UNIVERSITÄT PADERBORN
*Die Universität der Informationsgesellschaft*
Fakultät für Elektrotechnik, Informatik und Mathematik
Warburger Straße 100
33098 Paderborn

# The Fujisaki-Okamoto Transformation

Bachelor's Thesis

by
Jan Lippert

Thesis Supervisor:
Prof. Dr. Johannes Blömer
and
Prof. Dr. Gitta Domik

Paderborn, November 22, 2014

# Declaration

(Translation from German)

I hereby declare that I prepared this thesis entirely on my own and have not used outside sources without declaration in the text. Any concepts or quotations applicable to these sources are clearly attributed to them. This thesis has not been submitted in the same or substantially similar version, not even in part, to any other authority for grading and has not been published elsewhere.

## Original Declaration Text in German:

## Erklärung

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen worden ist. Alle Ausführungen, die wörtlich oder sinngemäß übernommen worden sind, sind als solche gekennzeichnet.

City, Date

Signature

# Contents

# List of Figures

# List of Tables

# Listings

# 1 Introduction

One important goal of cryptography is to ensure confidentiality by providing secure communications via encryption. To prove the security of different encryption schemes, a formalized model is used: Adversaries play "games" that model specific attacks against a challenger. In the main part of these games, the adversary chooses two messages of equal length; one of these is encrypted by the challenger and the ciphertext is given to the adversary. The adversary then has to guess which of his messages has been encrypted. Different attack types are modeled in the strength of the adversary – the stronger the attack, the more possibilities the adversary has. Furthermore, we only consider probabilistic polynomial-time adversaries.

The weakest attack is the so called eavesdropping attack in which the adversary only knows his two messages and the encryption of one message – and in the case of public key encryption the corresponding public key. If the adversary can distinguish between the encryptions of his messages with a probability that is only negligible better than random guessing, the encryption scheme is called to have indistinguishable encryptions in the presence of an eavesdropper. In a chosen plaintext attack, the adversary either is given access to an encryption oracle or a public key [KL07, p. 338ff].

In a chosen-ciphertext attack (CCA), the adversary gains access to a decryption oracle in addition to his ability to encrypt messages. There are only two restriction in this game: The adversary has a runtime limit and must not submit the challenge ciphertext to the decryption oracle after it was chosen – otherwise the attacker would always win. An encryption scheme that is secure in this case has indistinguishable encryptions in the presence of a chosen ciphertext attack [KL07, p. 103]. CCA-indistinguishability implies that an adversary is unable to "logically manipulate" a given ciphertext, i.e. every result of that manipulation is either an invalid ciphertext or has no relation to the original message [KL07, p. 104].

There are basically two different types of encryption schemes. In a symmetric scheme, the participants need a secure channel to share a secret key in advance which is used for both encryption and decryption. In asymmetric schemes on the other hand, no secure channel to share keys is needed. Every participant has two keys: A public key which is used for encryption and a secret key which is used for decryption. Unfortunately, both symmetric and asymmetric schemes have drawbacks: Whereas symmetric schemes are often very fast, they require a previously shared key; asymmetric schemes on the other hand can be very slow when used for very long messages.

Therefore, hybrid encryption schemes are often used in practice. In a "straight-

forward" hybrid scheme the message is encrypted using a symmetric scheme and a randomly chosen one-time key k. Then, k is encrypted using the public key encryption scheme [KL07, p. 347]. One drawback of this straightforward approach is that in general these hybrid encryption schemes are CCA-indistinguishable only when the underlying asymmetric and symmetric encryption schemes both are CCA-indistinguishable [CS03].

Fujisaki and Okamoto introduced a transformation that removes this restriction [FO99, FO13]: CCA-indistinguishable hybrid encryption schemes can be obtained from encryption schemes that have much weaker security notions than CCA-indistinguishability: The symmetric scheme needs to be one-time secure which means that the scheme has secure encryption in the presence of an eavesdropper when every key is used only once [FO13, p. 89]. The asymmetric scheme on the other hand only has to be a one-way encryption scheme. This means that an adversary $\mathcal{A}$ has only a negligible probability to find any correct plaintext to a given ciphertext $c$, i.e. $\mathcal{A}$ outputs an $m$ that - when encrypted - can result in $c$ [FO13, p. 87f].

The Fujisaki-Okamoto (FO) scheme obtained by applying the Fujisaki-Okamoto Transformation (FOT) uses some hash functions which are modeled as random oracles in the security proof. A random oracle is an idealized model of cryptographic hash functions. Informally, a random oracle can be described as a public, randomly-chosen function that can be evaluated by "querying" values to it; for every query the oracle returns a value which is picked uniformly at random from its range. The Random Oracle Model (ROM) is called a "middle ground between a fully-rigorous proof of security [...] and no proof"[KL07, p. 460f].

The basic definitions and techniques will be explained in Chapter 2. This chapter provides the requirements to convert the notation in [FO13] to a more standard notation like the notation used in [KL07]. The reader is also introduced to the game-hopping technique and the ROM. The Fujisaki-Okamoto Transformation then will be introduced in Chapter 3. Its usage will be motivated and a proof of its CCA-security in the ROM will be given. Different versions of the Fujisaki-Okamoto Transformation will be compared to each other. Finally, a one-time pad based variant of the Fujisaki-Okamoto will be investigated.

# 2 Foundations

This section is used to provide the foundations necessary to understand the Fujisaki-Okamoto Transformation. First, we will introduce the notation. Then, after presenting a definition of encryption schemes, we will introduce the security definitions used in this thesis.

## 2.1 Notation and Basic Definitions

In this section we firstly provide the notation. We will also give some basic definitions and some lemmas.

**Logarithm** $\log x$ is used to denote the binary logarithm $\log_2 x$.

**Bit-length** Let $x$ be a bit string. We denote by $|x|$ the bit-length of $x$.

**Concatenation** Denote by "$a \parallel b$" the concatenation of two bit strings $a$ and $b$. We assume that the concatenation $a \parallel b$ can be parsed into $a$ and $b$ again.

**Deterministic algorithm** We will use $y := \mathcal{A}(x)$ to denote "$y$ is the result of a deterministic algorithm $\mathcal{A}$ executed on input $x$".

**Randomness** In many experiments, elements need to be picked uniformly at random from finite set $\mathcal{R}$. We denote by $r \leftarrow \mathcal{R}$ the experiment of "picking $r$ uniformly at random from the finite set $\mathcal{R}$".

**Probabilistic polynomial-time algorithm** A probabilistic polynomial-time (PPT) algorithm is a randomized algorithm which runs in time polynomial in $n$, where $n$ is a security parameter. $\mathcal{A}$ may sample random coins $r \leftarrow \mathcal{R}$ to be used in its computations, where $\mathcal{R}$ is specified in the description of the algorithm. We write

$$y \leftarrow \mathcal{A}(x)$$

to denote that the random variable $y$ is defined as the output of $\mathcal{A}$ when run on input $x$ and with fresh random coins $r \leftarrow \mathcal{R}$. [HK09, 316]

For convenience we also define a deterministic variant of executing $\mathcal{A}$. The random coins can be sampled beforehand and passed to $\mathcal{A}$ as a parameter:

$$y := \mathcal{A}(x; r).$$

In this variant, $\mathcal{A}$ does not draw new random coins and is deterministic. For better readability we will separate the random coins from the normal parameters by semicolon.

**Output set** We will use $[\mathcal{A}(x)]$ to denote the set of all possible outputs of the PPT algorithm $\mathcal{A}$ that samples random coins $r \leftarrow \mathcal{R}$ on input $x$, i.e.

$$[\mathcal{A}(x)] := \{y \mid \exists r \in \mathcal{R} : y = \mathcal{A}(x; r)\},$$

where the set $\mathcal{R}$ is specified in the description of the algorithm.

**Probabilities** Let $\mathcal{A}$ be a probabilistic algorithm and let $Y$ be a random variable fo the output of algorithm $\mathcal{A}$ on input $(x_1, \ldots, x_n)$. We will use the following notation to denote the "probability that the output of $\mathcal{A}$ executed on $(x_1, \ldots, x_n)$ is equal to $y$". Note that we have two alternatives to write this probability as we can sample the random coins of $\mathcal{A}$ beforehand.

$$Pr[Y = y : y \leftarrow \mathcal{A}(x_1, \ldots, x_n)] = Pr[Y = y : y := \mathcal{A}(x_1, \ldots, x_n; r); r \leftarrow \mathcal{R}]$$

**Negligibility** A function $\varepsilon$ is negligible if for every polynomial $p(n)$ there exists an $N \in \mathbb{N}$ such that for all integers $n > N$ it holds that

$$\varepsilon(n) < \frac{1}{p(n)}.$$

For convenience we will use $\varepsilon$ to denote negligible functions.

**Oracle access** Oracle access of an algorithm $\mathcal{A}$ to some algorithm $B$ means that $\mathcal{A}$ may query $B$ to compute a function. $\mathcal{A}$ may submit queries with the argument $x$ to the $B$ and will receive $B(x)$ in return. Querying the oracle is one computational step for $\mathcal{A}$ because the computation of $B(x)$ is done by the oracle. We will write this as $\mathcal{A}^{B(\cdot)}$. If $\mathcal{A}$ has oracle access to multiple algorithms $B$ and $C$ we will write this as $\mathcal{A}^{B(\cdot),C(\cdot)}$.

**Experiments** Experiments are modeled as games between an adversary and a challenger. We will now look at a contrived experiment $Experiment_{\mathcal{A},\mathbf{C}}(n)$ to fix the notation. Note that $\mathcal{A}$ denotes the adversary and $\mathbf{C}$ denotes the challenger. Let $f(x)$ denote a secret function with domain and range $\{0, \ldots, n\}$ which is only known to the challenger, but not to the adversary. Let $\Omega_f$ be the set of such functions, i.e. $\Omega_f := \{\{0, \ldots, n\} \to \{0, \ldots, n\}\}$.

**The example experiment** $Experiment(n)$**:**

1. **C**: $f \leftarrow \Omega_f$
2. $x_1, x_2 \leftarrow \mathcal{A}(1^n)$ with $x_1, x_2 \in \{0, \ldots, n\}$
3. **C**: $b \leftarrow \{0,1\}$; $c^* := f(x_b)$.
4. $b' \leftarrow \mathcal{A}(c^*)$
5. The output of this experiment is defined to be 1 if $b = b'$, or 0 otherwise. This is written as $Experiment(n) = 1$ or $Experiment(n) = 0$ respectively.

This experiment reads as "The challenger initializes the game by choosing a function $f \in \Omega_f$ uniformly at random. The so-called security parameter $n$ is then given to the adversary. The adversary chooses two values $x_1$ and $x_2$ from the domain of $f$. The adversary then gives $x_1$ and $x_2$ to the challenger. The challenger picks a bit $b$ and assigns the result of $f(x_b)$ to the challenge $c^*$. After fixing the challenge, the adversary $\mathcal{A}$ is given $c^*$ and outputs another bit $b'$. The adversary wins if he guessed the challenger's bit $b$ correctly.". Note, that we use the same adversary $\mathcal{A}$ in Steps 2 and 4 – $\mathcal{A}$ therefore remembers $x_1$ and $x_2$ and all other information he could retrieve in 2.

In experiments that are used to define some security notion for an encryption scheme $\Pi$, we will write $Experiment_{\mathcal{A},\mathbf{C}}(n)$ instead. For convenience, all challenges are marked with a star, i.e. $c^*$ is a challenge. Also we will mark every part of the challenge except for $b$ and the adversary's input with a star.

**Working with negligible functions** In some cases we want to know if a function $\varepsilon'$ is negligible. We then try to construct an inequation to relate $\varepsilon'$ to some negligible function $\varepsilon$ and apply the following lemma.

**Lemma 2.1.** *Let $\varepsilon_1$ and $\varepsilon_2$ be negligible functions. Let $p(n)$ denote an arbitrary polynomial. Then the functions $\varepsilon_3$ and $\varepsilon_4$ defined below are also negligible [KL07, 57]:*

$$\varepsilon_3(n) := \varepsilon_1(n) + \varepsilon_2(n)$$
$$\varepsilon_4(n) := p(n) \cdot \varepsilon_1(n)$$

## 2.2 Encryption Schemes

This section will present definitions for symmetric and asymmetric encryption schemes. Based on these definitions we will give the security definitions used in this thesis and define the Fujisaki-Okamoto Transformation. We will first define three important sets for encryption schemes.

**Definition 2.2** (Message Space)**.** *Denote by $\mathcal{M}$ the set of all messages that can be encrypted by a specified encryption scheme. The message space is usually defined by the security parameters of the encryption scheme.*

**Definition 2.3** (Ciphertext Space)**.** *Denote by $\mathcal{C}$ the set of possible inputs of the decryption algorithm. The ciphertext space is usually defined by the security parameters of the encryption scheme.*

We will need to use the coin space of an encryption algorithm in the definition of well-spread encryption and the definition of the Fujisaki-Okamoto Transformation.

**Definition 2.4** (Coin Space of an encryption algorithm)**.** *Denote by $\mathcal{R}$ the coin-space of an encryption algorithm. The encryption algorithm draws new random coins $r \in \mathcal{R}$ on every execution. Note that $\mathcal{R}$ may be the empty set – then the encryption algorithm is purely deterministic.*

We will give the definition of symmetric encryption schemes and introduce some semantics. The following definition of asymmetric encryption schemes and later definitions will be kept brief because they are similar. Note, that it is common practice to write the key which is used in encryption or decryption as an index and not as function parameter.

**Definition 2.5** (Symmetric encryption schemes [KL07, 60])**.** *A symmetric encryption scheme $\Pi$ is a tuple of three PPT algorithms $(Gen, Enc, Dec)$ such that:*

*$Gen(1^n)$:* *The key-generation algorithm Gen takes as input the security parameter encoded in a bit-string of length n that consists purely of ones. The key-generation algorithm outputs a key k; we write this as $k \leftarrow Gen(1^n)$.*

*$Enc_k(m)$:* *The encryption algorithm Enc takes as input a key $k \in [Gen(1^n)]$ and a plaintext message $m \in \mathcal{M}$, and outputs a ciphertext c. Since Enc may be randomized, we write this as $c \leftarrow Enc_k(m)$.*

*$Dec_k(c)$:* *The decryption algorithm Dec takes as input a key $k \in [Gen(1^n)]$ and a ciphertext c, and outputs a message $m' \in \mathcal{M}$ or a special symbol $\perp \notin \mathcal{M}$ denoting failure. We assume that Dec is deterministic, therefore we write $m' := Dec_k(c)$.*

*Correctness: We will require that for every n, every $k \in [Gen(1^n)]$, and every $m \in \mathcal{M}$ it holds that $Dec_k(Enc_k(m)) = m$ [KL07, 60].*

We will assume without loss of generality that $|k| \geq n$. This assures that the encryption and decryption algorithms are given input of length polynomial in $n$. We also assume, that $Gen(1^n)$ select a key $k$ uniformly at random from the key space, i.e. $k \leftarrow Gen(1^n)$ is equivalent to $k \leftarrow [Gen(1^n)]$.

