CY 2550 Foundations of Cybersecurity

Cryptography Part 4

January 30

Alina Oprea

Associate Professor, Khoury College

Northeastern University

Outline

- IND-CPA secure encryption
- Public-key crypto
- Key exchange and the Diffie-Hellman protocol
- RSA public-key encryption
- Hash functions

- Announcement
  - CIO of Children’s Hospital in Boston – Dan Nigrin – will be on campus to give a talk on **Feb 5 from 11:45-12:45 in 655 ISEC**

IND-EAV / IND-CPA security

- In CPA Adv can encrypt messages of its choice

Charlie

k, Enc_k

Adv

Query: Encrypt m

Reply: Ciphertext c

Round 1: Charlie chooses k and encryption algo

Round 2: Adv can encrypt messages

Round 3: Adv chooses two plaintext messages $m_0, m_1 \in \boldsymbol{M}$

Round 4: Charlie chooses a random binary number $b \leftarrow_R \{0, 1\}$

Round 5: Charlie encrypts the corresponding message $c = Enc_k(m_b)$

Round 6: Adv can encrypt messages

Query: Encrypt m

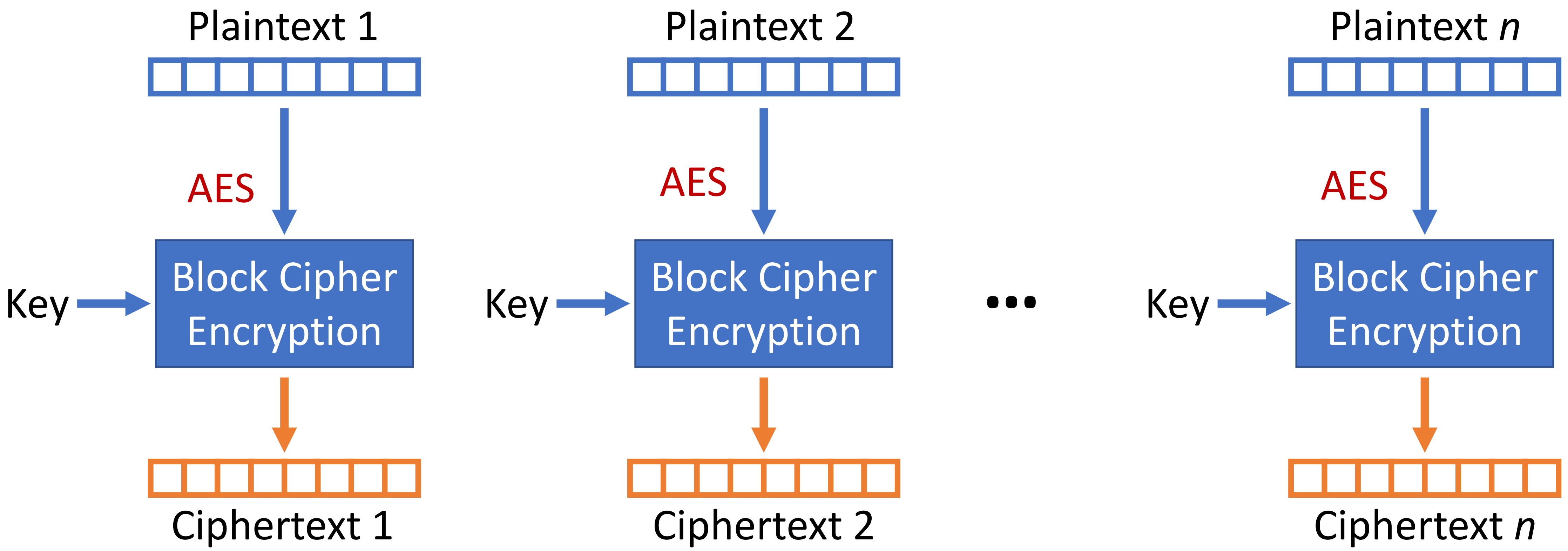
Reply: Ciphertext c

Round 7: Adv guesses the value of *b* $b' \in \{0, 1\}$

Adversary wins if $b = b'$

ECB Encryption Mode

- Message is broken into independent blocks
- Electronic Code Book (ECB): each block is encrypted separately



