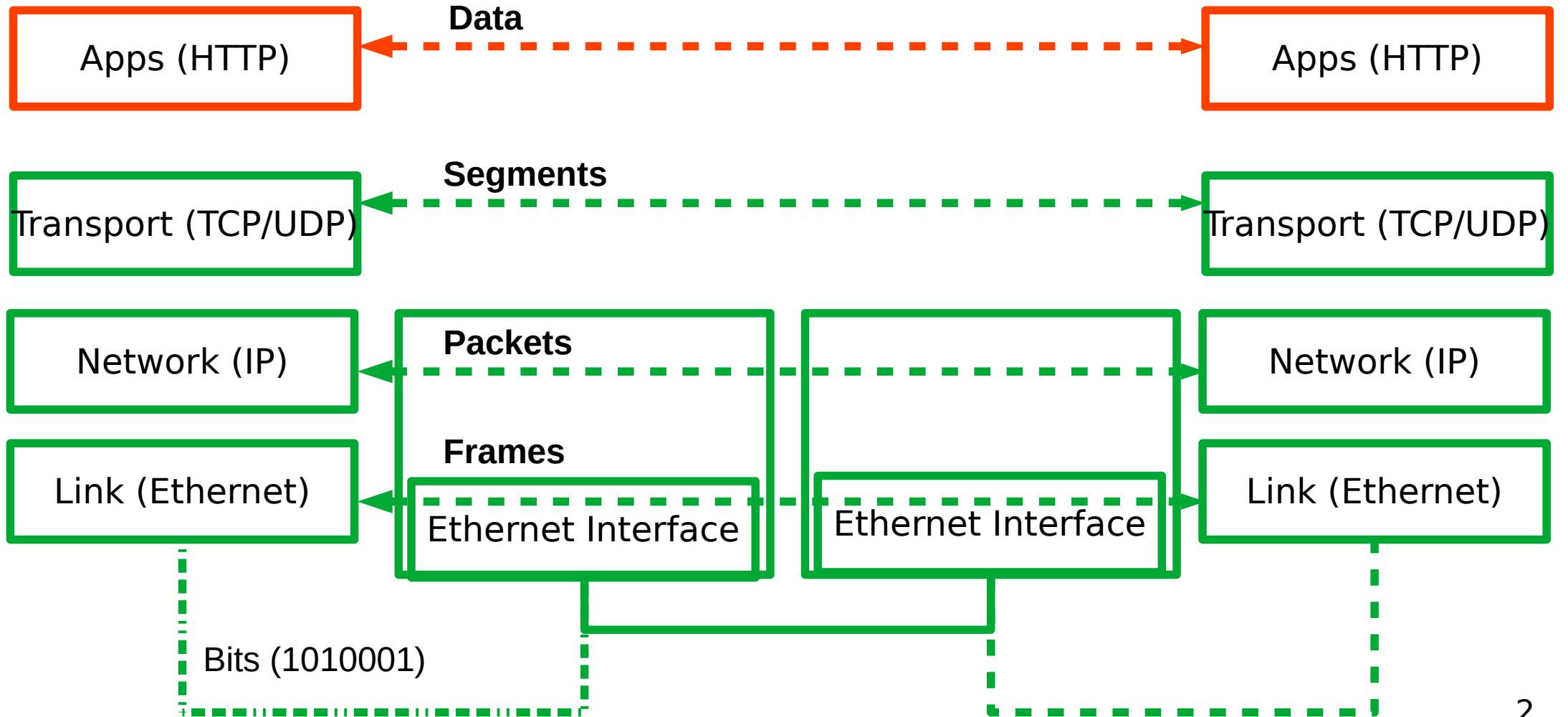
Instructor: Susmit Shannigrahi

NETWORK SECURITY
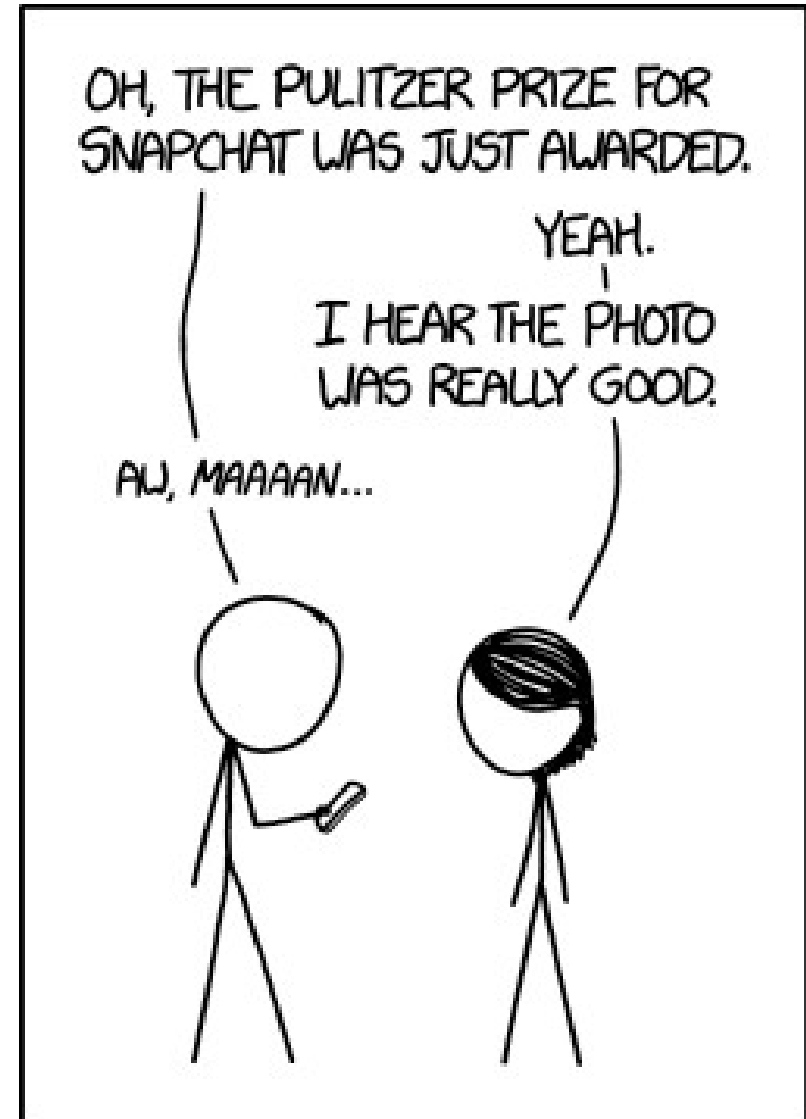
sshannigrahi@tntech.edu

GTA: dereddick42@students.tntech.edu

Tennessee
TECH

Apps (HTTP) ← Data → Apps (HTTP)

Transport (TCP/UDP) ← Segments → Transport (TCP/UDP)

Network (IP) **Packets** Network (IP)

Link (Ethernet) **Frames** Link (Ethernet)

Ethernet Interface   Ethernet Interface

Bits (1010001)

2

How do you ~~send~~ secure the cat picture?

Network Security

Goals

- understand principles of network security:
  - cryptography and its *many* uses beyond "confidentiality"
  - authentication
  - message integrity
- security in practice:
  - firewalls and intrusion detection systems
  - security in application, transport, network, link layers

