# Contents

# Chapter 1

# Private Key Encryption

## 1.1 Stronger Security Notions

지금까지 우리는 상대가 정직한 당사자들 사이에 전송되는 하나의 암호문을 수동적으로만 도청하는 보안에 대한 비교적 약한 정의를 고려해왔습니다. 여기서 우리는 더 강력한 보안 개념을 고려합니다.

### Chosen-Plaintext Attacks and CPA-Security

공식적인 정의에서 우리는 공격자 $\mathcal{A}$가 선택한 메시지를 $\mathcal{A}$가 알 수 없는 키 $k$를 사용하여 암호화하는 블랙박스로 간주되는 암호화 오라클에 대한 접근 권한을 $\mathcal{A}$에게 부여하여 COA를 모델링합니다. 즉, $\mathcal{A}$가 암호화 $\mathsf{Enc}_k$에 접근할 수 있다고 상상합니다. $\mathcal{A}$가 이 오라클에 메시지 $m$을 입력으로 제공하여 질의(query)하면, 오라클은 암호문 $\mathsf{Enc}_k(m)$를 응답으로 반환합니다. $\mathcal{A}$는 원하는 횟수만큼 암호화 오라클과 상호 작용할 수 있습니다.

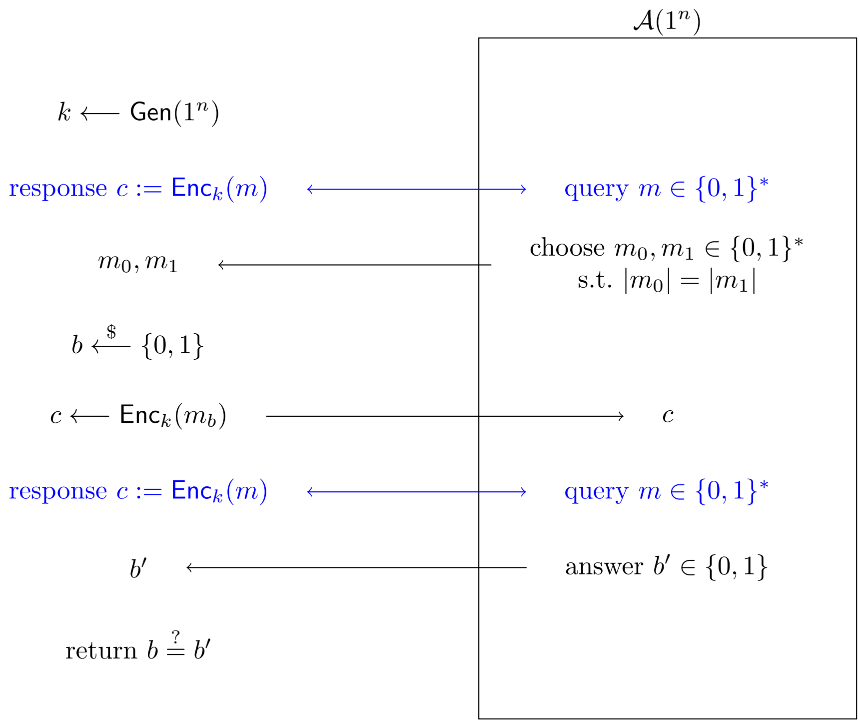


Figure 1.1: CPA indistinguishability experiment of private-key encryption scheme

**Definition 1.1.1.** A private-key encryption scheme $\Pi$ has **indistinguishable encryptions under a chosen-plaintext attack**, or is **CPA-secure**, if for all PPT distinguishers $\mathcal{A}$ there is a negligible function $\varepsilon$ such that

$$\Pr\left[\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\Pi}(n) = 1\right] \leq \frac{1}{2} + \varepsilon(n). \tag{1.1}$$

where the probability is taken over the randomness used by $\mathcal{A}$, as well as the randomness used in the experiment.

이 정의에서 사용한 부등식은 다음과 같이 바꿔서 사용할 수 있습니다. 증명은 EAV-secure에서 사용한 증명과 비슷하므로 생략합니다.

$$|\Pr[\mathcal{A}(1^n) = 1 \mid b = 0] - \Pr[\mathcal{A}(1^n) = 1 \mid b = 1]| \leq \varepsilon(n). \tag{1.2}$$

CPA-secure는 오늘날 암호화 체계가 만족해야 할 보안의 최소한의 개념이지만, 더 강력한 보안 개념(CCA)을 요구하는 것이 점점 더 일반화되고 있습니다.

### CPA-Security for Multiple Encryptions

CPA-secure 정의는 다중 암호화의 경우로 확장할 수 있습니다. 여기서는 암호화할 평문 쌍을 선택할 수 있는 공격자를 모델링할 수 있는 다소 단순하고 장점이 있는 다른 접근 방식을 취합니다. 특히, 공격자에게 동일한 길이의 메시지 $m_0, m_1$을 입력하면 암호문 $\mathsf{Enc}k(m_b)$를 계산하고 반환하는 좌우 오라클 $\mathrm{LR}_{k,b}$에 대한 액세스 권한을 부여합니다.