Cryptanalysis of ECB Mode

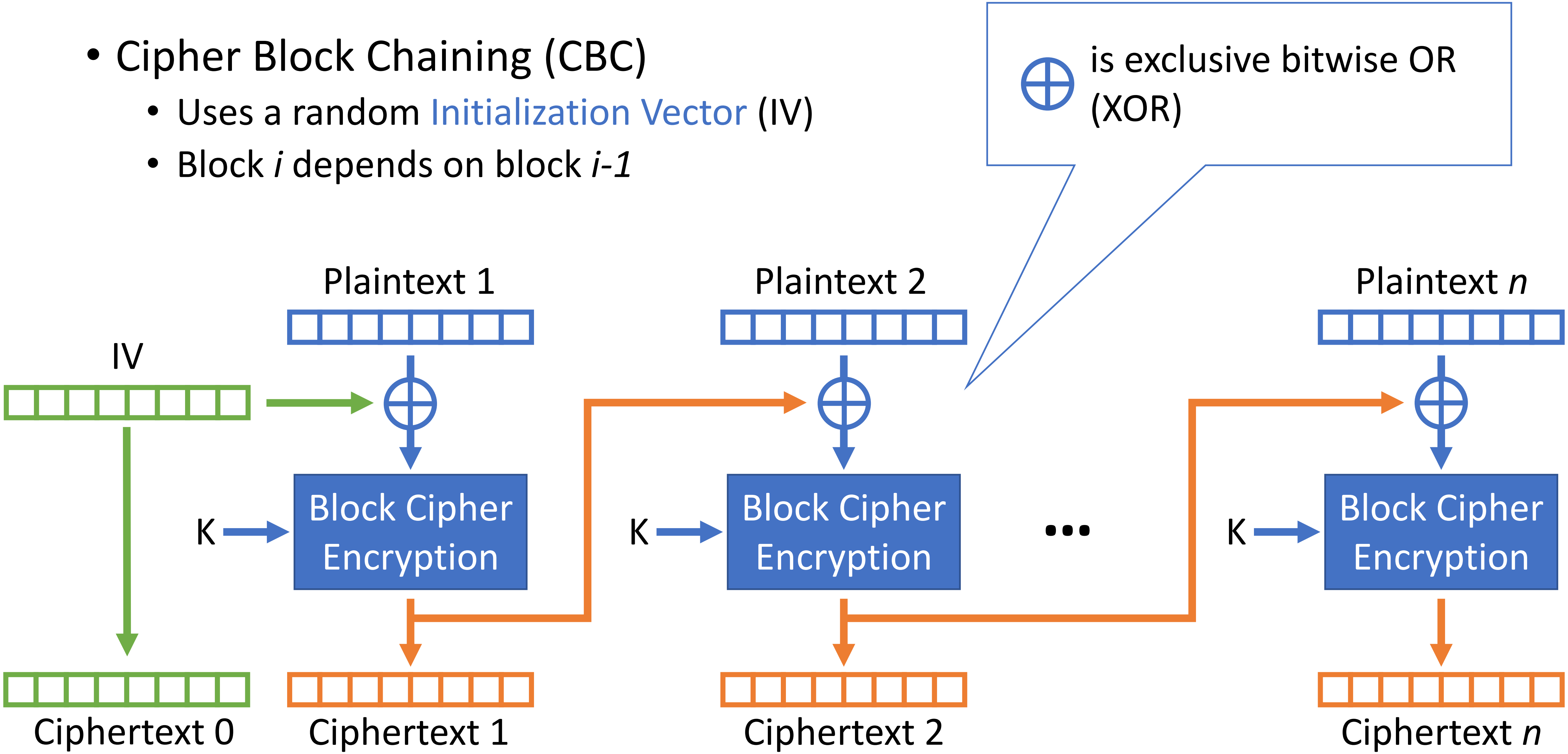
- Deterministic
  - The same data block always gets encrypted the same way
    - Reveals patterns when data repeats!
  - *m* encrypted with *k* always produces the same *c*
  - This is the same problem we had with the Vigenère cipher
- Is the ECB mode IND-CPA secure?
- Is the ECB mode IND-EAV secure?
- **Do not use ECB mode in practice**