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 and Alice want to communicate "securely"
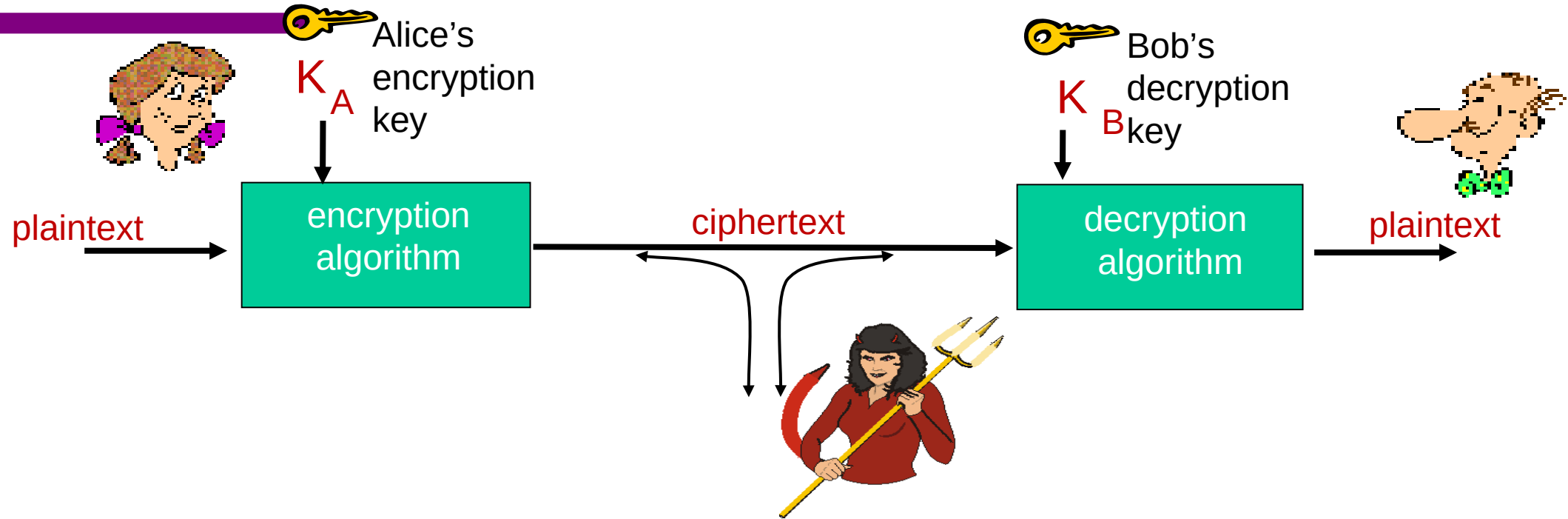- Trudy may intercept, delete, add messages
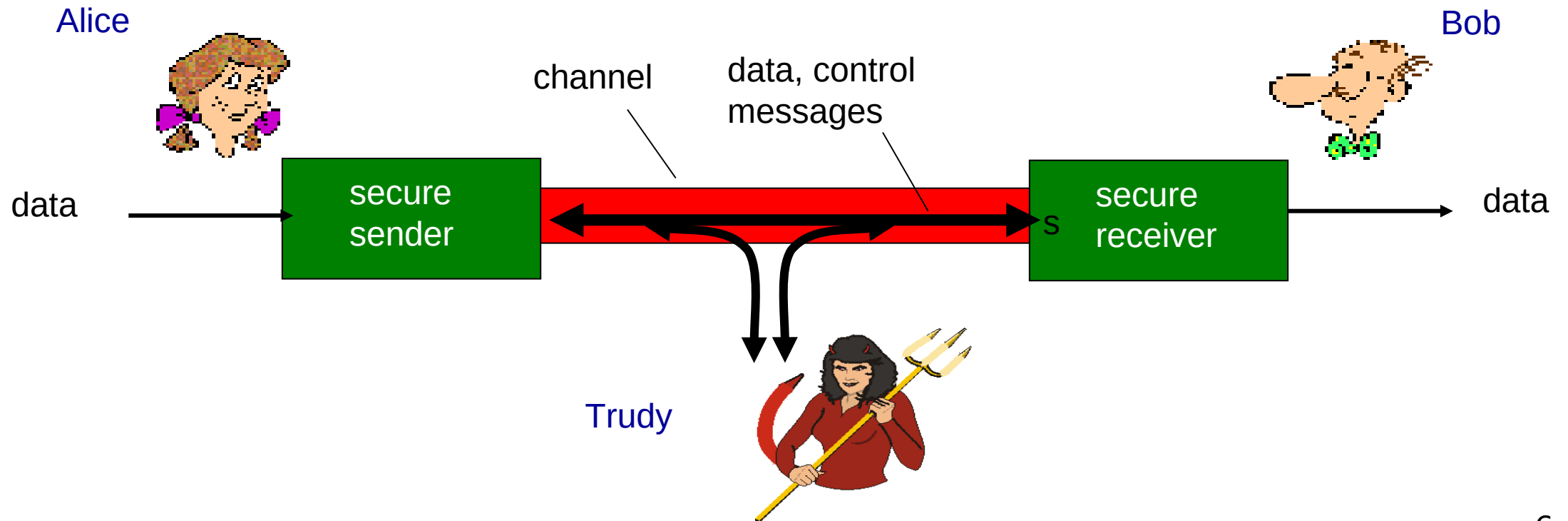
Where do we need security?