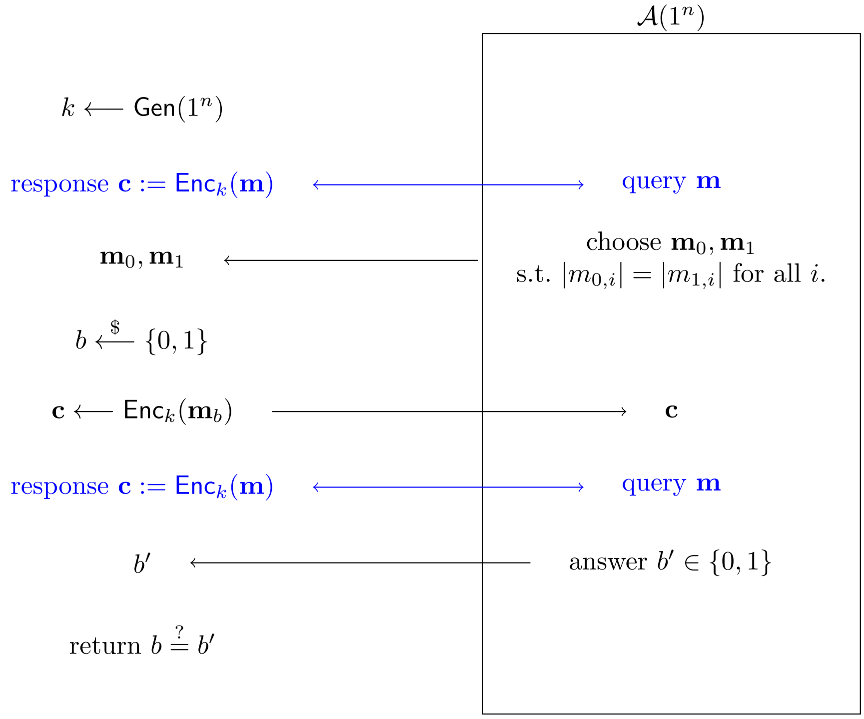


Figure 1.2: LR-CPA indistinguishability experiment of private-key encryption scheme

**Definition 1.1.2.** Private-key encryption scheme $\Pi$ has **indistinguishable multiple encryptions under a chosen-plaintext attack**, or is **CPA-secure for multiple encryptions**, if for all PPT distinguishers $\mathcal{A}$ there is a negligible function $\varepsilon$ such that

$$a \tag{1.3}$$

where the probability is taken over the randomness used by $\mathcal{A}$ and the randomness used in the experiment.

이 정의에서 사용한 부등식은 IND-PASS의 정의와 비슷한 방법으로 다음과 같이 바꿔서 사용할 수 있습니다.

$$\left|\Pr\left[\mathcal{A}(1^n) = 1 \mid b = 0\right] - \Pr\left[\mathcal{A}(1^n) = 1 \mid b = 1\right]\right| \leq \varepsilon(n). \tag{1.4}$$

다음의 정리 1.1.1에 의해, 어떤 개인 키 암호 체계 $\Pi$가 단일 암호화의 경우 CPA-secure하다는 것을 증명하는 것으로 그 체계가 다중 암호화의 경우에도 CPA-secure하다는 결론을 내릴 수 있습니다.

**Theorem 1.1.1.** Any private-key encryption scheme that is CPA-secure is also CPA-secure for multiple encryptions.

*Proof.* $\Pi$이 다중 IND-CPA secure하지 않다고 가정합니다. 즉, 공격자 $\mathcal{A}$에 대해, 다음을 만족하는 negligible 함수 $\varepsilon$이 존재합니다. (대우 증명)

$$\left|\Pr\left[\mathcal{A}(1^n) = 1 \mid b = 0\right] - \Pr\left[\mathcal{A}(1^n) = 1 \mid b = 1\right]\right| > \varepsilon(n). \tag{1.5}$$

$t = t(n)$을 구분자 $\mathcal{A}$가 실험 1.3에서 암호화 오라클에 질의하는 최대 다항 횟수라고 합시다. 키 $k$와 $1 \leq i \leq t$를 만족하는 $i$에 대해, $i = t$이라면 $\mathcal{A}$관점에서 볼 때 이 실험은 $b = 0$일 때의 다중 IND-CPA 실험과 같고, $i = 0$이라면 $b = 1$일 때의 다중 IND-CPA 실험과 같습니다. 따라서 처음의 식 1.5를 다음과 같이 바꿀 수 있습니다.