Lessons on IND-CPA Security

- ECB uses deterministic encryption
  - Encryption of a message m is always the same
  - Adv can trivially win the IND-CPA game
- Deterministic encryption is not IND-CPA secure!
- CPA secure encryption needs to be randomized!
  - How is that achieved?

CBC Encryption Mode

- Cipher Block Chaining (CBC)
  - Uses a random Initialization Vector (IV)
  - Block *i* depends on block *i-1*



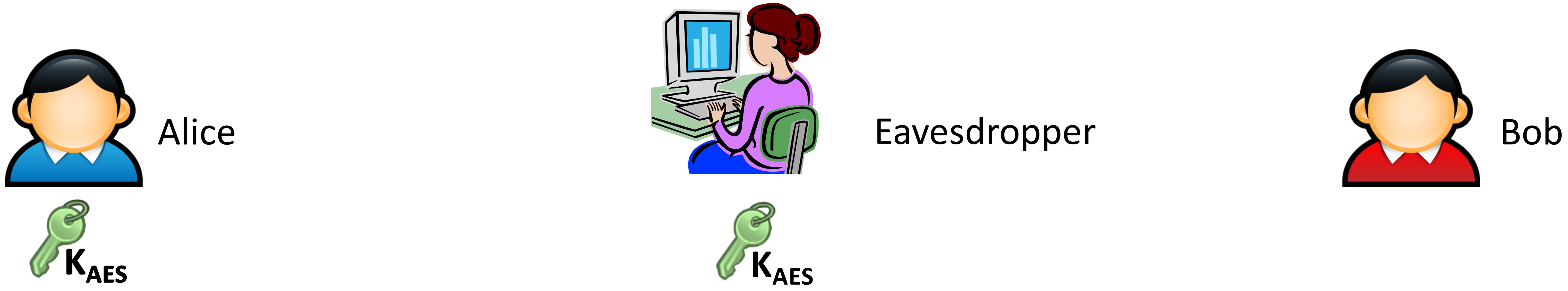
Cryptanalysis of CBC Mode

- CBC randomizes the encryption
  - IV ensures initial block is randomized
  - Dependency between blocks propagates randomness
- CBC is IND-CPA secure assuming
  - Block cipher itself is secure (pseudorandom permutation)
  - IV is truly random
  - IV is sufficiently large
  - Use the key for limited number of encryptions (key needs to be changed afterwards)
- Usage in practice: choose random IV and protect its integrity
  - The IV is not secret (it becomes part of the ciphertext)
  - Do not let the adversary control the IV (needs to be unpredictable)!

Public Key Cryptography

Weakness of Symmetric Key Crypto

- How do you securely exchange keys with someone?
- Easy(ish) to do if you can meet them in person
- However, the Internet is untrusted
  - You can't exchange shared secrets over an untrusted medium



