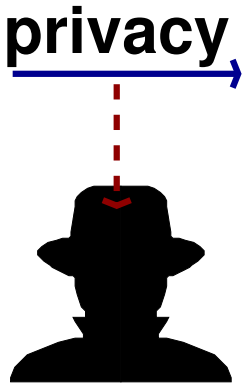
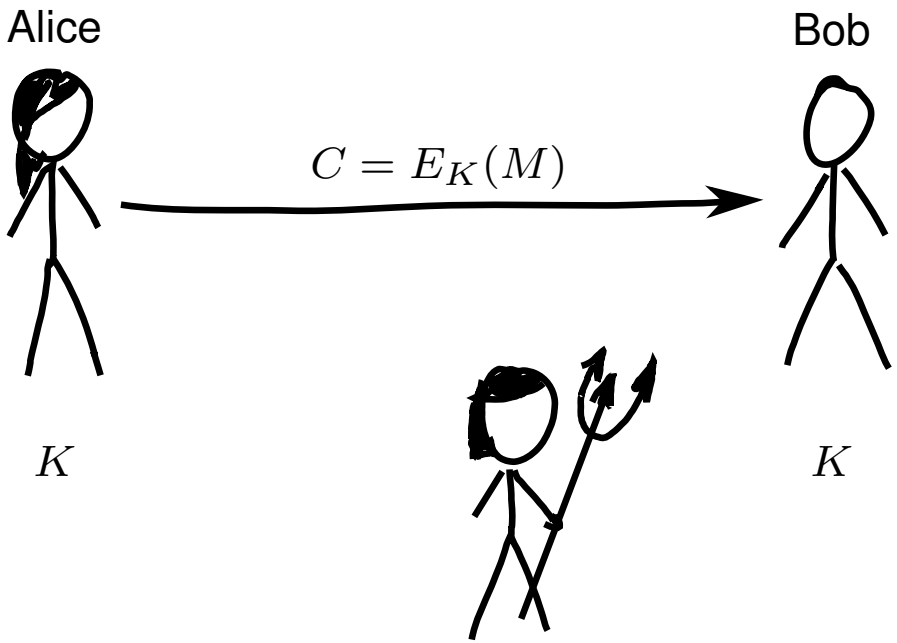
1: Encryption

- Alice sends a message to Bob.
- Alice and Bob know a secret key $\mathbf{K}$.
- Curious Eve does not know $\mathbf{K}$. She would like to find $\mathbf{K}$, or the message, or just *some information* about the message.





Bottom illustration: Christopher Lübbemeier.

Adversary Models

Depending on the adversary model, Eve is allowed to...

- *Known-Ciphertext Attack* (KCA): ... only observe ciphertexts
- *Known-Plaintext Attack* (KPA): ... observe plaintexts and ciphertexts
- *Chosen-Plaintext Attack* (CPA): ... choose plaintexts and ask for their encryptions.
- *Chosen-Ciphertext Attack* (CCA): ... choose plaintexts and ciphertexts (except the one at interest), ask for their encryption/decryption, and observe the ciphertexts/plaintexts

Occasionally, we will also consider some more fancy type of attacks (blockwise adaptive, padding oracle, ...).

Ciphers (1)

- Vernam Cipher (*One-Time Pad*, OTP):
  - Parameter n (message and key length)
  - KEYS $=$ MESSAGES $=$ CIPHERTEXTS $= \{0,1\}^n$
  - Gen: $K := \{0,1\}^n$ return K; $(*$ uniformly at random $*)$
  - $E_K(M) := M \oplus K$
  - $D_K(C) := C \oplus K$
- Simple stream ciphers
  - Inputs: (long) n-bit message M and a (short) k-bit key K
  - Usage: generate n-bit keystream $S(K)$ and use it like a OTP key

Ciphers (2)

- Block ciphers (DES, AES, . . .):
  - Block size n, key size k
  - MESSAGES $=$ CIPHERTEXTS $= \{0,1\}^n$
  - KEYS $= \{0,1\}^k$
  - GEN : $K \xleftarrow{\$} \{0,1\}^k$; return K;

    $X \xleftarrow{\$} \mathcal{X}$ means draw X uniformly at random from $\mathcal{X}$
  - $E_K(M), D_K(C)$:
    - $E_K : \{0,1\}^n \to \{0,1\}^n$ permutes over $\{0,1\}^n$
    - $D_K : \{0,1\}^n \to \{0,1\}^n$ is the inverse permutation
  - Security definition: “mimic a random permutation over $\{0,1\}^n$”

More formal: What Is a “Cipher”?

A cipher is defined by

- Sets:
  - MESSAGES $\subseteq \{0,1\}^*$
  - KEYS $\subseteq \{0,1\}^*$
  - CIPHERTEXTS $\subseteq \{0,1\}^*$
- Functions (efficient algorithms):
  - Key generation: GEN : $\ldots \to$ KEYS
  - Encryption: E : KEYS $\times$ MESSAGES $\to$ CIPHERTEXTS
  - Decryption D : KEYS $\times$ CIPHERTEXTS $\to$ MESSAGES
  - Common Notation: $E_K(\ldots)$, $D_K(\ldots)$ for $E(K,\ldots)$, $D(K,\ldots)$
- For all $K \in$ KEYS and for all $M \in$ MESSAGES holds:

$$D_K(E_K(M)) = M$$

Ciphers Are Not Suited for Privacy

Secure Cipher $\neq$ Secure Encryption Scheme

"Eve would like to find *some information* about the message."

So we try to deny Eve as much of *some information* as possible and feasible:

- Eve is allowed find out the length of the message.
  (Else, we'll have to flood our communication channel with fake messages and artificially long messages.)
- Eve must not find out anything else. Given two ciphertexts of equal length, Eve must not even be able to find out if they would decrypt to the same plaintext.

$\Rightarrow$ **The same message must likely encrypt to many different ciphertexts under the same key!**

Encyption Schemes: Stateful or Random

A good **encryption operation Encr** must be either

- *Stateful*:
  - $C_i := \mathsf{Encr}_K(M)$ depends on previous calls of Encr_K
- or *Probabilistic*:
  - $C_j := \mathsf{Encr}_K(M)$ depends on M, K, and some *coin flips*

The **decryption operation Decr** is **deterministic**. For

$C_1 := \mathsf{Encr}_K(M)$, $C_2 := \mathsf{Encr}_K(M)$, . . . , we always require

$$\mathsf{Decr}_K(C_1) = \mathsf{Decr}_K(C_2) = \cdots = M.$$

What About the Ciphers We Know?

- Vernam cipher ("One-Time Pad") and simple stream ciphers:
  - The key must not be used more than once
  - That makes state and randomness a non-issue
  - If the key is used more than once, attacks are easy anyways
- Block ciphers: deterministic

But, as we will later see, block ciphers (and even stream ciphers) can be excellent building blocks for secure encryption schemes.

Probabilistic Encryption

- Sets:
  - COINS $= \{0,1\}^*$
  - MESSAGES $\subseteq \{0,1\}^*$
  - KEYS $\subseteq \{0,1\}^*$
  - CIPHERTEXTS $\subseteq \{0,1\}^*$
- Functions:
  - GEN : COINS $\to$ KEYS
  - ENCR : KEYS $\times$ MESSAGES $\times$ COINS $\to$ CIPHERTEXTS
  - DECR : KEYS $\times$ CIPHERTEXTS $\to$ MESSAGES
  - Common Notation: $\text{ENCR}_K(\ldots)$, $\text{DECR}_K(\ldots)$
- For all $K \in$ KEYS, $R \in$ COINS, and $M \in$ MESSAGES holds:

$$\text{DECR}_K(\text{ENCR}_K(M,R)) = M.$$

Stateful Encryption

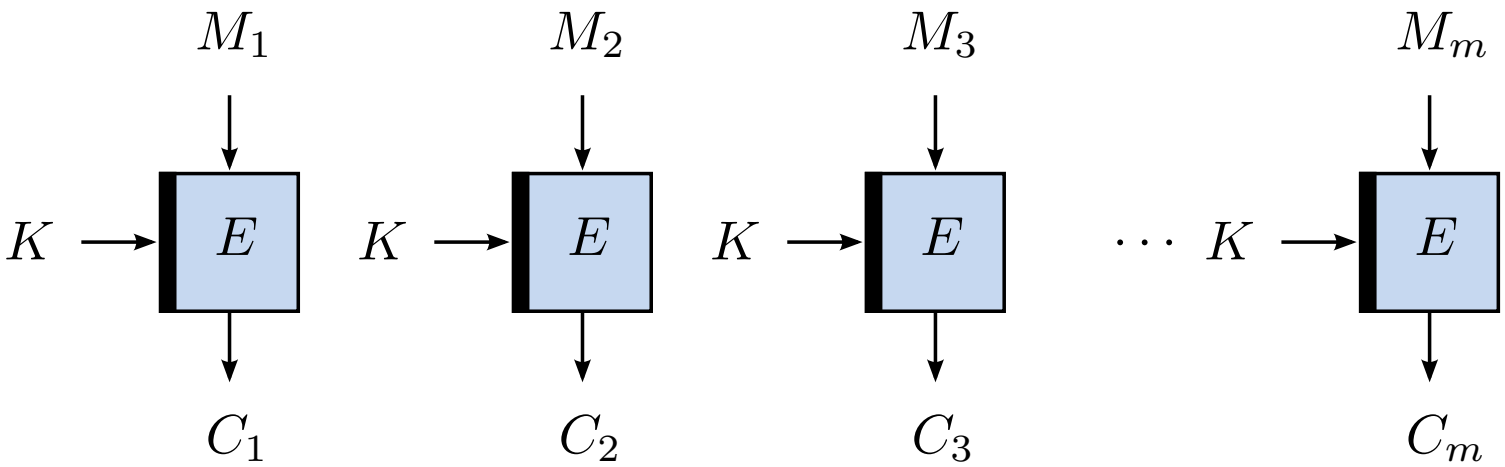
Can Also Be Probabilistic, But Unusually Is Not: COINS $= \{\}$

