CY 2550 Foundations of Cybersecurity

Cryptography Part 3

January 27

Alina Oprea

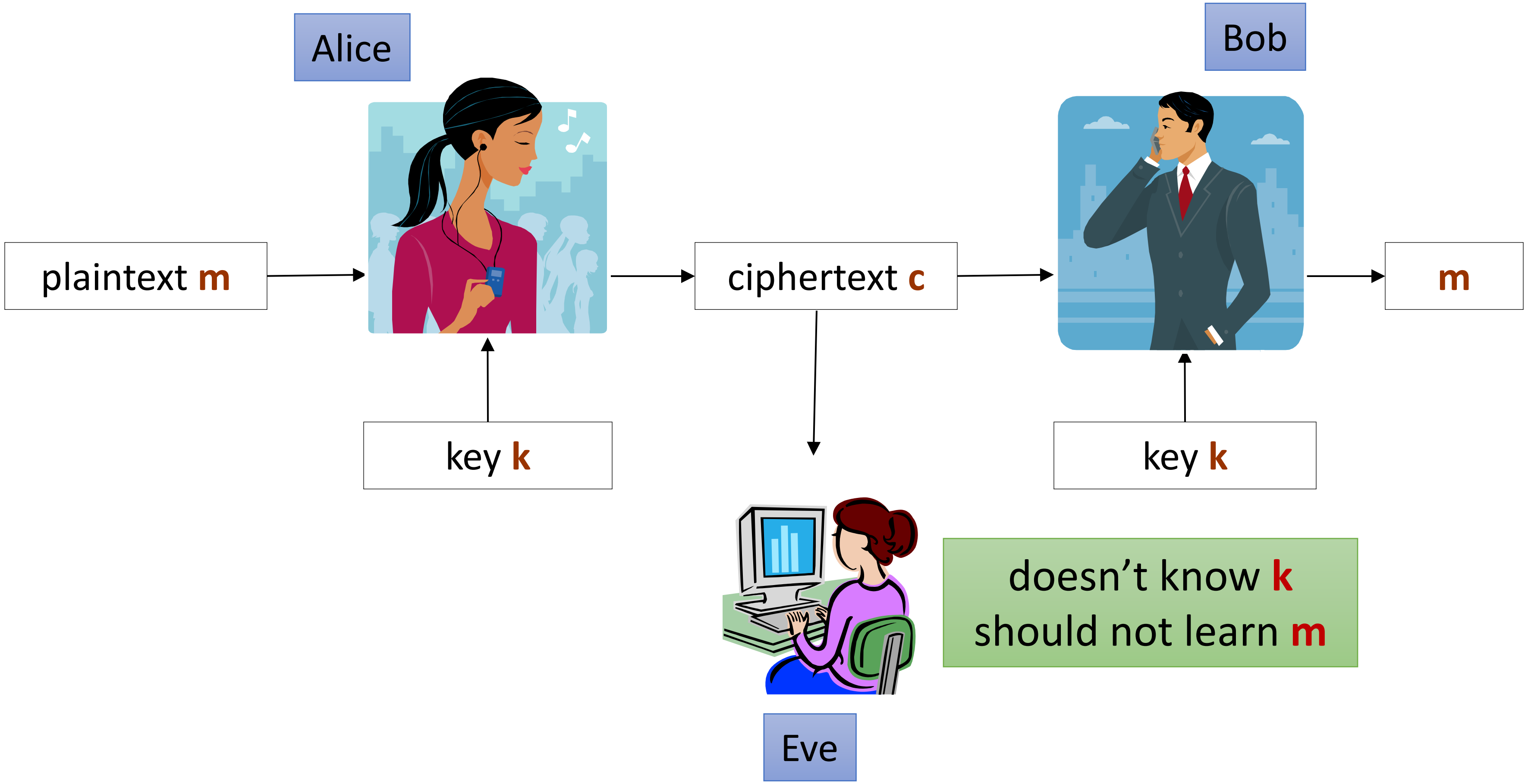
Associate Professor, Khoury College

Northeastern University

Outline

- Symmetric-key crypto
- Definitions of security for encryption
- Block ciphers and modes of operation
- Public-key crypto
- Key exchange

Encryption Terminology



Alice

Bob

plaintext **m**

ciphertext **c**

m

key **k**

key **k**

doesn't know **k**
should not learn **m**

Eve

Encryption scheme = encryption & decryption procedures

One-time pad is perfectly secure!

