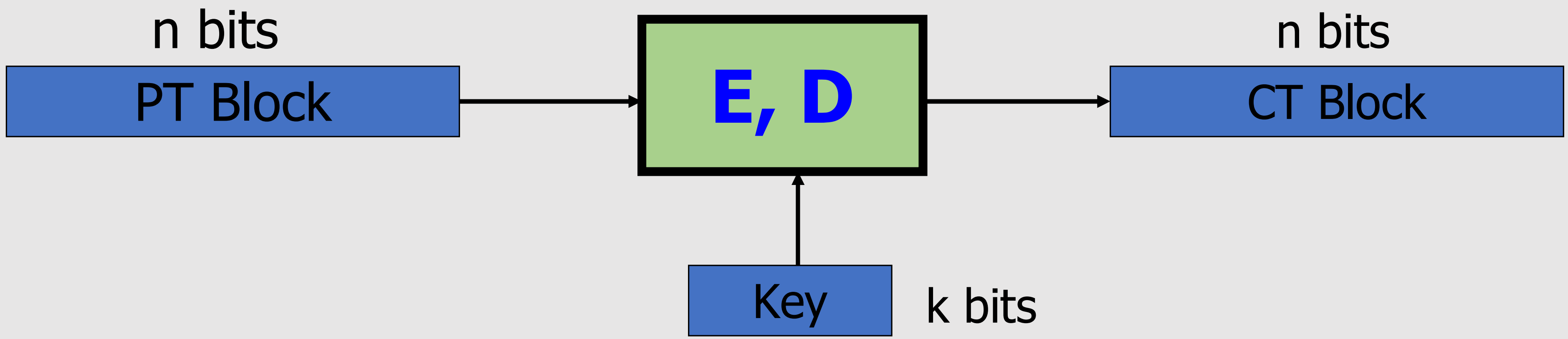
Modes of Operation (using block ciphers)

Outline

- One-Time Key
  - Semantic Security
  - **Electronic Code Book (ECB)**
  - Deterministic Counter Mode (DETCTR)
- Many-Time Key
  - Semantic Security for Many-Time Key:
    Semantic Security under Chosen-Plaintext Attack (CPA)
  - **Cipher Block Chaining (CBC)**
    - Randomized
    - Nonce-based

Review: PRPs and PRFs

Block Ciphers



Canonical examples:

- **DES**: n= 64 bits, k = 56 bits
- **3DES**: n= 64 bits, k = 168 bits
- **AES**: n=128 bits, k = 128, 192, 256 bits

Abstractly: PRPs and PRFs

- **Pseudo Random Function** (**PRF**) defined over (K,X,Y):

  $\mathbf{F}: K \times X \rightarrow Y$

  such that there exists "efficient" algorithm to evaluate F(k,x)

- **Pseudo Random Permutation** (**PRP**) defined over (K,X):

  $E: K \times X \rightarrow X$

  such that:

  1. There exists "efficient" <u>deterministic</u> algorithm to evaluate E(k,x)
  2. The function $E(k, \cdot)$ is one-to-one, for every k
  3. There exists "efficient" inversion algorithm D(k,y)

Using block ciphers

- Don’t think about the **inner-workings** of AES and 3DES.
- We assume both are **secure PRPs** and will see how to use them

Modes of Operation

How to use a **block cipher** on **messages consisting of <u>more than one block</u>**

- **One-Time Key**
  - Electronic Code Book
  - Deterministic Counter Mode

- **Many-Time Key**
  - Cipher Block Chaining
  - Counter Mode

Modes of Operation
One-Time Key

(example: encrypted email, new key for every message)

Using PRPs and PRFs

Goal: build "secure" encryption from a secure PRP (e.g., AES).

This segment: **one-time key**

1. **Adversary's power:** Adversary sees only one ciphertext (one-time key)

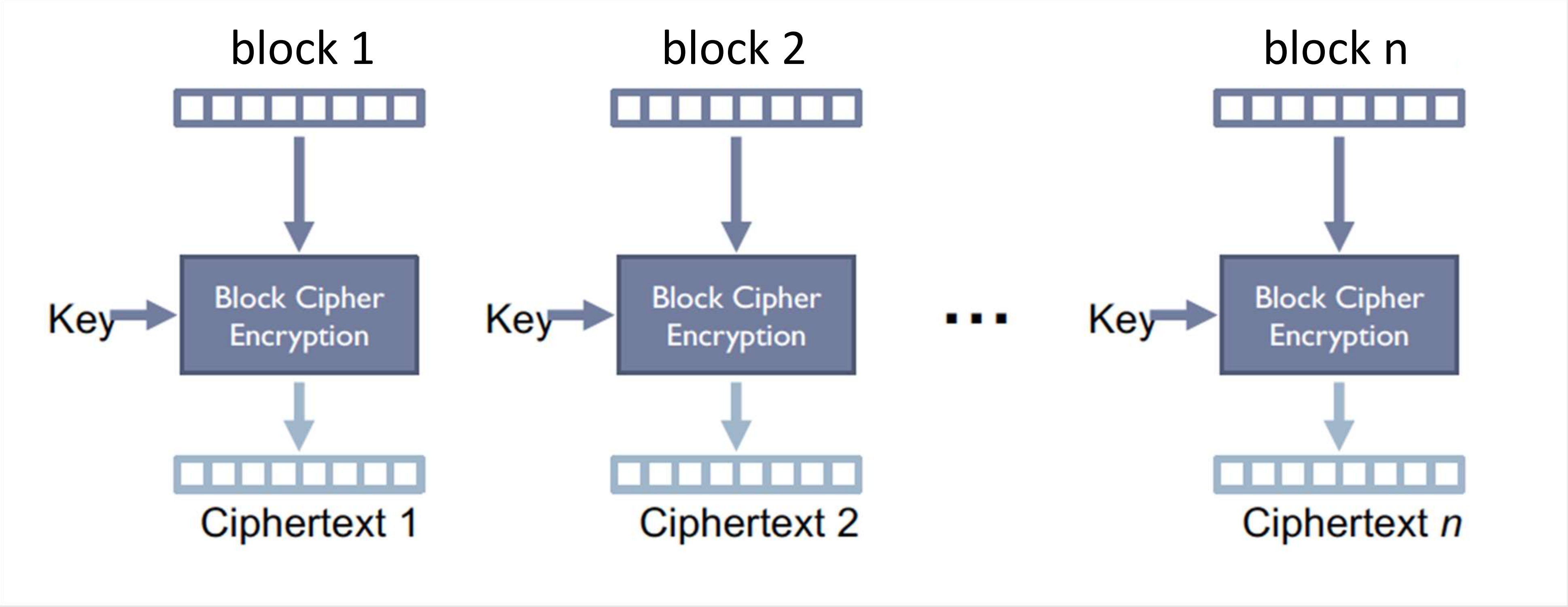
2. **Adversary's goal:** Learn info about PT from CT (semantic security)

Next segment: many-time keys (a.k.a. *chosen-plaintext security*)