- Sets:
  - COINS, MESSAGES, KEYS, CIPHERTEXTS as before
  - STATE $\subseteq \{0,1\}^*$
- Functions:
  - GEN, DECR as before
  - ENCR : KEYS $\times$ MESSAGES $\times$ STATE $\times$ COINS $\to$ CIPHERTEXTS $(\times$ STATE$)$
  - Notation: (as before)
- For all $K \in$ KEYS, $R \in$ COINS, $M \in$ MESSAGES, and $\mathbf{S} \in$ STATE :

$$\text{DECR}_K(\text{ENCR}_K(M, \mathbf{S}, R)) = M.$$

1.1: Some Blockcipher Modes for Encryption

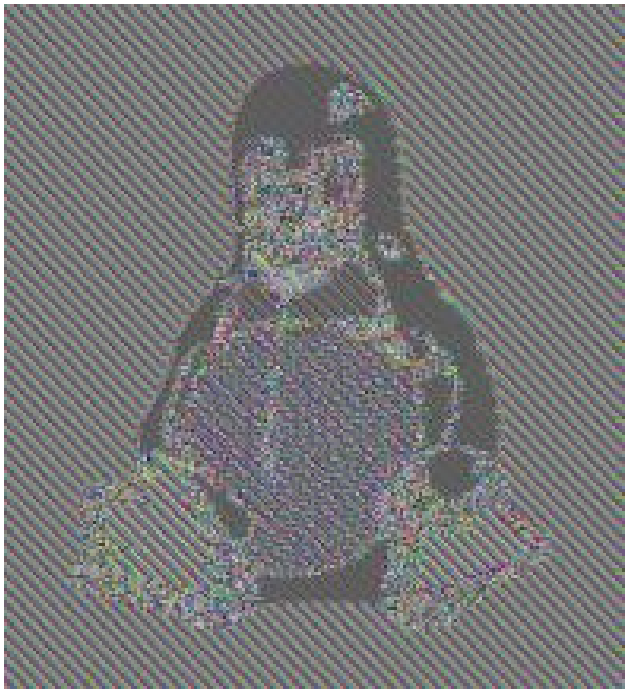
Electronic Codebook (ECB): $C_i \leftarrow E_K(M_i)$



- Conforms to cryptographic standards ("State-of-the-art AES encryption")
- Neither randomized nor stateful $\Longrightarrow$ clearly inappropriate for encryption of more than one message per key
- But even when encrypting a single (long) message, ECB is a security disaster

Weakness of ECB

ECB Encryption:



Plaintext:

Good Encryption:

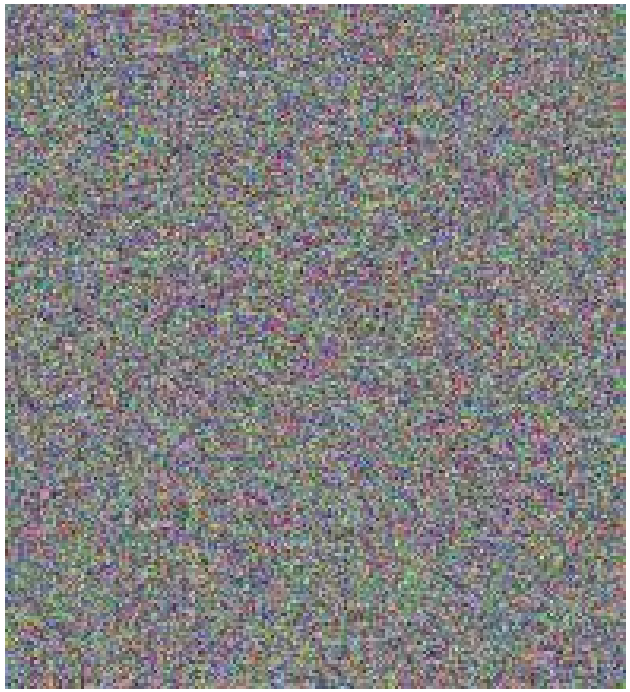
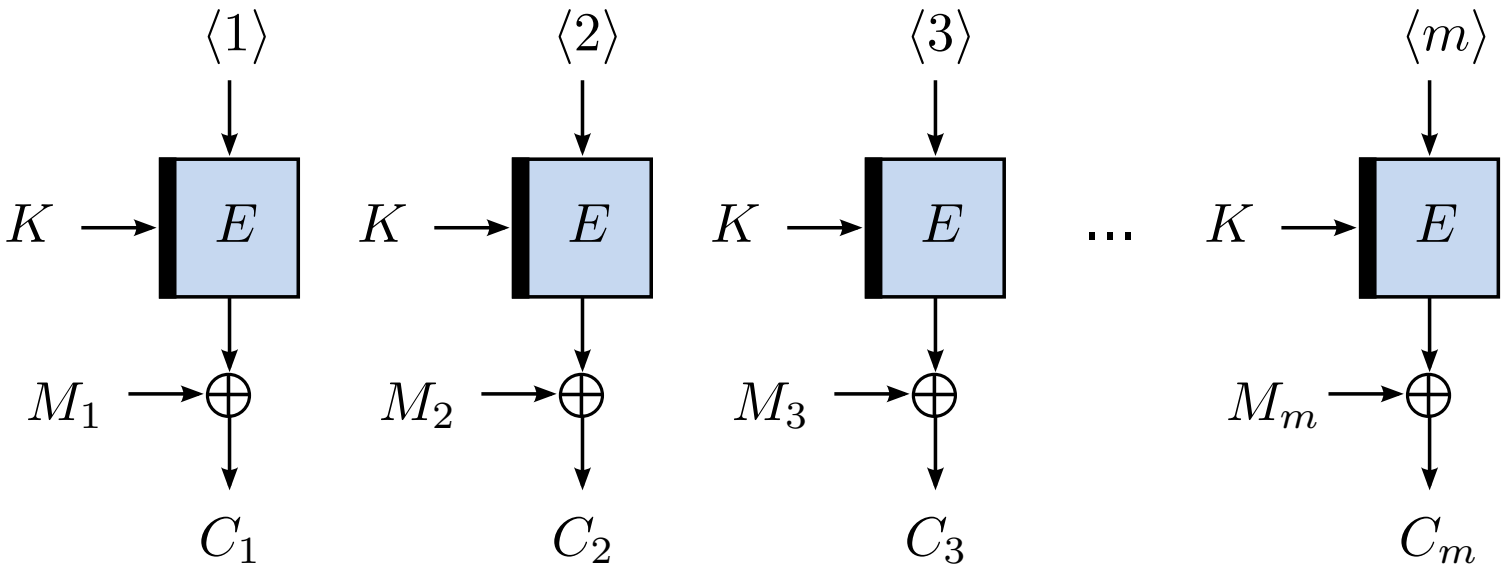