Public Key Cryptography

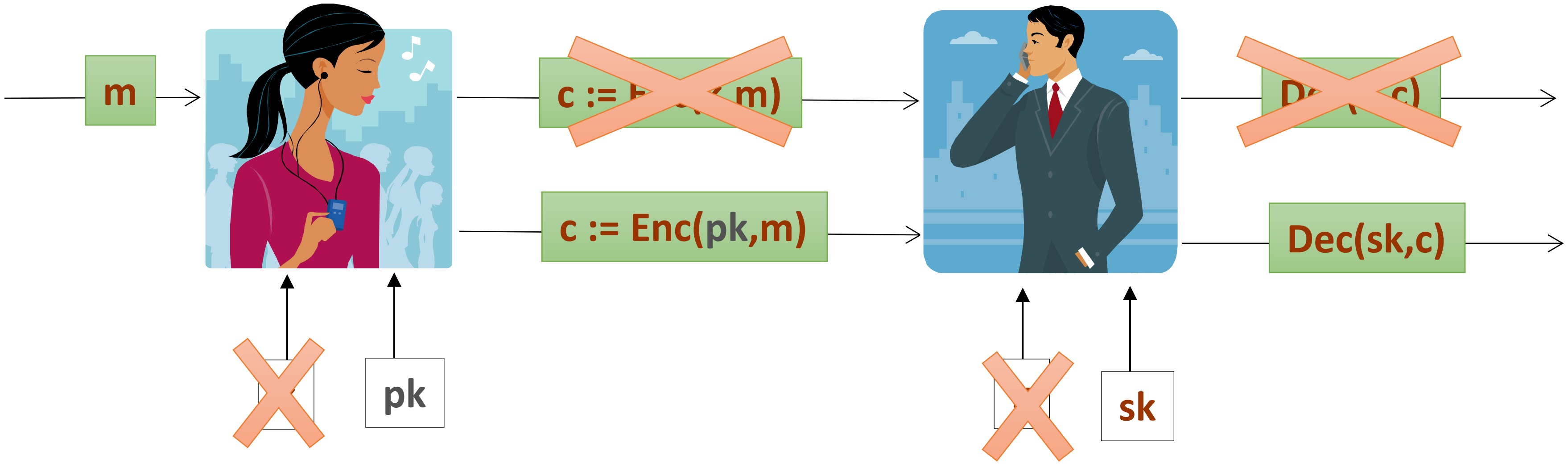
- Public key cryptography, a.k.a. asymmetric cryptography
  - Each principal has two keys: private (secret) and public
  - A message encrypted with one key must be decrypted by the other
  - Thus, the public key can be sent in-the-clear over the Internet
- Security is based on Very Hard Math Problems
  - Fast to verify a given solution for a given instance
  - Hard to finds solutions for a given instance in polynomial time
- Many different algorithms that offer different security properties
  - Diffie-Hellman, RSA, Goldwasser-Micali, ElGamal
- Forms the basis for most modern secure protocols
  - IPsec, SSL, TLS, S/MIME, PGP/GPG, etc.

Public Key Encryption

Instead of using one key **k**,
use **2** keys **(pk,sk)**, where
pk is used for **encryption**,
sk is used for **decryption**.