$$\left|\Pr\left[\mathcal{A}(1^n) = 1 \mid \mathbf{c} = (c_{0,0}, \cdots, c_{0,t-1})\right] - \Pr\left[\mathcal{A}(1^n) = 1 \mid \mathbf{c} = (c_{1,0}, \cdots, c_{1,t-1})\right]\right| > \varepsilon(n). \tag{1.6}$$

$$
\begin{array}{ccccc}
\text{challenger } \mathcal{C} & \xleftrightarrow{\textsf{IND-CPA}} & \text{adversary } \mathcal{B} & \xleftrightarrow[\textsf{IND-CPA}]{\text{(multiple)}} & \text{adversary } \mathcal{A} \\
k \longleftarrow \textsf{Gen}(1^n) & & & & \\
 & & i \xleftarrow{\$} \{1, \cdots, t\} & & \\
c_j \longleftarrow \textsf{Enc}_k(m_j) & \xleftrightarrow[\text{response } c_j]{\text{query } m_j} & \mathbf{m} & \xleftrightarrow[\text{response } \mathbf{c}]{\text{query } \mathbf{m}} & \text{choose } \mathbf{m} \\
 & & \mathbf{m}_0, \mathbf{m}_1 & \longleftarrow & \text{choose } \mathbf{m}_0, \mathbf{m}_1 \\
c_j \longleftarrow \textsf{Enc}_k(m_{0,j}) & \xleftrightarrow[\text{response } c_j]{\text{query } m_{0,j}} & m_{0,j}\ (j < i) & & \\
m_{0,i}, m_{1,i} & \longleftarrow & m_{0,i}, m_{1,i}\ (j = i) & & \\
b \xleftarrow{\$} \{0,1\} & & & & \\
c_i \longleftarrow \textsf{Enc}_k(m_{b,i}) & \longrightarrow & c_i & & \\
c_j \longleftarrow \textsf{Enc}_k(m_{1,j}) & \xleftrightarrow[\text{response } c_j]{\text{query } m_{1,j}} & m_{1,j}\ (j > i) & & \\
 & & \mathbf{c} = (c_{0,1}, \cdots, c_{0,i-1}, c_i, c_{1,i+1}, \cdots, c_{1,t}) & \longrightarrow & \mathbf{c} \\
c_j \longleftarrow \textsf{Enc}_k(m_j) & \xleftrightarrow[\text{response } c_j]{\text{query } m_j} & \mathbf{m} & \xleftrightarrow[\text{response } \mathbf{c}]{\text{query } \mathbf{m}} & \text{choose } \mathbf{m} \\
b' & \longleftarrow & b' & \longleftarrow & \text{choose } b' \in \{0,1\} \\
\text{return } b \overset{?}{=} b' & & & &
\end{array}
$$

Figure 1.3: A CPA-secure experiment with $\mathcal{A}$ and $\mathcal{A}'$

실험 1.3에서, $b = 0$인 경우 $\mathcal{B}$은 어떤 $i = i^*$에 대해 $i^* \geq j$라면 $m_{j,0}$의 암호문을, $i^* < j$라면 $m_{j,1}$의 암호문을 $\mathcal{A}$에게 전달하게 됩니다. 따라서 다음이 성립합니다.

$$
\begin{aligned}
\Pr[\mathcal{B}(1^n) = 1 \mid b = 0] &= \sum_{i^*=1}^{t} \Pr[\mathcal{B}(1^n) = 1 \mid b = 0 \wedge i = i^*] \cdot \Pr[i = i^*] \\
&= \frac{1}{t} \cdot \sum_{i^*=1}^{t} \Pr[\mathcal{A}(1^n) = 1 \mid \mathbf{c} = (\cdots, c_{0,i^*-1}, c_{0,i^*}, c_{1,i^*+1}, \cdots)]
\end{aligned} \tag{1.7}
$$

$$
\begin{aligned}
\Pr[\mathcal{B}(1^n) = 1 \mid b = 1] &= \sum_{i^*=1}^{t} \Pr[\mathcal{B}(1^n) = 1 \mid b = 1 \wedge i = i^*] \cdot \Pr[i = i^*] \\
&= \frac{1}{t} \cdot \sum_{i^*=1}^{t} \Pr[\mathcal{A}(1^n) = 1 \mid \mathbf{c} = (\cdots, c_{0,i^*-1}, c_{1,i^*}, c_{1,i^*+1}, \cdots)] \\
&= \frac{1}{t} \cdot \sum_{i^*=0}^{t-1} \Pr[\mathcal{A}(1^n) = 1 \mid \mathbf{c} = (\cdots, c_{0,i^*}, c_{1,i^*+1}, c_{1,i^*+2}, \cdots)]
\end{aligned} \tag{1.8}
$$

식 1.7과 1.8에 의해, 다음이 성립합니다.