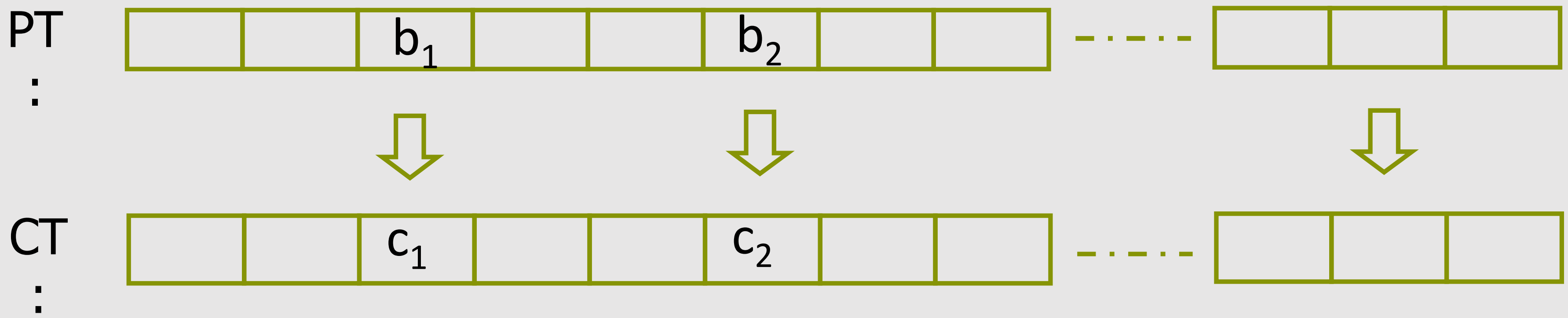
ECB encryption mode

Message is broken into independent blocks

Electronic Code Book (**ECB**): Each block is encrypted separately



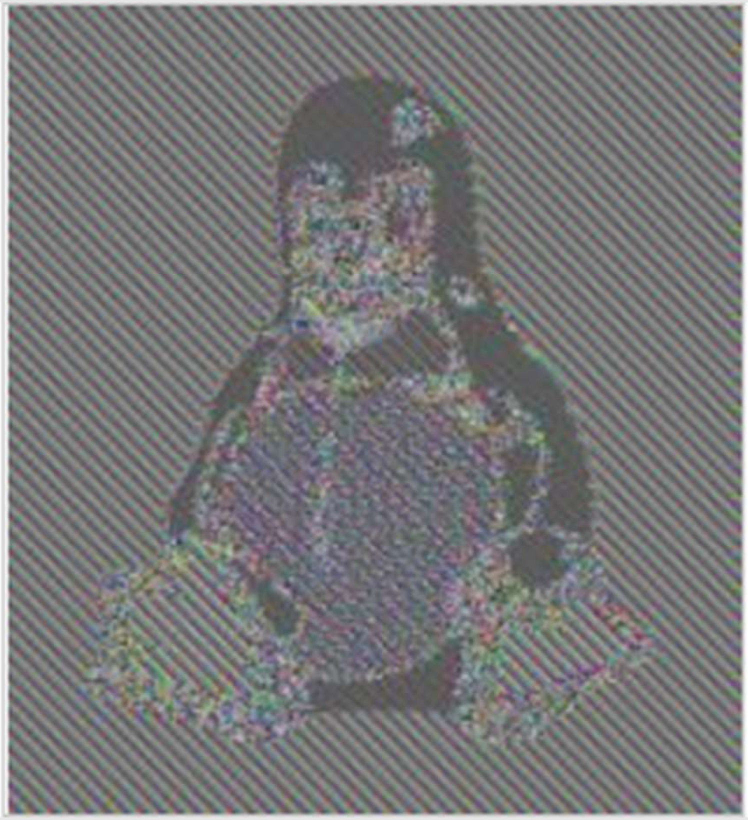
ECB: incorrect use of a PRP



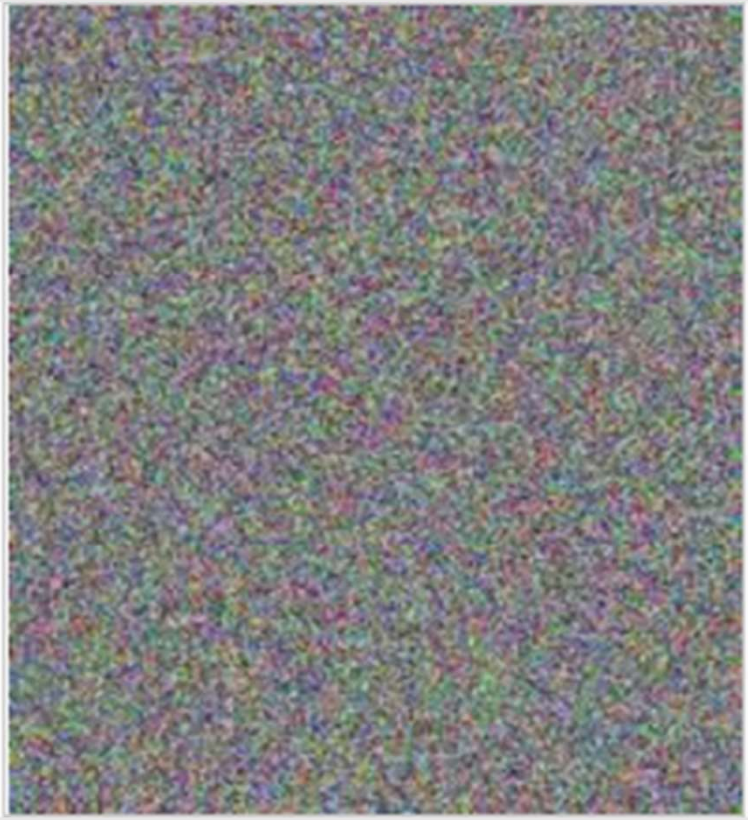
Problem: if $b_1 = b_2$ then $c_1 = c_2$

In pictures

Plain text



Cipher text with **ECB**



Cipher text with
other modes of operation

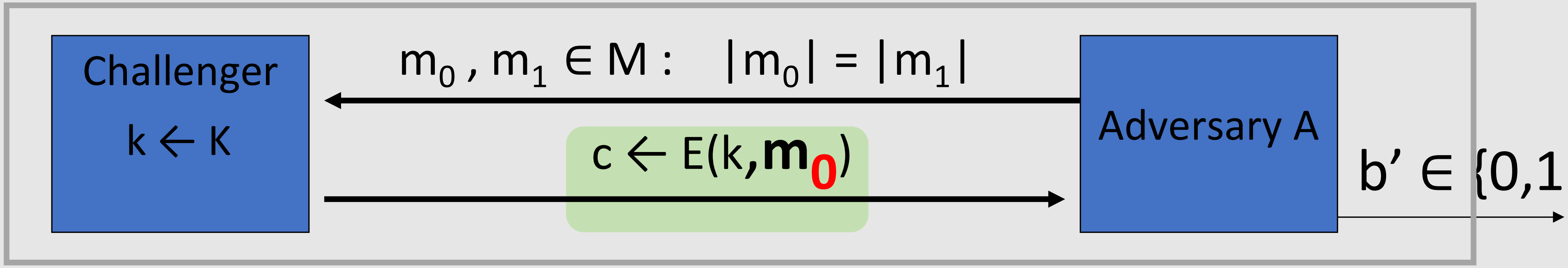
Cryptanalysis of ECB

- 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 semantically secure?

- Do not use ECB mode in practice

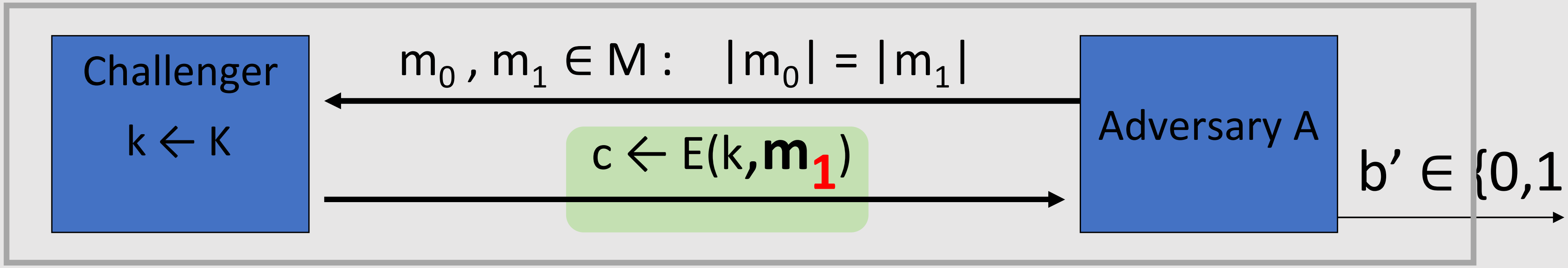
Definition of Semantic Security (one-time key)