**Definition 2.6** (Asymmetric encryption schemes [KL07, 336f])**.** *An asymmetric encryption scheme $\Pi$ is a tuple of PPT algorithms $(Gen, Enc, Dec)$ such that:*

$Gen(1^n)$: $(pk, sk) \leftarrow Gen(1^n)$. *We refer to* $pk$ *as public key and* $sk$ *as secret key.*

$Enc_{pk}(m)$: $c \leftarrow Enc_{pk}(m)$ *for* $m \in \mathcal{M}$ *and* $(pk, \cdot) \in [Gen(1^n)]$.

$Dec_{sk}(c)$: $m' := Dec_{sk}(c)$ *for* $c \in \mathcal{C}$ *and* $(\cdot, sk) \in [Gen(1^n)]$. $m'$ *may be a special symbol* $\perp$ *denoting failure.*

*Correctness: We will assume that for a large enough* $n \in \mathbb{N}$, *every* $(pk, sk) \in [Gen(1^n)]$ *and every* $m \in \mathcal{M}$ *we have* $Dec_{sk}(Enc_{pk}(m)) = m$ *[FO13, 85].*

A more theoretical correctness assumption states that the probability of a decryption error on an encrypted message is negligible when using asymmetric encryption schemes, i.e.

$$Pr[Dec_{sk}(Enc_{pk}(m)) \neq m] \leq \varepsilon(n)$$

with $\varepsilon$ being a negligible function [KL07, 337]. Note that most practically used schemes fulfill our correctness assumption. We also assume that $pk$ and $sk$ each have length at least $n$, and that $n$ can be determined from $pk$ and $sk$[1].

**Notes about ciphertext space** The set of valid ciphertexts $\mathcal{C}_{valid}$ is the output set of the encryption algorithm. It may be a subset of the ciphertext space, i.e. $\mathcal{C}_{valid} \subseteq \mathcal{C}$. $\mathcal{C}_{valid}$

## 2.3 Security Definitions

In this section we will introduce the security definitions that we need to understand the Fujisaki-Okamoto Transformation. First we will introduce the standard security notions: Eavesdropping indistinguishability, chosen plaintext indistinguishability, and chosen ciphertext indistinguishability. Based on these properties we will explain the "non-standard" security notion called one-way asymmetric encryption. We will also investigate a property called $\gamma$-spread.

The definitions for eavesdropping indistinguishability, chosen plaintext indistinguishability, and chosen ciphertext indistinguishability are based on [KL07] and have been adapted to use the $(t, \varepsilon)$ notation as in [FO13]. In [KL07] this notation is also shortly mentioned but quickly switch to the so-called asymptotic approach: An encryption scheme $\Pi$ is called secure if for all PPT adversaries there exists a negligible function $\varepsilon$ such that

$$Pr[Experiment_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \varepsilon(n),$$

[1] This property is needed in some simulations.

where $\varepsilon$ is often called the advantage of the adversary. Example 2.7 shows some potential values for $t$ and $\varepsilon$. This example is only given to give some basic understanding of the $(t, \varepsilon)$-notation. We will not use concrete values outside of examples.

**Example 2.7.** *This example is based on [KL07, 50ff]. Assume that there is a $(t, \varepsilon)$-secure encryption scheme for some security notion, where*

$$t^{asy}(n) = n^3 \text{ minutes and}$$
$$\varepsilon^{asy}(n) = 2^{20-n}.$$

*This informally means that an adversary $\mathcal{A}$ running for $n^3$ minutes has an advantage of $2^{40-n}$. Table 2.7 shows some example values for such a scheme. One*

Table 2.1: Example values for a $(n^3, 2^{40-n})$-secure scheme

| $n$ | $t^{asy}(n)$ | $\varepsilon^{asy}(n)$ |
|---|---|---|
| $\leq 40$ | $\leq 44,44$ days | 1 |
| 45 | 63.28 days | $2^{-5}$ |
| 80 | 355,56 days | $2^{-40}$ |
| 100 | 1,90 years | $2^{-60}$ |
| 120 | 3,29 years | $2^{-80}$ |

*can easily see, that $n \leq 40$ is a very insecure choice: The scheme can be broken with probability 1 in around 44 days. One can use this table to choose a value for $n$, depending on the security level to achieve. For some low security application, it may be sufficient that an adversary running for 63 days has an advantage of $2^{-5} = \frac{1}{32}$ which would be the case for $n = 45$. A high level security application may require that an adversary succeeds at most with probability $2^{-60}$ which would require $n \geq 100$.*

### 2.3.1 Eavesdropping Attack

The weakest of the standard security notions we will use in this theses is the eavesdropping indistinguishability (EAV). This notion is used to show that an adversary that eavesdrops on a ciphertext cannot easily distinguish between encryptions of two different messages – even if the adversary can choose those messages himself. Formally, this is modeled in a game between an adversary and a challenger.

**The eavesdropping indistinguishability experiment $PrivK^{eav}_{\mathcal{A},\Pi}(n)$:**

1. **C**: $k \leftarrow Gen(1^n)$.
2. $m_0, m_1 \leftarrow \mathcal{A}(1^n)$ with $m_0, m_1 \in \mathcal{M}$ and $|m_0| = |m_1|$.

3. **C**: $b \leftarrow \{0,1\}$; $c^* \leftarrow Enc_k(m_b)$. $c^*$ is called **challenge ciphertext**.
4. $b' \leftarrow \mathcal{A}(c^*)$.
5. The output of $PrivK^{eav}_{\mathcal{A},\Pi}(n)$ is defined to be 1 if $b = b'$, and 0 otherwise.

If $PrivK^{eav}_{\mathcal{A},\Pi}(n) = 1$, we say that $\mathcal{A}$ succeeded. Based on the experiment above, we will define eavesdropping indistinguishability:

**Definition 2.8** (Eavesdropping indistinguishability)**.** *Let $\Pi = (Gen, Enc, Dec)$ be a symmetric encryption scheme. The encryption scheme $\Pi$ is called $(t,\varepsilon)$-EAV secure if for every polynomial $t$-time adversary $\mathcal{A}$ there exists a negligible function $\varepsilon$ such that*

$$Pr[PrivK^{eav}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \varepsilon(n).$$

### 2.3.2 Chosen-Plaintext Attack

The chosen-plaintext attack is a stronger notion of security than eavesdropping. In a chosen plaintext attack, the adversary is able to obtain the encryptions for messages of his choice. In the case of symmetric encryption, the adversary gets access to an encryption oracle $Enc_k(\cdot)$. In an asymmetric scheme, the adversary knows the public key $pk$ and can encrypt messages on his own.

**CPA indistinguishability experiment $PubK^{cpa}_{\mathcal{A},\Pi}(n)$:**

1. **C**: $(pk, sk) \leftarrow Gen(1^n)$.
2. $m_0, m_1 \leftarrow \mathcal{A}(pk)$ with $m_0, m_1 \in \mathcal{M}$ and $|m_0| = |m_1|$.
3. **C**: $b \leftarrow \{0,1\}$; $c^* \leftarrow Enc_{pk}(m_b)$.
4. $b' \leftarrow \mathcal{A}(c^*)$.
5. The output of $PubK^{cpa}_{\mathcal{A},\Pi}(n)$ is defined to be 1 if $b = b'$, and 0 otherwise.

Furthermore we will give the definition of CPA indistinguishability as in [KL07, 339].

**Definition 2.9** (CPA indistinguishability)**.** *Let $\Pi = (Gen, Enc, Dec)$ be an asymmetric encryption scheme and let $\mathcal{A}$ be an adversary that works on $\Pi$. The encryption scheme $\Pi$ is called $(t,\varepsilon)$-CPA secure if, for every polynomial $t$-time adversary $\mathcal{A}$ there exists a negligible function $\varepsilon$ such that*

$$Pr[PubK^{cpa}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \varepsilon(n).$$

**Lemma 2.10.** *Deterministic encryption schemes cannot be CPA-secure.*

*Proof of Lemma 2.10.* A chosen-plaintext attack (CPA) adversary can compute the encryptions of $m_0$ and $m_1$ in Step 4 of the CPA indistinguishability experiment. These encryption can then be compared to the challenge ciphertext and the adversary always succeeds. □

### 2.3.3 Chosen-Ciphertext Attack

The CCA is the strongest notion of security we will work with. In addition to his access to the public key $pk$ or an encryption oracle, the adversary is now granted access to a decryption oracle. He may contrive possible ciphertexts (or use ciphertexts he knows) to request their decryption. The only limitations of the adversary are the given runtime limit and that the decryption of the challenge ciphertext must not be requested.

**The chosen-ciphertext indistinguishability experiment** $PubK^{cca}_{\mathcal{A},\Pi}(n)$**:**

1. **C**: $(pk, sk) \leftarrow Gen(1^n)$.
2. $m_0, m_1 \leftarrow \mathcal{A}^{Dec_{sk}(\cdot)}(pk)$ with $m_0, m_1 \in \mathcal{M}$ and $|m_0| = |m_1|$.
3. **C**: $b \leftarrow \{0,1\}$; $c^* \leftarrow Enc_{pk}(m_b)$.
4. $b' \leftarrow \mathcal{A}^{Dec_{sk}(\cdot)}(c^*)$.
   $\mathcal{A}^{Dec_{sk}(\cdot)}(c^*)$ is not allowed to submit $c^*$ to the decryption oracle.
5. The output of $PubK^{cca}_{\mathcal{A},\Pi}(n)$ is defined to be 1 if $b = b'$, and 0 otherwise.

Similar to the definitions of eavesdropping and chosen plaintext indistinguishability, we will now adapt the definition of chosen ciphertext indistinguishability [KL07, 371] to the notation of [FO13, 89].

**Definition 2.11** (CCA indistinguishability)**.** *Let* $\Pi = (Gen, Enc, Dec)$ *be an asymmetric encryption scheme and let* $\mathcal{A}$ *be an adversary against* $\Pi$*. An adversary is called a* $(t, q_{Dec})$ *if it is a* $t$*-time adversary that accesses the decryption oracle at most* $q_{Dec}$ *times. The encryption scheme* $\Pi$ *is called* $(t, q_{Dec}, \varepsilon)$*-CCA secure if, for every polynomial* $(t, q_{Dec})$*-adversary* $\mathcal{A}$ *there exists a negligible function* $\varepsilon$ *such that*

$$Pr[PubK^{cca}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \varepsilon(n).$$

### 2.3.4 Well-Spread Encryption

Well-spread encryption is a term defined in [FO13, 88]. In this section we will give the basics to understand the definition of well-spread encryption as well as further explanation on its meaning. At the end of the section we will give different examples of this notion in practice.

We will start with some foundations. Let $\| X \|_\infty$ denote the infinity norm of a probability space $X$ on a finite set $S$, i.e.,

$$\| X \|_\infty := \max_{a \in S} Pr[x = a : x \leftarrow X].$$

The infinity norm can be described as the probability of the most likeliest event in probability space $X$. In the case of encryption, we are interested in how well the

outputs of the encryption algorithm are distributed in the corresponding output set. Remember that $\mathcal{R}$ is the random coin space. We then adapt the infinity norm as the probability of the most likeliest ciphertext $c$ when encrypting a fixed message $m \in \mathcal{M}$ with a fixed public key $pk$:

$$\| Enc_{pk}(m) \|_\infty := \max_{c \in \mathcal{C}} Pr[c = Enc_{pk}(m; r) : r \leftarrow \mathcal{R}] \tag{2.1}$$

Based on the adapted infinity norm, we will define the function

$$\Gamma(pk, m) := -\log \| Enc_{pk}(m) \|_\infty$$

to be used in the definition of $\gamma$-spread.

**Definition 2.12** ($\gamma$-spread [FO13, 88])**.** *An asymmetric encryption scheme $\Pi$ is called $\gamma$-spread if we have*

$$\Gamma(pk, m) \geq \gamma(n)$$

*for every $(pk, sk) \in Gen(1^n)$ and every $m \in \mathcal{M}$. The scheme is said to be "well-spread" if $\gamma(n) = \omega(\log n)$.*

Above definition can be rephrased into the following Corollary.

**Corollary 2.13.** *Let $\Pi = (Gen, Enc, Dec)$ be an asymmetric encryption scheme with message space $\mathcal{M}$ and random coin space $\mathcal{R}$. $\Pi$ is called $\gamma$-spread if for all $(pk, sk) \in [Gen(1^n)]$ and every $m \in \mathcal{M}$, for all $c \in \mathcal{C}$ we have*

$$Pr[c = Enc_{pk}(m; r) : r \leftarrow \mathcal{R}] \leq 2^{-\gamma(n)},$$

*and the number of possible encryptions for every $m \in \mathcal{M}$ is at least $2^{\gamma(n)}$.*

As one can see in this corollary, $\gamma$-spread can be seen as a measurement for the minimum number of random bits used in the encryption algorithm.

*Proof of Corollary 2.13.* We will start with the definition of $\gamma$-spread:

$$\Gamma(pk, m) = -\log \| Enc_{pk}(m) \|_\infty = \log \frac{1}{\| Enc_{pk}(m) \|_\infty} \geq \gamma(n)$$

After applying Equation 2.1 we have

$$\log \frac{1}{\max_{c \in \mathcal{C}} Pr[c = Enc_{pk}(m; r) : r \leftarrow \mathcal{R}]} \geq \gamma(n).$$

By exponentiating the inequation we get

$$\frac{1}{\max_{c \in \mathcal{C}} Pr[c = Enc_{pk}(m; r) : r \leftarrow \mathcal{R}]} \geq 2^{\gamma(n)}.$$

And therefore we finally get

$$\max_{c \in \mathcal{C}} Pr[c = Enc_{pk}(m; r) : r \leftarrow \mathcal{R}] \leq \frac{1}{2^{\gamma(n)}}. \tag{2.2}$$

The encryption algorithm cannot fail for valid $(pk, \cdot) \in [Gen(1^n)]$ and $m \in \mathcal{M}$ and therefore we have

$$\sum_{c \in \mathcal{C}} Pr[c = Enc_{pk}(m; r) : r \leftarrow \mathcal{R}] = 1. \tag{2.3}$$

To fulfill both Inequation 2.2 and Equation 2.3, there must be at least $2^{\gamma(n)}$ different ciphertexts for every message $m$. □

According to [FO13, 90], any asymmetric encryption scheme $\Pi = (Gen, Enc, Dec)$ that is $\gamma$-spread and OWE-secure can be transformed to a $(\gamma + \gamma')$-spread encryption scheme $\Pi' = (Gen', Enc', Dec')$ by appending random coins $r' \in \{0,1\}^{\gamma'}$ to the end of the encrypted message, i.e. the encryption algorithm works as follows:

1. $r \leftarrow \mathcal{R}$
2. $r' \leftarrow \{0,1\}^{\gamma'}$
3. $c_m := Enc_{pk}(m; r)$
4. Output $c'_m := c_m \parallel r'$

The adapted decryption algorithm removes the last $\gamma'(n)$ bits from $c$ and then executes $Dec_{sk}$ on the remaining bits. As one can see, this method is a simple way to increase the $\gamma$-spread of an encryption scheme. Therefore we will state the following corollary.