ℓ – a parameter

$\mathcal{K} = \mathcal{M} = \{0,1\}^\ell$

component-wise **xor**

Vernam's cipher:

$$\mathrm{Enc}_k(m) = k \oplus m$$
$$\mathrm{Dec}_k(c) = k \oplus c$$

Gilbert Vernam

(1890 –1960)

Correctness:

$$\mathrm{Dec}_k(\mathrm{Enc}_k(m)) \;=\; k \oplus (k \oplus m)$$
$$m$$

Computational security

Restriction: **Eve is computationally-bounded**

We will construct schemes that in **principle can be broken** if the adversary has a huge computing power or is extremely lucky.

- E.g., break the scheme by enumerating all possible secret keys. (“**brute force attack**”)
- E.g., break the scheme by guessing the secret key.

Goal: cannot be broken with reasonable computing power with reasonable probability.

Eavesdropping security

- Ciphertext INDistinguishability under an EAVesdropping attacker (IND-EAV)

Charlie (Challenger)

Adv

Round 1: Charlie chooses k and encryption algo k, Enc_k

Round 2: Adv chooses two plaintext messages $\leftarrow$ $m_0, m_1 \in \boldsymbol{M}$

Round 3: Charlie chooses a random binary number $b \leftarrow_R \{0, 1\}$

Round 4: Charlie encrypts the corresponding message $c = Enc_k(m_b)$ $\rightarrow$