$$
\begin{aligned}
&\left|\Pr\left[\mathcal{B}(1^n) = 1 \mid b = 0\right] - \Pr\left[\mathcal{B}(1^n) = 1 \mid b = 1\right]\right| \\
&= \frac{1}{t} \cdot \left| \sum_{i^*=1}^{t} \Pr\left[\mathcal{A}(1^n) = 1 \mid \mathbf{c} = (\cdots, c_{0,i^*}, c_{1,i^*+1}, c_{1,i^*+2}, \cdots)\right] - \sum_{i^*=0}^{t-1} \Pr\left[\mathcal{A}(1^n) = 1 \mid \mathbf{c} = (\cdots, c_{0,i^*-1}, c_{0,i^*}, c_{1,i^*+1}, \cdots)\right] \right| \\
&= \frac{1}{t} \cdot \left|\Pr\left[\mathcal{A}(1^n) = 1 \mid b = 0\right] - \Pr\left[\mathcal{A}(1^n) = 1 \mid b = 1\right]\right| > \varepsilon'(n).
\end{aligned}
\tag{1.9}
$$

$t$는 다항식이므로, $\left|\Pr\left[\mathcal{A}'(1^n) = 1 \mid b = 0\right] - \Pr\left[\mathcal{A}'(1^n) = 1 \mid b = 1\right]\right|$는 negligible하지 않습니다. 따라서 $\Pi$는 $\mathcal{A}'$에 대해 CPA-secure하지 않습니다. 결론적으로 $\Pi$이 CPA-secure하다면, 다중 암호화의 경우에도 CPA-secure합니다. □

## 1.2 Constructing a CPA-Secure Encryption Scheme

이 장에서는 Pseudorandom Function(PRF)과 Pseudorandom Permutation(PRP)을 정의합니다.

### Pseudorandom Functions

균일한 키 $k \in \{0,1\}^n$에 대한 효율적이고 결정론적인 키 함수 $F_k : \{0,1\}^{l_{in}} \to \{0,1\}^{l_{out}}$가, 동일한 정의역과 치역을 가지는 모든 함수의 집합 $\mathsf{Func}_l$에서 균일하게 임의로 선택된 함수 $f$와 구별할 수 없는 경우, $F_k$를 PRF라고 합니다. 공식적인 정의를 위해 다음과 같이 실험 1.4를 구성합니다. 이 실험에서 오라클 $\mathcal{O}$은 결정론적입니다.

$$
\begin{array}{lcl}
\text{challenger } \mathcal{C} & \xleftrightarrow{\text{PRF experiment}} & \text{adversary } \mathcal{A} \\[2ex]
k \longleftarrow \mathsf{Gen}(1^n) & \xrightarrow{\hspace{8em}} & 1^n \\[2ex]
b \overset{\$}{\longleftarrow} \{0,1\} & & \\[4ex]
\begin{array}{l} \text{if } b = 1 \\ \quad y \longleftarrow F_k(x) \\ \text{otherwise} \\ \quad y \longleftarrow f(x) \end{array} & \underset{\text{response } y}{\overset{\text{query } x}{\longleftrightarrow}} & \text{choose } x \\[4ex]
b' & \xleftarrow{\hspace{8em}} & \text{choose } b' \in \{0,1\} \\[2ex]
\text{return } b \overset{?}{=} b' & &
\end{array}
$$

Figure 1.4: Indistinguishability experiment for a PRF

$\mathsf{Func}_l$에서 $f$를 균등하게 뽑는 과정은 효율적이지 않습니다. 따라서 적절한 실험 구성을 위해 알고리듬 1와 같이 on-the-fly 방식으로 $f$를 구현합니다.

**Algorithm 1** Random Function (on-the-fly)

**Input:** $x$
**Output:** $y$

```
procedure f(x)
    if ∃(x, y') ∈ Tab then
        y ⟵ y'
    else if ∄(x, y') ∈ Tab then
        y ⟵$ {0,1}^n
        Tab ⟵ Tab ∪ (x, y)
    end if
    return y
end procedure
```

실험 1.4을 바탕으로 PRF를 정의합니다.

**Definition 1.2.1.** An efficient, keyed function $F_k : \{0,1\}^{l_{in}} \to \{0,1\}^{l_{out}}$ is a **pseudorandom function** if for all PPT distinguishers $\mathcal{A}$, there is a negligible function $\varepsilon$ such that:

$$\Pr\left[\mathrm{Exp}^{\mathsf{PRF}}_{\mathcal{A},F_k}(n) = 1\right] \le \frac{1}{2} + \varepsilon(n). \tag{1.10}$$

명제 **??**와 비슷한 방법으로, 식 1.10을 다음과 같이 바꿀 수 있습니다.

$$\left|\Pr\left[\mathcal{A}^{F_k}(1^n)=1\right]-\Pr\left[\mathcal{A}^{f}(1^n)=1\right]\right|\le\varepsilon(n). \tag{1.11}$$

## Pseudorandom Permutations

$P_k$가 PRF이면서 전단사 함수라면, 우리는 $P_k$를 PRP라고 합니다. 즉, 균일한 키 $k\in\{0,1\}^n$에 대한 효율적이고 결정론적인 키 함수 $P_k:\{0,1\}^l\to\{0,1\}^l$가, 동일한 정의역과 치역을 가지는 모든 일대일 함수(순열)의 집합 $\mathsf{Perm}_l$에서 균일하게 임의로 선택된 함수 $p$와 구별할 수 없는 경우, $P_k$를 PRP라고 합니다. PRP의 정의 및 실험은 PRF와 유사하므로 생략하며, $p$를 on-the-fly 방식으로 구현한 알고리듬만 소개합니다.