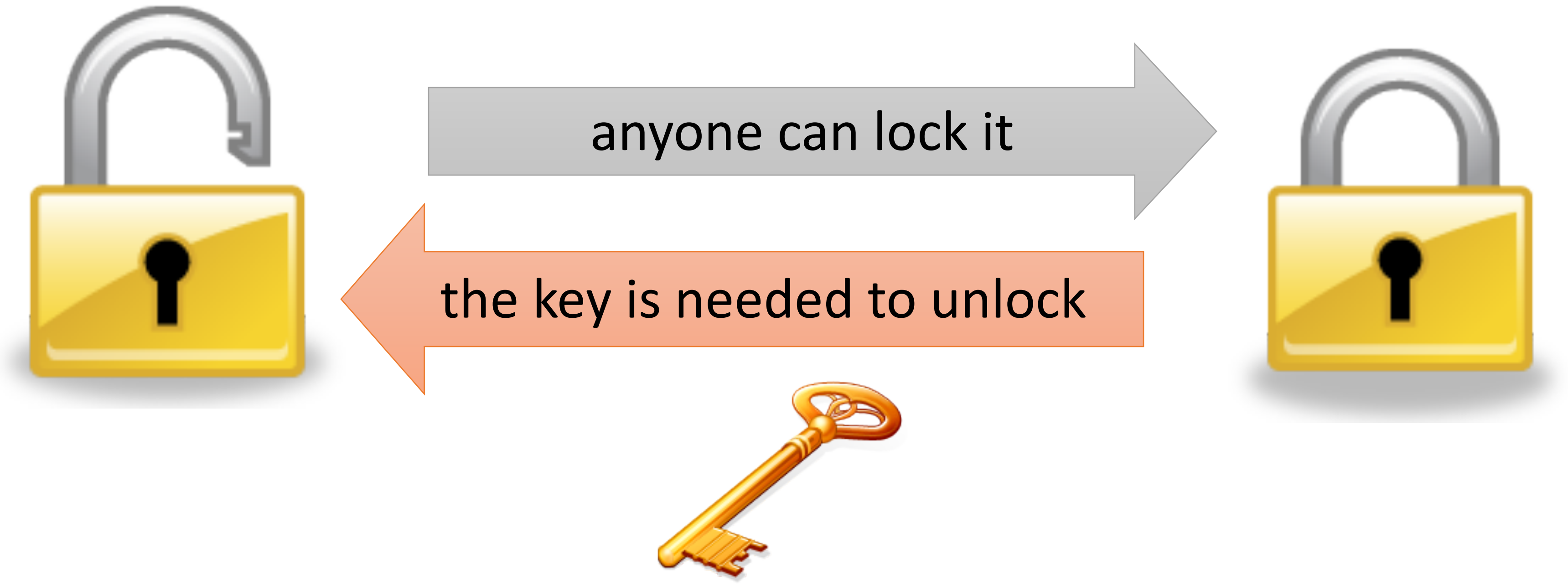
pk can be public, and only **sk** has to be kept secret!

That's why it's called:
public-key cryptography



Analogy

Examples padlocks:



Public-Key Cryptography

- Public-Key Encryption
  - Examples: RSA, ElGamal
- Digital Signatures:
  - Authenticate messages
  - Examples: RSA, DSA
- Key Exchange
  - Protocols to establish a secret key between two parties
  - Examples: key exchange
- Intuition for all these
  - Computation in one direction is “easy”, but “hard” in the reverse
  - Hardness assumptions imply that adversary cannot reverse computation

A little bit of history

- **Diffie and Hellman** were the first to publish a paper containing the idea of the public-key cryptography:

  W.Diffie and M.E.Hellman,
  New directions in cryptography
  IEEE Trans. Inform. Theory, IT-22, 6, **1976**, pp.644-654.

- A similar idea was described by **Ralph Merkle**:
  - in **1974** he described it in a project proposal for a Computer Security course at UC Berkeley
    (it was rejected)
  - in **1975** he submitted it to the CACM journal (it was rejected)

  (see http://www.merkle.com/1974/)

- 1977: R. Rivest, A. Shamir and L. Adelman published the first construction of public-key encryption (RSA)

- It 1997 the GCHQ (the British equivalent of the NSA) revealed that they knew it already in **1973**.

Diffie-Hellman Key Exchange

- Goal
  - Share a secret key over a public channel in presence of eavesdropping adversary
- Really should be called Diffie-Hellman-Merkle
  - Ralph Merkle developed the mathematical theories
  - Whitfield Diffie and Martin Hellman developed the protocol
- Security is based on the discrete logarithm problem
  - Compute *k* such that $b^k = g$ mod p, where *b*, *g*, and *k* are all integers and p is a large prime
  - Possible that no solution exists given arbitrary *b* and *g*
  - Best known algorithms are exponential time