Round 5: Adv guesses the value of *b* $b' \in \{0, 1\}$

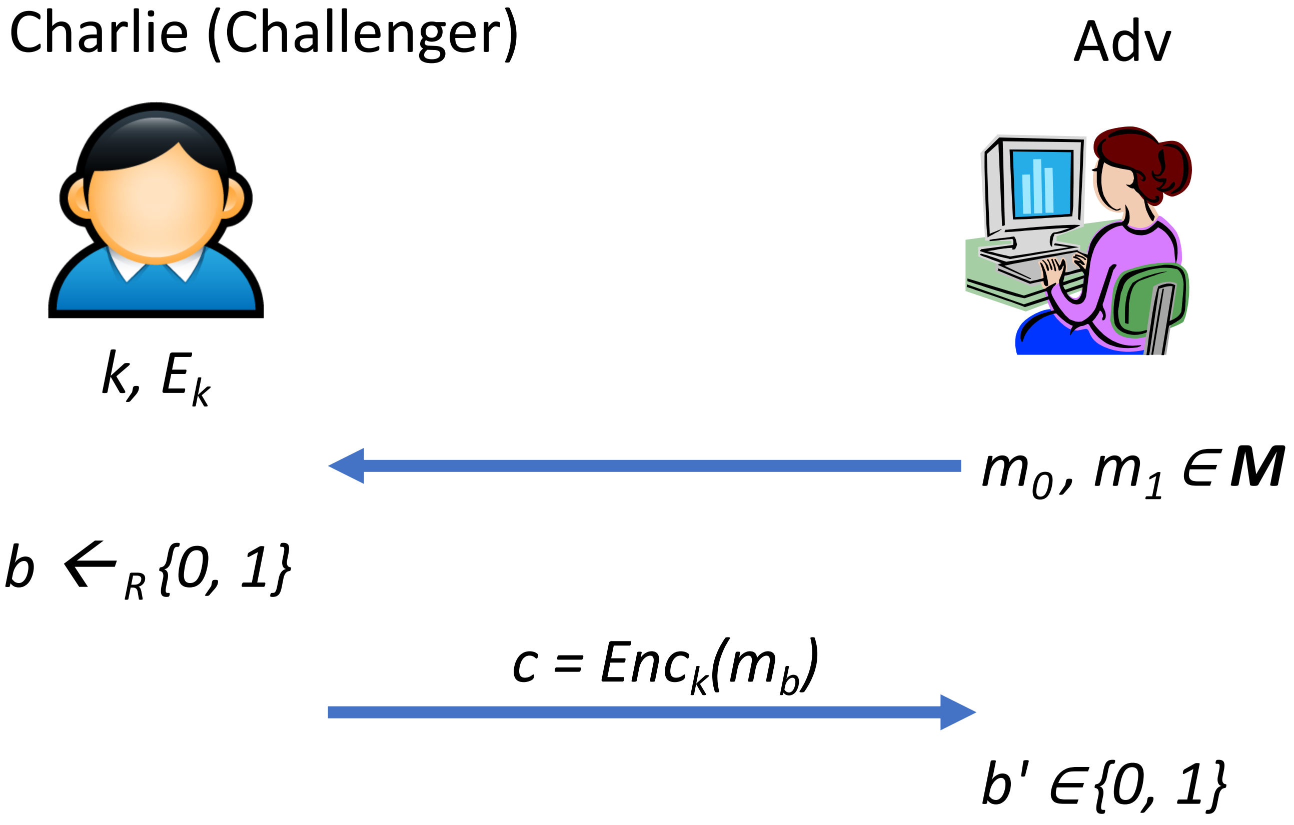
Adversary wins if $b = b'$

Computational secure IND-EAV

- *Enc* is computationally secure if for any Adv running in polynomial time:

P(Adv wins) = ½ + negligible(|k|)

How to achieve this?



Charlie (Challenger)