**Algorithm 2** Random Permutation (on-the-fly)

**Input:** $x$
**Output:** $y$

**procedure** $p(x)$
  **if** $\exists(x,y')\in\mathsf{Tab}$ **then**
    $y\longleftarrow y'$
  **else if** $\nexists(x,y')\in\mathsf{Tab}$ **then**
    **while** $\nexists(x',y)\in\mathsf{Tab}$ **do**
      $y\xleftarrow{\$}\{0,1\}^n$
    **end while**
    $\mathsf{Tab}\longleftarrow\mathsf{Tab}\cup(x,y)$
  **end if**
  **return** $y$
**end procedure**

$P_k$가 키 순열이라면 $P_k$를 기반으로 한 암호 체계는 $P_k$ 자체를 계산하는 것 외에도 $P_k^{-1}$를 계산해야 할 수도 있습니다. 이는 잠재적으로 새로운 보안 문제를 야기할 수 있습니다. 특히, 이제는 구분자에게 역 순열에 대한 오라클 접근 권한이 추가로 주어져도 $P_k$와 $p$을 구별할 수 없다는 더 강력한 요구 사항을 부과해야 할 수도 있습니다. $F_k$가 이러한 속성을 가지고 있다면, 우리는 이를 strong PRP(SPRP)이라고 부릅니다. 즉, 다음의 식을 만족하는 negligible 함수 $\varepsilon$가 존재한다면, $P_k$는 SPRP입니다.

$$\left|\Pr\left[\mathcal{A}^{P_k,P_k^{-1}}(1^n)=1\right]-\Pr\left[\mathcal{A}^{p,p^{-1}}(1^n)=1\right]\right|\le\varepsilon(n). \tag{1.12}$$

## 1.3 Modes of Operation and Encryption

### Stream Ciphers and Stream-Cipher Modes of Operation

Formally, a stream cipher is a pair of deterministic algorithms $(\mathsf{Init}, \mathsf{Next})$ where:

- $\mathsf{Init}$ takes as input a seed sand an optional initialization vector $IV$, and outputs some initial state $\mathsf{st}$.
- $\mathsf{Next}$ takes as input a current state $\mathsf{st}$ and outputs a bit $y$ along with updated state $\mathsf{st}'$.

We define an algorithm $\mathsf{GetBits}$ that takes as input an initial state $\mathsf{st}_0$ and a desired output length $1^l$ and then does:

- For $i = 1$ to $l$, compute $(y_i, \mathsf{st}_i) := \mathsf{Next}(\mathsf{st}_{i-1})$.
- Return the $l$-bit string $y = y_1 \cdots y_l$ as well as the final state $\mathsf{st}_l$.

We let $\mathsf{GetBits}_1$ be the algorithm that runs $\mathsf{GetBits}$ and only returns its initial output (namely, the $l$-bit string $y$).

Given a stream cipher and a parameter $l = l(n) > n$, we may define the deterministic function $G^l$ as

$$G^l(s) := \mathsf{GetBits}_1(\mathsf{Init}(s), 1^l). \tag{1.13}$$

Then the stream cipher is secure if $G^l$ is a pseudorandom generator for any polynomial $l$.

Security for a stream cipher that does take an $IV$ can be defined in multiple ways. Given a stream cipher where $\mathsf{Init}$ takes an $n$-bit $IV$ and a parameter $l = l(n)$, we may define the keyed function $F^l$ as

$$F(s, IV) = \mathsf{GetBits}_1(\mathsf{Init}(s, IV), 1^l). \tag{1.14}$$

Then the stream cipher is secure if $F^l$ is a pseudorandom function for any polynomial $l$.

We discuss two modes of operation for encrypting arbitrary-length messages using a stream cipher: synchronized mode and un-synchronized mode.

(synchronized mode) The two parties are synchronized and the following method can be used to encrypt a series of messages from a sender $S$ to a receiver $R$:

- Both parties call $\mathsf{Init}(k)$ to obtain the same initial state $\mathsf{st}_0$.
- Let $\mathsf{st}_S$ be the current state of $S$. If $S$ wants to encrypt a message $m$, it computes $(y, \mathsf{st}'_S) := \mathsf{GetBits}(\mathsf{st}_S, 1^{|m|})$, sends $c := m \oplus y$ to the receiver, and updates its local state to $\mathsf{st}'_S$.
- Let $\mathsf{st}_R$ be the current state of $R$. When $R$ receives a ciphertext $c$ from the sender, it computes $(y, \mathsf{st}'_R) :=$ $\mathsf{GetBits}(\mathsf{st}_R, 1^{|c|})$, outputs the message $m := c \oplus y$, and updates its own local state to $st'_R$.