**Corollary 2.14.** *There is a transformation to increase transform any $\gamma$-spread asymmetric encryption scheme to a $(\gamma + \gamma')$-spread encryption scheme.*

In the following paragraphs, we will investigate the $\gamma$ values for some standard schemes: Textbook RSA and ELGamal.

**Definition 2.15** (Textbook RSA [KL07, 355f])**.** *Let $GenRSA$ be a PPT algorithm that, on input $1^n$ outputs $N = pq$ where $p$ and $q$ are $n$-bit primes, along with integers $e$, $d$ satisfying $ed = 1 \mod (p-q)(q-1)$.*

*$Gen(1^n)$ Generate a key pair $(pk, sk)$ as follows:*

1. *$(N, e, d) \leftarrow GenRSA(1^n)$*
2. *$pk := (N, e)$*
3. *$sk := (N, d)$*
4. *return $(pk, sk)$*

*$Enc_{pk}(m)$ On input $pk = (N, e)$ and $m \in \mathbb{Z}_N^*$, return*

$$c := [m^e \mod N].$$

*$Dec_{sk}(c)$ On input $sk = (N, d)$ and $c \in \mathbb{Z}_N^*$, return*

$$m' := \left[c^d \mod N\right].$$

The textbook-RSA encryption scheme uses a deterministic encryption algorithm as displayed in Definition 2.15. Therefore textbook RSA is a 0-spread encryption scheme, i.e. $\gamma(n) = 0$:

$$\Gamma(pk, m) = -\log \max_{c \in \mathbb{Z}_N^*} Pr[c = Enc_{pk}(m)] = -\log 1 = 0$$

The ElGamal encryption scheme is randomized; its definition as in [KL07, 364f] is given below.

**Definition 2.16** (ElGamal)**.** *Let $GenGroup(1^n)$ be a PPT algorithm that outputs a tuple $(\mathbb{G}, q, g)$ where $\mathbb{G}$ is a cyclic group of order $q$ (with bit-length $|q| = n$) and $g$ as a generator of $\mathbb{G}$.*

*$Gen(1^n)$:* *Generate a key pair $(pk, sk)$ as follows:*

1. $(\mathbb{G}, q, g) \leftarrow GenGroup(1^n)$
2. $x \leftarrow \mathbb{Z}_q$
3. $h := g^x$
4. $pk := (\mathbb{G}, q, g, h)$
5. $sk := (\mathbb{G}, q, g, x)$
6. *return* $(pk, sk)$

*$Enc_{pk}(m)$:* *On input $pk = (\mathbb{G}, q, g, h)$ and $m \in \mathbb{G}$:*

1. $r \leftarrow \mathbb{Z}_q$
2. $c_m := h^r \cdot m$
3. $c_r := g^r$
4. *return* $c := c_m \parallel c_r$

*$Dec_{sk}(c)$:* *On input $sk = (\mathbb{G}, q, g, x)$ and $c = c_m \parallel c_r$:*

1. $m = c_m / c_r^x$
2. *return* $m$

**Lemma 2.17.** *Let $\mathbb{G}$ be a finite group and let $m \in \mathbb{G}$ be an arbitrary element. Then choosing random $g \leftarrow \mathbb{G}$ and setting $g' := g \cdot m$ gives the same distribution for $g'$ as choosing random $g' \leftarrow \mathbb{G}$. I.e. for any $g^* \in \mathbb{G}$*

$$Pr[m \cdot g = g^* : g \leftarrow \mathbb{G}] = \frac{1}{|\mathbb{G}|}.$$

With the definition of the encryption algorithm and Lemma 2.17, we can compute the $\gamma$-spread of the ElGamal encryption scheme.

$$\begin{aligned}\Gamma(pk, m) &= -\log \max_{c \in \mathbb{G}\times\mathbb{G}} Pr[c = Enc_{pk}(m; r) : r \leftarrow \mathbb{Z}_q] \\ &= -\log \max_{c_m \in \mathbb{G}, c_r \in \mathbb{G}} Pr[(c_m, c_r) = (h^r \cdot m, g^r) : r \leftarrow \mathbb{Z}_q] \qquad (2.4)\end{aligned}$$

For every tuple of the form

$$(c_m, c_r) = (h^r \cdot m, g^r),$$

there is at most one $r \in \mathbb{Z}_q$ that fulfills above equation. Therefore we have

$$Pr[(c_m, c_r) = (c_r^x \cdot m, g^r) : r \leftarrow \mathbb{Z}_q] \leq \frac{1}{|\mathbb{Z}_q|}, \qquad (2.5)$$

and Equation 2.4 can be simplified to

$$\begin{aligned}\Gamma(pk, m) &= -\log \max_{c \in \mathbb{G}\times\mathbb{G}} Pr[(c_m, c_r) = (c_r^x \cdot m, g^r) : r \leftarrow \mathbb{Z}_q] \\ &= -\log \frac{1}{|\mathbb{Z}_q|} \\ &= \log q.\end{aligned}$$

Per definition of the ElGamal encryption scheme, the bit-length of $q$ is $n$:

$$\Rightarrow \Gamma(pk, m) \geq \log 2^n = n = \gamma(n)$$

ElGamal is a well-spread encryption scheme because $\gamma(n) = n = \omega(\log n)$.

### 2.3.5 One-way Asymmetric Encryption

In [FO13, 87f] a security notion called one-way attack (OWE) is defined. It is mentioned in [KL07, 357] as a very weak notion that just means that an "adversary does not learn everything about $m$" when given a public key and the ciphertext of a randomly chosen $m$, i.e. the adversary could learn $n - 1$ of $n$ bits. We will investigate OWE and relate it to the standard security notions CPA and CCA.

This notion is often used in one-time key settings like hybrid schemes. $\mathcal{M}$ must be restricted to an efficiently sampleable set, i.e. the challenger must be able to pick $m$ uniformly in time polynomial in $n$.

**The One-way encryption experiment** $PubK^{owe}_{\mathcal{A},\Pi}(n)$**:**

1. **C**: $(pk, sk) \leftarrow Gen(1^n)$; $m \leftarrow \mathcal{M}$; $c^* \leftarrow Enc_{pk}(m)$.
2. $m' \leftarrow \mathcal{A}(pk, c^*)$.
3. The output of the experiment is defined to be 1 if $m = m'$, and 0 otherwise.

Based on this experiment, we can define OWE-security.

**Definition 2.18** (One-way asymmetric encryption [FO13, 87f])**.** *An asymmetric encryption scheme* $\Pi = (Gen, Enc, Dec)$ *is* $(t, \varepsilon)$*-OWE secure if for all polynomial* $t$*-time adversaries there exists a negligible function* $\varepsilon$ *such that*

$$Pr[PubK^{owe}_{\mathcal{A},\Pi}(n) = 1] \leq \varepsilon(n).$$

We will now investigate the relationship between one-way asymmetric encryption and CPA-security.

**Lemma 2.19.** *Let* $\Pi$ *be an asymmetric encryption scheme with message space* $\mathcal{M} = \mathcal{M}$*. Suppose there is a* $t^{owe}$*-time adversary* $\mathcal{A}^{owe}$ *that attacks* $\Pi$ *in the sense of OWE and succeeds with probability* $\frac{1}{2} + \varepsilon^{owe}$*. Then we can construct an adversary* $\mathcal{A}^{cpa}$ *that attacks* $\Pi$ *in the sense of CPA and succeeds with probability* $\frac{1}{2} + \frac{1}{2}\varepsilon^{owe}(n)$*. The running time of* $\mathcal{A}^{cpa}$ *is* $t^{cpa}(n) = O(t^{owe}(n))$*.*

*Proof of Lemma 2.19.* Let $\Pi$ be an asymmetric encryption scheme and let $\mathcal{A}^{owe}$ be a $t^{owe}$-time adversary attacking $\Pi$ in the sense of OWE that succeeds with probability $\varepsilon^{owe}(n)$. We will use $\mathcal{A}^{owe}$ to construct a CPA adversary $\mathcal{A}^{cpa}$. The construction is depicted Figure 2.1 and works as follows:

- On input $pk$, $\mathcal{A}^{cpa}$ picks two distinct messages $m_0, m_1$ uniformly at random from the message space, i.e. "$m_0 \leftarrow \mathcal{M}$; $m_1 \leftarrow \mathcal{M} \setminus \{m_0\}$". Both messages are given to the challenger.

- On input $c^*$, $\mathcal{A}^{cpa}$ runs $\mathcal{A}^{owe}$ on input $(pk, c^*)$. When $\mathcal{A}^{owe}$ halts and outputs a message $m'$, $\mathcal{A}^{cpa}$ converts $m'$ to a bit $b'$ as shown in Listing 2.1. $\mathcal{A}^{cpa}$ succeeds if it outputs $b' = b$.

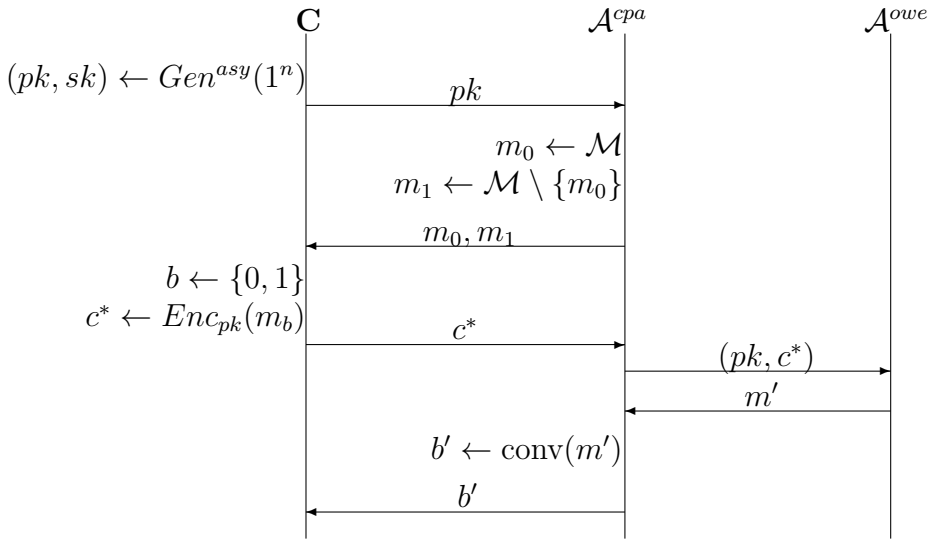


Figure 2.1: A CPA adversary constructed from an OWE adversary

Listing 2.1: Conversion algorithm

```
conv(m'){
  if (m' = m_0)
    return 0
  endif
  if (m' = m_1)
    return 1
  endif
  guess ← {0,1}
  return guess
}
```

We now have to prove that $\mathcal{A}^{cpa}$'s simulation is distributed identically to a non-simulated OWE experiment. Let $X$ be a random variable that is the result of the experiment "$m_0 \leftarrow \mathcal{M}; m_1 \leftarrow \mathcal{M} \setminus \{m_0\}; b \leftarrow \{0,1\}; X := m_b$". We have to show that we have $Pr[X = m] = \frac{1}{|\mathcal{M}|}$ for all $m \in \mathcal{M}$. Let $m \in \mathcal{M}$ be an arbitrary message.

$$
\begin{aligned}
Pr[X = m] &= Pr[b = 0] \cdot Pr[X = m_0 \mid b = 0] + Pr[b = 1] \cdot Pr[X = m_1 \mid b = 1] \\
&= \frac{1}{2} \cdot Pr[X = m_0] + \frac{1}{2} \cdot Pr[X = m_1 \mid b = 1] \\
&= \frac{1}{2} \cdot \frac{1}{|\mathcal{M}|} + \frac{1}{2} \cdot \sum_{x \in \mathcal{M} \setminus \{m_0\}} Pr[X = m_1 \mid x = m_0] \cdot Pr[x = m_0] \\
&= \frac{1}{2} \cdot \frac{1}{|\mathcal{M}|} + \frac{1}{2} \cdot \sum_{x \in \mathcal{M} \setminus \{m_0\}} \frac{1}{|M| - 1} \cdot \frac{1}{|M|} \\
&= \frac{1}{2} \cdot \frac{1}{|\mathcal{M}|} + \frac{1}{2} \cdot \frac{1}{|\mathcal{M}|} \\
&= \frac{1}{|\mathcal{M}|}
\end{aligned}
$$

As we have assumed that $\mathcal{A}^{owe}$ succeeds with probability $\varepsilon^{owe}(n)$, we can state the following equations.

$$
\begin{aligned}
Pr[m' = m_b] = Pr[\mathcal{A}^{owe} \text{ succeeds}] &= \varepsilon^{owe}(n) \\
Pr[m' \neq m_0 \wedge m' \neq m_1] \leq Pr[\mathcal{A}^{owe} \text{ fails}] &= 1 - \varepsilon^{owe}(n).
\end{aligned}
$$

Per construction, $\mathcal{A}^{cpa}$'s output depends on $\mathcal{A}^{owe}$'s output and we can construct the following equation

$$
\begin{aligned}
Pr[b' = b] &= Pr[b' = b \mid m' = m_b] Pr[m' = m_b] \\
&\quad + Pr[b' = b \mid m' \neq m_0 \wedge m' \neq m_1] Pr[m' \neq m_0 \wedge m' \neq m_1] \\
&= 1 \cdot Pr[m' = m_b] + \frac{1}{2} \cdot Pr[m' \neq m_0 \wedge m' \neq m_1] \\
&\leq \varepsilon^{owe}(n) + \frac{1}{2} \left(1 - \varepsilon^{owe}(n)\right) \\
&= \frac{1}{2} + \frac{1}{2} \varepsilon^{owe}(n)
\end{aligned}
$$

The runtime of $\mathcal{A}^{cpa}$ is $t^{cpa} = O(t^{owe})$ because picking two random messages and converting the output of $\mathcal{A}^{owe}$ can be done in time polynomial in $n$. □

Lemma 2.19 also implies, that an encryption scheme $\Pi$ which is secure in the sense of CPA is also secure in the sense of OWE. We will also give the intuition, that OWE does not imply CPA. For example, the textbook RSA encryption scheme is described as a one-way secure encryption scheme by Katz and Lindell *if* the RSA assumption holds:

> "If the RSA assumption holds [...] one can show that if a message $m$ is chosen *uniformly at random* from $\mathbb{Z}_N^*$, then no PPT adversary

given the public key $(N, e)$ and $c := Enc_{pk}(m)$ can recover the entire message $m$." [KL07, 357]

Nonetheless RSA cannot be CPA-secure because the encryption algorithm as shown in Definition 2.15 is deterministic.