Illustration: `https://de.wikipedia.org/wiki/Electronic_Code_Book_Mode`

Counter Mode

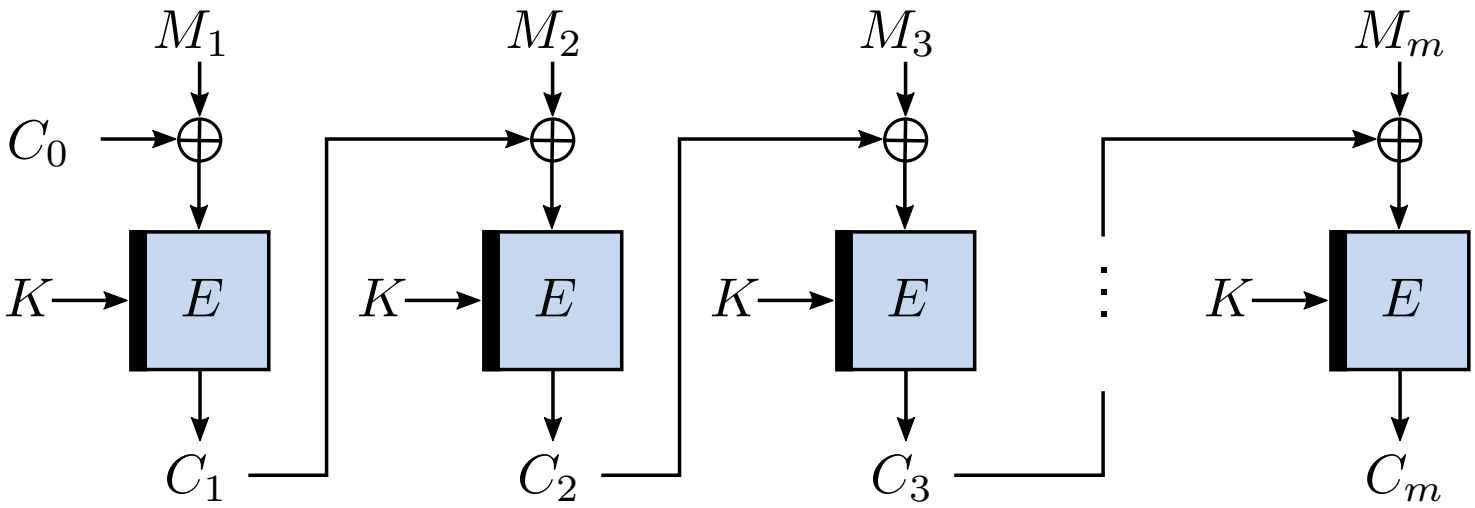
$C_i \leftarrow M_i \oplus E_K(\langle i \rangle)$, $\langle i \rangle$ = Conversion into binary strings



- **How do you decrypt $(C_1, \ldots, C_m)$?**
- Neither randomized nor stateful, but can safely be used to encrypt a single long message ($\ll 2^{n/2}$ blocks) under a fixed key.
- Can be transformed into a stateful or a probabilistic encryption scheme: $C_i := M_i \oplus E_K(\langle i \rangle + R \bmod 2^n)$
- Secure if the counters never overlap and if total #blocks $\ll 2^{n/2}$. (But we still have to prove that – after defining *secure*!)

Cipher Block Chaining (CBC)

$C_i \leftarrow E_K(M_i \oplus C_{i-1})$



Encrypt message $(M_1, \ldots, M_m)$:

1. Choose a random *initialization vector* $C_0 \xleftarrow{\$} \{0,1\}^n$
2. For $1 \le i \le m$: $C_i := E_K(M_i \oplus C_{i-1})$.
3. Return the ciphertext $(C_0, \ldots, C_m)$.

How do you decrypt $(C_0, \ldots, C_m)$?

1.2: Four Security Definitions

For the moment, we focus on *Chosen-Plaintext Attacks* (CPA):

> Eve chooses messages $M_1, M_2, \ldots, M_q$ and receives ciphertexts
>
> $$C_i := E_K(M_i)$$
>
> from an "oracle".

Eve's resources:

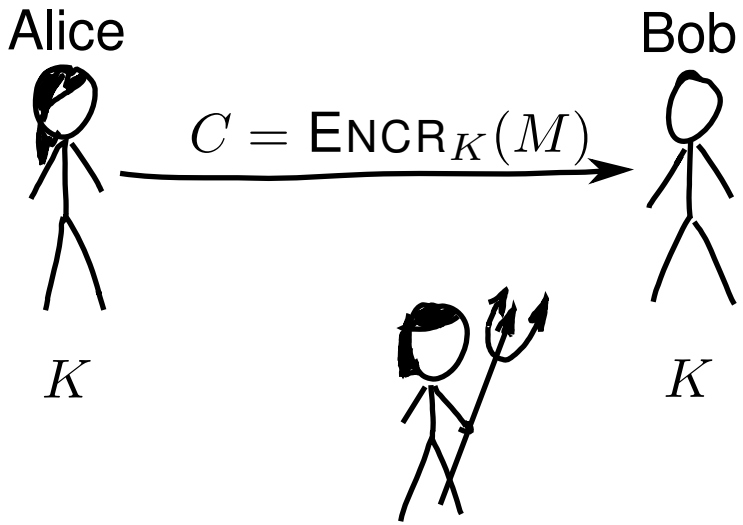
- $q =$ #Oracle queries $M_1, \ldots, M_q$
- $\sigma =$ Total message length (in full blocks)

$$\sigma = \sum_{1 \le i \le q} \lceil |M_i|/n \rceil$$

  (or the product $q \cdot \max\{M_i\}$ as an upper bound)
- $t =$ Run time

Also important: Eve's *success probability* or rather her *advantage*.

Potential Goals of an Adversary



- **Message Recovery:** Given C, recover M.
  Too weak – even partial information about M can be interesting, cf. the ECB case.
- **Key Recovery:** Given C, recover K. *Even weaker; if we find K, we can decrypt, but not necessarily vice versa.*
- We want to prevent that Eve **learns anything useful about** M except for its length, which may be too expensive to hide.

The General Structure of our Security Definitions

- Two parties:
  1. An adversary (*Eve*) who tries to *win* a game,
  2. A neutral oracle, interacting with Eve (representing the system under attack)
- Rules how to play the game:
  - how the oracle initializes itself
  - how the oracle responds to Eve's queries
- A rule to decide if Eve has *won* the game or not

Semantic Security

Oldest (and seemingly most natural) privacy definition
Proposed 1983 by Goldwasser and Micali for Public-Key Cryptography

Eve wins if she can **learn anything useful** about M from C:

- Messages are distributed according to some distribution $\mathcal{M}$.
- The "useful-information" is determined by a function

$$f : \text{MESSAGES} \to \{0, 1\}.$$

- Eve learns something = she can predict the bit $f(D_K(C))$
- The specific choice for $\mathcal{M}$ and f depends on application, user, and the adversaries motivation – not on the cryptosystem.
- When developing a general-purpose cryptosystem, we must be pessimistic:
  - The adversary can choose both $\mathcal{M}$ and f.

Semantic-CPA-Security

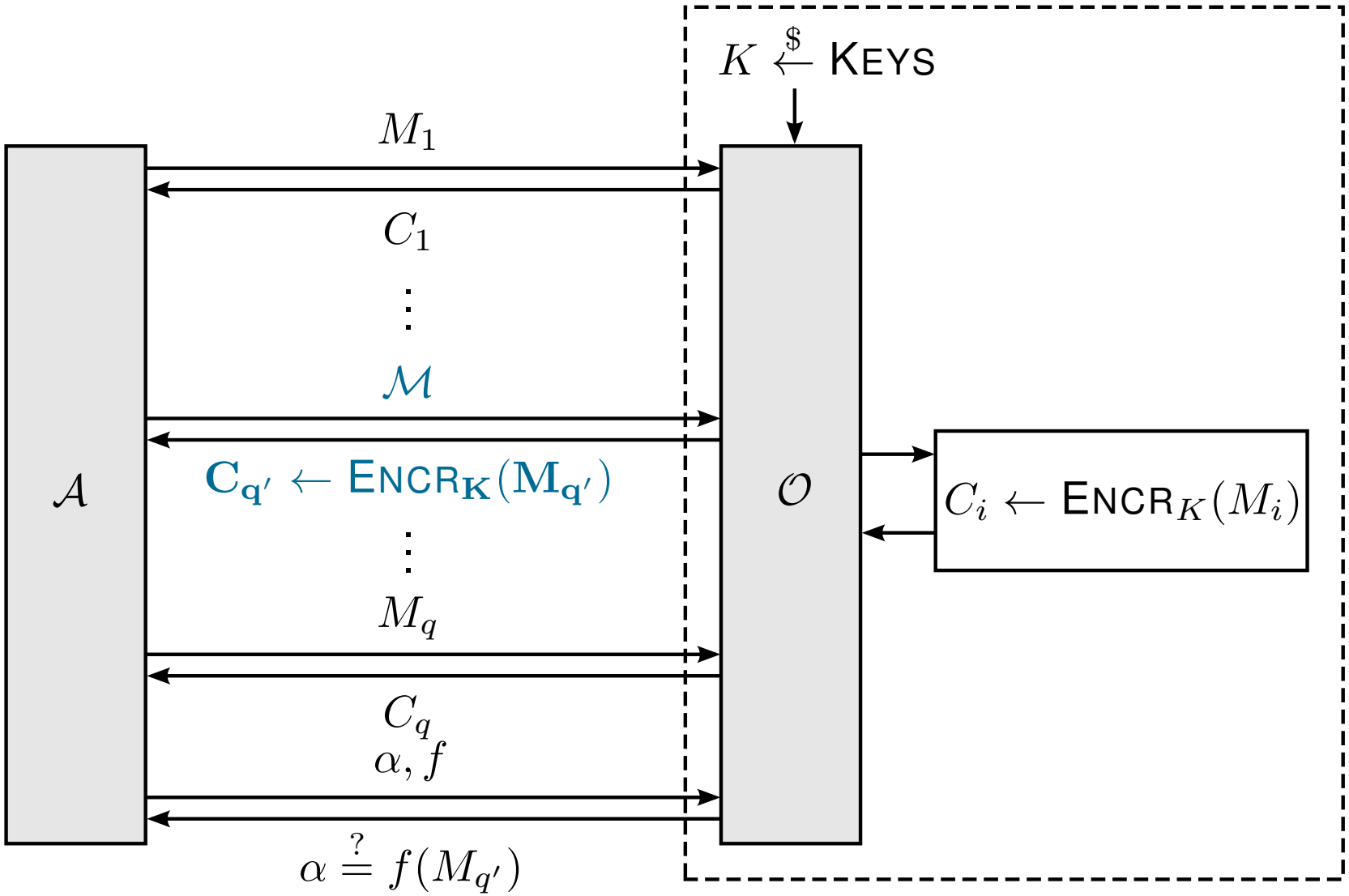
Definition 1 (The Sem-CPA Experiment)