EXP(**0**):



one time key $\Rightarrow$ adversary sees only one ciphertext

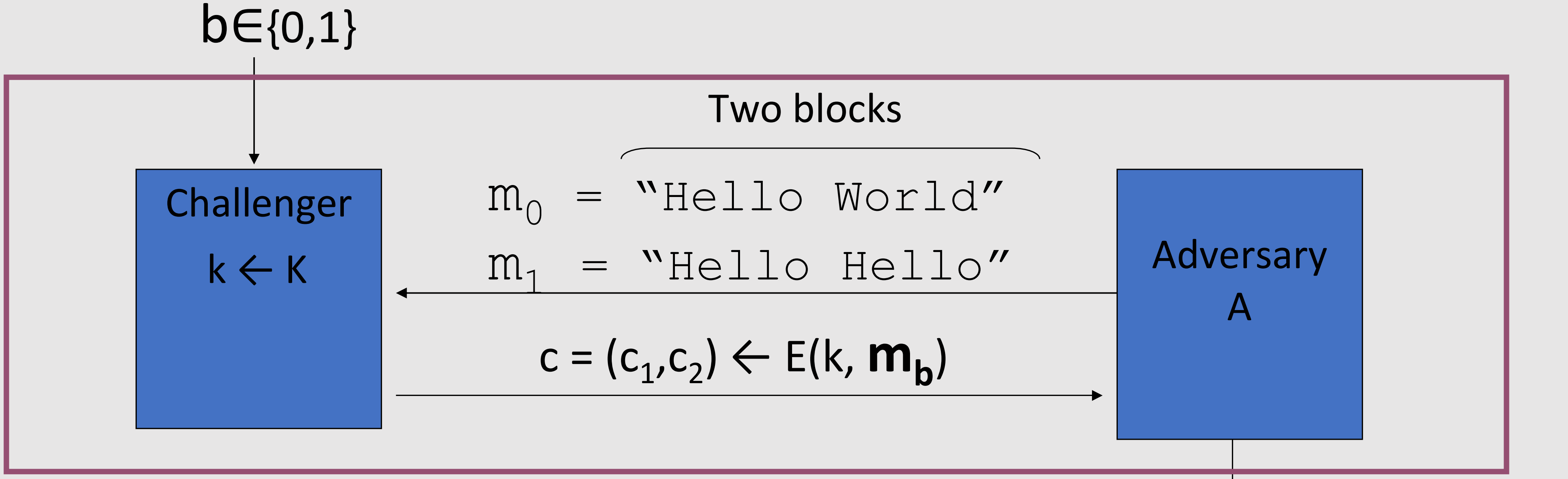
EXP(**1**):



$\text{Adv}_{SS}[A,\text{Cipher}] = \Big|\, \Pr[\,\textbf{EXP(0)}=1\,] - \Pr[\,\textbf{EXP(1)}=1\,] \,\Big|$ should be "negligible" for all "efficient" A

ECB is not Semantically Secure

ECB **is not semantically secure** for messages that contain **more than one block** (known-plaintext attack)



If $c_1=c_2$ output 1, else output 0

Then $\text{Adv}_{SS}[A, ECB] =$

Deterministic Counter Mode (Secure Construction)

- **PRF** F : K × $\{0,1\}^n \rightarrow \{0,1\}^n$ (e.g., n=128 with AES)
- **E_{DETCTR} (k, m) =** (Encryption)

	m[0]	m[1]	…	m[L]
$\oplus$	**F(k,0)**	**F(k,1)**	…	**F(k,L)**
	c[0]	c[1]	…	c[L]

⇒ Stream cipher built from a PRF (e.g., AES, 3DES)

Deterministic Counter Mode (Secure Construction)

- **PRF** F : K × $\{0,1\}^n \rightarrow \{0,1\}^n$ (e.g., n=128 with AES)
- **D_{DETCTR} (k, c) =** (Decryption)

	c[0]	c[1]	…	c[L]
$\oplus$	**F(k,0)**	**F(k,1)**	…	**F(k,L)**
	m[0]	m[1]	…	m[L]

No need to **invert** F when decrypting

Deterministic Counter Mode Security

Theorem: For any L>0,

If **F** is a **secure PRF** over (K,X,X) then

DETCTR is **semantically secure** over (K,X^L,X^L).

In particular, for every efficient adversary **A** **attacking DETCTR** there exists an efficient adversary **B** **attacking F** s.t.:

$$\mathrm{Adv}_{SS}[A, \mathrm{DETCTR}] = 2 \cdot \mathrm{Adv}_{PRF}[B, F]$$

$\mathrm{Adv}_{PRF}[B, F]$ is negligible (since F is a secure PRF)

Hence, $\mathrm{Adv}_{SS}[A, \mathrm{DETCTR}]$ must be negligible.

Modes of Operation
Many-Time Key

Examples:

- **File systems**: Same AES key used to encrypt many files.
- **IPsec**: Same AES key used to encrypt many packets.

Semantic Security for Many-Time Key

Key used **more than once** $\Rightarrow$ adversary sees many CTs with the same key (i.e., used for **multiple messages**)

Adversary's power: **Chosen-Plaintext Attack (CPA)**

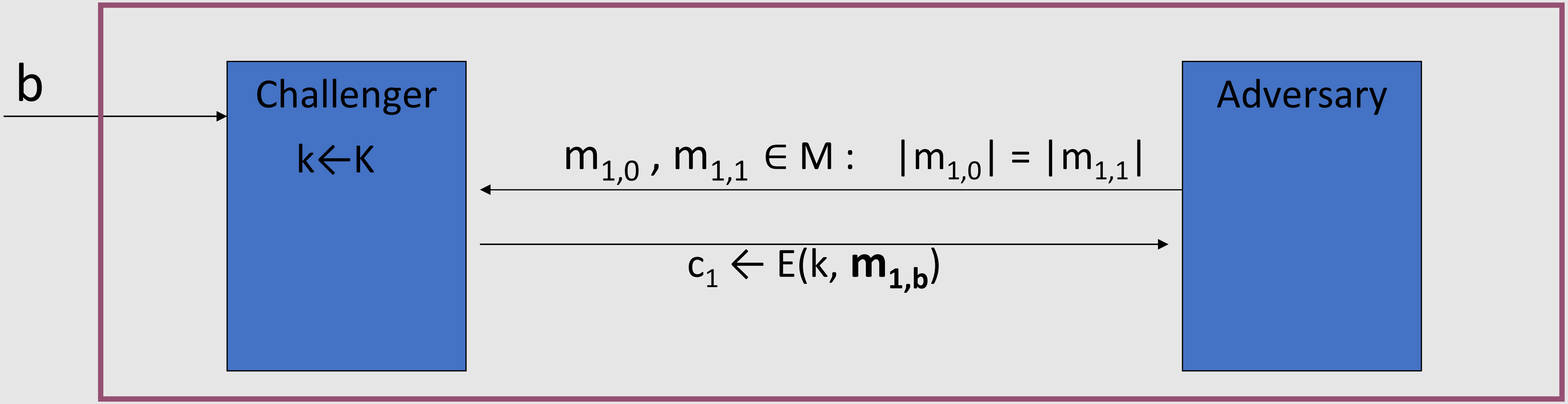
Adversary can obtain the encryption of arbitrary messages of his choice (conservative modeling of real life)

Adversary's goal: Break semantic security

Semantic Security for Many-Time Key (CPA Security)

Q = (E,D) a cipher defined over (K,M,C)