(un-synchronized mode) When a stream cipher does take an IV, it can be used to construct a stateless encryption scheme:

- $\mathsf{Gen}$: on input $1^n$, choose a uniform $k \in \{0,1\}^n$ and output it.
- $\mathsf{Enc}$: on input a key $k$ and a message $m \in \{0,1\}^*$, choose uniform $IV \in \{0,1\}^n$, and output the ciphertext $\langle IV, \mathsf{GetBits}_1(\mathsf{Init}(k, iv), 1^{|pt|}) \oplus m \rangle$.
- $\mathsf{Dec}$: on input a key $k$ and a ciphertext $\langle iv, c \rangle$, output the message $m := \mathsf{GetBits}_1(\mathsf{Init}(k, IV), 1^{|c|}) \oplus c$.

## Block Ciphers and Block-Cipher Modes of Operation

블록 암호 $F_k : \{0,1\}^l \to \{0,1\}^l$ 은 (strong) PRP 의 다른 이름이다. 블록 암호는 임의의 길이를 지원하는 PRP와 다르게 특정한 키/블록 길이 $n, l$을 사용한다는 점이다. 이 장에서는 블록 암호의 운영 모드 5개를 소개한다. 단순화를 위해 모든 메시지 벡터를 $m_i \in \{0,1\}^l$에 대해 $\mathbf{m} := (m_1, m_2, \cdots, m_l)$로 정의한다. (실제 메시지의 길이가 $l$의 배수가 아니더라도 패딩으로 해결 가능하므로, 이 가정은 일반성을 잃지 않는다.)

- ECB: $c_i := F_k(m_i)$ and $m_i := F_k^{-1}(c_i)$.
- CBC: $c_i := F_k(c_{i-1} \oplus m_i)$ and $m_i := F_k^{-1}(c_i) \oplus c_{i-1}$. $c_0 := IV \in \{0,1\}^n$ is uniform. $i = 1, 2, \cdots, l$.
- OFB:
- CFB:
- CTR: $y_i = F_k(IV||\langle i \rangle)$. $IV \in \{0,1\}^{3n/4}$ is uniform. $i = 1, 2, \cdots, l \le 2^{n/4}$. (일반적인 $IV$의 크기는?)

**Theorem 1.3.1.** ECB Mode is not IND-PASS secure.

*Proof.* (안 어려우므로 생략.) To prove something is not secure we need to exhibit an attack within the model. The attack on ECB mode is very simple:

Let $\mathbf{0}$ denote the block of all zeros, and $\mathbf{1}$ denote the block of all ones. Call the encryption oracle with $\mathbf{m_0} = \mathbf{0}||\mathbf{1}$ and $\mathbf{m_1} = \mathbf{1}||\mathbf{1}$ The challenge ciphertext $\mathbf{c}^*$ is returned which is the encryption of $\mathbf{m}_b$, for the hidden bit $b$. The challenge ciphertext consists of two blocks $\mathbf{c}_0$ and $\mathbf{c}_1$. If $\mathbf{c}_0 \ne \mathbf{c}_1$ then output $b' = 0$, else return $b' = 1$. □

**Lemma 1.3.1.** Fix a positive integer $N$, and say $q \le \sqrt{2N}$ elements $y_1, \cdots, y_q$ are chosen uniformly and independently from a set of size $N$. Then

$$\frac{q \cdot (q-1)}{4N} \le \mathsf{coll}(q, N) \le \frac{q \cdot (q-1)}{2N} \tag{1.15}$$

*Proof.* (일단 생략.) □

**Theorem 1.3.2.** If $P_k$ is a pseudorandom permutation, then CBC mode is IND-CPA secure.

*Proof.* $P_k$는 PRP, $p$는 RP라고 하고, $q(n)$을 임의의 PPT 구분자 $\mathcal{A}$가 암호화 오라클에 질의하는 최대 다항 횟수라고 하자. 마지막으로 $\mathsf{CBC}[P_k]$를 $P_k$를 블록암호로 사용하는 CBC 운영 모드, $\mathsf{CBC}[p]$를 $p$를 블록암호로 사용하는 CBC 운영 모드라고 하자. 증명은 다음과 같은 단계로 진행한다.