1. The oracle randomly chooses $K \xleftarrow{\$} \text{KEYS}$.
2. For $1 \leq i < q' < q$:

   Eve chooses M_i and asks for its encryption.
   The oracle returns $C_i \leftarrow \text{ENCR}_K(M_i)$.
3. Eve chooses a distribution $\mathcal{M}$ over n-bit plaintexts.
4. The oracle chooses $M_{q'} \in_{\mathcal{M}} \{0,1\}^n$ and responds with $C_{q'} \leftarrow \text{ENCR}_K(M_{q'})$.
5. For $q' + 1 \leq i \leq q$:

   Eve chooses M_i and asks for its encryption.
   The oracle returns $C_i \leftarrow \text{ENCR}_K(M_i)$.
6. Eve outputs a function f and a bit $\alpha \in \{0,1\}$.
   Eve wins iff $\boxed{f(M_{q'}) = \alpha}$.

A Graphical Look at Semantic Security



Note: $\mathcal{O}$ chooses $\mathbf{M_{q'}}$ according to $\mathcal{M}$

Eve's Sem-CPA Advantage

- Note: Eve can easily win with probability 1:
  Set f constant, e.g. $f(x) = 0$ for all x.
- But: We want to measure *how much better* Eve can predict $f(x)$, given the encryption of x.
- Let $x \xleftarrow{\$} \mathcal{M}$ and $K \xleftarrow{\$}$ KEYS. Eve's **SEM-CPA advantage** wrt. ENCR is defined as

$$\mathbf{Adv}^{\text{SEM-CPA}}_{\text{ENCR}}(\mathcal{A}) := \left|\Pr\left[\mathcal{A} \Rightarrow \alpha\right] - \Pr\left[f(M_{q'}) = \alpha\right]\right|,$$

  where $\Pr[\mathcal{A} \Rightarrow \alpha]$ denotes the probability that Eve outputs α in the given scenario.

Issues of the Semantic Security Notion

1. The advantage as the difference between two probabilities – why can't we directly deal with a probability?
2. The adversary is allowed to chose the message distribution $\mathcal{M}$.
3. The adversary is allowed to choose the "useful" function f.

The first issue is difficult to come by – our definition of the advantage is standard for cryptographic security definitions.

But we can overcome the second and third issue by turning our security definition into something "less natural" but simpler.

Find-then-Guess-CPA Security

Definition 2 (FtG-CPA Experiment)

1. The oracle chooses $K \xleftarrow{\$} \textsc{Keys}$ and a bit $\beta \xleftarrow{\$} \{0,1\}$.
2. For $1 \le i < q' < q$:

   Eve chooses M_i and asks for $C_i \leftarrow \textsc{Encr}_K(M_i)$.
3. Eve chooses two messages M^0, M^1 of equal length, $|M^0| = |M^1|$, and sends them to the oracle.
4. The oracle responds $C_{q'} \leftarrow \textsc{Encr}_K(M^\beta)$.
5. For $q' + 1 \le i \le q$:

   Eve chooses M_i and asks for $C_i \leftarrow \textsc{Encr}_K(M_i)$.
6. Eve outputs a guess $\beta' \in \{0,1\}$ for β. She wins iff $\beta' = \beta$.

Let $K \xleftarrow{\$} \textsc{Keys}$. Eve's **FtG-CPA advantage** wrt. $\textsc{Encr}$ is defined as

$$\mathbf{Adv}^{\text{FtG-CPA}}_{\textsc{Encr}}(\mathcal{A}) := \left|\Pr\left[\mathcal{A} \Rightarrow 1 \,|\, \beta = 1\right] - \Pr\left[\mathcal{A} \Rightarrow 1 \,|\, \beta = 0\right]\right| .$$

Left-or-Right-CPA Security

Definition 3 (LoR-CPA Experiment)

1. The oracle chooses $K \xleftarrow{\$} \textsc{Keys}$ and a bit $\beta \xleftarrow{\$} \{0,1\}$.
2. For $1 \le i \le q$:

   Eve chooses two messages M_i^0, M_i^1 of equal length, $|M_i^0| = |M_i^1|$.
   Eve sends M_i^0 and M_i^1 to the oracle.
   The oracle returns $C_i \leftarrow \textsc{Encr}_K(M_i^\beta)$.
3. Eve outputs a bit β'. She wins iff $\beta' = \beta$.

Let $K \xleftarrow{\$} \textsc{Keys}$. Eve's **LoR-CPA advantage** wrt. $\textsc{Encr}$ is defined as

$$\mathbf{Adv}^{\text{LoR-CPA}}_{\textsc{Encr}}(\mathcal{A}) := \left|\Pr\left[\mathcal{A} \Rightarrow 1 \,|\, \beta = 1\right] - \Pr\left[\mathcal{A} \Rightarrow 1 \,|\, \beta = 0\right]\right| .$$

Real-or-Random-CPA Security

Definition 4 (RoR-CPA Experiment)

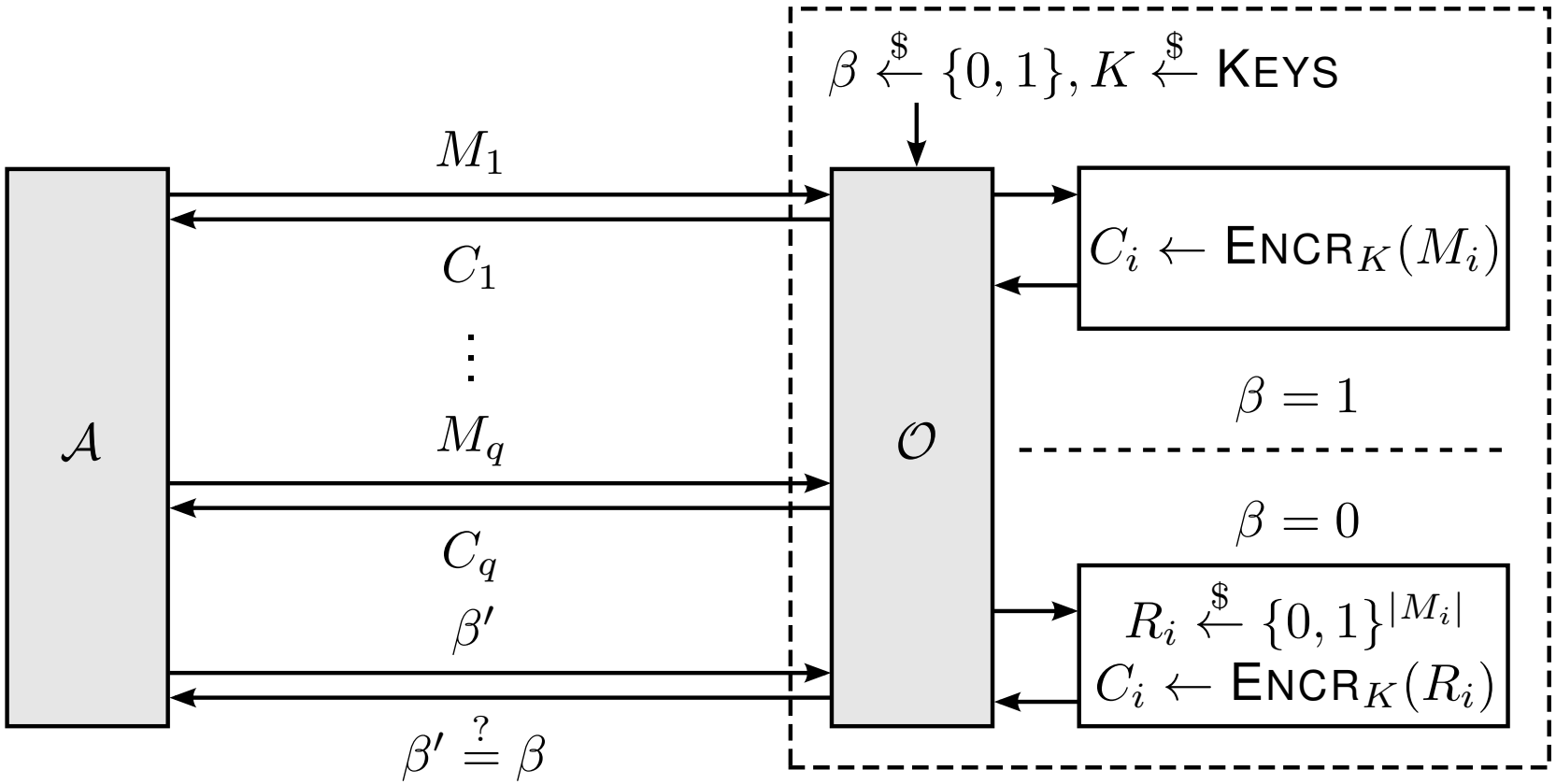
1. The oracle chooses $K \xleftarrow{\$} \textsc{Keys}$ and a bit $\beta \xleftarrow{\$} \{0,1\}$.
2. For $1 \le i \le q$:

   Eve chooses M_i and sends it to the oracle.
   If $\beta = 1$, the oracle returns $C_i \leftarrow \textsc{Encr}_K(M_i)$.
   Else it chooses $R_i \xleftarrow{\$} \{0,1\}^{|M_i|}$ and returns $C_i \leftarrow \textsc{Encr}_K(R_i)$.
3. Eve outputs a bit β'. She wins iff $\beta' = \beta$.

