

---

# **Network Security**

# **What is network security?**

---

- Computer network security consists of measures taken by business or some organizations to monitor and prevent unauthorized access from the outside attackers.
- Different approaches to computer network security management have different requirements depending on the size of the computer network. For example, a home office requires basic network security while large businesses require high maintenance to prevent the network from malicious attacks.

# What is network security?

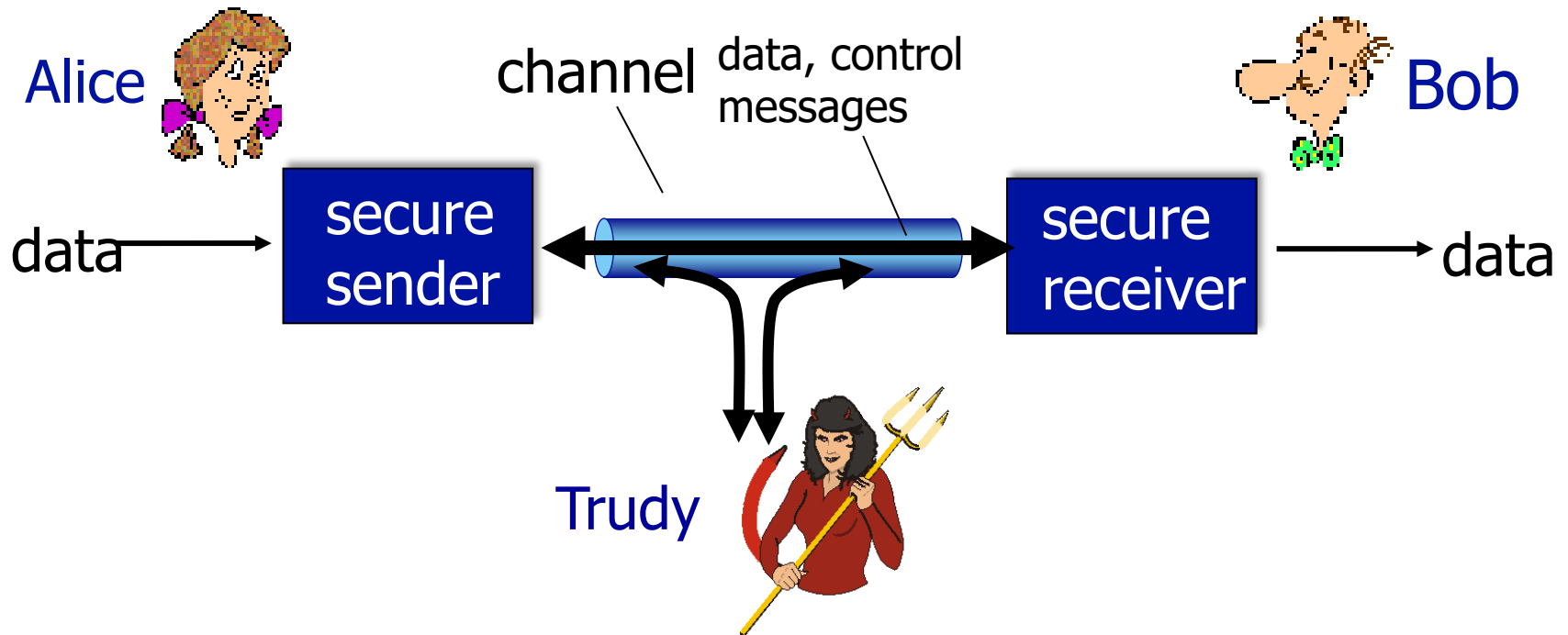
---

- **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

---

- well-known in network security world
- Bob, Alice (lovers!) want to communicate “securely”
- Trudy (intruder) may intercept, delete, add messages



# **Friends and enemies: Alice, Bob, Trudy**

---

Who might Bob and Alice be?