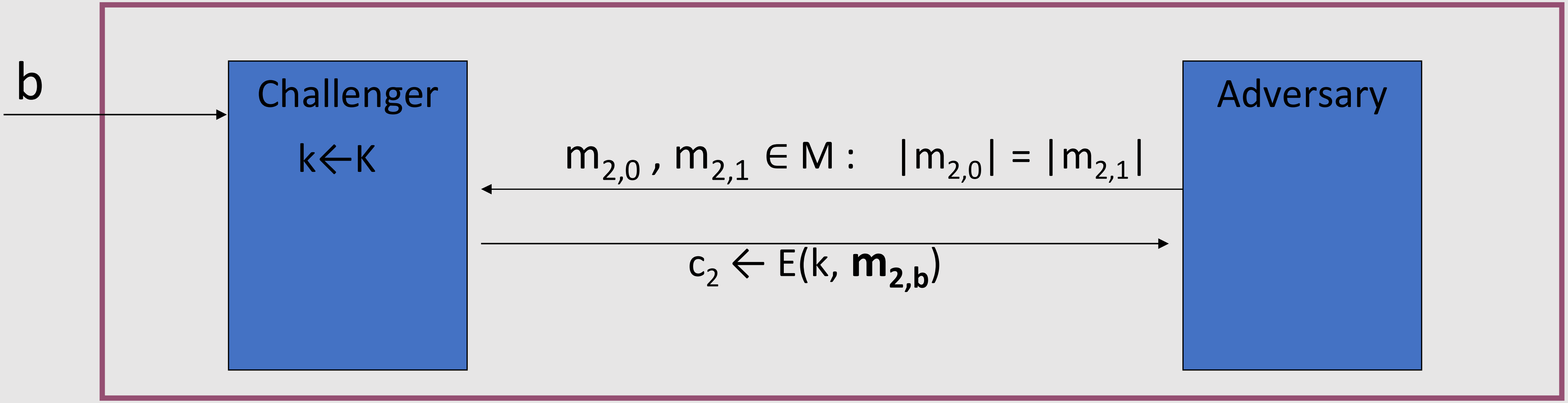
For b=0,1 define EXP(b) as:



Semantic Security for Many-Time Key (CPA Security

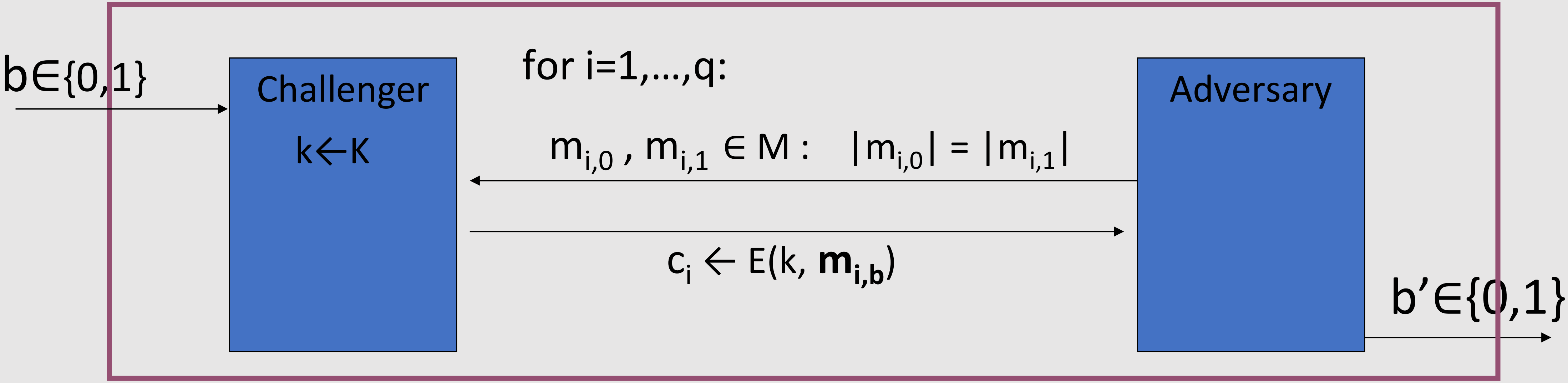
Q = (E,D) a cipher defined over (K,M,C)

For b=0,1 define EXP(b) as:



Semantic Security for Many-Time Key (CPA Security

Q = (E,D) a cipher defined over (K,M,C)
For b=0,1 define EXP(b) as:



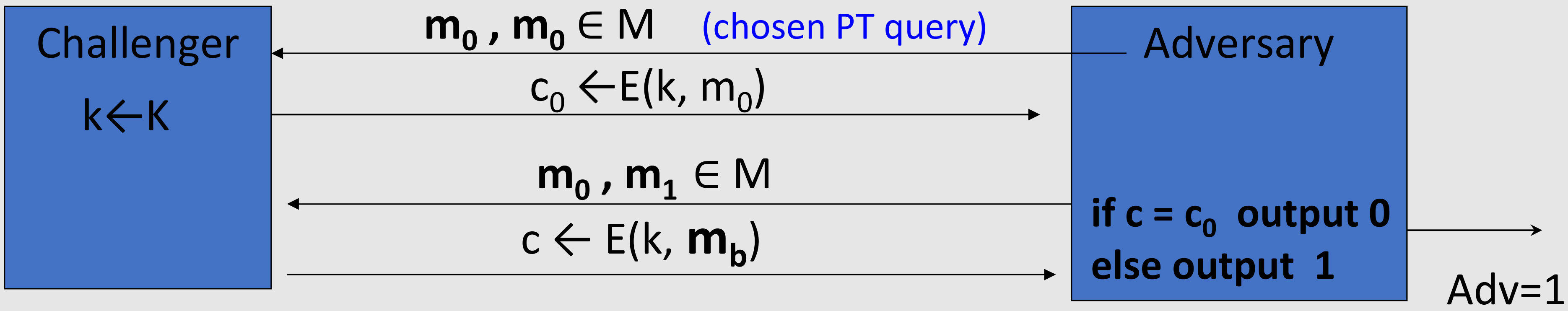
CPA ⇒ if adversary wants $c = E(k, m)$ it queries with $m_{j,0} = m_{j,1} = m$

Definition: Q is **semantically secure under CPA** if for all "efficient" adversary A:

$$\mathrm{Adv}_{\mathrm{CPA}}[A,Q] = \left|\Pr[\mathrm{EXP}(0)=1] - \Pr[\mathrm{EXP}(1)=1]\right| \quad \text{is “negligible”.}$$

Ciphers Insecure under CPA

Suppose E(k,m) **always outputs <u>same ciphertext</u> for msg m and key k**. Then:

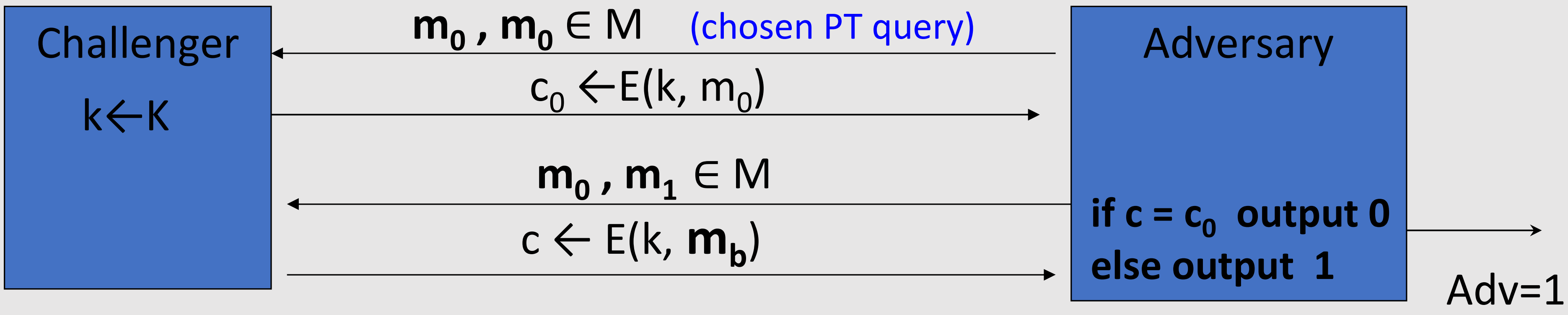


So what? an attacker can learn that two encrypted files are the same, two encrypted packets are the same, etc.