1. 먼저, 다음을 만족하는 negligible 함수 $\varepsilon$가 있음을 보인다.

$$\left| \Pr\left[\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{CBC}[P_k]}(n) = 1\right] - \Pr\left[\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{CBC}[p]}(n) = 1\right] \right| \le \varepsilon(n). \tag{1.16}$$

2. 이후 다음을 만족함을 보인다.

$$\Pr\left[\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{CBC}[p]}(n) = 1\right] \le \frac{1}{2} + \frac{q^2}{2^{n+1}}. \tag{1.17}$$

3. 위의 두 식을 통해 다음을 만족함을 알 수 있고, $q$는 다항식이므로, 증명이 완성된다.

$$\Pr\left[\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{CBC}[P_k]}(n) = 1\right] \le \frac{1}{2} + \frac{q^2}{2^{n+1}} + \varepsilon(n). \tag{1.18}$$

첫 번째 식 1.16을 증명하기 위해, 실험 1.5을 구성하자. 만약 $\mathcal{O} = P_k$이라면, $\mathcal{A}$ 관점에서 볼 때, 이 실험은 $\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{CBC}[P_k]}$ 실험과 같다. 따라서 다음이 성립한다.

$$\Pr\left[\mathcal{B}^{\mathsf{CBC}[P_k]}(1^n) = 1\right] = \Pr\left[\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{CBC}[P_k]}(n) = 1\right]. \tag{1.19}$$

비슷한 방식으로, 만약 $\mathcal{O} = p$라면, 이 실험은 $\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{CBC}[p]}$ 실험과 같다. 따라서 다음이 성립한다.

$$\Pr\left[\mathcal{B}^{\mathsf{CBC}[p]}(1^n) = 1\right] = \Pr\left[\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{CBC}[p]}(n) = 1\right]. \tag{1.20}$$

$P_k$는 PRP이므로, 다음을 만족하는 negligible 함수 $\varepsilon$가 존재한다.

$$\left|\Pr\left[\mathcal{B}^{\mathsf{CBC}[P_k]}(1^n) = 1\right] - \Pr\left[\mathcal{B}^{\mathsf{CBC}[p]}(1^n) = 1\right]\right| \le \varepsilon(n). \tag{1.21}$$

이 식과 위에서 보인 두 식 1.19과 1.20을 통해 증명하기로한 첫 번째 식을 증명할 수 있다.

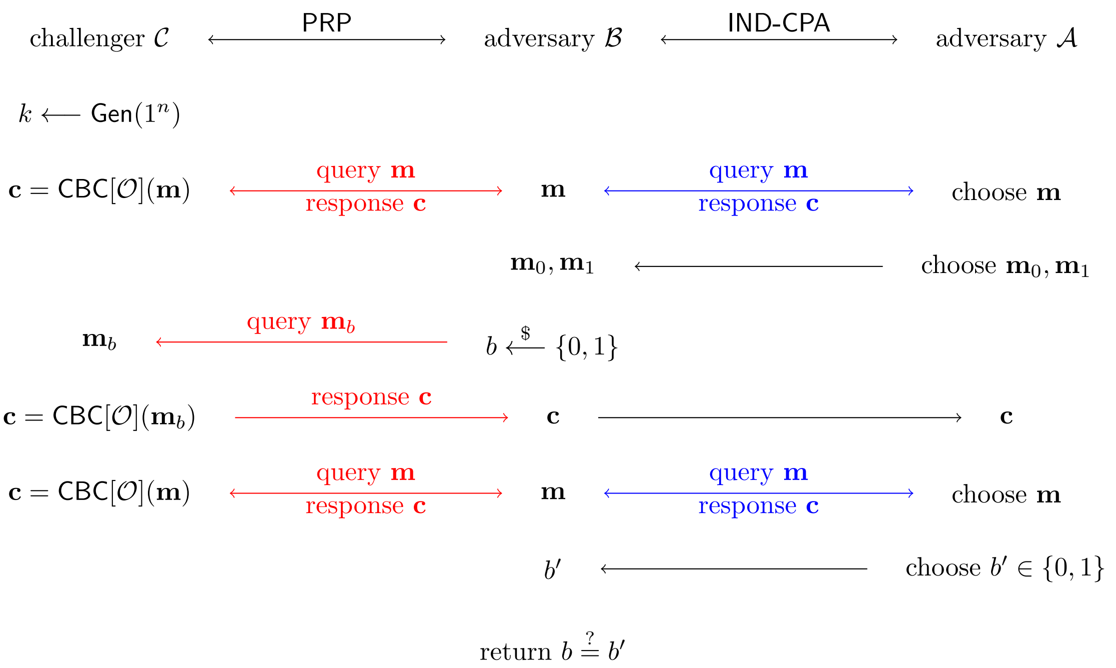


Figure 1.5: IND-CPA experiment of private-key encryption scheme with CBC mode

다음으로 두 번째 식 1.17을 증명한다. repeat을 $\mathcal{A}$가 암호화 오라클에 질의할 때, 초기화 벡터 $IV$가 적어도 한 번 이전과 같은 값을 사용하는 사건이라고 하자. $\mathcal{A}$가 모든 질의을 마쳤을 때, 다음의 두 사건이 있을 수 있다.