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

# There are bad guys (and girls) out there!

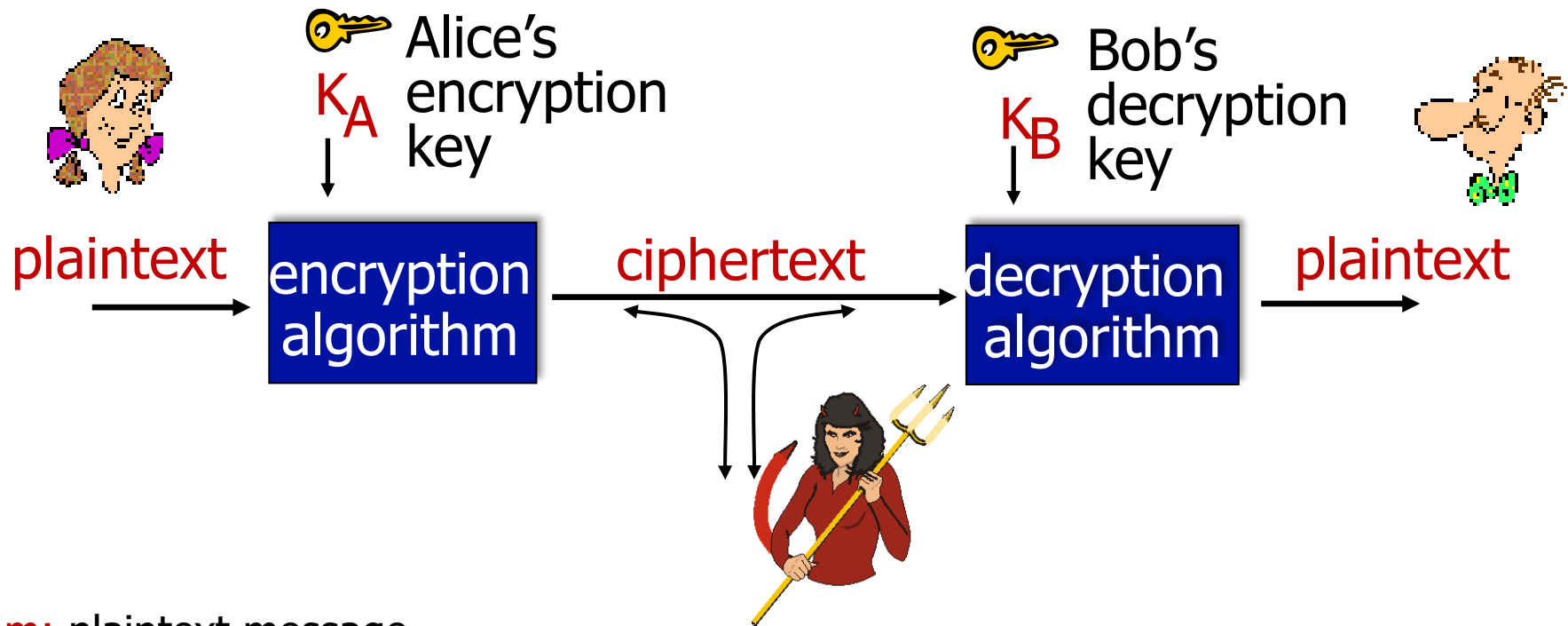
---

Q: What can a “bad guy” do?

A: A lot! (recall section 1.6)