Leads to significant attacks when the message space M is small

Ciphers Insecure under CPA

Suppose E(k,m) **always outputs <u>same ciphertext</u> for msg m and key k**. Then:

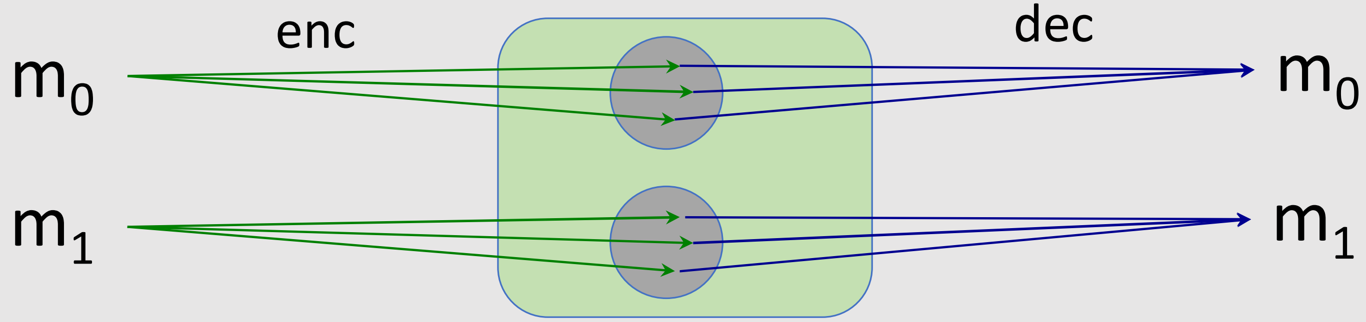


If secret key is to be used multiple times $\Rightarrow$

given **the same plaintext message twice**,
encryption must produce different outputs.

Solution 1: Randomized Encryption

- E(k,m) is a randomized algorithm:

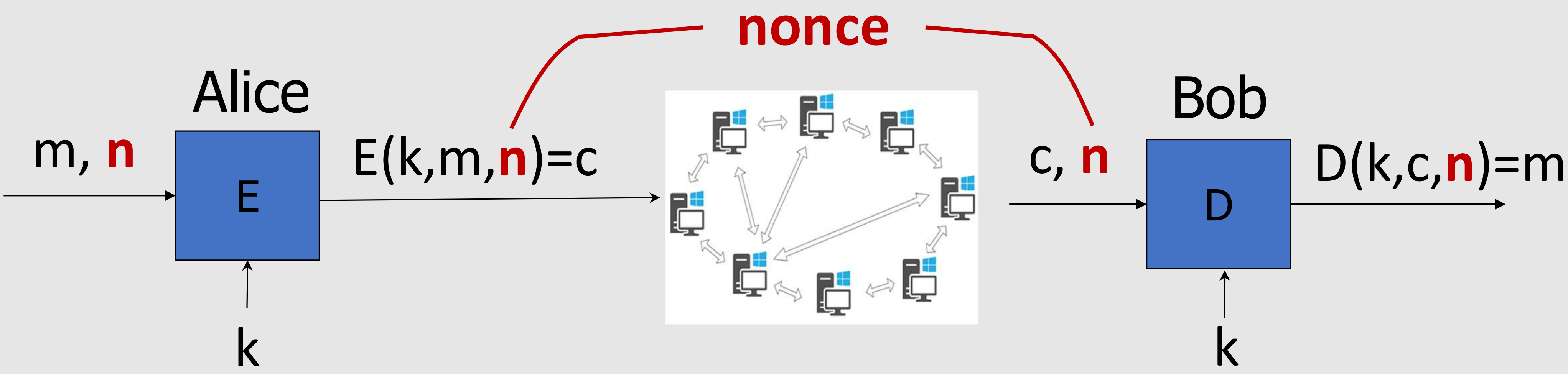


⇒ encrypting same msg twice gives different ciphertexts (w.h.p.)

⇒ ciphertext must be longer than plaintext

Roughly speaking: CT-size = PT-size + “# random bits”

Solution 2: Nonce-based Encryption



Nonce n:

- a value that changes from msg to msg
- (k,n) pair **<u>never</u> used more than once**
- n does **not** need to be **secret** and does **not** need to be **random**

Solution 2: Nonce-based Encryption

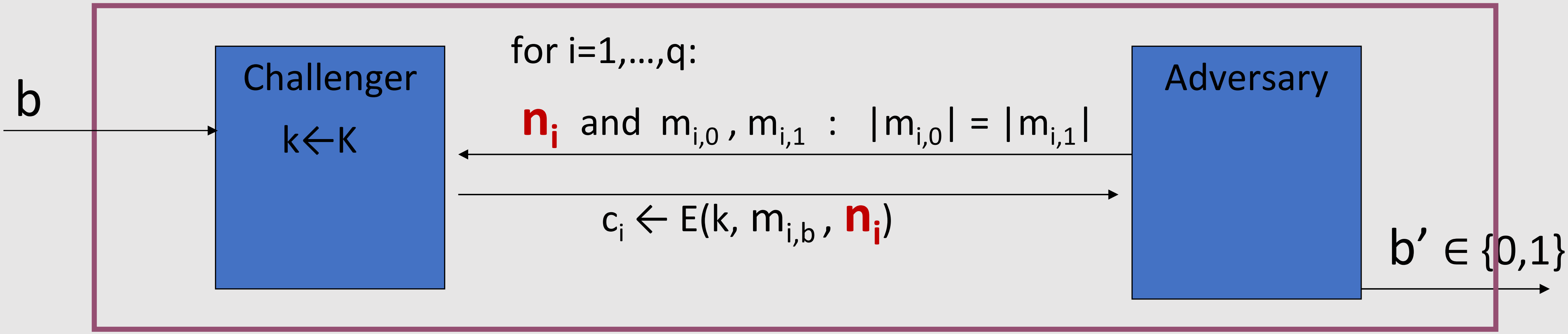
Nonce

- **Method 1:** nonce is a **counter** (e.g., packet counter)
  - used when encryptor keeps state from msg to msg
  - if decryptor has same state, need not send nonce with CT

- **Method 2:** encryptor chooses a **random nonce**, $n \leftarrow \mathcal{N}$
  (It's like randomized encryption)
  (ex. Multiple devices encrypting with the same key)
  - $\mathcal{N}$ must be large enough to ensure that the same nonce is not chosen twice with high probability

CPA Security for Nonce-based Encryption

System should be secure when **nonces are chosen adversarially.**



All nonces $\{n_1, …, n_q\}$ must be distinct.

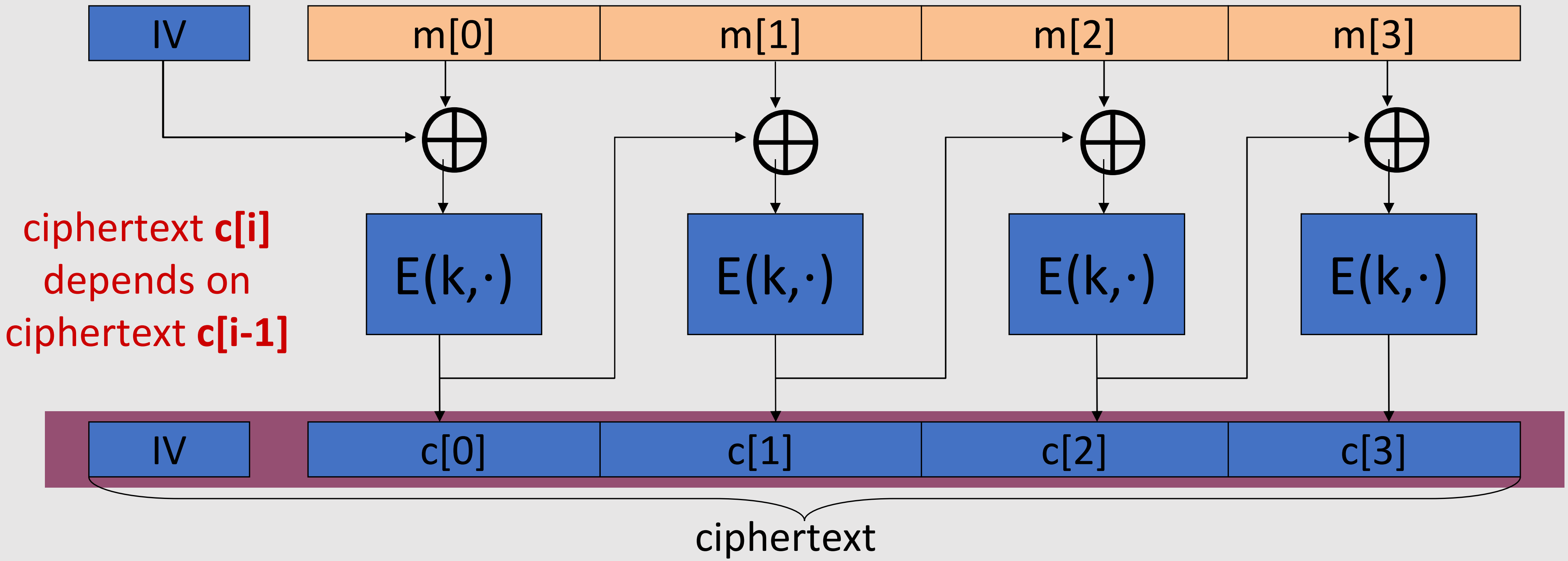
Definition. Nonce-based **Q** is **semantically secure under CPA** if for all "efficient" adversary A:

$$\mathrm{Adv}_{\mathrm{nCPA}}[A,Q] = |\Pr[\mathrm{EXP}(0)=1] - \Pr[\mathrm{EXP}(1)=1]|$$ is "negligible".