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

Some example problems

- 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)

The Principle of cryptography

plaintext → encryption algorithm → ciphertext → decryption algorithm → plaintext

K_A Alice's encryption key

K_B Bob's decryption key
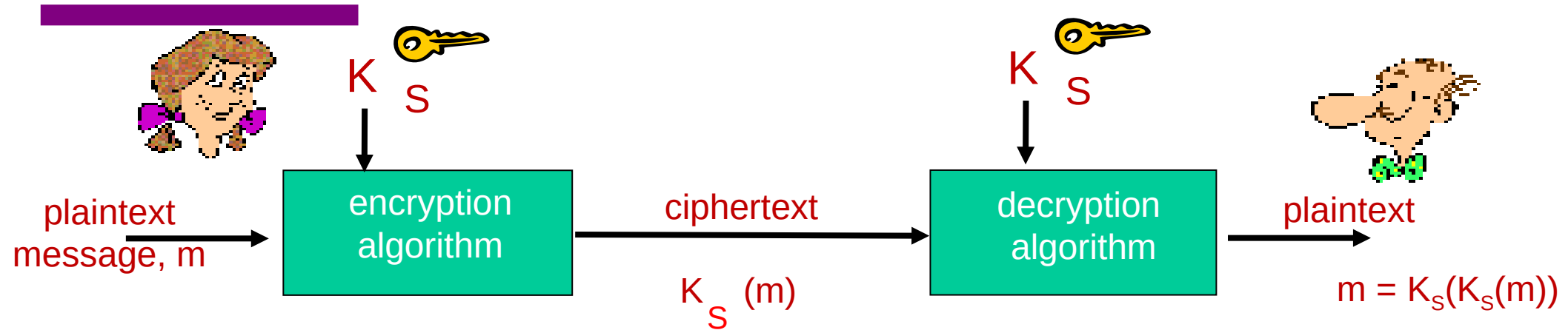
m plaintext message

$K_A(m)$ 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: someone** has plaintext corresponding to ciphertext
  - Enigma machine
  - Weather and Hilter in same position in every message

- **chosen-plaintext attack: someone** can get ciphertext for chosen plaintext
  - The battle of Midway
  - Planning to attack AF
  - AF has water supply problem
  - Repeat – AF has water supply problems

    Source: https://www.history.navy.mil/content/dam/museums/nmas/education/Codes%20and%20Ciphers%20Activity.pdf

Symmetric key cryptography

plaintext message, m → **encryption algorithm** → ciphertext $K_S(m)$ → **decryption algorithm** → plaintext $m = K_S(K_S(m))$

K_S (keys shown above encryption and decryption algorithms)

symmetric key crypto: Bob and Alice share same (symmetric) key: K
e.g., key is knowing substitution pattern in mono alphabetic substitution cipher – caesar cypher
Q: how do Bob and Alice agree on key value?