- 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 language of cryptography



$m$ : plaintext message

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

$m = K_B(K_A(m))$

# Breaking an encryption scheme

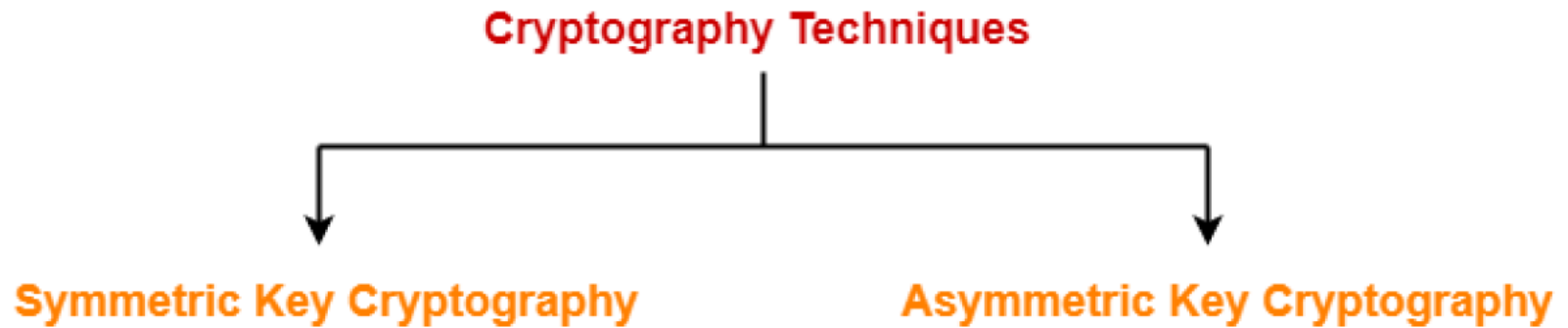
---

- **cipher-text only attack:** Trudy has ciphertext she can analyze
- **two approaches:**
  - brute force: search through all keys
  - statistical analysis
- **known-plaintext attack:** Trudy has plaintext corresponding to ciphertext
  - *e.g.*, in monoalphabetic cipher, Trudy determines pairings for a,l,i,c,e,b,o,
- **chosen-plaintext attack:** Trudy can get ciphertext for chosen plaintext

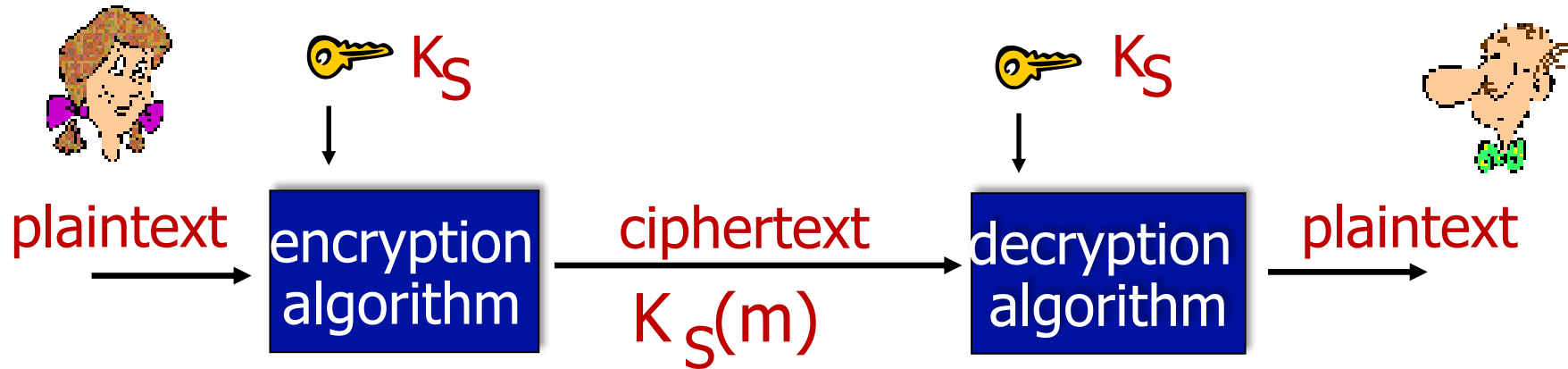


# **Cryptography Techniques**

---



# Symmetric key cryptography



**symmetric key crypto:** Bob and Alice share same (symmetric) key:  $K$

• *e.g.*, key is knowing substitution pattern in mono alphabetic substitution cipher

**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:	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
		↓																								↓
ciphertext:	m	n	b	v	c	x	z	a	s	d	f	g	h	j	k	l	p	o	i	u	y	t	r	e	w	q

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,  $M_1, M_2, \dots, M_n$   
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 substitution 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

# 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

# Advantage

---

- **Efficient:** The secret key algorithms are more efficient as it takes less time to encrypt the message than to encrypt the message by using a public key encryption algorithm. The reason for this is that the size of the key is small. Due to this reason, Secret Key Algorithms are mainly used for encryption and decryption.