Many-time Key Mode of Operation:
Cipher Block Chaining (**CBC**)

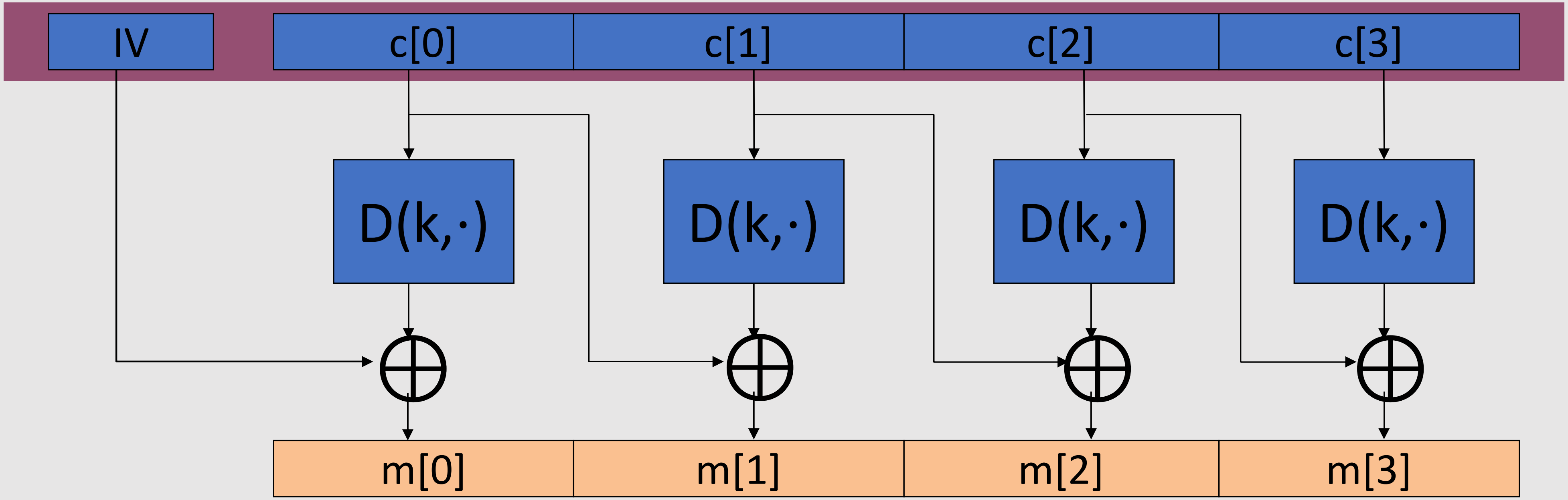
Construction 1: CBC with random Initialization Vector (IV)

- **PRP E** : $K \times \{0,1\}^n \rightarrow \{0,1\}^n$
- (Encryption) $\mathbf{E_{CBC}(k,m)}$**:** choose **random** $IV \in \{0,1\}^n$ and do:



Construction 1: CBC with random IV

- $D : K \times \{0,1\}^n \to \{0,1\}^n$ **inversion algorithm** of E
- (Decryption) $\mathbf{D_{CBC}(k,c)}$**:**



(Randomized) CBC Security

Theorem: For any L>0 (length of the message we are encrypting),
If **E** is a **secure PRP** over (K,X) then
CBC is **semantically secure under CPA** over (K, X^L, X^{L+1}).

In particular, for every efficient **q**-query adversary **A** **attacking CBC** there exists an efficient PRP adversary **B** **attacking E** s.t.

$$\mathrm{Adv}_{\mathrm{CPA}}[A, \mathrm{CBC}] \le 2 \cdot \mathrm{Adv}_{\mathrm{PRP}}[B, E] + \mathbf{2\, q^2\, L^2 / |X|}$$

Note: CBC is only secure as long as $q^2L^2 \ll |X|$

(the error term should be negligible)

An example

$$\text{Adv}_{\text{CPA}}[A, \text{CBC}] \le 2\cdot \text{Adv}_{\text{PRP}}[B, E] + \mathbf{2\, q^2 L^2 / |X|}$$

q = # messages encrypted with the same key **k** , **L** = max length of a message in blocks

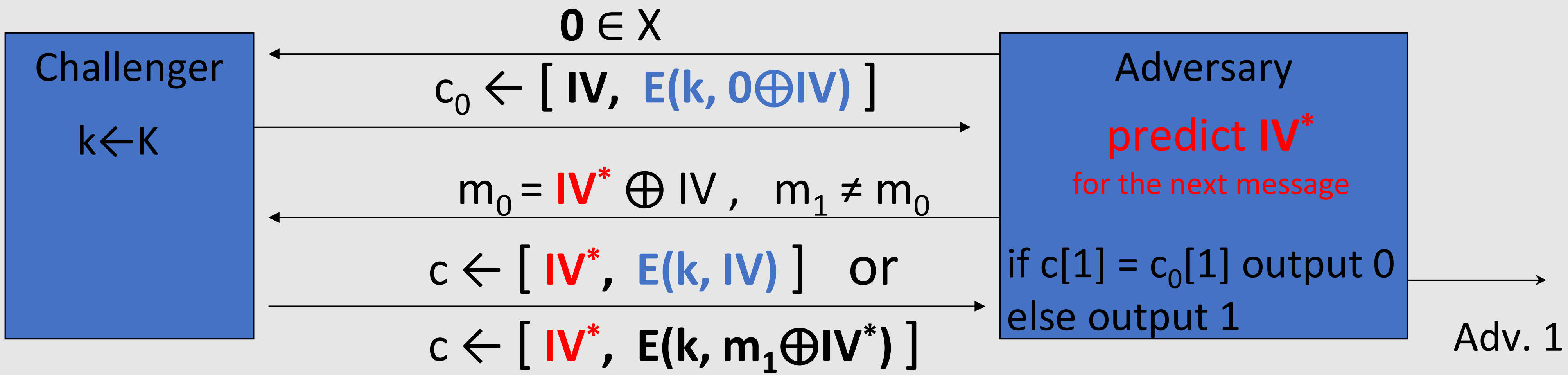
Suppose we want $\text{Adv}_{\text{CPA}}[A, \text{CBC}] \le 1/2^{32}$ $\Leftarrow$ $q^2 L^2 / |X| < 1/2^{32}$

- AES: $|X| = 2^{128}$ $\Rightarrow$ $q\,L < 2^{48}$
  So, after 2^{48} AES blocks, must change key
- 3DES: $|X| = 2^{64}$ $\Rightarrow$ $q\,L < 2^{16}$
  So, after 2^{16} DES blocks, must change key
  $\Rightarrow$ after 2^{16} blocks (each of 8 bytes) need to change key $\Rightarrow$ $2^{16} \times 8 = ½$ MB !!!

Warning: an attack on CBC with rand. IV

CBC where adversary can **predict** the IV is not CPA-secure !!