**Lemma 2.20.** *One-way security does not imply CPA-security.*

Because an encryption scheme that is not chosen-plaintext secure cannot be chosen ciphertext secure, we can conclude:

**Corollary 2.21.** *One-way security does not imply CCA-security.*

## 2.4 Game-Hopping Technique

The game-hopping technique can be used to estimate the adversary's probability of success $Pr[S_0]$ in an attack game denoted by $G_0$. Game-hopping is used when the direct computation of $Pr[S_0]$ would be too hard or too complicated – Game $G_0$ then is gradually changed following the rules below until one can compute the adversary's probability of success $Pr[S_l]$ in a final Game $G_l$. Denote by $S_i$ the event that "the adversary succeeds in Game $G_i$".

As mentioned, the changes between two Games $G_i$ and $G_{i-1}$ follow some rules which bound the difference $|Pr[S_i] - Pr[S_{i-1}]|$. This can be used to estimate the difference

$$|Pr[S_l] - Pr[S_0]| \leq \sum_{i=1}^{l} |Pr[S_i] - Pr[S_{i-1}]|.$$

After $Pr[S_l]$ has been computed, one can also find an upper bound for $Pr[S_0]$ using above inequation:

$$Pr[S_0] \leq Pr[S_l] + |Pr[S_l] - Pr[S_0]|.$$

We will now outline different types of game-hops and explain how these changes bound the probability difference between two games.

**Bridging steps** Bridging steps are conceptual changes that change the game in a way that both probability distributions are exactly the same. These changes are used to prepare other changes and make it simpler to follow the proof. As the probability distribution does not change, we have $Pr[S_i] = Pr[S_{i-1}]$ after applying a bridging step [Sho04, 3].

**Transition on negligible failure events** In this transition, the simulation of the attack behaves exactly the same in both games – except when a specific failure event $F$ occurs. We cannot know how $\mathcal{A}$ will respond in the case that $F$ occurs but this does not matter as long as the probability of $F$ is negligible. We can bound the difference between both $S$'s by using the Difference Lemma.

**Lemma 2.22** (Difference Lemma [Sho04, 3])**.** *Let A,B,F be events defined in some probability distribution, and suppose that $A \wedge \neg F \Leftrightarrow B \wedge \neg F$. Then it holds that $|Pr[A] - Pr[B]| \leq Pr[F]$.*

*Proof of Lemma 2.22 [Sho04, 3].*

$$|Pr[A] - Pr[B]| = |Pr[A \wedge F] + Pr[A \wedge \neg F] - Pr[B \wedge F] - Pr[B \wedge \neg F]|$$

We assumed that $A \wedge \neg F \Leftrightarrow B \wedge \neg F$ and as such $Pr[A \wedge \neg F] = Pr[B \wedge \neg F]$.

$$|Pr[A] - Pr[B]| = |Pr[A \wedge F] - Pr[B \wedge F]|$$

We then have

$$|Pr[A \wedge F] - Pr[B \wedge F]| \leq Pr[F], \tag{2.6}$$

because $0 \leq Pr[A \wedge F] \leq Pr[F]$ and $0 \leq Pr[B \wedge F] \leq Pr[F]$. □

Assume that we have two games $G_i$ and $G_{i+1}$. These games behave exactly the same except when an event $F$ occurs – we therefore define $A$ as the event that "the adversary succeeds in Game $G_i$" and $B$ as the event that "the adversary succeeds in Game $G_{i+1}$". One can then apply the Difference Lemma and see that the difference between the probabilities of those events is

$$|Pr[A] - Pr[B]| \leq Pr[F].$$

The following types of transition are not used in this thesis and therefore will be kept very brief.

**Transitions based on indistinguishability** In this transition we make a small change between the games that could enable the adversary to easily break the challenge. We then argue that the probability to distinguish between those two games is negligible and therefore the difference $Pr[S_i] - Pr[S_{i-1}]$ is also negligible [Sho04, 2].

Note that a "hybrid argument" is essentially a sequence of transitions based on indistinguishability [Sho04, 4].

**Transitions on large failure events** As in a transition on negligible failure events, this transition is also done over a failure event $F$. But in this case we consider that the probability of the failure event $F$ may be "very large, but not totally overwhelming". There are two cases mentioned in [Den06]: The adversary directly tries to compute an output as in an OWE attack or he tries to distinguish between two possibilities as in an eavesdropping indistinguishability. In the first case we have

$$Pr[S_i] = \frac{1}{Pr[\neg F]} \cdot Pr[S_{i+1}],$$

where $Pr[S_i]$ is only negligible if $Pr[S_{i+1}]$ is negligible.

In the latter case we have

$$|Pr[S_i] - \frac{1}{2}| = \frac{1}{Pr[\neg F]} \cdot |Pr[S_{i+1}] - \frac{1}{2}|,$$

where $|Pr[S_i] - \frac{1}{2}|$ is negligible if, and only if, $|Pr[S_{i+1}] - \frac{1}{2}|$ is negligible.

## 2.5 The Random Oracle Model

The ROM is a model which assumes the existence of an idealized hash function. One can use this model to design new encryption schemes that rely on one or

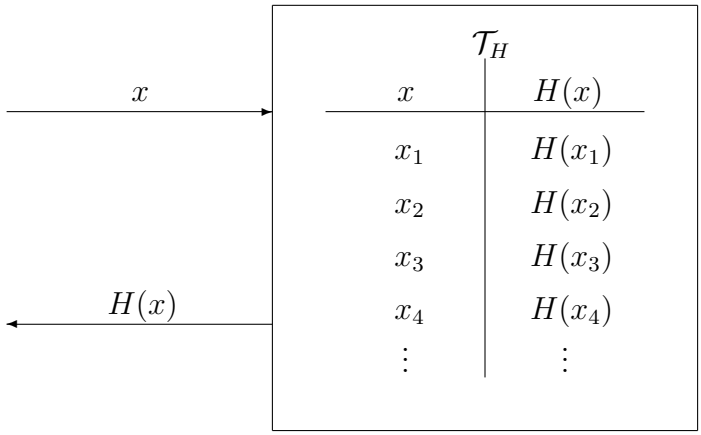


Figure 2.2: A Random Oracle imagined as a black box.

more hash functions. Using this abstraction, the cryptographer can focus in the ROM on the scheme itself. On the one hand this can be seen as the first step in the design of novel cryptographic schemes. On the other hand using instantiation of a Random Oracle (RO) with an appropriate hash function yields encryption schemes that are very efficient. In the standard model where no Random Oracle is available, provable security is often gained at the cost of efficiency. Also it seems "that people prefer to use *no encryption* rather than to use inefficient encryption schemes" [KL07, 458f].

A RO can be described as a black box that takes bit strings as input and outputs bit strings. The Random Oracle is known to all participants in an experiment – honest parties and adversaries alike – but they do not know the Random Oracle's internal function $H$. Everyone may submit queries $x$ from a domain $D$ to the black box to receive the value $H(x)$ from the oracle's range $R$. Before the oracle can answer to queries, a random function $H$ must be picked to map the inputs to random outputs.

Let $\Omega := \{D \to R\}$ for some finite sets $D$ and $R$. We can then define the experiment "$H \leftarrow \Omega$". Fujisaki and Okamoto use the infinite set $D = \{0,1\}^*$ but state that this is an imaginary experiment [FO13, 88]. In [BR93] a function $H$ is picked at random by sampling every single bit of an answer $H(x)$ uniformly at random for every possible query $x \in D$. [FO13] uses a similar definition but picks every answer uniformly at random from the range, i.e. the experiment "$H(x) \leftarrow R$" is carried out for every $x \in D$. These pairs $(x, H(x))$ are then saved in a lookup table called $\mathcal{T}_H$ as displayed in Figure 2.2.

Another thing is, that true random oracles do not exist in reality. To use an encryption scheme that was designed in the ROM, a so-called instantiation has to be made: The oracle $H$ has to be replaced by a real cryptographic hash function

$H'$. The RO methodology now assumes that the encryption scheme is secure in reality if $H'$ is good enough at emulating a Random Oracle. This last step is one of the drawbacks of the ROM as there is no reason to assume that the security proof carries over into reality. In fact, one can construct *contrived* schemes that are insecure in the real world, no matter how they are instantiated. Canetti et al. show a construction to modify an encryption scheme $\Pi$ that is secure both in the ROM and in the standard model into a scheme that is only secure in the ROM – no matter how the scheme is instantiated [CGH04].

Nonetheless, the ROM has been very successful in practice. It enables cryptographers to design schemes that are more efficient than those created in the standard model. Many schemes that are used today were designed and proven secure in the ROM. Examples include Optimal Asymmetric Encryption [BR94], Rapid Enhanced-security Asymmetric Cryptosystem Transform [OP01], and the Boneh–Franklin scheme [BF01].

An equivalent view on ROs is to pick function values $H(x)$ on demand. The RO starts with an empty lookup table $\mathcal{T}_H$. Whenever the RO receives a query $x$ it first looks into $\mathcal{T}_H$ if $x$ has been answered before; if so, the same $H(x)$ is returned. If $x$ was not encountered yet, a new value $H(x)$ is picked uniformly at random from the range of the oracle, $(x, H(x))$ is added to $\mathcal{T}_H$, and $H(x)$ is returned. From the participants' point of view, picking a random function beforehand and generating function values on demand is equivalent.

We will now adapt the chosen-ciphertext indistinguishability experiment to the Random Oracle Model. Let $\Pi = (Gen, Enc, Dec)$ be an asymmetric encryption scheme that uses a hash function $H$ in its encryption algorithm and in its decryption algorithm. The domain of $H$ is set to $\{0,1\}^\alpha$ and the range is set to $\{0,1\}^\beta$ and $H$ is modeled as a Random Oracle. We then can define the chosen-ciphertext attack in the random oracle model (CCA-RO) as follows.

**The CCA-RO indistinguishability experiment** $PubK^{cca-ro}_{\mathcal{A},\Pi}(n)$**:**

1. $H \leftarrow \left\{\{0,1\}^\alpha \rightarrow \{0,1\}^\beta\right\}$
2. **C**: $(pk, sk) \leftarrow Gen(1^n)$.
3. $m_0, m_1 \leftarrow \mathcal{A}^{Dec_{sk}(\cdot),H(\cdot)}(pk)$ with $m_0, m_1 \in \mathcal{M}$ and $|m_0| = |m_1|$.
4. **C**: $b \leftarrow \{0,1\}$; $c^* \leftarrow Enc_{pk}(m_b)$.
5. $b' \leftarrow \mathcal{A}^{Dec_{sk}(\cdot),H(\cdot)}(pk, c^*)$.
6. $PubK^{cca-ro}_{\mathcal{A},\Pi}(n)$ is defined to be 1 if $b = b'$, and 0 otherwise.

Figure 2.3 shows an example execution of the CCA-RO experiment. In this example, the adversary queried the RO $H$ two times. We assumed that $H$ models a cryptographic hashfunction which is used in the encryption algorithm, and as

such the oracle is queried once while creating the challenge. Note, that we will depict ROs as boxes to denote their black-box characteristic. We also define CCA-security in the ROM as follows.

**Definition 2.23** (CCA-RO indistinguishability [FO13, 89])**.** *Let* $\Pi$ *be an asymmetric encryption scheme and let* $\mathcal{A}$ *be an adversary that works on* $\Pi$*. Let* $H$ *be a hash function modeled as RO. The encryption scheme* $\Pi$ *is called* $(t, q_{Dec}, q_{Hash}, \varepsilon)$*-CCA-RO secure if, for every polynomial* $(t, q_{Dec}, q_{Hash})$*-adversary* $\mathcal{A}$ *it holds that*

$$Pr[PubK^{cca-ro}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \varepsilon(n),$$

*where* $\mathcal{A}$ *is a* $t$*-time adversary that accesses the decryption oracle at most* $q_{Dec}$ *times and the Random Oracle at most* $q_{Hash}$ *times.*

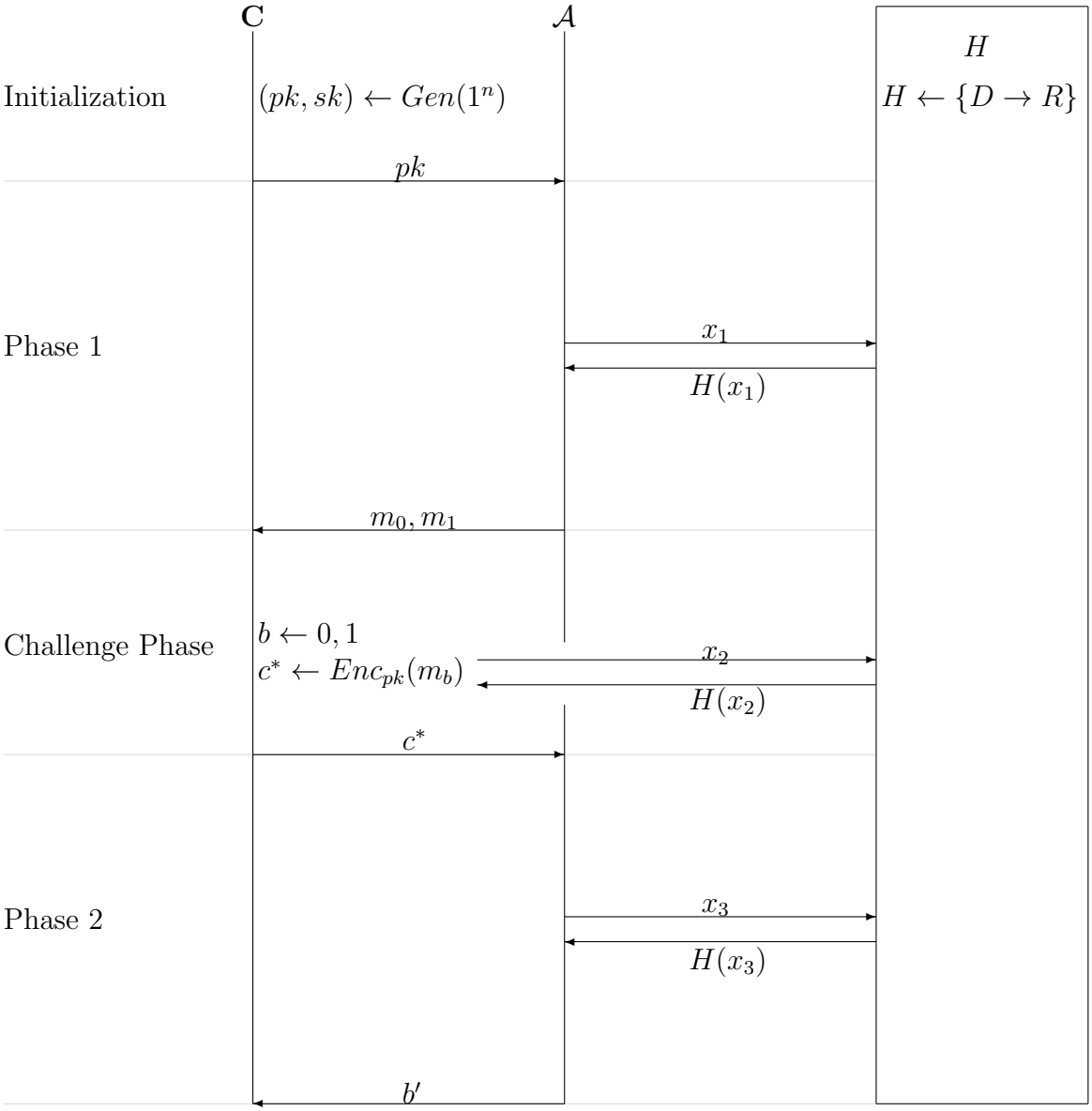


Figure 2.3: A possible execution of the chosen-ciphertext attack in the ROM

# 3 Fujisaki-Okamoto-Transformation

The FOT was first introduced in [FO99] by Fujisaki and Okamoto and revised in [FO13]. It is a special way to construct CCA-secure hybrid encryption schemes. We will first motivate hybrid encryption schemes, Hybrid encryption schemes are used to solve some problems of pure asymmetric and pure symmetric encryption schemes by combining both types. Whereas asymmetric encryption schemes are too slow to be practically used for big messages, symmetric encryption schemes must have the secret key exchanged beforehand. We will provide a slightly adapted straight-forward transformation. This scheme is used as an example for hybrid schemes in [KL07, 348]:

**Definition 3.1** (Hybrid encryption schemes)**.** *Let $\Pi^{asy}$ be an asymmetric encryption scheme, and let $\Pi^{sy}$ be a symmetric encryption scheme such that $\mathcal{K}^{sy} = \mathcal{M}^{asy}$. Construct an asymmetric encryption scheme $\Pi^{hy}$ as follows:*

$Gen^{hy}(1^n)$**:** *The key generation algorithm works as follows:*

1. $(pk, sk) \leftarrow Gen^{asy}(1^n)$
2. *Output* $(pk, sk)$

$Enc^{hy}_{pk}(m)$**:** *The encryption algorithm works as follows:*

1. $k \leftarrow \mathcal{M}^{asy}$
2. $c_m \leftarrow Enc^{sy}_r(m)$
3. $c_k \leftarrow Enc^{asy}_{pk}(k)$
4. *output the ciphertext* $c_m \parallel c_k$

$Dec^{hy}_{sk}(c)$**:** *The decryption algorithm works as follows:*

1. *Parse $c$ into $c_m$ and $c_k$; if $c$ cannot be parsed into $c_m$ and $c_k$, output $\perp$.*
2. $k' := Dec^{asy}_{sk}(c_k)$
3. *if* $(k' \notin \mathcal{M}^{asy})$ *return* $\perp$[1]
4. $m' := Dec^{sy}_{k'}(c_m)$. *$m'$ may be a special symbol $\perp$ denoting failure.*

[1] This step was added for clarification. Step 2 may set $k' = \perp$ which would lead to an error in the final step.

If $\Pi^{asy}$ is a CPA-secure asymmetric encryption scheme and $\Pi^{sy}$ is an EAV-secure symmetric scheme, then the hybrid scheme $\Pi^{hy}$ obtained by above transformation is a CPA-secure asymmetric encryption scheme [KL07, 351]. Unfortunately, to get a CCA-secure hybrid scheme one generally needs to start with both an CCA-secure asymmetric scheme and a CCA-secure symmetric scheme [CS03]. There are some CCA-secure asymmetric schemes based on the decisional Diffie-Hellman like the Cramer-Shoup scheme (see [CS03]), but no CCA-secure schemes based on the RSA assumption are currently known[2] [KL07, 473]. The Fujisaki Okamoto transformation removes these restrictions at least in the ROM: One can now construct a CCA-secure hybrid scheme from encryption schemes that are secure in much weaker security models.

## 3.1 The Fujisaki-Okamoto Scheme

In this section we will present the revised FOT as given in [FO13, 86] – it is used to construct CCA-secure encryption schemes in the ROM.

Let $\Pi^{asy} = (Gen^{asy}, Enc^{asy}, Dec^{asy})$ be an asymmetric encryption scheme and let $\Pi^{sy} = (Gen^{sy}, Enc^{sy}, Dec^{sy})$ be a symmetric encryption scheme $\Pi^{sy}$. We require two hash functions

$$G : \mathcal{M}^{asy} \to \mathcal{K}^{sy} \text{ and } H : \mathcal{M}^{asy} \times \mathcal{C}^{sy} \to \mathcal{R}^{asy}$$

We will now define the FO encryption scheme $\Pi^{fo}$ such that $\mathcal{M}^{fo} = \mathcal{M}^{sy}$ and $\mathcal{R}^{fo} = \mathcal{M}^{asy}$.

**Definition 3.2** (FO scheme [FO13, 86f])**.** *Define the FO encryption scheme* $\Pi^{fo} = (Gen^{fo}, Enc^{fo}, Dec^{fo})$ *as follows:*

$Gen^{fo}(1^n)$**:** *The key generation algorithm works as follows:*

1. $(pk, sk) \leftarrow Gen^{asy}(1^n)$
2. *Output* $(pk, sk)$

*We assume that the public key pk can be easily derived from sk.*

$Enc^{fo}_{pk}(m)$**:** *The encryption algorithm takes as input a public key pk and a message* $m \in \mathcal{M}^{sy}$ *and works as follows:*

1. $r \leftarrow \mathcal{M}^{asy}$
2. $k := G(r)$
3. $c_m \leftarrow Enc^{sy}_k(m)$
4. $h := H(r, c_m)$

[2] It is also stated in [KL07] that there are no CCA-secure asymmetric schemes based on the factoring assumption. Hofheinz and Kiltz have since then introduced such a scheme in [HK07].

5. $c_r := Enc^{asy}_{pk}(r; h)$
6. *return* $c := c_m \parallel c_r$

$Dec^{fo}_{sk}(c)$**:** *The decryption algorithm takes as input a secret key sk and a ciphertext* $c \in \{0,1\}^*$*. For convenience we assume that the public key pk is can be derived from the secret key sk. It works as follows:*

1. *Parse the given ciphertext c into* $c_m$ *and* $c_r$*. If c cannot be parsed, return the error symbol* $\perp$*.*
2. $r' := Dec^{asy}_{sk}(c_r)$
3. *if* $(r' \notin \mathcal{M}^{asy})$*, return* $\perp$
4. $h' := H(r', c_m)$
5. $c_r{}' := Enc^{asy}_{pk}(r'; h')$
6. *if* $(c_r \neq c_r{}')$*, return* $\perp$
7. $k' := G(r')$
8. *return* $Dec^{sy}_{k'}(c_m)$*.*

We now have to show that the $\Pi^{fo}$ is a correct encryption scheme, i.e. for every $m \in \mathcal{M}^{fo}$ and every $(pk, sk) \in Gen^{fo}(1^n)$ we have $Dec^{fo}_{sk}(Enc^{fo}_{pk}(m)) = m$.

*Proof of Correctness.* Let $n$ be an arbitrary security parameter and let $(pk, sk) \in \left[Gen^{fo}(1^n)\right]$ be an arbitrary key-pair. Let $m \in \mathcal{M}^{fo}$ be an arbitrary but fixed message. Let $c \in \left[Enc^{fo}_{pk}(m)\right]$. We then have

$$c = c_m \parallel c_r, \text{ where } c_m = Enc^{sy}_{G(r)}(m) \text{ and } c_r = Enc^{asy}_{pk}(r; H(r, c_m)).$$

We then can decrypt $c_r$ into $r' := Dec^{asy}_{sk}(c_r)$. Per correctness of the asymmetric scheme we then have

$$r' = Dec^{asy}_{sk}(Enc^{asy}_{sk}(r; H(r, c_m))) = r.$$

$G$ is a hash function. Therefore we have $G(r) = G(r')$. Then – using the correctness of the symmetric scheme – we have

$$Dec^{sy}_{G(r')}(c_m) = Dec^{sy}_{G(r')}(Enc^{sy}_{G(r)}(m)) = m.$$

□

As we can see in the definition of the decryption algorithm from above, not all inputs are decrypted: The given ciphertext $c$ must be correctly formed and fulfill some constraints.

**Definition 3.3** (Correctly formed ciphertexts)**.** *A ciphertext c is correctly formed if, and only if, c can be parsed into* $c_m \parallel c_r$ *such that* $c_m \in \mathcal{C}^{sy}$ *and* $c_r \in \mathcal{C}^{asy}$*.*

**Definition 3.4** (Valid ciphertexts)**.** *Let $c = c_m \parallel c_r$ be a correctly formed ciphertext and let $r$ be the decryption of $c_r$, i.e. $r := Dec^{asy}_{sk}(c_r)$. $c$ is called valid if, and only if, we have*

$$r \in \mathcal{M}^{asy} \wedge c_r = Enc^{asy}_{pk}(r; H(r, c_m)).$$

We will now give two examples for the FO decryption algorithm executed on some invalid ciphertexts. Example 3.5 corresponds to the first check in Step 3 whereas Example 3.6 corresponds to the check in Step 6.

**Example 3.5** (Rejection of invalid ciphertexts 1)**.** *Remember that the set of valid ciphertexts, i.e. encryptions of real messages, may be a subset of the corresponding ciphertext space. Let $(pk, sk) \in \left[Gen^{fo}(1^n)\right]$ be an arbitrary key pair and let $c_m \in \mathcal{C}^{sy}$ be an arbitrary symmetric ciphertext. Let $c_r \in \mathcal{C}^{asy}$ be an invalid asymmetric ciphertext with respect to $pk$.*

*Now run $Dec^{fo}_{sk}(c_m \parallel c_r)$. The submitted ciphertext can be parsed into $c_m$ and $c_r$ and $Dec^{asy}_{sk}(c_r)$ is run. $Dec^{asy}_{sk}$ is a deterministic algorithm and will output $\perp \notin \mathcal{M}^{asy}$ because $c_r$ is invalid. The FO decryption algorithm will then reject $c_m \parallel c_r$ in Step 3 since $\perp$ is not in $\mathcal{M}^{asy}$.*

**Example 3.6** (Rejection of invalid ciphertexts 2)**.** *Let $(pk, sk) \in \left[Gen^{fo}(1^n)\right]$ be an arbitrary key pair. Choose $r \in \mathcal{M}^{asy}$, $c_m \in \mathcal{C}^{sy}$, and $h \in \mathcal{R}^{asy}$ such that $h \neq H(r, c_m)$. Let $c_r := Enc^{asy}_{pk}(r; h)$ and run $Dec^{fo}_{sk}$ on $c_m \parallel c_r$. Since $c_r$ is a valid output of the asymmetric encryption algorithm, it will not be rejected in Step 3 of the FO decryption algorithm.*

*The decryption algorithm continues and creates $c'_r := Enc^{asy}_{pk}(r; H(r, c_m))$. Remember that the encryption algorithm is deterministic when the random coins are passed as an argument. We assumed that $h \neq H(r, c_m)$ and therefore the encryption algorithm will output $c'_r = c_r$ with a probability less or equal to $2^{-\gamma}$. Therefore the ciphertext will probably get rejected in Step 6 of the FO decryption algorithm.*

The decryption algorithm has been changed for better readability. In particular, *goto* operations were inlined and the same error symbol $\perp$ is used in all failure cases. Another adaption has been made in the definition of the random oracles: The explicit domains $\mathcal{M}^{asy}$ and $\mathcal{C}^{sy}$ are used instead of $\{0,1\}^*$. This change will be further explained in Section 3.4.

**Comparison of FO scheme to straight-forward hybrid scheme** In the FO scheme, the adversary's possibility to construct new ciphertexts is restricted. This is important to ensure CCA-security of the FO scheme. In the straight-forward hybrid encryption scheme as in Definition 3.1, there are no such restrictions as can be seen in Listings 3.1 and 3.2.

The FO scheme utilizes two hash functions $G$ and $H$. $G$ is used to hash the one-time key candidate onto a one-time key. This is an additional step to secure the one-time key. As long as an adversary does not find out all the bits of the $r'$, the adversary gains no information about the actual one-time key $G(r')$ if $G$ is modeled as a RO.

The straight-forward hybrid encryption scheme on the other hand does not hash the one-time key. It may be enough for the adversary to find some bits of the one-time key to partly decrypt the message.

The second hash function $H$ is used to construct a validity check. This is done to restrict an adversary's ability to contrive new ciphertexts. Assume now that $\Pi^{asy}$ is $\gamma$-spread in both the straight-forward scheme and the FO scheme. In the straight-forward hybrid encryption scheme, the encryption of the one-time key is independent of $c_m$ and there are at least $2^\gamma$ valid values for $c_k$. The FO scheme on the other hand restricts the number of valid one-time key encryptions: Only one of those $2^\gamma$ possible encryption is valid under fixed $r$ and $c_m$.

Listing 3.1: FO decryption $Dec^{fo}_{sk}(\cdot)$

```
Dec^fo_sk(c_m || c_r){
  r' := Dec^asy_sk(c_r)
  if (r' ∉ M^asy) then
    return ⊥
  endif
  h' := H(r, c_m)
  c'_r := Enc^asy_pk(r'; h')
  if (c'_r ≠ c_r) then
    return ⊥
  endif
  k' := G(r')
  m' := Dec^sy_k'(c_m)
  return m'
}
```

Listing 3.2: Hybrid decryption

```
Dec^hy_sk(c_m || c_r){
  r' := Dec^asy_sk(c_r)
  if (r' ∉ M^asy) then
    return ⊥
  endif

  // No validity check for c_r

  k' := r' // No hashing
  m' := Dec^sy_k'(c_m)
  return m'
}
```

## 3.2 Security of the Fujisaki-Okamoto scheme

**Theorem 3.7.** *Let $\Pi^{asy}$ be a $\gamma$-spread $(t^{asy}, \varepsilon^{asy})$-OWE secure asymmetric encryption scheme and let $\Pi^{sy}$ be a $(t^{sy}, \varepsilon^{sy})$-EAV secure symmetric encryption scheme. Denote by $T^{asy}(n)$ the worst case of the running time of $Enc^{asy}_{pk}(m)$ for every $m \in \mathcal{M}^{asy}$ and every $(pk, sk) \in [Gen^{asy}(1^n)]$. Let $\Pi^{fo}$ be the FO scheme and let $\mathcal{A}$ be a $(t^{fo}, q_{Hash}, q_{Dec})$-adversary.*

*$\Pi^{fo}$ is $(t^{fo}, q_{Hash}, q_{Dec}, \varepsilon^{fo})$-CCA secure in the Random Oracle Model where*

$$t^{fo}(n) = \min(t^{asy}(n), t^{sy}(n)) - q_{Hash}O(n) - (q_{Dec}+1)T^{asy}(n) \text{ and}$$
$$\varepsilon^{fo}(n) = \varepsilon^{sy}(n) + q_{Hash}\varepsilon^{asy}(n) + 2^{-\gamma}q_{Dec}.$$

The $(t^{fo}, q_{Hash}, q_{Dec}, \varepsilon^{fo})$-notation is useful for two main reasons: It relates the adversary's advantage to his power, while it also relates the security bounds of the FO scheme to the security bounds of the underlying encryption schemes.