Adv

k, E_k

$m_0, m_1 \in \boldsymbol{M}$

$b \leftarrow_R \{0, 1\}$

$c = Enc_k(m_b)$

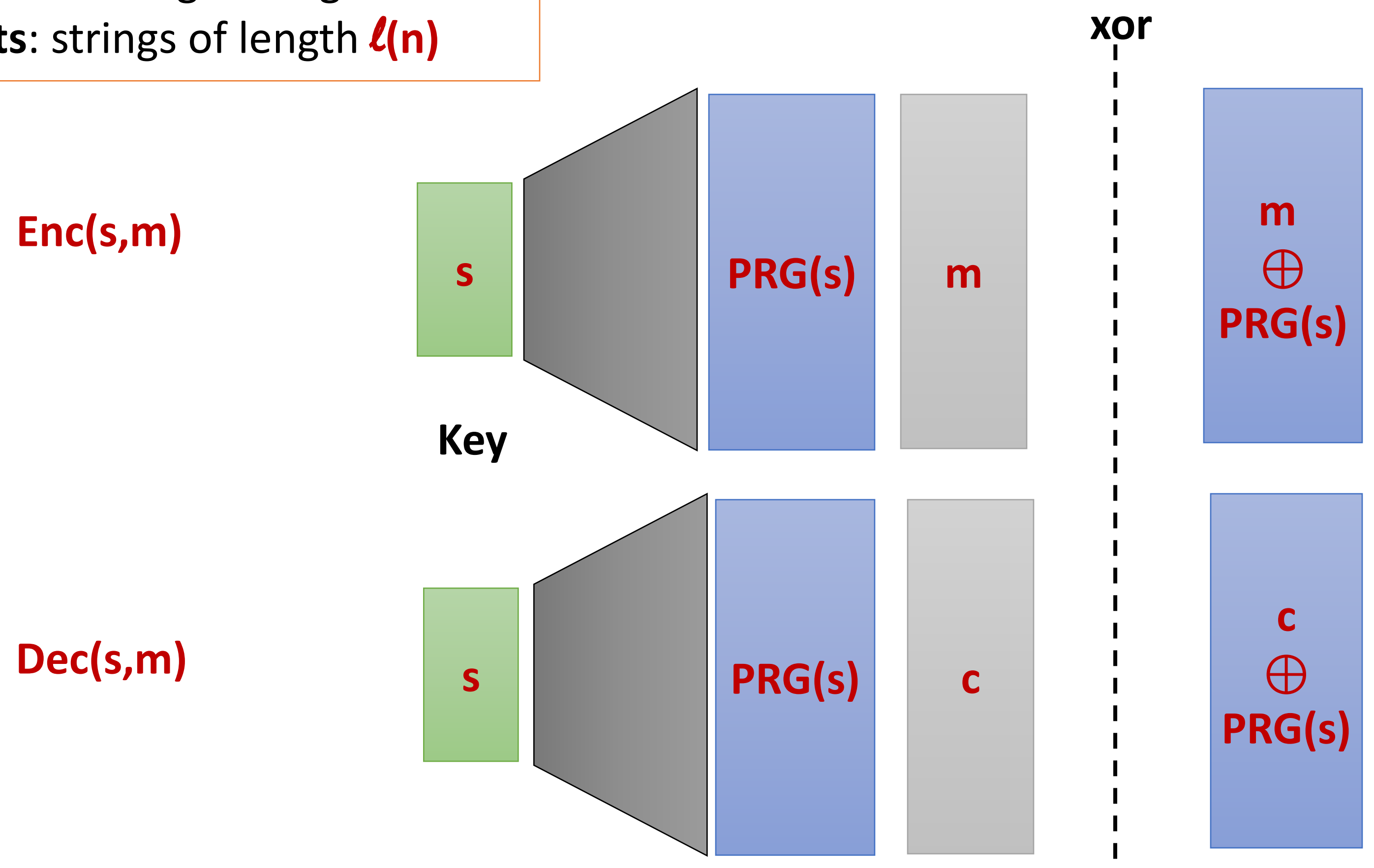
$b' \in \{0, 1\}$

Adversary wins if *b* = *b’*

Using a PRG to build efficient OTP

Use PRGs to “shorten” the key in the one time pad

Key: random string of length **n**
Plaintexts: strings of length **ℓ(n)**

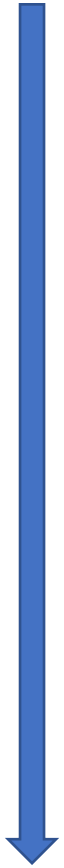


STREAM
CIPHER
Examples:
RC4, Salsa20

IND-EAV secure one-time pad

Adversarial capability

- Ciphertext-only attack: Perfect security, IND-EAV
  - Adversary observes one ciphertext
  - Cannot infer information about plaintext
- Chosen-plaintext attack: IND-CPA
  - Adversary can encrypt messages of his choice
  - Cannot infer information about plaintext by observing ciphertext
- Chosen-ciphertext attack: IND-CCA
  - Adversary can decrypt ciphertexts of its choice
  - Cannot learn plaintext information on other ciphertext



Stronger attacker

IND-CPA security

- Ciphertext Indistinguishability under Chosen-Plaintext Attack (CPA)
- Adv can encrypt messages of its choice

Round 1: Charlie chooses k and encryption algo

Round 2: Adv can encrypt messages

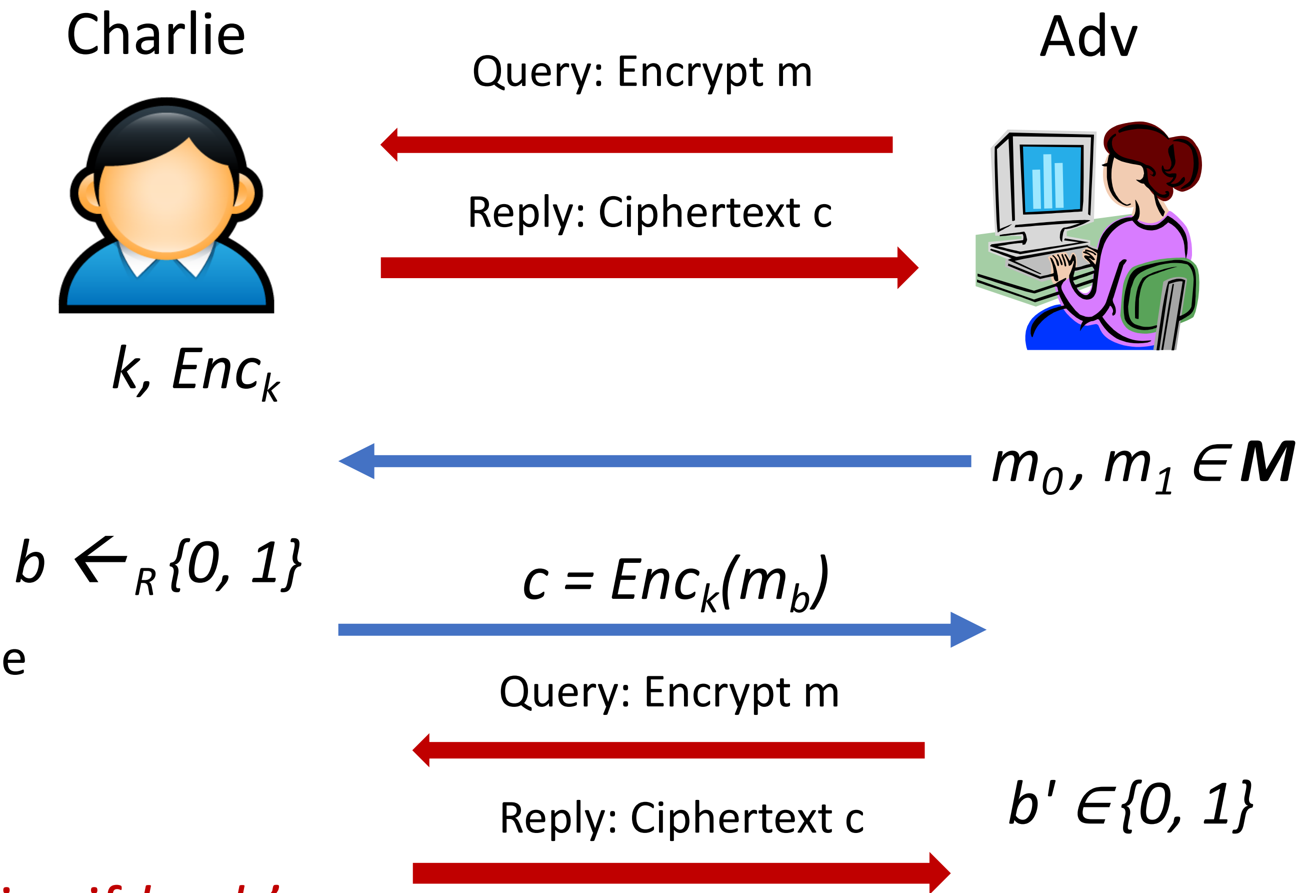
Round 3: Adv chooses two plaintext messages