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 subsitution 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:
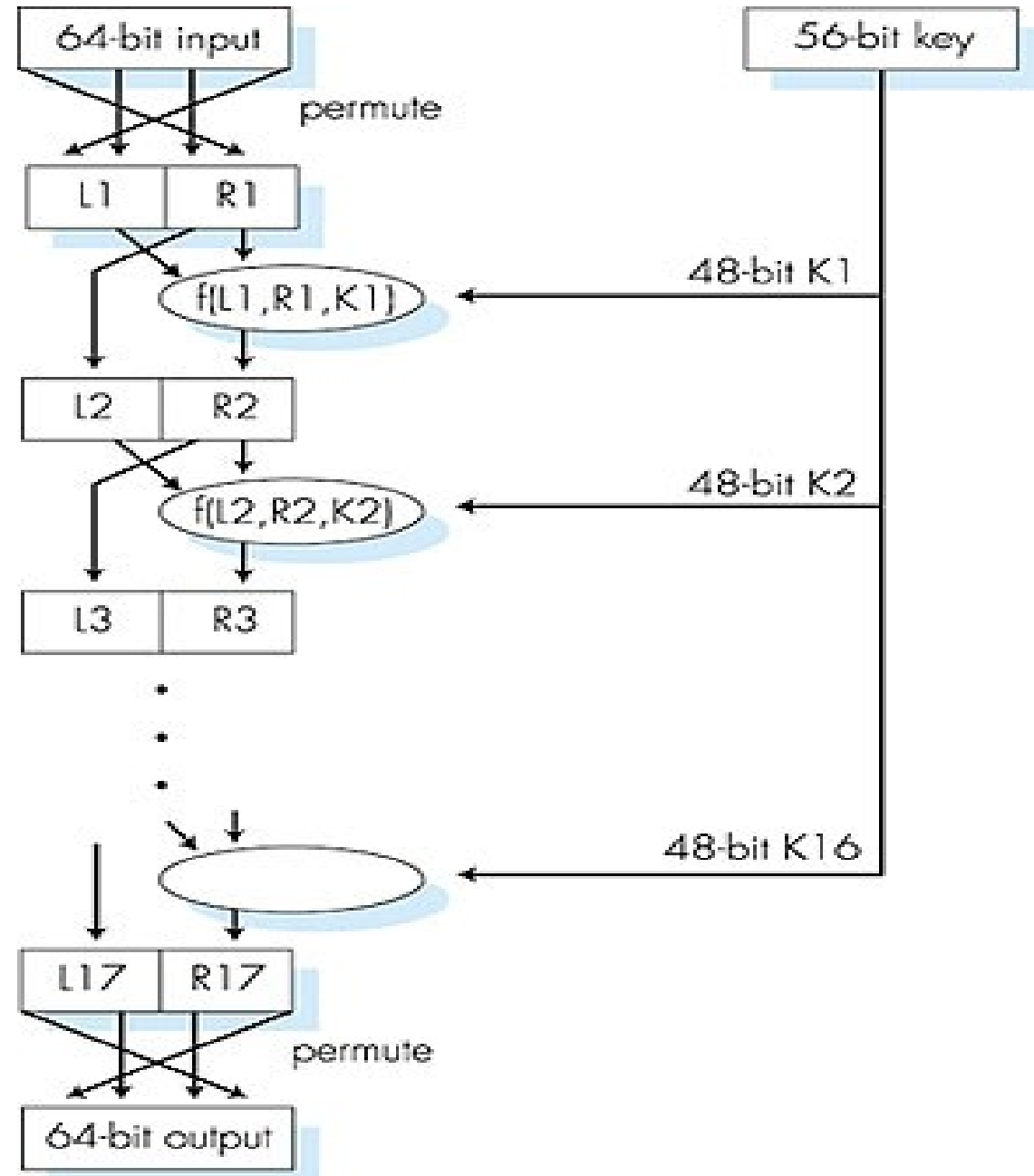 - 3DES: encrypt 3 times with 3 different keys

Symmetric key crypto: DES

DES operation

initial permutation

16 identical "rounds" of function application, each using different 48 bits of key

final permutation

How secure is DES - DES Challenges

- The first challenge began in 1997 and was solved in 96 days

- DES Challenge II-1 in 39 days in early 1998.
  - "Many hands make light work."