Our proof is based on [FO13, 91-95]. It is divided in two parts. The first part utilizes the game-hopping technique. We will make use of four games as depicted in Figure 3.1.

$G_0$ We will start with the CCA-RO experiment as given in Section 2.5.

$G_1$ We will first replace the ROs with simulations by the challenger. This change is a bridging step.

$G_2$ Then we will construct a new decryption oracle that does not need the secret key. We hereby introduce the failure event $Bad$.

$G_3$ Finally, we adapt the challenge to be more convenient for part two of the proof. We introduce the failure event $Ask$ in this transition.

Denote by $S_i$ the event that the adversary succeeds in Game $G_i$, i.e. the adversary outputs a bit $b'$ with $b' = b$. Above games then can be used to find an upper bound $|Pr[S_3] - Pr[S_0]| \leq Pr[Bad] + Pr[Ask]$. We then state an upper bound for $Pr[S_3]$ using the EAV-security of $\Pi^{sy}$ and bound $Pr[S_0]$. Hereby, we can also bound $\varepsilon^{fo}$.

In the second part of the proof, we will show upper bounds for $Pr[Bad]$, $Pr[Ask]$, and $Pr[S_3]$ using the assumptions that $\Pi^{asy}$ is $\gamma$-spread and OWE-secure, and that $\Pi^{sy}$ is EAV-secure.

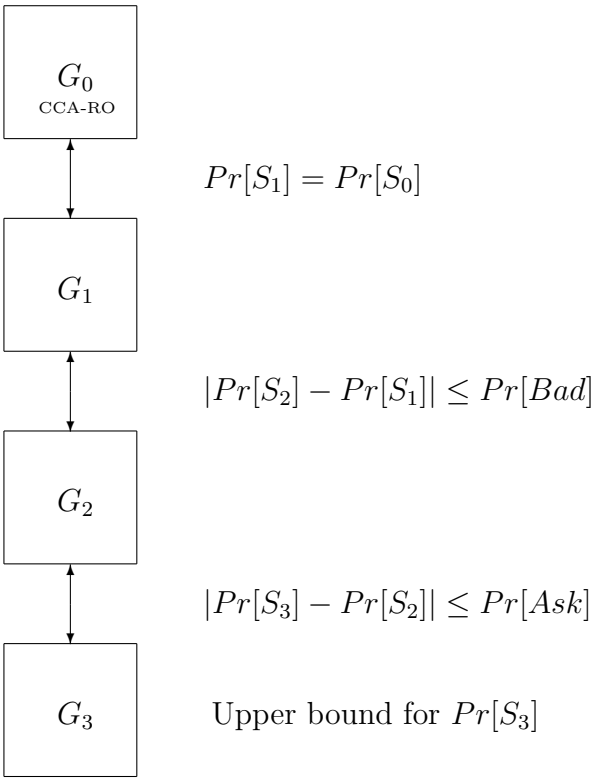


Figure 3.1: Outline of the security proof.

### Proof of Theorem 3.7

*Proof of Theorem 3.7.* We will use the game-hopping technique as described in Section 2.4.

#### Game $G_0$

The first Game $G_0$ is the CCA-RO experiment as defined in Section 2.5 and repeated in Listing 3.3. We assume without loss of generality that the adversary only submits correctly formatted ciphertexts to the decryption oracle, since we can always construct an appropriate check. Note that the random oracles are not simulated by the challenger in this game yet.

Listing 3.3: Game $G_0$ – CCA-RO

Initialization Phase:
$G \leftarrow \{\mathcal{M}^{asy} \rightarrow \mathcal{K}^{sy}\}$
$H \leftarrow \{\mathcal{M}^{asy} \times \mathcal{C}^{sy} \rightarrow \mathcal{R}^{asy}\}$
**C**: $(pk, sk) \leftarrow Gen(1^n)$

Phase 1:
$(m_0, m_1) \leftarrow \mathcal{A}^{Dec_{sk}^{fo}(\cdot), G(\cdot), H(\cdot)}(pk)$

Challenge Phase:
**C**: $b \leftarrow \{0,1\}$
$c^* \leftarrow Enc_{pk}^{fo}(m_b)$

Phase 2:
$b' \leftarrow \mathcal{A}^{Dec_{sk}^{fo}(\cdot), G(\cdot), H(\cdot)}(c^*)$

Let $\mathcal{A}^{fo}$ be a $(t^{fo}, q_{hash}, q_{dec})$-adversary that attacks $\Pi^{fo}$ in the sense of CCA-RO with advantage $\varepsilon^{fo}(n)$.

Per definition we then have

$$Pr[S_0] := \frac{1}{2} + \varepsilon^{fo}(n)$$

and therefore

$$\varepsilon^{fo}(n) = Pr[S_0] - \frac{1}{2}. \tag{3.1}$$

In our next steps we will modify the games using transitions based on failure events and one bridging step. We will then find an upper-bound for $Pr[S_0]$ and hereby also an upper bound for $\varepsilon^{fo}(n)$.

**Game** $G_1$

From this game onwards we will use the challenger $\mathbf{C}$ to simulate the random oracles $G$ and $H$ as on-demand ROs. All queries to the ROs are answered by the challenger. Therefore our first step is to adapt the Initialization Phase of the experiment: Instead of picking two random functions for $G$ and $H$, we initialize two empty query answer lists $\mathcal{G}$ and $\mathcal{H}$ as in Listing 3.4.

Listing 3.4: Initialization Phase of Game $G_1$

```
Initialization Phase:
  C : G := ∅
      H := ∅
      (pk, sk) ← Gen(1^n)
```

These query answer lists are gradually filled with every oracle query the challenger receives. Note, that the challenger needs to simulate two oracle queries for himself when creating the challenge ciphertext. These queries are not saved in $\mathcal{G}$ and $\mathcal{H}$, but need to be taken into account when answering the adversary's oracle queries. Whenever the challenger receives a query $r$ to $G$, the challenger answers as follows:

- If $r$ was queried to the oracle before, the same answer $k$ is returned.
- If $r$ was neither submitted by the adversary nor used by the challenger when creating the challenge, a new answer $k$ is sampled uniformly at random and the tuple $(r, k)$ is added to $\mathcal{G}$. Finally, $k$ is returned.

The oracle $H$ is simulated analogous. Figure 3.2 shows the simulation of $G$ answering to some queries.

Both Games – $G_0$ and $G_1$ – are equivalent from the adversary's point of view. The ROs are black boxes that return a random answer on a fresh query and that will return the same answer when a query is re-submitted. Since this is only a conceptual change, we have the same probability distribution as in Game $G_0$ and therefore

$$Pr[S_0] = Pr[S_1]. \tag{3.2}$$

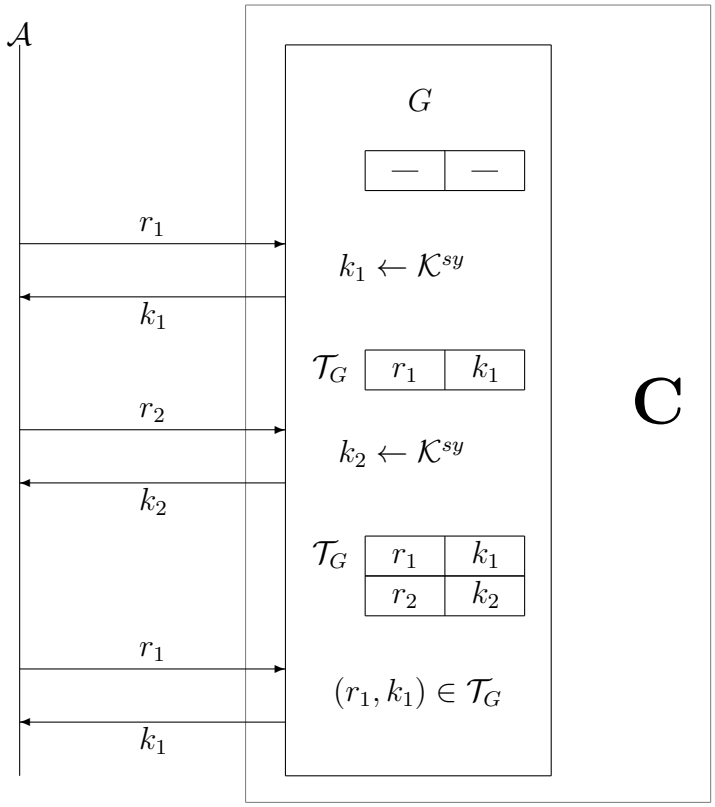


Figure 3.2: Example of some queries to an simulated on-demand oracle $G$.

**Game** $G_2$

This Game works exactly like $G_1$, except that we will now construct and use a decryption oracle that works without a secret key. A comparison between both versions of the decryption oracle can be seen in Listings 3.5 and 3.6.

Listing 3.5: Oracle $Dec^{fo}_{sk}(\cdot)$

```
Dec^fo_sk(c_m ‖ c_r){
  r' := Dec^asy_sk(c_r)
  if (r' ∉ M^asy) then
    return ⊥
  endif
  h' := H(r', c_m)
  c'_r := Enc^asy_pk(r'; h')
  if (c'_r ≠ c_r) then
    return ⊥
  endif
  k' := G(r')
  return Dec^sy_k'(c_m)
}
```

Listing 3.6: Oracle $Dec^{fo\prime}(\cdot)$

```
Dec^fo'(c_m ‖ c_r){
  Look up (r', c_m, h') ∈ H
    such that c_r = Enc^asy_pk(r'; h')
  if (no such tuple) then
    return ⊥
  endif
  k' := G(r')
  return Dec^sy_k'(c_m)
}
```

Remember that $Dec^{asy}_{sk}(c_r)$ deterministic. If $c_r$ is valid, there is exactly one $r' \in \mathcal{M}^{asy}$ which can fulfill the condition $c_r = Enc^{asy}_{pk}(r'; h')$ for some $h' \in \mathcal{R}^{asy}$. As such, there is at most one tuple $(r', c_m, h') \in \mathcal{H}$ which fulfills the lookup conditions if the adversary queried $H$ for $(r', c_m)$ before. If the adversary did not query $H$ for these values, the lookup fails and $c_m \parallel c_r$ gets rejected even if it would be valid.

Therefore, the new decryption oracle is not a perfect simulation. The original decryption oracle would submit $(r', c_m)$ to the oracle $H$ and receive a freshly sampled answer $h'$ in return. Since $c_m \parallel c_r$ is valid, the decryption oracle would then continue with the decryption and return the corresponding message $m'$ as displayed in Figure 3.3. Therefore, we define the following failure event:

**Definition 3.8** (*Bad*)**.** *The event Bad is the case that the adversary submits a valid ciphertext $c_m \parallel c_r$ to the decryption oracle whereas there is no tuple $(r, c_m, h)$ in the query-answer list $\mathcal{H}$.*

$\mathcal{A}$ is only able to generate such ciphertexts with a very small probability. One can also describe this as the adversary trying to guess a valid combination $(r, c_m, h)$ such that $h = H(r, c_m)$. The underlying asymmetric encryption scheme is $\gamma$-spread and as such there are at least $2^\gamma$ possible values for $H(r, c_m)$. Also the adversary may submit up to $q_{Dec}$ decryption oracle queries – we therefore state the following Lemma.

**Lemma 3.9.** *The probability of event Bad is $Pr[Bad] \leq 2^{-\gamma} q_{Dec}$.*

The proof of Lemma 3.9 will be delayed until later. Both Games – $G_1$ and $G_2$ – have the same probability distribution and proceed identically until the event $Bad$ occurs. We therefore have

$$S_1 \wedge \neg Bad \Leftrightarrow S_2 \wedge \neg Bad$$

and can apply the Difference Lemma (Lemma 2.22):

$$|Pr[S_2] - Pr[S_1]| \leq 2^{-\gamma} q_{Dec}. \tag{3.3}$$

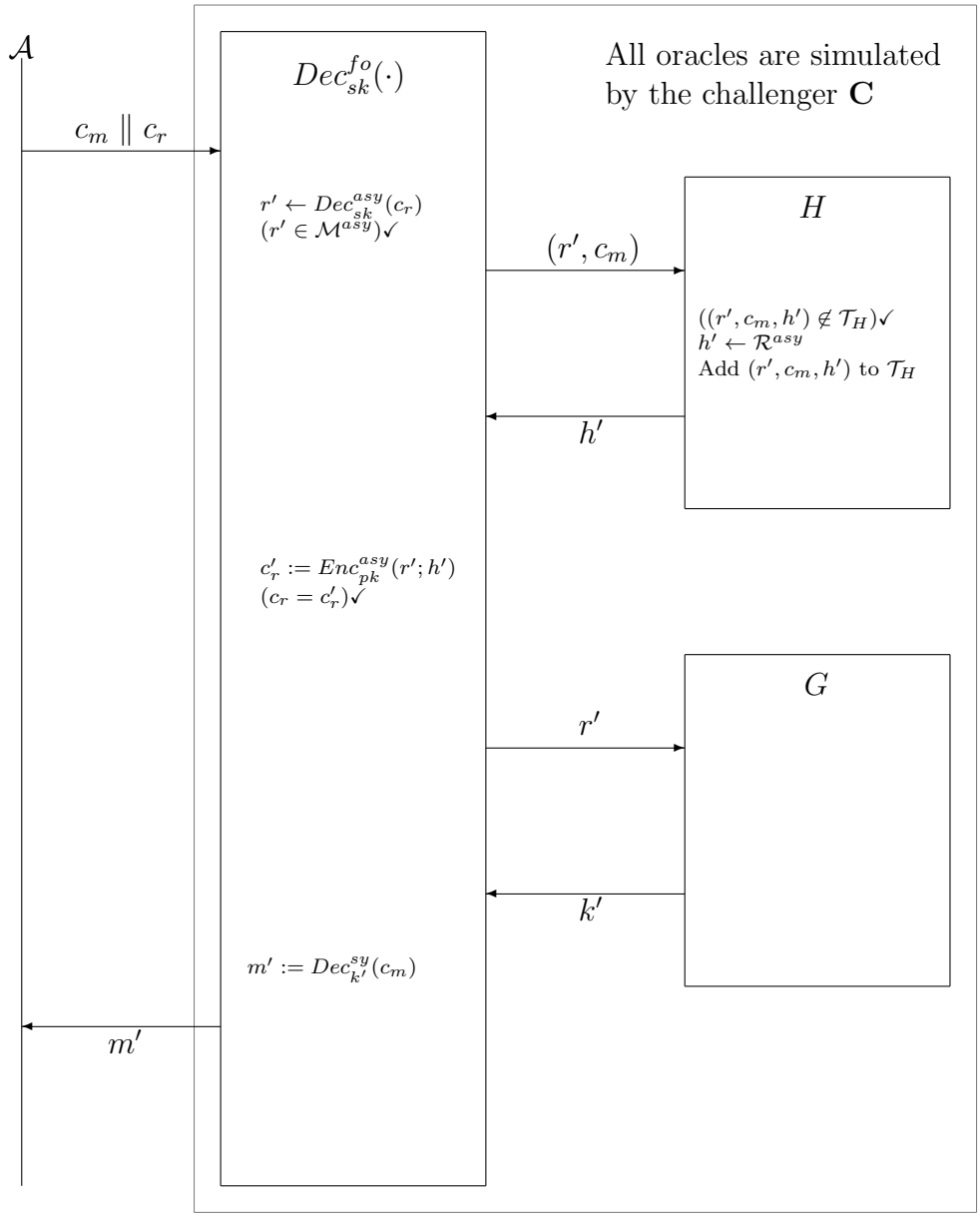


Figure 3.3: Event *Bad* in Game $G_1$.