Let $K \xleftarrow{\$} \textsc{Keys}$. Eve's **RoR-CPA advantage** wrt. $\textsc{Encr}$ is defined as

$$\mathbf{Adv}^{\text{RoR-CPA}}_{\textsc{Encr}}(\mathcal{A}) := \left|\Pr\left[\mathcal{A} \Rightarrow 1 \,|\, \beta = 1\right] - \Pr\left[\mathcal{A} \Rightarrow 1 \,|\, \beta = 0\right]\right| .$$

A Graphical Look at the RoR-CPA Experiment



Security Parameters

Definition 5 (Security Parameters)

Let $\Pi = (\textsc{Gen}, \textsc{Encr}, \textsc{Decr})$ denote an encryption scheme.
Let $\mathbf{I} \in \{\textsc{Sem}\text{-CPA}, \textsc{FtG}\text{-CPA}, \textsc{LoR}\text{-CPA}, \textsc{RoR}\text{-CPA}\}$ be a security notion.

We say that Π is $(\mathbf{t}, \mathbf{q}, \sigma, \mathbf{a})$**-secure wrt.** $\mathbf{I}$ if and only if there exists **no efficient adversary** $\mathcal{A}$ which

- runs in **time at most** $\mathbf{t}$,
- asks **at most** $\mathbf{q}$ **queries** with a
- **total message length of at most** σ **blocks**, and
- achieves an **advantage greater than** $\mathbf{a}$.

Relations among Security Definitions (1)

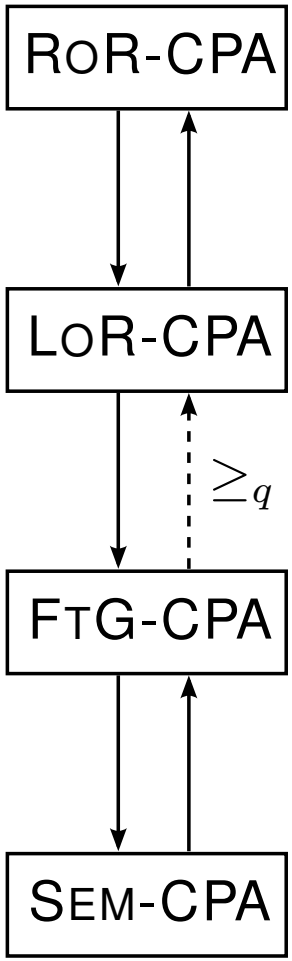
Let $\mathbf{I}, \mathbf{J} \in \{\text{SEM-CPA}, \text{FTG-CPA}, \text{LOR-CPA}, \text{ROR-CPA}\}$ be two security definitions.

- *I is not weaker than J*: There exists a constant $c > 1$ such that any $(\mathbf{t}, \mathbf{q}, \sigma, \mathbf{a})$-secure encryption scheme wrt. $\mathbf{I}$ is at least $(\frac{\mathbf{t}}{\mathbf{c}}, \frac{\mathbf{q}}{\mathbf{c}}, \frac{\sigma}{\mathbf{c}}, \mathbf{ac})$-secure wrt. $\mathbf{J}$. We write $\mathbf{I} \geq \mathbf{J}$.
- *I and J are of equivalent strength*: $\mathbf{I} \geq \mathbf{J}$ and $\mathbf{J} \geq \mathbf{I}$. We write $\mathbf{I} \equiv \mathbf{J}$.
- *I and J are not of equivalent strength*: We write $\mathbf{I} \not\equiv \mathbf{J}$.
- *J is at most q times weaker than I:* There exists a constant $c > 1$, such that any $(\mathbf{t}, \mathbf{q}, \sigma, \mathbf{a})$-secure encryption scheme wrt. $\mathbf{J}$, is at least $(\frac{\mathbf{t}}{\mathbf{c}}, \frac{\mathbf{q}}{\mathbf{c}}, \frac{\sigma}{\mathbf{c}}, \mathbf{acq})$-secure wrt. $\mathbf{I}$. We write $\mathbf{I} \geq_{\mathbf{q}} \mathbf{J}$.

Relations among Security Definitions (2)

Theorem 6 (Bellare, Desai, Jokiph, Rogaway (1997))

1. RoR-CPA $\equiv$ LoR-CPA
2. Sem-CPA $\equiv$ FtG-CPA
3. FtG-CPA $\not\equiv$ LoR-CPA
4. LoR-CPA $\geq$ FtG-CPA
5. LoR-CPA $\geq_q$ FtG-CPA



1.3: Analyzing Block Cipher Modes of Operation

Security of Block Ciphers

- Given a block cipher $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$
- Let $\mathcal{P}_n = \{\pi \mid \pi : \{0,1\}^n \to \{0,1\}^n\}$ be the set of all n-bit permutations
- A secure block cipher should not be **efficiently distinguishable from a random permutation** $\pi \xleftarrow{\$} \mathcal{P}_n$
- Note: π can be efficiently instantiated by lazy sampling

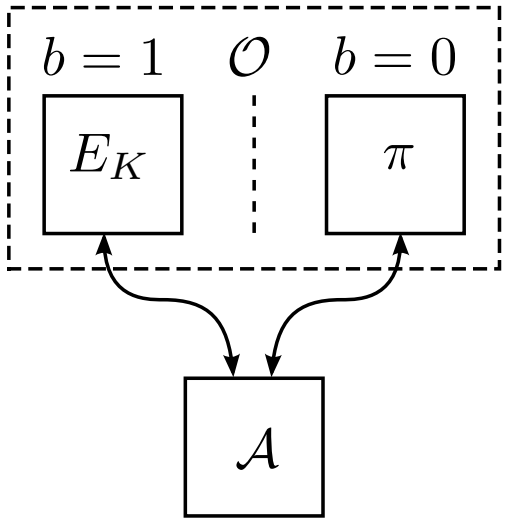
Random Permutation $\pi(X)$**:**

1: **if** $L[X] = \perp$ **then**
2: $\quad Y \xleftarrow{\$} \{0,1\}^n \setminus \mathcal{R}$
3: $\quad L[X] \leftarrow Y$
4: $\quad \mathcal{R} \leftarrow \mathcal{R} \cup \{Y\}$
5: **return** $L[X]$

PRP Security of Block Ciphers

Definition 7 (PRP Experiment)

1. The oracle chooses $K \xleftarrow{\$} \text{KEYS}$ and a bit $\beta \xleftarrow{\$} \{0,1\}$.
2. For $1 \le i \le q$:
   - Eve chooses M_i and sends it to the oracle.
   - If $\beta = 1$, the oracle returns $C_i \leftarrow E_K(M_i)$ and $C_i \leftarrow \pi_K(M_i)$ otherwise.
3. Eve outputs a bit β'. She wins iff $\beta' = \beta$.