- (repeat이 일어나지 않은 사건, 즉, $\neg$repeat) 만약 $\mathcal{O} = p$라면, $IV$는 균등하게 선택되었으므로, 블록암호의 출력도 균등분포를 따른다. 따라서 $\mathcal{A}$의 구분 성공 확률은 정확히 1/2이다. 정리하면 다음과 같다.

$$\Pr\left[\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{CBC}[p]}(n) = 1 \mid \neg\mathsf{repeat}\right] \cdot \Pr\left[\neg\mathsf{repeat}\right] = \frac{1}{2}. \tag{1.22}$$

- (repeat이 일어난 사건) $IV$를 균등하게 독립적으로 선택하므로, 보조정리 1.3.1에 의해, 이 사건이 발생할 확률은 다음과 같다.

$$\Pr\left[\mathsf{repeat}\right] \le \frac{q^2}{2^{n+1}}. \tag{1.23}$$

위의 두 식을 이용하여, 다음과 같이 두 번째 식을 증명할 수 있고, 이로써 증명은 완성된다.

$$\begin{aligned}
&\Pr\left[\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{CBC}[p]}(n)=1\right] \\
&= \underbrace{\Pr\left[\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{CBC}[p]}(n)=1 \mid \neg\mathsf{repeat}\right]\cdot\Pr\left[\neg\mathsf{repeat}\right]}_{=\frac{1}{2}} + \underbrace{\Pr\left[\mathrm{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{CBC}[p]}(n)=1 \mid \mathsf{repeat}\right]\cdot\Pr\left[\mathsf{repeat}\right]}_{\le\Pr[\mathsf{repeat}]\le\frac{q^2}{2^{n+1}}} \\
&\le \frac{1}{2}+\frac{q^2}{2^{n+1}}.
\end{aligned} \tag{1.24}$$

□

**Theorem 1.3.3.** If $F_k$ is a pseudorandom function, then CTR mode is IND-CPA secure for multiple encryptions.

*Proof.* 정리 1.3.2과 유사하므로, 생략. □

## Nonce-Based Encryption

**Definition 1.3.1.** A nonce-based (private-key) encryption scheme consists of PPT algorithms $(\mathsf{Gen},\mathsf{Enc},\mathsf{Dec})$ such that:

- $\mathsf{Gen}$ takes as input $1^n$ and outputs a key $k$ with $|k|\ge n$.
- $\mathsf{Enc}$ takes as input a key $k$, a nonce $\mathsf{non}\in\{0,1\}^*$, and a message $m\in\{0,1\}^*$, and outputs a ciphertext $c$.
- $\mathsf{Dec}$ takes as input a key $k$, a nonce $\mathsf{non}$, and a ciphertext $c$, and outputs a message $m$ or $\perp$.

We require that for every $n$, every $k$ output by $\mathsf{Gen}(1^n)$, every $\mathsf{non}\in\{0,1\}^*$, and every $m\in\{0,1\}^*$, it holds that $\mathsf{Dec}(\mathsf{non},\mathsf{Enc}(\mathsf{non},m))=m$.

$$\begin{array}{ccc}
\text{challenger } \mathcal{C} & \xleftrightarrow{\quad\mathsf{IND\text{-}CPA}\text{ experiment}\quad} & \text{adversary } \mathcal{A} \\
k \longleftarrow \mathsf{Gen}(1^n) & & \\
b \xleftarrow{\$} \{0,1\} & & \\
c=\mathsf{Enc}(m,\mathsf{non}) & \xleftrightarrow[\text{response } c]{\text{query } (m,\mathsf{non})} & \text{choose } m,\mathsf{non} \\
m_0,m_1 & \longleftarrow & \text{choose } m_0,m_1,\mathsf{non} \\
c=\mathsf{Enc}(m_b,\mathsf{non}) & \longrightarrow & c \\
c=\mathsf{Enc}(m,\mathsf{non}) & \xleftrightarrow[\text{response } c]{\text{query } (m,\mathsf{non})} & \text{choose } m,\mathsf{non} \\
b' & \longleftarrow & \text{choose } b'\in\{0,1\} \\
\text{return } b\overset{?}{=}b' & &
\end{array}$$

Figure 1.6: IND-CPA experiment of nonce-based private-key encryption scheme

**Theorem 1.3.4.** With a nonce as the IV, CBC mode is not IND-CPA secure.

*Proof.* (안 어려우므로 생략.) Let $\mathbf{0}$ be the all-zero block and $\mathbf{1}$ be the all-one block. The attack on the $\mathsf{IND\text{-}CPA}$ security is as follows: Send the message $\mathbf{0}$ with the nonce $\mathbf{0}$ to the encryption oracle. The adversary obtains the ciphertext $\mathbf{0}||c$ in return, where $c = \mathsf{Enc}(\mathbf{0}, \mathbf{0})$. Now send the messages $m_0 = \mathbf{0}$ and $m_1 = \mathbf{1}$ to the encryption oracle, with nonce $\mathbf{1}$. Notice this is a new nonce and so the encryption is allowed in the game. Let $\mathbf{1}||c^*$ be the returned ciphertext. If $c^* = c$ then return $b' = 1$, else return $b' = 0$. $\square$