Round 4: Charlie chooses a random binary number

Round 5: Charlie encrypts the corresponding message

Round 6: Adv can encrypt messages

Round 7: Adv guesses the value of *b*



Charlie

Adv

Query: Encrypt m

Reply: Ciphertext c

k, Enc_k

$m_0, m_1 \in \boldsymbol{M}$

$b \leftarrow_R \{0, 1\}$

$c = Enc_k(m_b)$

Query: Encrypt m

Reply: Ciphertext c

$b' \in \{0, 1\}$

Adversary wins if *b* = *b'*

IND-CPA Security

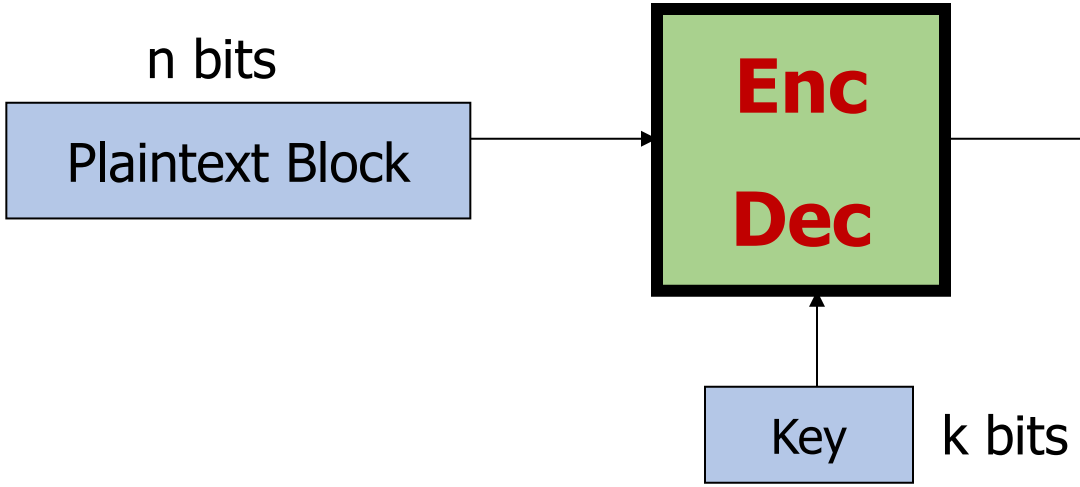
- Adversary can encrypt messages of his choice
  - Including m_0, m_1
- Adversary can encrypt any message before and after seeing the ciphertext c
- CPA adversary is stronger than EAV
- A scheme secure under CPA is also secure under EAV
- But not the other way around!
  - The One-time pad is IND-EAV secure, but not IND-CPA secure
  - IND-CPA is strictly stronger than IND-EAV (for symmetric-key encryption)
- How to design IND-CPA secure ciphers?