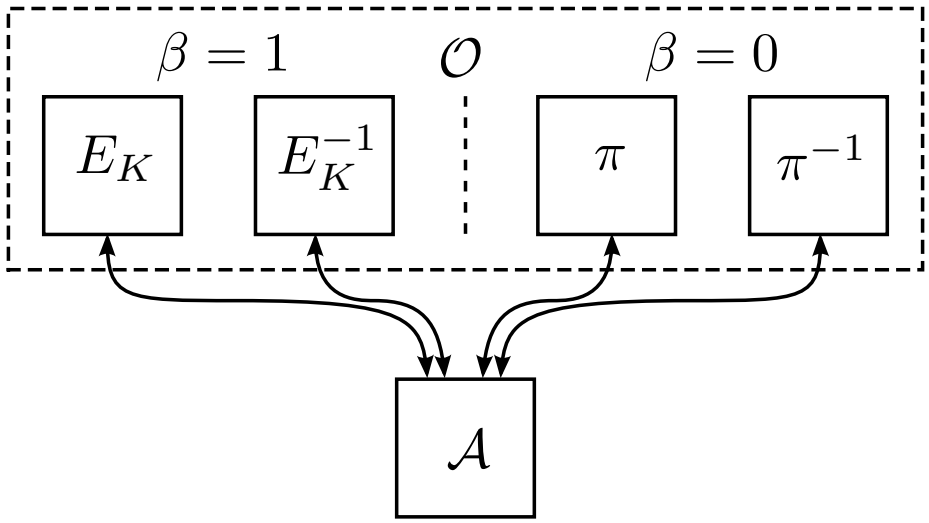


Let $K \xleftarrow{\$} \text{KEYS}$. Eve's **PRP advantage** wrt. E is defined as

$$\mathbf{Adv}_E^{\text{PRP}}(\mathcal{A}) := \left|\Pr\left[\mathcal{A}^{E_K} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\pi} \Rightarrow 1\right]\right|.$$

SPRP Security of Block Ciphers

- Eve has also oracle access to a decryption oracle $E_K^{-1}(\cdot)$ or $\pi^{-1}(\cdot)$, respectively
- There exist more specific attack models (e. g. related key, ...)



Let $K \xleftarrow{\$} \text{KEYS}$. Eve's **SPRP advantage** wrt. E, E^{-1} is defined as

$$\mathbf{Adv}^{\text{SPRP}}_{E,E^{-1}}(\mathcal{A}) := \left| \Pr\left[\mathcal{A}^{E_K, E_K^{-1}} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\pi, \pi^{-1}} \Rightarrow 1\right] \right| .$$

Random Functions

- Let $\mathcal{F}_{m,n} = \{f \mid f : \{0,1\}^m \to \{0,1\}^n\}$ be the set of all functions which map m-bit inputs to n-bit outputs
- We call $f \xleftarrow{\$} \mathcal{F}_{n,n}$ an n-bit random function
- Note: f can also be efficiently implemented by lazy sampling

Random Function $f(X)$:

1: **if** $L[X] = \perp$ **then**
2: $\quad Y \xleftarrow{\$} \{0,1\}^n$
3: $\quad L[X] \leftarrow Y$
4: **return** $L[X]$

Random Permutation P(X):

1: **if** $L[X] = \perp$ **then**
2: $\quad Y \xleftarrow{\$} \{0,1\}^n \setminus \mathcal{R}$
3: $\quad L[X] \leftarrow Y$
4: $\quad \mathcal{R} \leftarrow \mathcal{R} \cup \{Y\}$
5: **return** $L[X]$

Distinguishing between PRP and PRF

Definition 8 (PRP/PRF Distinguishing Experiment)

1. The oracle chooses a random function $\rho \xleftarrow{\$} \mathcal{F}_{n,n}$, a random permutation $\pi \xleftarrow{\$} \mathcal{P}_n$, and a bit $\beta \xleftarrow{\$} \{0,1\}$.
2. For $1 \leq i \leq q$:
   - Eve chooses M_i and sends it to the oracle.
   - If $\beta = 1$, the oracle returns $\pi(M_i)$ and $\rho(M_i)$ otherwise.
3. Eve outputs a bit β'. She wins iff $\beta' = \beta$.

The PRP-PRF-Switching Lemma

Lemma 9 (PRP-PRF-Switching Lemma)

Let $\mathcal{A}$, ρ, π, and integers $n, q \geq 1$ be defined as on the previous slide. Then, the advantage of $\mathcal{A}$ is upper bounded by

$$|\Pr[\mathcal{A}^{\rho} \Rightarrow 1] - \Pr[\mathcal{A}^{\pi} \Rightarrow 1]| \leq \frac{q(q-1)}{2 \cdot 2^n} < \frac{q^2}{2^{n+1}}.$$

So, if $q \ll 2^{n/2}$, $\mathcal{A}$ cannot distinguish ρ from π with significant advantage.

PRP-PRF-Switching Lemma

Core Ideas of the Proof

- A random permutation π is a random function with a constraint:

$$\text{For all distinct } x_i, x_j \in \{0,1\}^n : \pi(x_i) \neq \pi(x_j)$$

- Consider the probability for an unconstrained random function ρ to violate the constraint by making q distinct queries $x_1, \ldots, x_q$:

$$\Pr[\exists i < j \leq q : \rho(x_i) = \rho(x_j)].$$

- There are $\binom{q}{2}$ pairs x_i, x_j
- All outputs of ρ are chosen independently uniformly at random
- Each pair can collide with probability $1/2^n$:

$$\Pr[\exists i < j \leq q : \rho(x_i) = \rho(x_j)] \leq \binom{q}{2} \cdot \frac{1}{2^n} = \frac{q(q-1)}{2^{n+1}}.$$

RoR-CPA-Security of Random-IV Counter Mode

Random-IV Counter Mode;

- Plaintext: $M^i = (M^i_1, \ldots, M^i_{m_i})$
- Initial Value: $R^i \xleftarrow{\$} \{0,1\}^n$
- Keystream: $X^i_j = E_K(R^i + \langle j-1 \rangle \bmod 2^n)$
- Encryption: $C^i_j = X^i_j \oplus M^i_j$ for $1 \le i \le q$ and $1 \le j \le m_i$
- Ciphertext: $(R^i, C^i_1, \ldots, C^i_{m_i})$

Theorem 10 (RoR-CPA-Security of Random-IV Counter Mode)

Let CTR$[\pi]$ *denote random-IV counter mode, using an ideal* n*-bit permutation* $\pi \xleftarrow{\$} \mathcal{P}_n$ *internally. Then,* CTR$[\pi]$ *is* (∞, q, σ, a)-RoR-CPA*-secure with* $q \le \sigma$ *and* $a \le \sigma^2/2^n$*.*

RoR-CPA-Security of Random-IV Counter Mode

Proof Sketch:

- Replace π by a random function $\rho \xleftarrow{\$} \mathcal{F}_{n,n}$. The advantage for distinguishing both settings is upper bounded by

$$\sigma^2/(2 \cdot 2^n).$$

- The advantage to distinguish the key streams from σ indep. random n-bit blocks is at most the probability that the key streams overlap:

$$\Pr\left[\exists i, j, k, \ell : i \neq j, N^i + j = N^k + \ell\right] \leq \sigma^2/(2 \cdot 2^n).$$

- If the keystreams do not overlap, the message is XORed with independent random blocks, like the one-timepad. The advantage to distinguish between the ciphertext and random bits is 0.
- Add the advantages:

$$\sigma^2/(2 \cdot 2^n) + \sigma^2/(2 \cdot 2^n) = \sigma^2/2^n.$$

RoR-CPA-Security of Random-IV CBC

Theorem 11

Let $\pi \xleftarrow{\$} \mathcal{P}_n$. *Let* CBC$[\pi]$ *denote* CBC *which uses* π *internally with a random IV* C_0. *Then,* CBC$[\pi]$ *is* (∞, q, σ, a)-RoR-CPA*-secure with* $q \le \sigma, a \le \sigma^2/2^n$.

Proof Sketch:

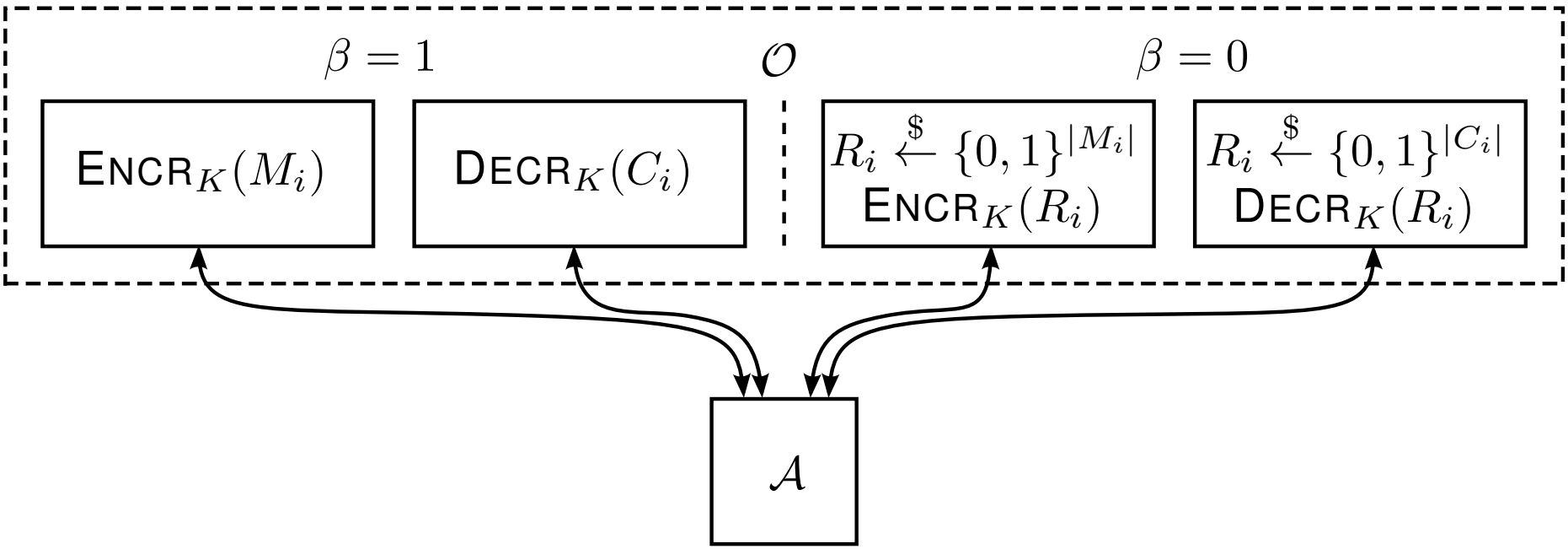
- Replace π by a random function $\rho \xleftarrow{\$} \mathcal{F}_{n,n}$. The advantage of distinguishing between both settings is upper bounded by $\sigma^2/(2 \cdot 2^n)$.
- Starting with random C_0^i, the C_{j-1}^i are independent random values, except when a collision occurs:

$$M_j^i \oplus C_{j-1}^i = M_\ell^k \oplus C_{\ell-1}^k.$$

- The probability for such a collision is at most $\sigma^2/(2 \cdot 2^n)$.
- Finally, add both advantages: $\sigma^2/(2 \cdot 2^n) + \sigma^2/(2 \cdot 2^n) = \sigma^2/2^n$.

Chosen-Ciphertext Security

- $\beta = 1$: Eve sees **real** encryption *and* decryption
- $\beta = 0$: Eve sees only **random** encryptions *and* decryptions
- Eve is not allowed to query
  - C_i's or M_i's from earlier queries



RoR-CCA Security

Definition 12 (RoR-CCA Experiment)

1. The oracle chooses $K \xleftarrow{\$} \text{KEYS}$ and a bit $\beta \xleftarrow{\$} \{0,1\}$.
2. For $1 \le i \le q$:
   - **Encryption query:** Eve chooses $M_i \notin \{M_1, \ldots, M_{i-1}\}$. If $\beta = 1$, the oracle returns $C_i \leftarrow \text{ENCR}_K(M_i)$ and $C_i \leftarrow \text{ENCR}_K(R_i)$ for $R_i \xleftarrow{\$} \{0,1\}^{|M_i|}$ otherwise.
   - **Decryption query:** Eve chooses $C_i \notin \{C_1, \ldots, C_{i-1}\}$. If $\beta = 1$, the oracle returns $M_i \leftarrow \text{DECR}_K(C_i)$ and $M_i \leftarrow \text{ENCR}_K(R_i)$ for $R_i \xleftarrow{\$} \{0,1\}^{|C_i|}$ otherwise.
3. Eve outputs a bit β'. She wins iff $\beta' = \beta$.

Let $K \xleftarrow{\$} \{0,1\}^k$. Eve's **RoR-CCA advantage** wrt. ENCR is defined as

$$\mathbf{Adv}^{\text{RoR-CCA}}_{\text{ENCR},\text{ENCR}^{-1}}(\mathcal{A}) := \left|\Pr\left[\mathcal{A} \Rightarrow 1 | \beta = 1\right] - \Pr\left[\mathcal{A} \Rightarrow 1 | \beta = 0\right]\right| .$$

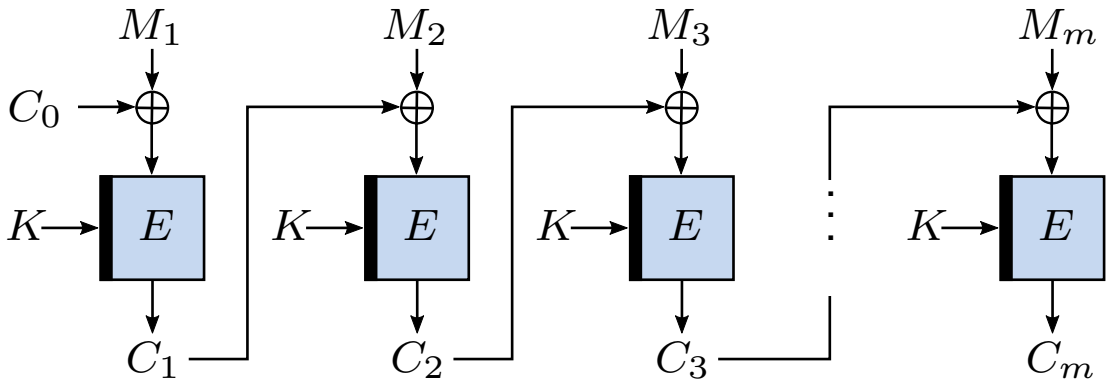
Why Should We Care About CCA-Security?

- **This attack model appears completely absurd!**
- **If Bob is so foolish to decrypt messages for Eve, then we cannot help him, anyway!**

Cryptographers believe the CPA model is too weak for practical purposes.

- Sometimes, it is hard *not* to leak any information about plaintexts.
- If something is “wrong” with the plaintext, you would like to send an error message.
- Even if you don’t, Eve may observe if you had found something wrong or not (e.g., by the timing of sending your next message).
- This kind of leakage is very difficult to prevent.
- So: better assume the adversary does learn *whatever* about the plaintexts you read.

Properties of CBC Encryption



- The plaintext length *must* be a multiple of the block size n for example: $n = 128$ bits for the AES
- Plaintexts whose length not a multiple of 16 will be **padded** to the next multiple of n.
- CBC Ciphertexts are *malleable*
  - Cutting the last i ciphertext blocks the plaintext will be the same – except for the last i plaintext blocks.
  - If you flip any bit in C_0, you flip the corresponding bit in M_1.

1.4: Nonces: Secure Encryption as Stateless and Deterministic Functions

. . . kind of

- Sets:
  - NONCES $\subseteq \{0,1\}^*$
  - MESSAGES $\subseteq \{0,1\}^*$
  - KEYS $\subseteq \{0,1\}^*$
  - CIPHERTEXTS $\subseteq \{0,1\}^*$
- Functions:
  - Gen: $\ldots \to$ KEYS
  - Encr: NONCES $\times$ KEYS $\times$ MESSAGES $\to$ CIPHERTEXTS
  - Decr: NONCES $\times$ KEYS $\times$ CIPHERTEXTS $\to$ MESSAGES
  - write as AEncr$_K(\ldots)$, ADecr$_K(\ldots)$
- For all $K \in$ KEYS, $N \in$ NONCES, and $M \in$ MESSAGES holds:

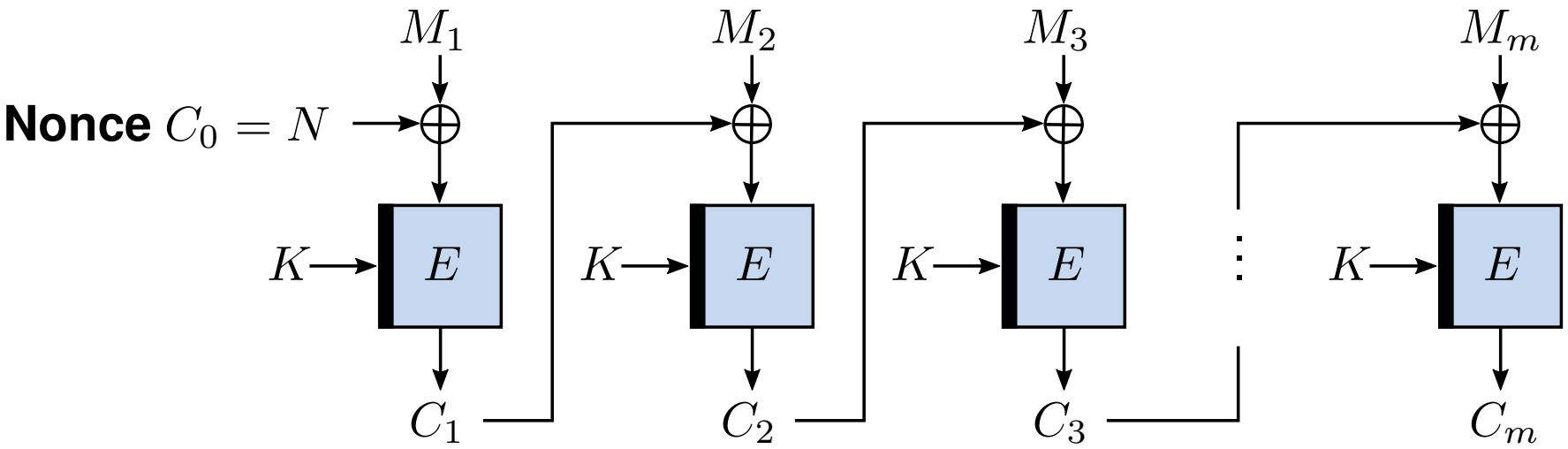
$$\text{ADECR}_K(N, \text{AENCR}_K(N, M)) = M.$$

Nonce-Respecting Adversaries

- A **nonce** represents the state or the randomness of an Encryption scheme.
- It enables the application programmer to decide how to store the state or how to generate the random bits.
- The application programmer *must* ensure that the nonce really reflects some persistent (during the life time of the key) state, or is really random:

  Nonces must not repeat!
- This is formally modelled by assuming a **nonce-respecting** adversary: Eve can choose arbitrary nonces, but all encryption queries must have different nonces.