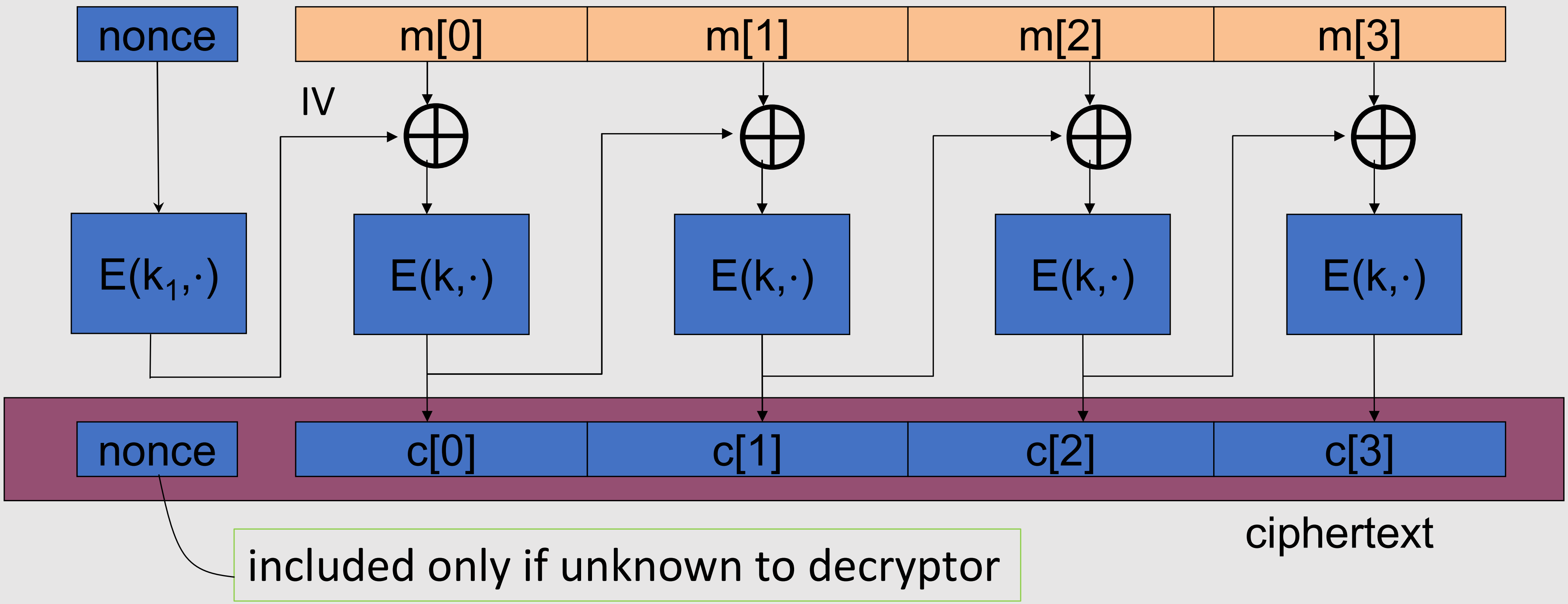
Suppose given $c \longleftarrow E_{CBC}(k,m)$ adversary can predict IV for next message



Bug in SSL/TLS 1.0: IV for record #i is last CT block of record #(i-1)

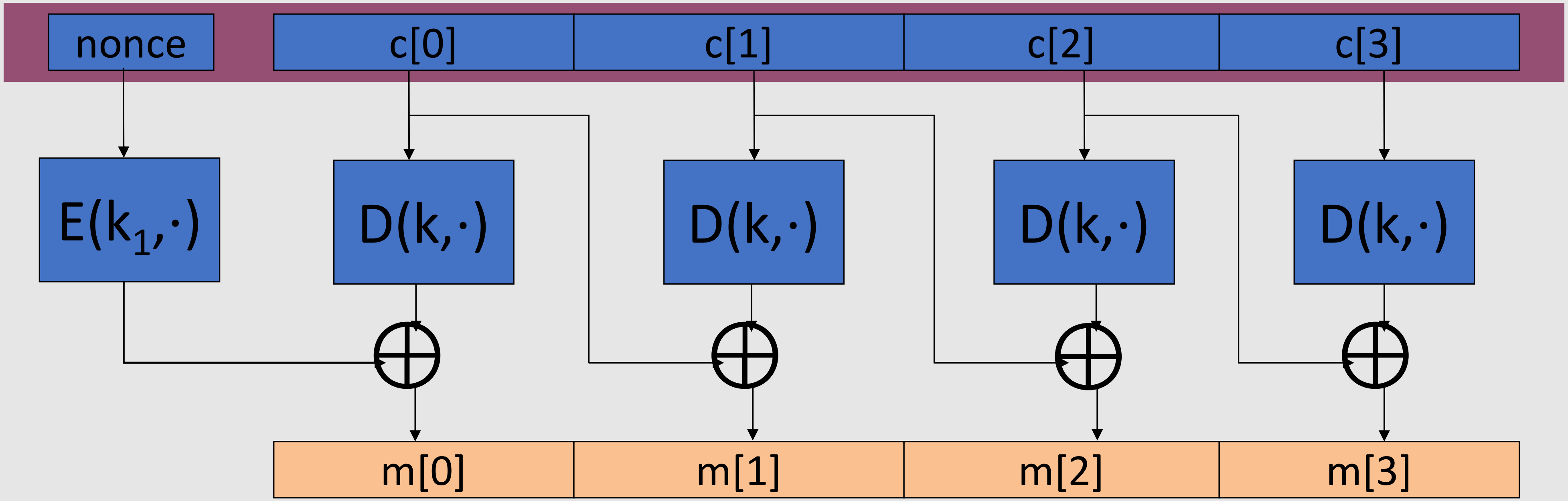
Construction 2: Nonce-based CBC

- key = (**k, k$_1$**)
- (key, nonce) pair is used for only one message
- **Encryption:**



Construction 2: Nonce-based CBC

- **Decryption:**



An example Crypto API (OpenSSL)

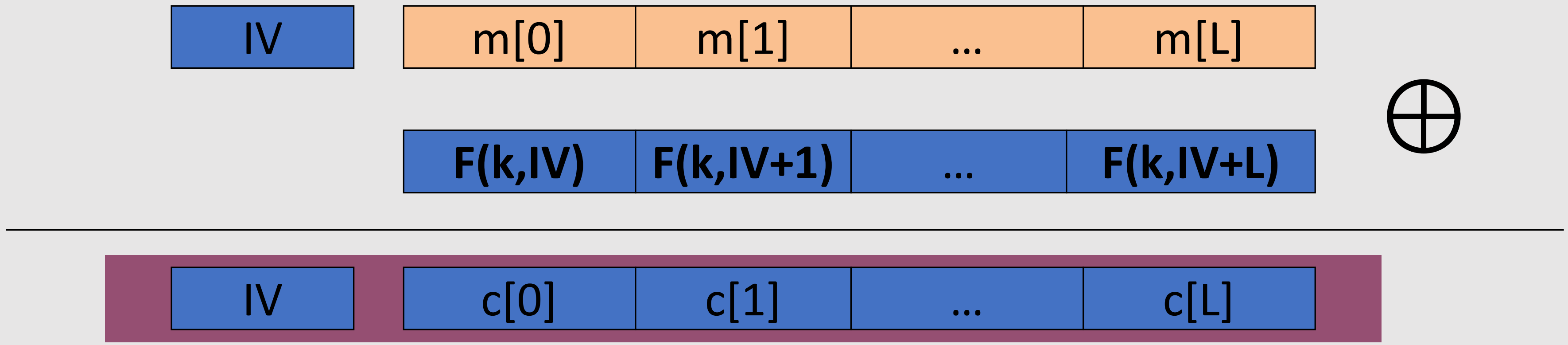
```
void AES_cbc_encrypt(
        const unsigned char *in,
        unsigned char *out,
        size_t length,
        const AES_KEY *key,
        unsigned char *ivec,        ⟵  user supplies IV
        AES_ENCRYPT or AES_DECRYPT);
```

When IV is non-random need to encrypt it before use (Otherwise, no CPA security!!)