**Game** $G_3$

In this game we will create the challenge ciphertext $c^* = c^*_m \parallel c^*_r$ without using the ROs. In all other aspects, Game $G_2$ and $G_3$ are exactly the same. Listings 3.7 and 3.8 show the difference between the challenge ciphertexts. In Game $G_2$ we had

$$c^*_m = Enc^{sy}_{k^*}(m_b) \text{ and } c^*_r = Enc^{asy}_{pk}(r^*; h^*),$$

whereas $k^*$ and $h^*$ were obtained by simulating two oracle queries

$$k^* := G(r^*) \text{ and } h^* := H(r^*, c^*_m).$$

Instead of doing these queries, we define two new variables – $\hat{k}$ and $\hat{h}$ – and pick their values uniformly at random from the corresponding sets.

Listing 3.7: Game $G_1$, Challenge

$\mathbf{C}$: $b \leftarrow \{0,1\}$
$r^* \leftarrow \mathcal{M}^{asy}$
$k^* := G(r)$
$c^*_m \leftarrow Enc^{sy}_k(m_b)$
$h^* := H(r, c_m)$
$c^*_r := Enc^{asy}_{pk}(r^*; h^*)$
$c^* := c^*_m \parallel c^*_r$

Listing 3.8: Game $G_2$, Challenge

$\mathbf{C}$: $b \leftarrow \{0,1\}$
$r^* \leftarrow \mathcal{M}^{asy}$
$\hat{k} \leftarrow \mathcal{K}^{sy}$ // **not** $G(r^*)$
$c^*_m \leftarrow Enc^{sy}_{\hat{k}}(m_b)$
$\hat{h} \leftarrow \mathcal{R}^{asy}$ // **not** $H(r^*, c^*_m)$
$c^*_r := Enc^{asy}_{pk}(r^*; \hat{h})$
$c^* := c^*_m \parallel c^*_r$

This change drops the consistency of the ROs: Since $\hat{k}$ and $\hat{h}$ are picked at random, we have $\hat{k} = G(r^*) \wedge \hat{h} = H(r^*, c^*_m)$ with only negligible probability. We omit this good case in our simulation and define the following failure event:

**Definition 3.10** (Ask)**.** *Ask is the event that the adversary queries $G$ for $r^*$ or that the adversary queries $H$ for $(r^*, c^*_m)$.*

We also state the following Lemma and will give proof of this lemma later.

**Lemma 3.11.** *Let $\Pi^{asy}$ be a $(t^{asy}, \varepsilon^{asy})$-OWE asymmetric encryption scheme, where $t^{asy}(n) \geq t^{fo}(n) + q_{Hash}O(n) + q_{Dec}T^{asy}(n)$. Then $Pr[Ask] \leq q_{hash}\varepsilon^{asy}(n)$.*

Games $G_2$ and $G_3$ are identical except for the challenge ciphertext. Both simulations proceed identically except in the case that the event $Ask$ occurs. Therefore we have

$$G_2 \wedge \neg Ask \Leftrightarrow G_3 \wedge \neg Ask$$

and can apply the Difference Lemma (Lemma 2.22. Combined with Lemma 3.11 we then get

$$|Pr[S_3] - Pr[S_2]| \leq q_{hash}\varepsilon^{asy}(n). \tag{3.4}$$

□

### Bounding $t^{fo}$ and $\varepsilon^{fo}$

*Bounding $t^{fo}$ and $\varepsilon^{fo}$.* Using Equations 3.2, 3.3, and 3.4 we finally can estimate the difference between $Pr[S_3]$ and $Pr[S_0]$ as

$$|Pr[S_3] - Pr[S_0]| \leq q_{hash}\varepsilon^{asy}(n) + 2^{-\gamma}q_{Dec}. \tag{3.5}$$

We will also state an upper-bound for $Pr[S_3]$ via the following Lemma; we will prove this upper bound later.

**Lemma 3.12.** *Let $\Pi^{sy}$ be a $(t^{sy}, \varepsilon^{sy})$-EAV secure encryption scheme, where $t^{sy} \geq t^{fo} + q_{hash}O(n) + (q_{dec}+1)T^{asy}(n)$. Then, we have $Pr[S_3] \leq \frac{1}{2} + \varepsilon^{sy}(n)$.*

By Equation 3.5 and Lemma 3.12, we can find an upper-bound for $Pr[S_0]$:

$$Pr[S_0] \leq \frac{1}{2} + \varepsilon^{sy}(n) + q_{Hash}\varepsilon^{asy}(n) + 2^{-\gamma}q_{Dec}$$

We then can find an upper bound for $\varepsilon^{fo}(n)$ from Equation 3.1:

$$\begin{aligned}\varepsilon^{fo}(n) &= Pr[S_0] - \frac{1}{2} \\ &\leq \frac{1}{2} + \varepsilon^{sy}(n) + q_{hash}\varepsilon^{asy}(n) + 2^{-\gamma}q_{Dec} - \frac{1}{2} \\ &= \varepsilon^{sy}(n) + q_{hash}\varepsilon^{asy}(n) + 2^{-\gamma}q_{Dec}\end{aligned}$$

After bounding $\varepsilon^{fo}$, we must find a bound for $t^{fo}$. Lemmas 3.11 and stated that

$$\begin{aligned}t^{asy}(n) &\geq t^{fo}(n) + q_{Hash}O(n) + q_{Dec}T^{asy}(n) \text{ and} \\ t^{sy}(n) &\geq t^{fo}(n) + q_{hash}O(n) + (q_{dec}+1)T^{asy}(n).\end{aligned}$$

To fulfill both inequations, we must have

$$\min\left(t^{asy}(n), t^{sy}(n)\right) \geq t^{asy}(n) + q_{Hash}O(n) + (q_{Dec}+1)T^{asy}(n),$$

which is equivalent to

$$t^{fo}(n) \leq \min\left(t^{asy}(n), t^{sy}(n)\right) - q_{hash}O(n) - (q_{dec}+1)T^{asy}(n).$$

□

### Proofs of Lemmas 3.9, 3.11, and 3.12

In Game $G_1$ we constructed a decryption oracle without using the secret key which rejects valid ciphertexts that were created without asking oracle $H$. We hereby introduced the failure event $Bad$:

**Definition 3.8 ($Bad$)** *The event Bad is the case that the adversary submits a valid ciphertext $c_m \parallel c_r$ to the decryption oracle whereas there is no tuple $(r, c_m, h)$ in the query-answer list $\mathcal{H}$.*

We then stated, that the adversary is able to create such ciphertexts only with a very low probability – depending on the $\gamma$-spread of the underlying asymmetric encryption scheme:

**Lemma 3.9** *The probability of event Bad is $Pr[Bad] \leq 2^{-\gamma} q_{Dec}$.*

In the following proof, we will use the fact that the adversary must not submit the challenge ciphertext to the decryption oracle. The event $Bad$ is then equivalent to the event that "the adversary correctly guesses a tuple $(r, c_m, h)$ such that $h = H(r, c_m)$". For convenience, we repeat the creation of the challenge ciphertext in Listing 3.9.

Listing 3.9: Challenge

$\mathbf{C}$: $b \leftarrow \{0,1\}$
$r^* \leftarrow \mathcal{M}^{asy}$
$k^* := G(r)$
$c_m^* \leftarrow Enc_k^{sy}(m_b)$
$h^* := H(r, c_m)$
$c_r^* := Enc_{pk}^{asy}(r^*; h^*)$
$c^* := c_m^* \parallel c_r^*$

*Proof of Lemma 3.9.* We will show that the probability of the event that the adversary submits a valid ciphertext $c_m \parallel c_r$ to the decryption oracle such that

$$c_r = Enc_{pk}^{asy}(r; h) \text{ and } h = H(r, c_m)$$

is bounded by

$$Pr[Bad] \leq 2^{-\gamma} q_{Dec},$$

when the adversary does not query $H$ for $(r, c_m)$. The adversary is not allowed to submit the challenge ciphertext $c^* = c_m^* \parallel c_r^*$ (see Listing 3.9). Therefore we must have

$$(c_m, c_r) \neq (c_m^*, c_r^*)$$

for all ciphertexts that the adversary submits. We will now show, that this implies

$$r \neq r^* \vee c_m \neq c_m^*$$

for all of the adversary's decryption queries.

If we have $c_r = c_r^*$, we must have $c_m \neq c_m$. If we have $c_m = c_m^*$, we must have $c_r \neq c_r^*$. The latter implies that we have $r \neq r^*$ which will be proven via contradiction. Assume that we have $r = r^*$. We then have $(r, c_m) = (r^*, c_m^*)$ which implies that $(r, H(r, c_m)) = (r^*, H(r^*, c_m^*))$. Since the encryption algorithm is deterministic when given pre-sampled random coins, we would have

$$c_r = Enc_{pk}^{asy}(r; H(r, c_m)) = Enc_{pk}^{asy}(r^*, H(r^*, c_m^*)) = c_r^*.$$

This is a contradiction to our assumptions, and therefore we must have $r \neq r^*$ in the case that $c_m = c_m^*$. Therefore all decryption oracle queries fulfill the condition

$$r \neq r^* \vee c_m \neq c_m^* \Leftrightarrow (r, c_m) \neq (r^*, c_m^*).$$

Remember now, that $H$ is modeled as a RO. The answer to query $(r, c_m)$ is sampled independently of the answer to query $(r^*, c_m^*)$. The adversary tries to construct a valid $c_r$ without querying $H$, i.e. he tries to guess a tuple $(r, c_m, h)$ such that $c_r$ is valid. Since $H$ is modeled as a RO, the value $h = H(r, c_m)$ is random from the adversary's point of view:

$$Pr[c_r = Enc_{pk}^{asy}(r; h) \wedge h = H(r, c_m)] \leq Pr[c_r = Enc_{pk}^{asy}(r; h) : h \leftarrow \mathcal{R}^{asy}] \quad (3.6)$$

Remember that $\Pi^{asy}$ is $\gamma$-spread. Corollary 2.13 stated that for all $(pk, sk) \in [Gen(1^n)]$ and all $r \in \mathcal{M}^{asy}$, we have at least $2^\gamma$ possible encryptions which occur with probability

$$Pr[c_r = Enc_{pk}(r; h) : h \leftarrow \mathcal{R}] \leq 2^{-\gamma} q_{Dec}.$$

We can use this fact to find an upper bound for Inequality 3.6.

$$Pr[c_r = Enc_{pk}^{asy}(r; h) : h \leftarrow \mathcal{R}^{asy}] \leq 2^{-\gamma}. \quad (3.7)$$

Note that the bound of Inequality 3.7 is for the case that the adversary submits exactly *one* query to the decryption oracle. The adversary is allowed to submit $q_{Dec}$ queries and may choose to submit a different ciphertext on each query. Therefore an upper bound for $Pr[Bad]$ is

$$Pr[Bad] \leq 2^{-\gamma} q_{Dec}.$$

□

In Game $G_3$, the two parts of the challenge ciphertext – $c_m^*$ and $c_r^*$ – were made independent of each other. This was accomplished by creating the challenge without querying the oracles $G$ and $H$ as in Listing 3.10.

Listing 3.10: Challenge of $G_3$ with independent $c_m^*$ and $c_r^*$

$\mathbf{C}$: $b \leftarrow \{0,1\}$
$r^* \leftarrow \mathcal{M}^{asy}$
$\hat{k} \leftarrow \mathcal{K}^{sy}$ // **not** $G(r^*)$
$c_m^* \leftarrow Enc_{\hat{k}}^{sy}(m_b)$
$\hat{h} \leftarrow \mathcal{R}^{asy}$ // **not** $H(r^*, c_m^*)$
$c_r^* := Enc_{pk}^{asy}(r^*; \hat{h})$
$c^* := c_m^* \parallel c_r^*$

We hereby dropped the consistency of the ROs and introduced the failure event $Ask$:

**Definition 3.10 ($Ask$)** *$Ask$ is the event that the adversary queries $G$ for $r^*$ or that the adversary queries $H$ for $(r^*, c_m^*)$.*

We then stated the following Lemma:

**Lemma 3.11** *Let $\Pi^{asy}$ be a $(t^{asy}, \varepsilon^{asy})$-OWE asymmetric encryption scheme, where $t^{asy}(n) \geq t^{fo}(n) + q_{Hash}O(n) + q_{Dec}T^{asy}(n)$. Then $Pr[Ask] \leq q_{hash}\varepsilon^{asy}(n)$.*

To prove this bound, we will construct an adversary attacking an asymmetric scheme in the sense of OWE from an adversary that attacks the FO scheme in the sense of CCA-RO.

*Proof of Lemma 3.11.* Let $\Pi^{asy}$ be a $(t^{asy}, \varepsilon^{asy})$-OWE secure, asymmetric encryption scheme. Let $\mathcal{A}^{fo}$ be a $(t^{fo}, q_{Dec}, q_{Hash})$ adversary attacking $\Pi^{fo}$ in the sense of CCA-RO that succeeds with probability $\frac{1}{2} + \varepsilon^{fo}(n)$. Denote by $\mathcal{Q}$ the list of adversary queries made to both oracles $G$ and $H$. Every query that is submitted by the adversary to one of these oracles is added to $\mathcal{Q}$.