Example: Nonce-Based CBC-Encryption



- Why is it a bad idea to choose $C_0 := N$ as the nonce?
- Let L denote another secret key, independent of K.
  Why is $C_0 \leftarrow E_L(N)$ secure?
- How secure is $C_0 \leftarrow E_K(N)$, i.e., the same idea without an additional secret key?

1.5: Padding

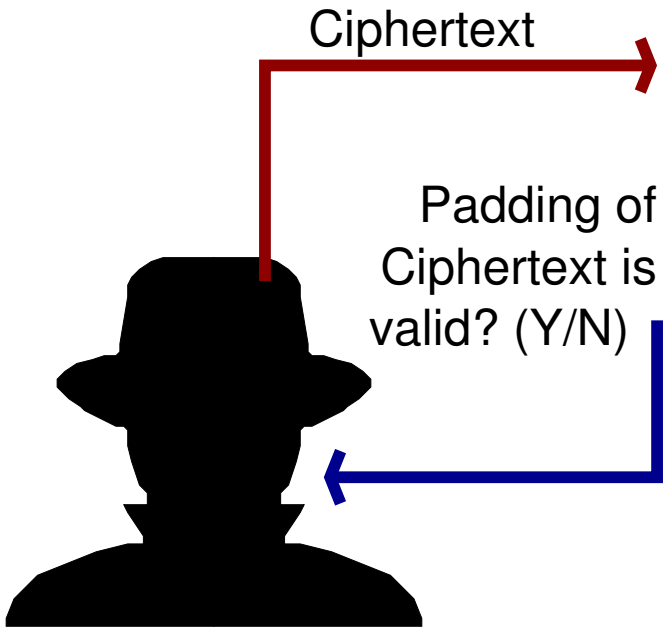
- Necessary in CBC (and many other modes) for handling messages whose length is not a multiple of n
- Padding must be injective $\Longrightarrow$ Decryption must be able to recover the message before padding
- For messages whose length is a multiple of n, the padding adds an additional block
- In practice: Most messages are byte streams (multiples of 8 bits)
- Practical padding schemes (for example):

Standard	**Padding Scheme**	**Example**
PKCS5, SSL/TLS, IPSec	$\mathbf{MMM\ldots Mnnn\ldots n}$	"Hello world" 55555
ESP	$\mathbf{MMM\ldots M1234\ldots n}$	"Hello world" 12345
XML Encr.	$\mathbf{MMM\ldots MRR\ldots Rn}$	"Hello world" ????5

$\mathbf{M}$: Message byte, $\mathbf{R}$: Random byte, $\mathbf{n}$: #Padded bytes.

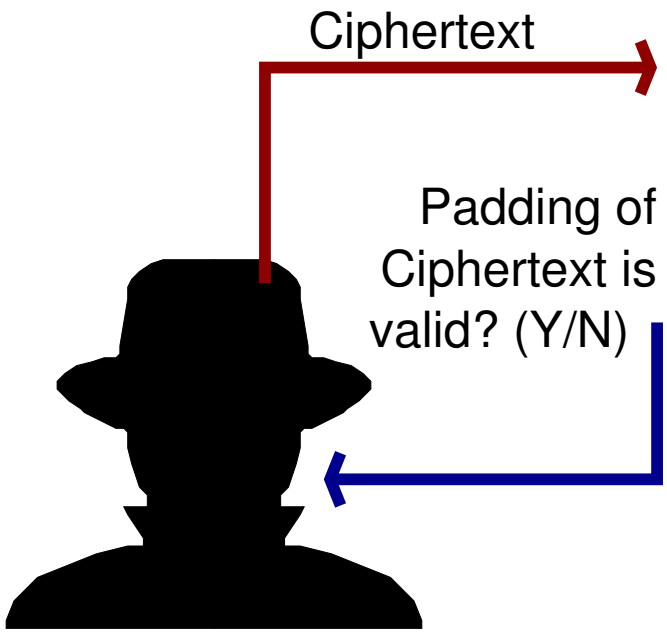
Padding-Oracle Attacks

In the past, the adversary could often learn information about the validity of her ciphertexts from side channels (e. g. error messages, server-response duration, . . .).



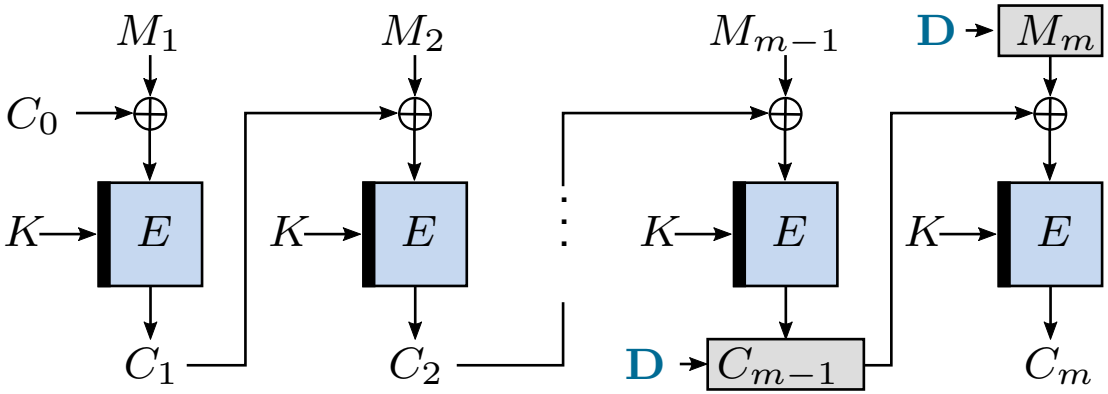
Attacking MMM...Mnnn...n

TLS and Friends



- System: AES-CBC-encryption (1 block = 16 bytes)
- Known: Ciphertext $(C_0, \ldots, C_m)$
- Goal: Recover the original plaintext $(M_1, \ldots, M_m)$

The Padding-Oracle Attack on CBC



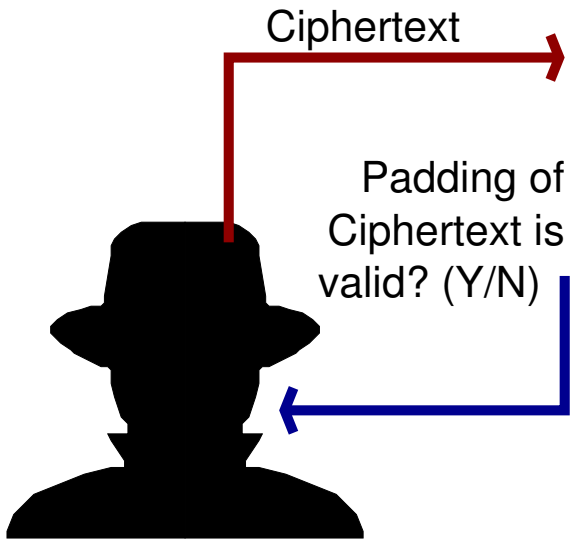
1: **for all** Blocks i from $m-1$ downto 0 **do**
2: $D := (D^{15}, \ldots, D^0) = (0, \ldots, 0)$
3: **for all** Bytes j from 0 to 15 **do**
4: **for** v from 0 to 255 **do**
5: Compute Byte $\mathbf{D}^j := \mathbf{v} \oplus (\mathbf{j}+\mathbf{1})$
6: Ask for the decryption of $C' := (C_0, \ldots, C_{i-1}, \mathbf{C_i} \oplus \mathbf{D}, C_{i+1})$
7: **if** C' is deemed *valid* **then**
8: Store byte $M^j_{i+1} := v$
9: For all $k \in \{0, \ldots, j\}$: $D^k := M^j_{i+1} \oplus (j+1) \oplus (j+2)$
10: Guess next byte (**goto** 3)
11: **return** The recovered plaintext $M = (M_1, \ldots, M_m)$

Assignment

- Describe a padding-oracle attack against XML Encryption Padding, i.e., for the padding scheme

$$\mathbf{MMM}\ldots\mathbf{MRR}\ldots\mathbf{Rn}$$

- Can the adversary find out the entire plaintext? Or does the attack stop when the first Plaintext block M_1 has been decrypted?



- Hint: Any bracket “<” must be closed by “>”. Assume the padding oracle to complain if the last byte of the message isn’t “>”.

Final Remarks

- Designing and implementing sound and secure **Encryption** is harder than it looks.
- Even after decades of research, new flaws are found.
- Some padding-oracle attacks are quite practical.
- Their discovery is relatively new.
- Even when performing authenticated encryption, where chosen-ciphertexts are rejected with high probability, timings may provide a practical padding oracle to the adversary.