Symmetric Block Ciphers

Symmetric Key Cryptography

- Algorithms that use a single key for encryption and decryption
  - i.e. the algorithm is reversible
  - $\forall k\ \forall m\quad Dec_k(Enc_k(m)) = m$ where *m* is a message, *k* is a key, and Dec_k and Enc_k are decryption and encryption using *k*
- Historic examples:
  - Caeser shift, mono and polyalphabetic substitution, OTP
- Modern examples (block ciphers):
  - DES, 3DES, RC4, Blowfish, Twofish, AES
  - **Warning**: many of these methods are known to be vulnerable

Block ciphers: crypto work horse



Canonical examples:

1. DES: n=64 bits, k=56 bits
2. Triple DES: n=64 bits, k=168 bits
3. AES: n=128 bits, k=128, 192, 256 bits

Desired properties:

1. Change one bit of plaintext completely changes ciphertext
2. Good mixing properties
3. Ciphertext looks random

The Data Encryption Standard (DES)

- Early 1970s: Horst Feistel designs Lucifer at IBM

  key-len = 128 bits ; block-len = 128 bits

- 1973: NBS asks for block cipher proposals.
  IBM submits variant of Lucifer.
- 1976: NBS adopts DES as a federal standard

  key-len = 56 bits ; block-len = 64 bits

- 1997: DES broken by exhaustive search (short keys)
- 2000: NIST adopts Rijndael as AES to replace DES

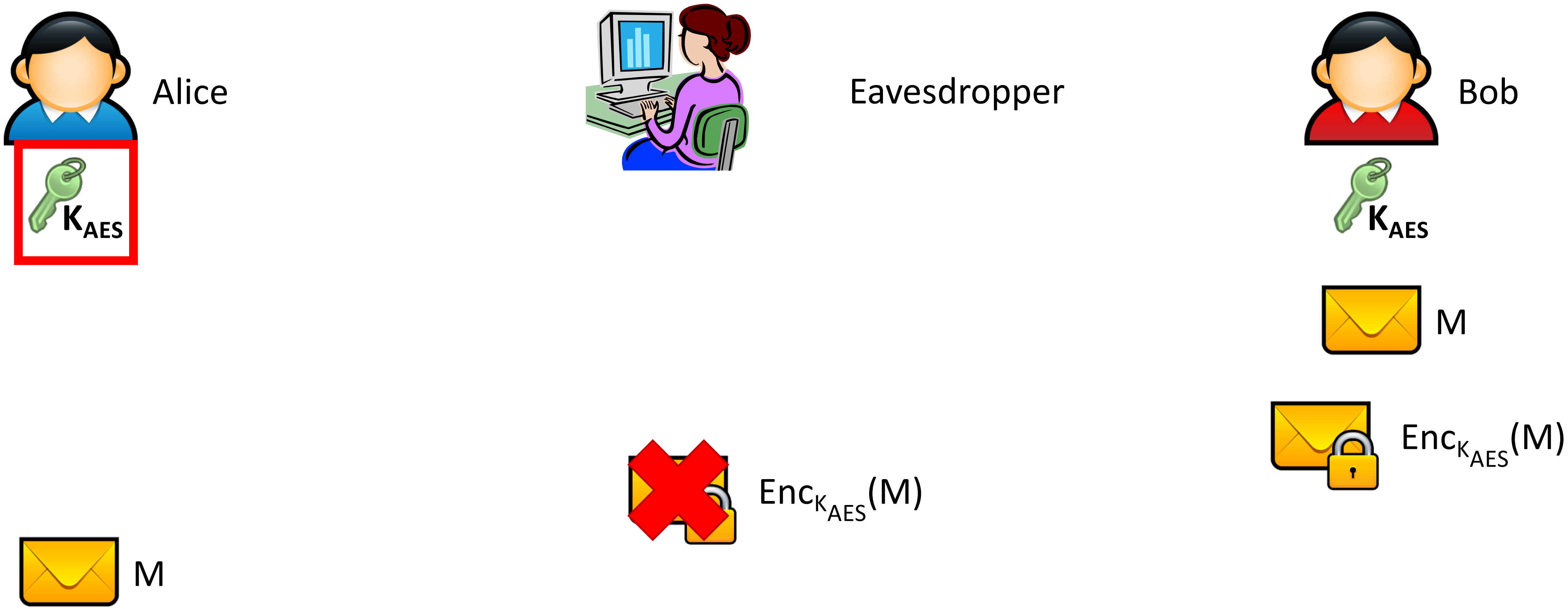
Data Encryption Standard (DES)