Many-time Key Mode of Operation:
Counter Mode (CTR)

Construction 1: Randomizied CTR

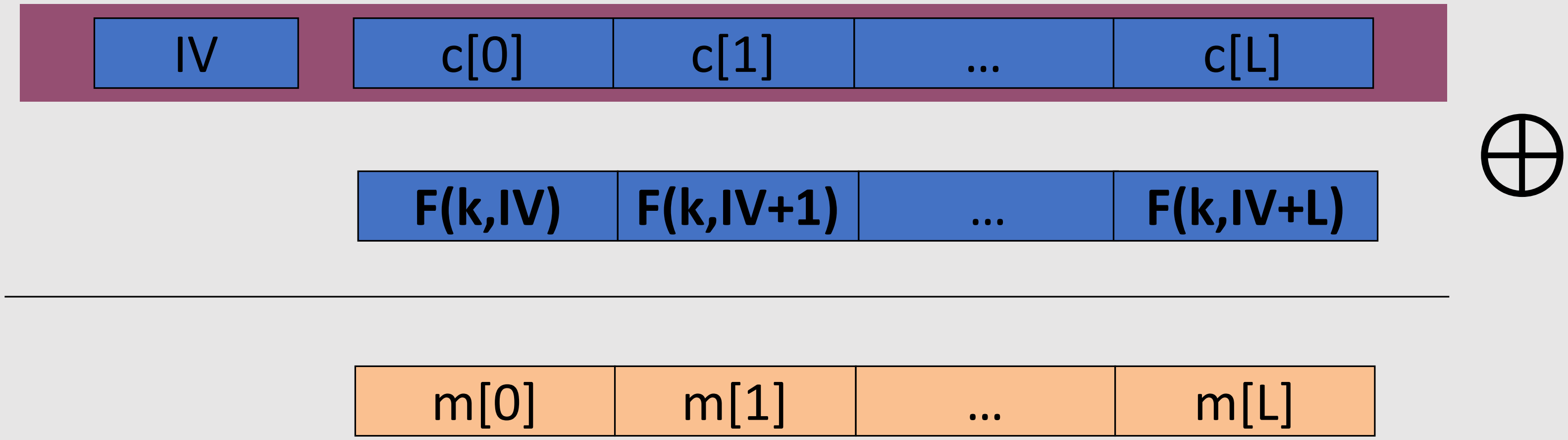
- **PRF** F: $K \times \{0,1\}^n \longrightarrow \{0,1\}^n$
- (Encryption) **E_{CTR}(k,m):** choose **random** IV$\in\{0,1\}^n$ and do:



- **IV – chosen at random for every message**
- **Parallelizable** (unlike CBC)

Construction 1: Randomized CTR

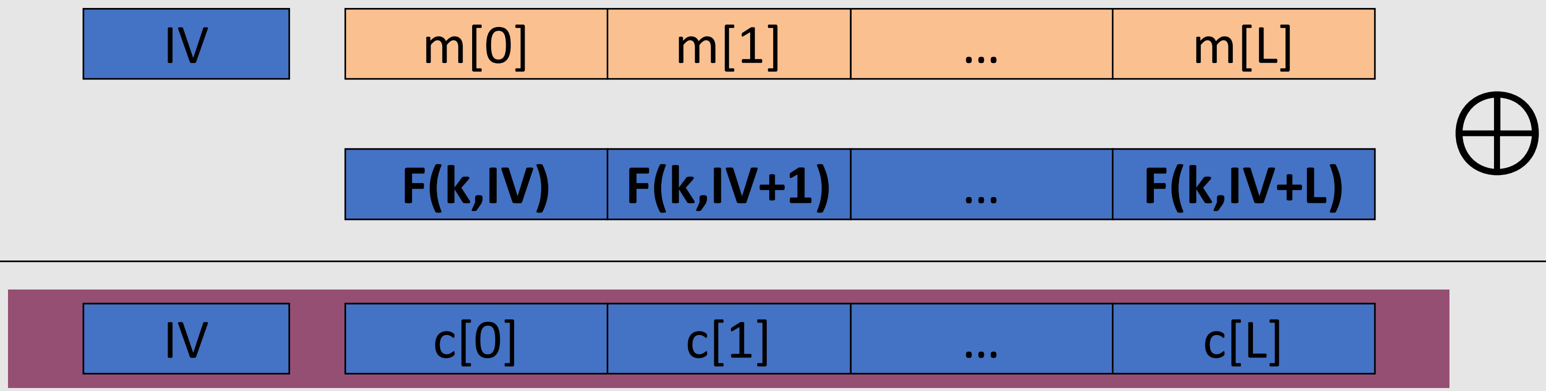
Decryption:



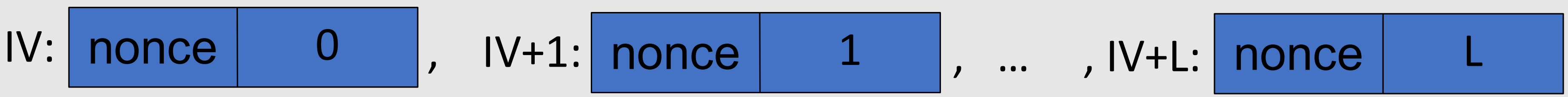
No need to **invert** F for decryption!

Construction 2: Nonce-based CTR

- **PRF** F: K × $\{0,1\}^n \longrightarrow \{0,1\}^n$
- (Encryption) **E_{CTR}(k,m,nonce):**



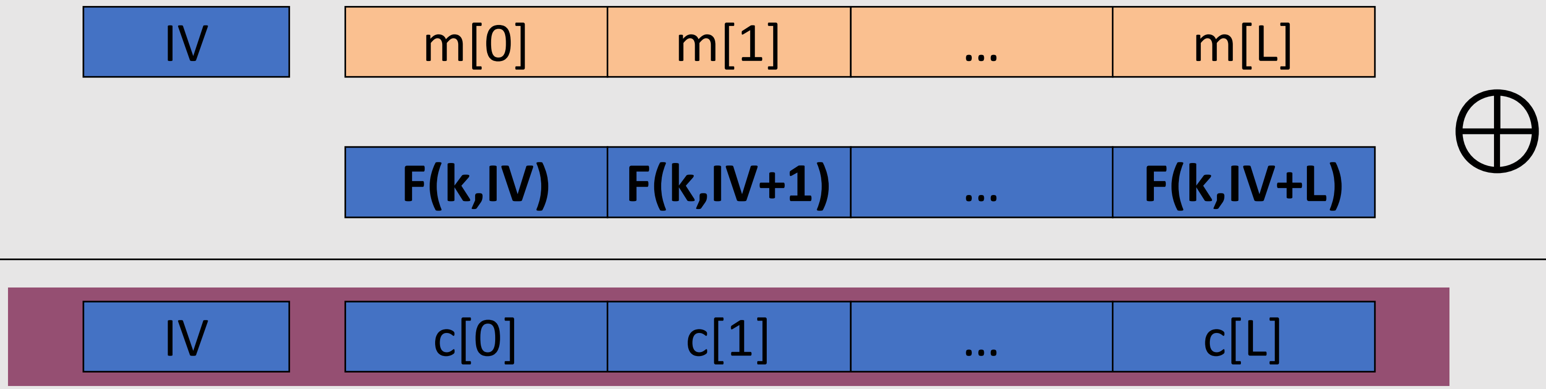
- To ensure F(k,x) is never used more than once, do:



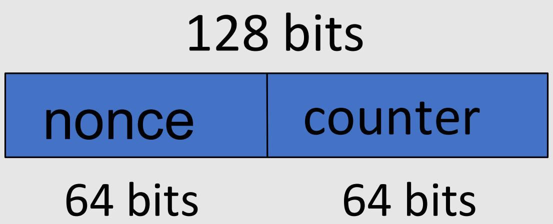
- **nonce** remains the same for the message (**but it varies from msg to msg**)

Construction 2: Nonce-based CTR

- **PRF** F: $K \times \{0,1\}^n \longrightarrow \{0,1\}^n$
- (Encryption) $\mathbf{E_{CTR}(k,m,nonce)}$**:**



- To ensure F(k,x) is never used more than once, do:



- **counter** starts at 0 for every msg and then varies for the different blocks of that message (it increments).
- **nonce** remains the same for that message (but it varies from msg to msg)
- Warning: #blocks of the msg < 2^{64}