Diffie-Hellman Protocol

- Red = secret, blue = public

1. Alice chooses a large prime p and a base g in {1, …, p}
2. Alice chooses a secret integer a;
3. Alice → Bob: p, g, $A = g^a \bmod p$;
4. Bob chooses secret b
5. Bob → Alice: $\mathrm{B} = g^b \bmod p$
6. Alice computes $s = B^a \bmod p$; Bob computes $s = A^b \bmod p$
7. Alice and Bob now share secret key s

The Diffie-Hellman protocol

Fix a large prime p (e.g. 600 digits)

Fix an integer g in {1, …, p}

Alice **Bob**

choose random **a** in {1,…,p-1}

choose random **b** in {1,…,p-1}

$p, g, A \leftarrow g^a \; mod \; p$ →

← $B \leftarrow g^b \; mod \; p$

B^a (mod p) = $(g^b)^a$ = $k_{AB} = g^{ab}$ (mod p) = $(g^a)^b$ = A^b (mod p)

Diffie-Hellman Example

Alice

Knows	Doesn’t Know
p = 23, g = 5	
a = 6	b = ?
$A = g^{a} \bmod p$ $A = 5^{6} \bmod 23$ $= 8$	
B = 19	
$s = B^{a} \bmod p$ $= 19^{6} \bmod 23$ $= 2$	

Eavesdropper

Knows	Doesn’t Know
p = 23, g = 5	
	a = ?, b = ?
A = 8, B = 19	

Bob

Knows	Doesn’t Know
p = 23, g = 5	
b = 15	a = ?
$B = g^{b} \bmod p$ $= 5^{15} \bmod 23$ $=19$	
$A = 8$	
$s = A^{b} \bmod p$ $= 8^{15} \bmod 23$ $= 2$	

Calculating *s* requires solving for *a* or *b*, which is the discrete logarithm problem

Man-in-the-middle Attacks

As described, the protocol is insecure against **active** attacks

Alice | **MiTM** | **Bob**

$A \leftarrow g^{a} \bmod p$ → a′ → $A' \leftarrow g^{a'} \bmod p$ →

← $B' \leftarrow g^{b'} \bmod p$ ← b′ ← $B \leftarrow g^{b} \bmod p$

$g^{ab'}$ ⟷ $g^{ab'}$ $g^{a'b}$ ⟷ $g^{a'b}$

Attacker relays traffic from Alice to Bob and reads it in clear

Public-key Encryption

- Encryption algorithm: Enc(pk, m); decryption Dec(sk, c)
- RSA algorithm invented by Rivest, Shamir, and Adleman in 1978
  - Equivalent system invented by Clifford Cox in 1973, but GCHQ classified it
- RSA is the dominant public key cryptosystem today
  - Algorithm was commercialized by RSA Security
  - RSA Security created a certificate authority that eventually became Verisign

RSA Algorithm

- Security is based on the difficulty of factoring the product of primes
  - Alice chooses two secret primes p and q, $n = pq$, $\phi(n) = (p-1)(q-1)$
  - Choose e such that $1 < e < \phi(n)$, and $gcd(e, \phi(n)) = 1$
  - <n, e> is Alice's public key
  - Private key $d = e^{-1} \bmod \phi(n)$; $d \cdot e = 1 \bmod \phi(n)$
- Encryption and decryption
  - Given a message M, $0 < M < n$
  - Compute ciphertext $C = M^e \bmod n$
  - To decipher, compute $C^d \bmod n = (M^e \bmod n)^d \bmod n = M^{ed} \bmod n = M$
  - Use Euler's theorem: $x^{\phi(n)} = 1 \bmod n$

RSA Example

$p = 11$, $q = 7$, $n = pq = 77$, $\phi(n) = 60$

$e = 37$, $d = 13$ ($ed = 481$, ed mod $60 = 1$)

If $M = 15$ then $C = M^e$ mod $n = 15^{37}$ mod $77 = 71$

C^d mod $n = 71^{13}$ mod $77 = 15 = M$