- Designed by IBM, with modifications proposed by the NSA
- US national standard from 1977 to 2001
- Block size is 64 bits
- Key size is 56 bits
- Has 16 rounds based on Feistel permutations
- Designed mostly for fast implementation in hardware
    - Software implementation is somewhat slow
- Considered insecure now
    - Vulnerable to brute-force attacks, key too short

Advanced Encryption Standard (AES)

- In 1997, NIST made a formal call for algorithms stipulating that the AES would specify an unclassified, publicly disclosed encryption algorithm, available royalty-free, worldwide
- Goal: replace DES for both government and private-sector encryption.
- The algorithm must implement symmetric key cryptography as a block cipher and (at a minimum) support block sizes of 128-bits and key sizes of 128-, 192-, and 256-bits.
- In 1998, NIST selected 15 AES candidate algorithms.
- In 2000, NIST selected Rijndael (invented by Joan Daemen and Vincent Rijmen) as the AES
- Designed to be efficient in both hardware and software

AES Example



Alice

Eavesdropper

Bob

K_{AES}

K_{AES}

M

$\mathrm{Enc}_{K_{AES}}(M)$

$\mathrm{Enc}_{K_{AES}}(M)$

M

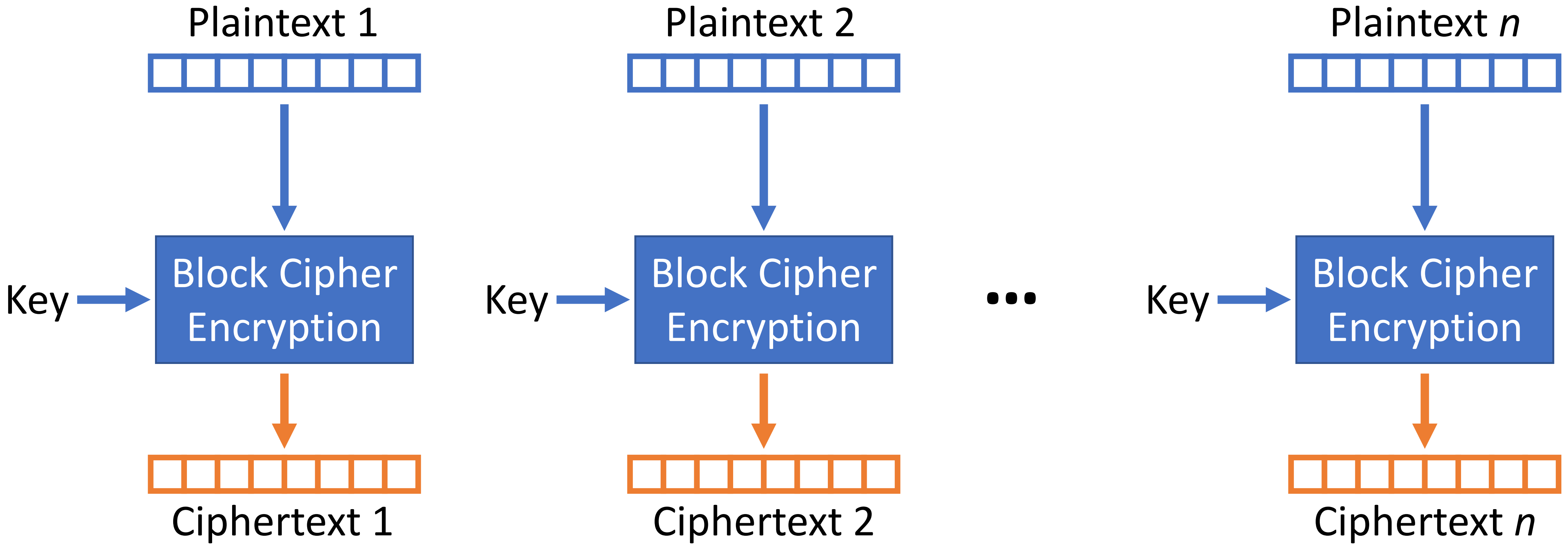
- AES is assumed to be secure (aka ciphertext is pseudorandom)!
- This is backed up by years of crytanalysis
- Block cipher: encrypts blocks of fixed size

Need for Encryption Modes

- A block cipher encrypts only one block
  - But a message may be longer than one block
- Need a way to extend the algorithm to encrypt arbitrarily long messages
- Need to ensure that if block cipher is secure, then whole encryption is secure
  - Whole operation should be secure if block cipher is secure

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?