# Disadvantages

---

- The number of keys required is very large.
  - In symmetric key cryptography,
    - Each pair of users require a unique secret key.
    - If  $N$  people in the world wants to use this technique, then there needs to be  $N(N-1) / 2$  secret keys.
    - For 1 million people to communicate, a half billion secret keys would be needed.
- Sharing the secret key between the sender and receiver is an important issue. While sharing the key, attackers might intrude.



# Public Key Cryptography/ Asymmetric

---

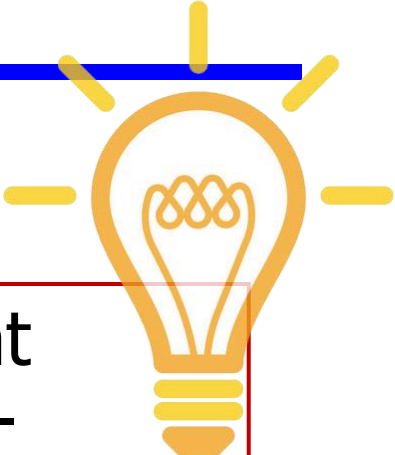
symmetric key

crypto:

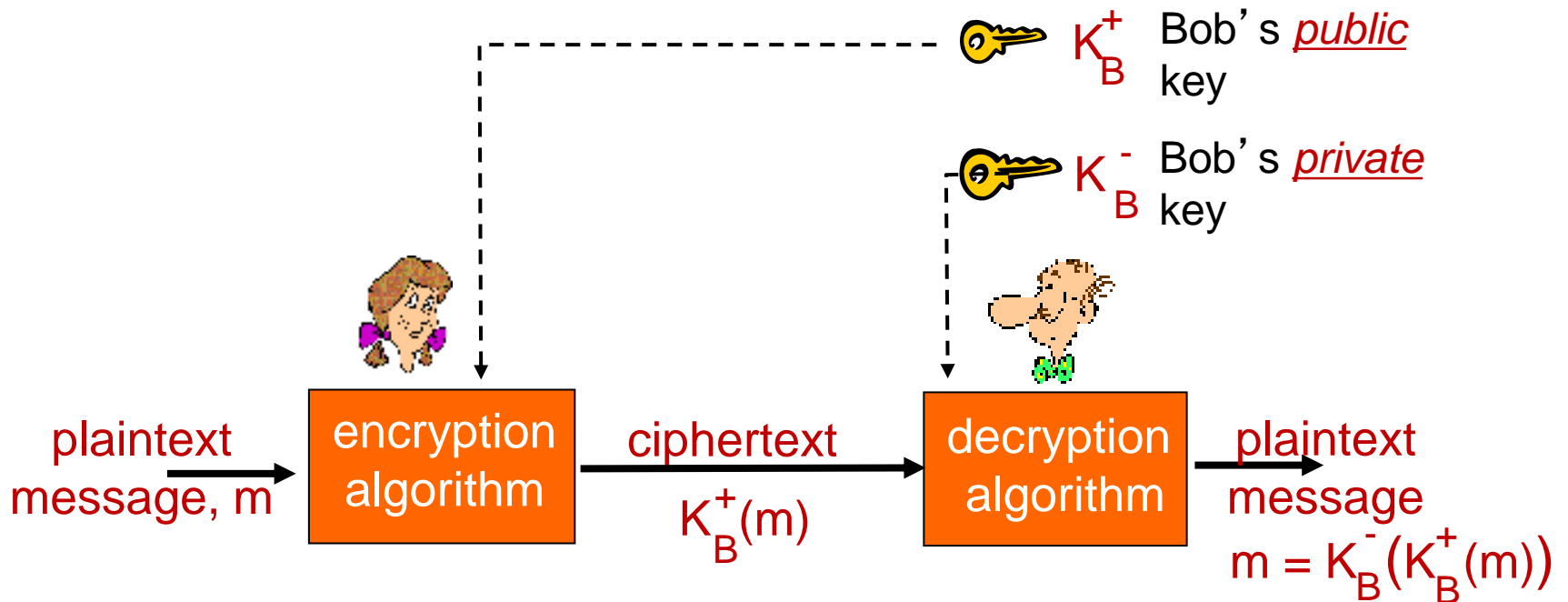
requires sender, receiver know  
shared secret key