- DES Challenge II-2 - 56 hours in July 1998,
- "It's time for those 128-, 192-, and 256-bit keys."

- DES Challenge III
  - 22 hours 15 minutes in January 1999,
  - "See you in Rome (second AES Conference, March 22-23, 1999)".

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
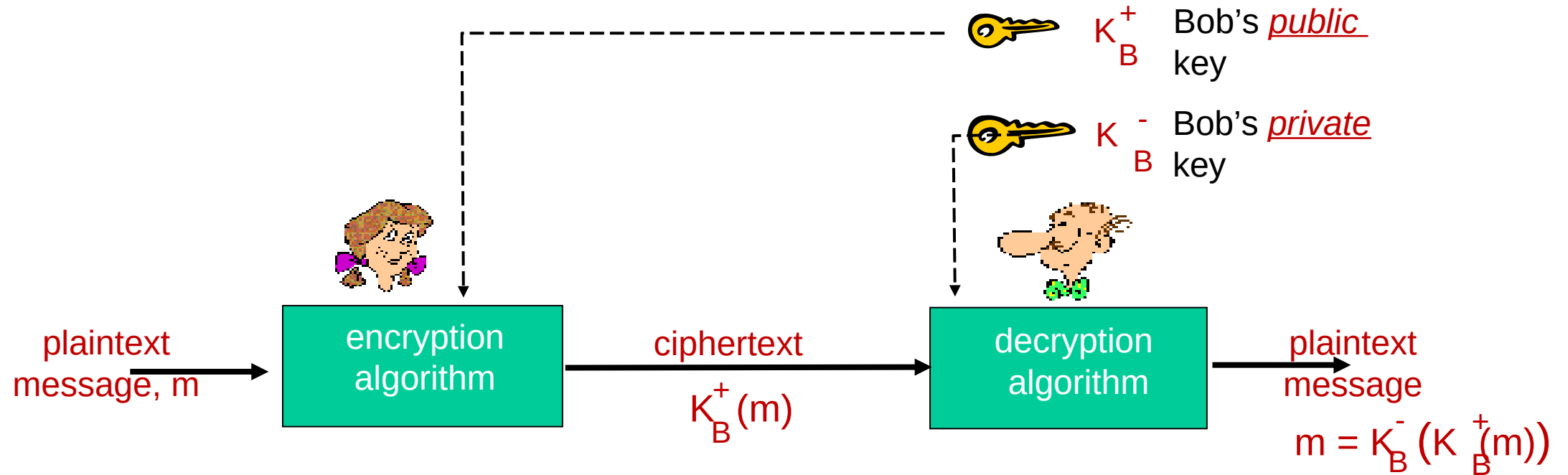
Public Key Cryptography
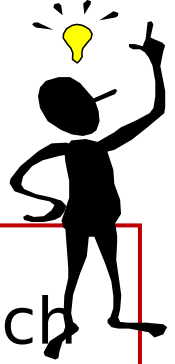
symmetric key crypto

- 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

K_B^+ Bob's *public* key

K_B^- Bob's *private* key

plaintext message, m → **encryption algorithm** → ciphertext $K_B^+(m)$ → **decryption algorithm** → plaintext message

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

Public key encryption algorithms

requirements:

$①$ need $K_B^+( )$ and $K_B^-( )$ 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 mod n = remainder of x when divide by n

❖ facts:

[(a mod n) + (b mod n)] mod n = (a+b) mod n

[(a mod n) - (b mod n)] mod n = (a-b) mod n

[(a mod n) * (b mod n)] mod n = (a*b) mod n

❖ thus

$(a \bmod n)^d \bmod n = a^d \bmod n$

❖ example: x=14, n=10, d=2:

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

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

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 mod $z = 1$).