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



**Wow** - public key cryptography revolutionized 2000-year-old (previously only symmetric key) cryptography!

- similar ideas emerged at roughly same time, independently in US and UK (classified)

# Public key encryption algorithms

---

requirements:

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

# Prerequisite: modular arithmetic

---

- $x \bmod n$  = remainder of  $x$  when divide by  $n$
- 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: 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:  $ed \bmod z = 1$  ).
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!*

$$m = \underbrace{(m^e \bmod n)}_c^d \bmod n$$

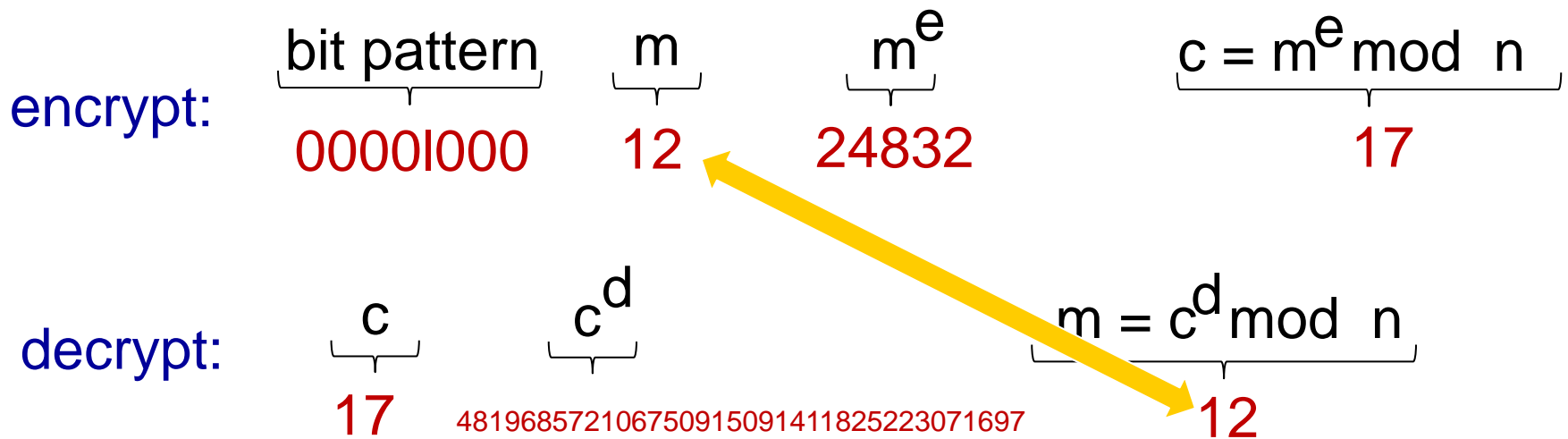
# RSA example:

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.





# Why does RSA work?

---

- 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,  
$$\begin{aligned} c^d \bmod n &= (m^e \bmod n)^d \bmod n \\ &= m^{ed} \bmod n \\ &= m^{(ed \bmod z)} \bmod n \\ &= m^1 \bmod n \\ &= m \end{aligned}$$

# **RSA: another important property**

The following property will be *very* useful later:

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

use public key first,  
followed by  
private key

use private key  
first, followed by  
public key

*result is the same!*

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

---

follows directly from modular arithmetic:

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

# 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

# **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 key  $K_S$
- once both have  $K_S$ , they use symmetric key cryptography

# Advantages

---

- The main restriction of private key encryption is the sharing of a secret key. A third party cannot use this key. In public key encryption, each entity creates a pair of keys, and they keep the private one and distribute the public key.
- The number of keys in public key encryption is reduced tremendously. For example, for one million users to communicate, only two million keys are required, not a half-billion keys as in the case of secret key encryption.

# Disadvantages

---

- **Speed:** One of the major disadvantage of the public-key encryption is that it is slower than secret-key encryption.
- **Authentication:** A public key encryption does not have a built-in authentication. Without authentication, the message can be interpreted or intercepted without the user's knowledge.
- **Inefficient:** The main disadvantage of the public key is its complexity.