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

$$K_B^+ \qquad K_B^-$$

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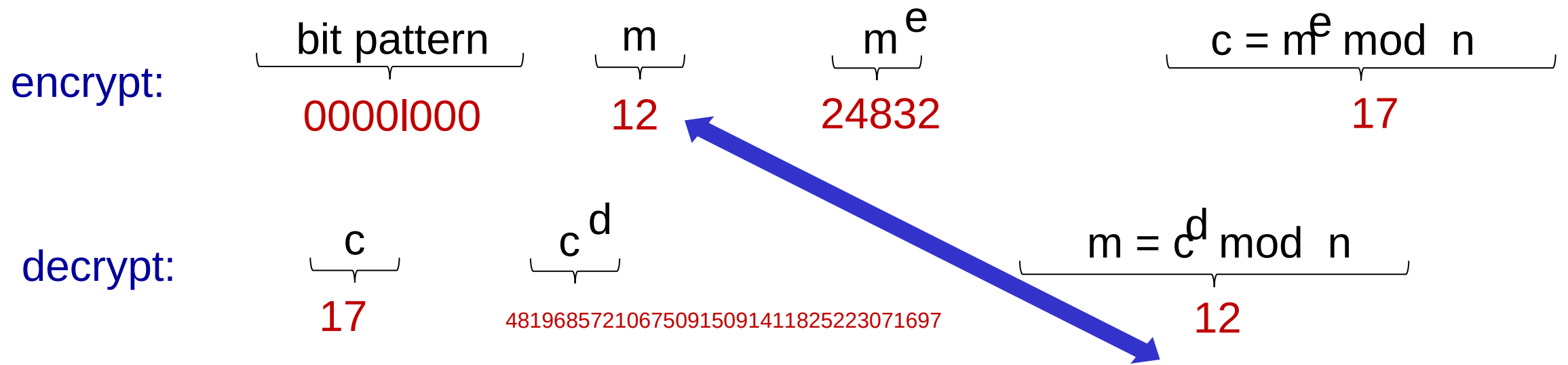$$m = c^d \bmod n$$

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

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.

encrypt:

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

decrypt:

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

25

RSA: another important property

The following property will be *very* useful later:

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

use public key first,
followed by private
key

use private key first,
followed by public
key

result is the same!

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

    https://listserv.nodak.edu/cgi-bin/wa.exe?
    A2=NMBRTHRY;dc42ccd1.2002
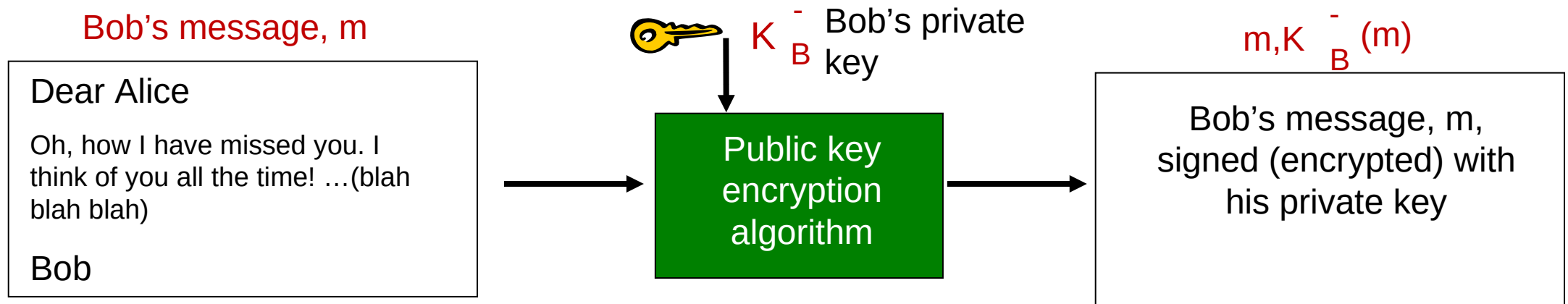
Digital signatures

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, $K_B^-(m)$

Bob's message, m

K_B^- Bob's private key

$m, K_B^-(m)$

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

→ Public key encryption algorithm →

Bob's message, m, signed (encrypted) with his private key

whoever signed m must have used Bob's private key.

Reading Material

https://book.systemsapproach.org/security.html
Read through 8.2.4