We will now construct an adversary $\mathcal{A}^{owe}$ attacking $\Pi^{asy}$ in the sense of OWE. $\mathcal{A}^{owe}$ will simulate the random oracles $G$ and $H$ and the decryption oracle as in Game $G_3$. $\mathcal{A}^{owe}$ first runs $\mathcal{A}^{cca-ro}(pk)$ to obtain two messages $m_0$ and $m_1$.

We then have to adapt the challenge to the CCA-RO experiment as in Listing 3.11. First, a one-time key $k^*$ and a bit $b$ are picked uniformly at random, then one of the two messages is encrypted symmetrically using $k^*$. The concatenation of $\mathcal{A}^{owe}$'s original challenge $c_m^*$ with $c_r^*$ then results in a challenge that is constructed as in Game $G_3$.

Listing 3.11: Adapting the challenge

$k^* \leftarrow \mathcal{K}^{sy}$
$b \leftarrow \{0,1\}$
$c_m^* \leftarrow Enc_{k^*}^{sy}(m_b)$
$c^* := c_m^* \parallel c_r^*$

After $\mathcal{A}^{cca-ro}$ outputs a bit $b'$, we generate a solution for the OWE experiment as follows:

1. Pick one of the queries that $\mathcal{A}^{cca-ro}$ made to the random oracles uniformly at random, i.e. $q \leftarrow \mathcal{Q}$.

2. $q$ is either of the form $(r', G(r'))$ or $(r', c'_m, H(r', c'_m))$. Pick the answer $r'$ as in that query.

The construction of $\mathcal{A}^{owe}$ is also depicted in Figure 3.4.

$\mathcal{A}^{owe}$ will only output a $r'$ that was contained in a query to one of the random oracles. The corresponding query is picked uniformly at random, and therefore we have

$$Pr[r = r' \mid Ask] = \frac{1}{q_{Hash}},$$

which implies that

$$Pr[r = r'] = \frac{1}{q_{Hash}} Pr[Ask]. \tag{3.8}$$

Because $\Pi^{asy}$ is $(t^{asy}, \varepsilon^{asy})$-OWE secure, we have an upper bound for Equation 3.8:

$$Pr[r = r'] = \frac{1}{q_{Hash}} Pr[Ask] \leq \varepsilon^{asy}(n), \tag{3.9}$$

which gives us an upper bound for $Pr[Ask]$:

$$Pr[Ask] \leq q_{Hash} \cdot \varepsilon^{asy}(n). \tag{3.10}$$

We also can estimate the minimum running time $t^{asy}$ of $\mathcal{A}^{owe}$: $\mathcal{A}^{cca-ro}$ is run completely, therefore $t^{fo}$ is included in the runtime. Whenever a query fresh is made to one of the oracles $G$ or $H$ we have to sample a response which takes up $O(n)$ time – since $q_{Hash}$ queries are made, we add $q_{Hash}O(n)$ to $t^{asy}$. Finally, $q_{Dec}$ decryption queries are made. As we are using the decryption oracle derived in Game $G_1$, for queries to the decryption oracle $c'_r = Enc_{pk}^{asy}(r; h)$ has to be computed and compared to the given $c_r$; we have to add $q_{Dec}T^{asy}(n)$ to $t^{asy}$.

$$\Rightarrow t^{asy}(n) \geq t^{fo}(n) + q_{Hash}O(n) + q_{Dec}T^{asy}(n) \tag{3.11}$$

□

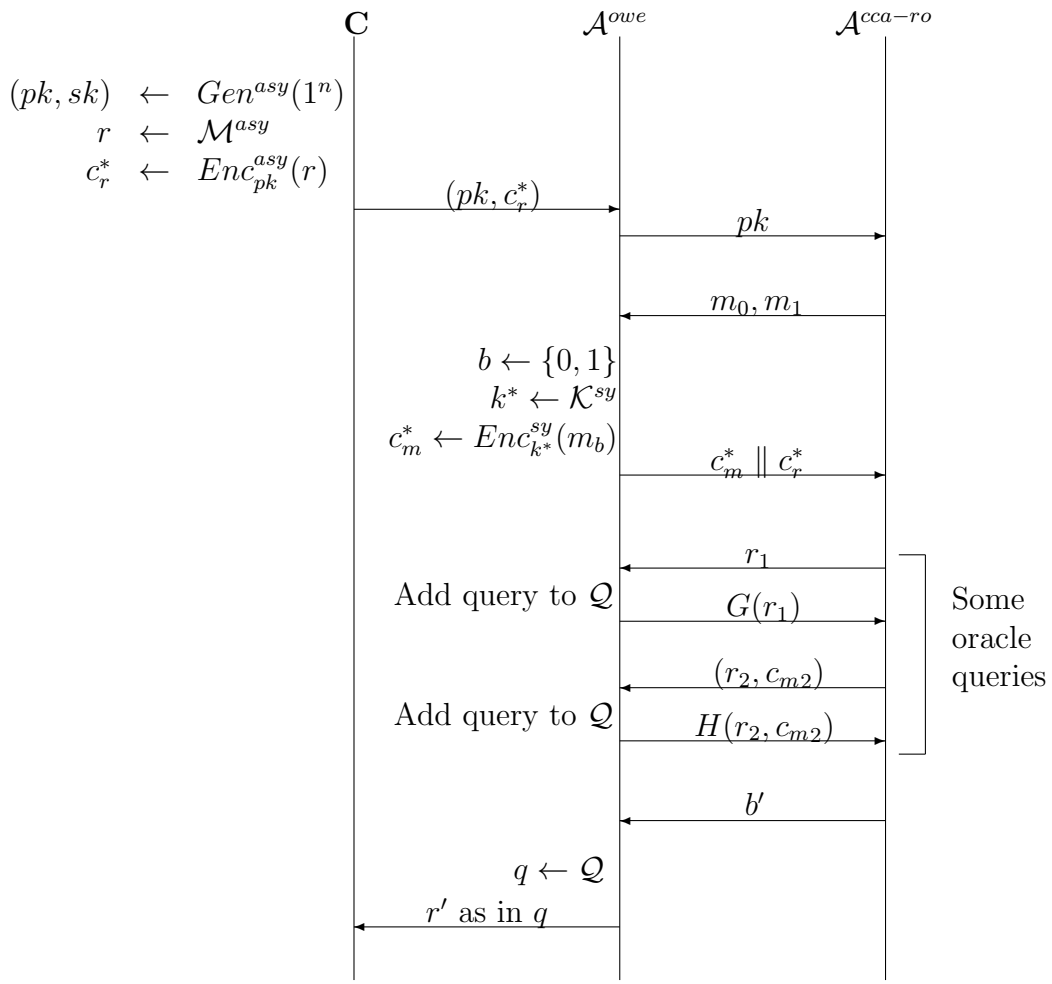


Figure 3.4: Construction of an OWE adversary from a CCA-RO adversary

Finally we will complete our proof by proving Lemma 3.12.

**Lemma 3.12** *Let $\Pi^{sy}$ be a $(t^{sy}, \varepsilon^{sy})$-EAV secure encryption scheme, where $t^{sy} \geq t^{fo} + q_{hash}O(n) + (q_{dec}+1)T^{asy}(n)$. Then, we have $Pr[S_3] \leq \frac{1}{2} + \varepsilon^{sy}(n)$.*

We will construct an adversary attacking a symmetric scheme in the sense of EAV that uses a CCA-RO adversary attacking the FO scheme. We can use this construction to find the upper-bounds of Lemma 3.12. For convenience, the symmetric eavesdropping indistinguishability and Game $G_3$ are given in Listings 3.12 and 3.13.

Listing 3.12: $PrivK^{eav}_{\mathcal{A}^{eav},\Pi^{sy}}(n)$

Init :
  $\mathbf{C} : k \leftarrow Gen^{sy}(1^n)$

Phase 1:
  $m_0, m_1 \leftarrow \mathcal{A}^{eav}(1^n)$

Challenge :
  $\mathbf{C} : b \leftarrow \{0,1\}$
    $c^*_m \leftarrow Enc^{sy}_k(m_b)$

Phase 2:
  $b' \leftarrow \mathcal{A}^{eav}(c^*_m)$

Listing 3.13: Game $G_3$

Init :
  $\mathbf{C} : (pk, sk) \leftarrow Gen^{fo}(1^n)$

Phase 1:
  $m_0, m_1 \leftarrow \mathcal{A}^{cca-ro}(pk)$

Challenge :
  $\mathbf{C} : b \leftarrow \{0,1\}$
    $r^* \leftarrow \mathcal{M}^{asy}$
    $k^* \leftarrow \mathcal{K}^{sy}$
    $c^*_m \leftarrow Enc^{sy}_{k^*}(m_b)$
    $h^* \leftarrow \mathcal{R}^{asy}$
    $c^*_r := Enc^{asy}_{pk}(r^*; h^*)$
    $c^* := c^*_m \parallel c^*_r$

Phase 2:
  $b' \leftarrow \mathcal{A}^{cca-ro}(c^*)$

*Proof of Lemma 3.12.* Let $\Pi^{sy}$ be a $(t^{sy}, \varepsilon^{sy})$-EAV secure, symmetric encryption scheme. Let $\mathcal{A}^{fo}$ be a $(t^{fo}, q_{Dec}, q_{Hash})$ adversary attacking $\Pi^{fo}$ in the sense of CCA-RO that succeeds with probability $\frac{1}{2} + \varepsilon^{fo}(n)$. Construct an adversary $\mathcal{A}^{eav}$ as shown in Figure 3.5.

- On input $1^n$, $\mathcal{A}^{eav}$ generates a key-pair: $(pk, sk) \leftarrow Gen(1^n)$. Then $\mathcal{A}^{eav}$ runs $\mathcal{A}^{cca-ro}(pk)$ to obtain two messages $m_0$ and $m_1$ which are passed to the challenger.

- On input $c^*_m$ the adversary $\mathcal{A}^{eav}$ adapts the challenge as depicted in Figure 3.5 and then runs $\mathcal{A}^{cca-ro}(c^*)$. When $\mathcal{A}^{cca-ro}$ halts and outputs a bit $b'$, the adversary $\mathcal{A}^{eav}$ outputs the same bit $b'$. The oracles are simulated as in Phase 1.

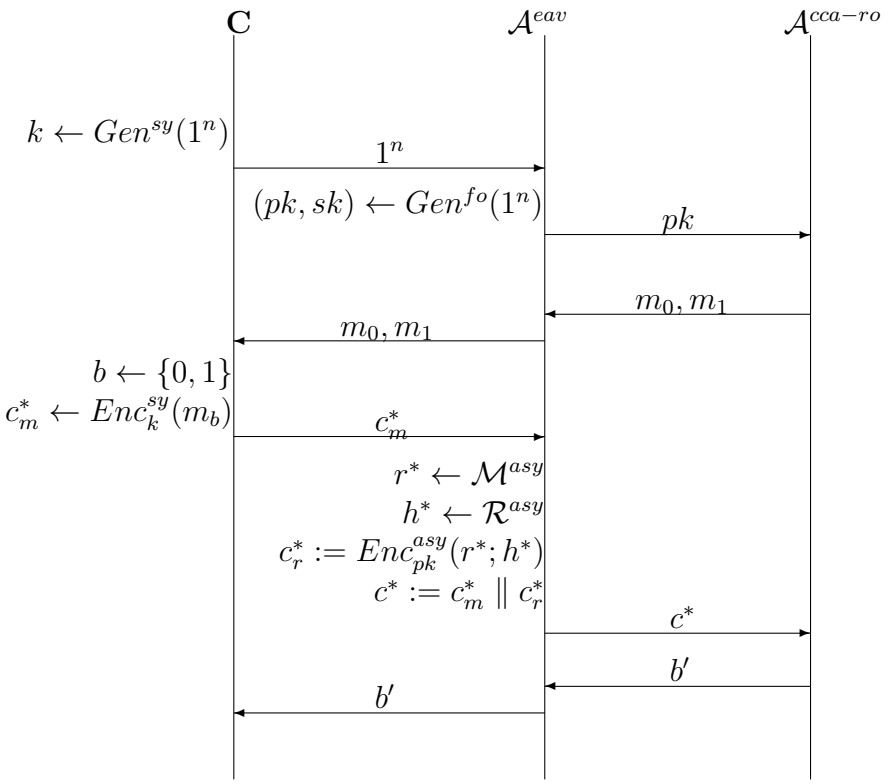


Figure 3.5: An EAV-adversary constructed from a CCA-RO adversary

$\mathcal{A}^{eav}$ will only output the correct bit $b'$ when $\mathcal{A}^{fo}$ outputs $b'$. Because $\Pi^{sy}$ is $(t^{sy}, \varepsilon^{sy})$-EAV, there is an upper bound for $Pr[b' = b]$ and we have:

$$Pr[b' = b] = \frac{1}{2} + \varepsilon^{fo}(n) \leq \frac{1}{2} + \varepsilon^{sy}(n) \tag{3.12}$$

Per construction, $\mathcal{A}^{fo}$'s perspective is the same as in Game $G_3$, therefore we have

$$Pr[S_3] = \frac{1}{2} + \varepsilon^{fo}(n) \leq \frac{1}{2} + \varepsilon^{sy}(n). \tag{3.13}$$

We first compute a lower bound for $t^{sy}$. $\mathcal{A}^{cca-ro}$ needs to be executed once, therefore the runtime has to be at least $t^{fo}$. We need to execute the asymmetric encryption algorithm ($q_{Dec}$+1) times because $\mathcal{A}^{cca-ro}$ queries the decryption oracle $q_{Dec}$ times and we had to adapt the challenge. Therefore we need to add $(q_{Dec} + 1)T^{asy}(n)$ to the runtime. As $\mathcal{A}^{cca-ro}$ may query the random oracles $q_{Hash}$ times, $q_{Hash}$ answers need to be sampled and $q_{Hash}O(n)$ is added to the runtime. The total runtime of $\mathcal{A}^{eav}$ therefore is bounded as

$$t^{sy}(n) \geq t^{fo}(n) + q_{Hash}O(n) + (q_{Dec} + 1)T^{asy}(n).$$

□

## 3.3 Different Versions of the FOT

The FOT was introduced in [FO99]. A revised version was then given in [FO13]. This section will outline and explain the differences between the versions introduced by Fujisaki and Okamoto, and the version used in this thesis.

**Construction** There are some differences between the original FOT in [FO99] and the revised version in [FO13]. Because the changes are affecting each other – one cannot change the definition of the hash functions without adapting encryption and decryption – all differences will be listed before being explained.

The first difference is shown in Table 3.1: In the original version [FO99], ROs were defined over sets like $\mathcal{M}^{asy}$ and $\mathcal{M}^{sy}$. The ROs in the revised version always used $\{0,1\}^*$. The definite sets have been reintroduced in this thesis.

Table 3.1: Hashfunctions used in the different versions of the FOT

| | |
|---|---|
| First version [FO99] | $G : \mathcal{M}^{asy} \to \mathcal{K}^{sy}$<br>$H : \mathcal{M}^{asy} \times \mathcal{M}^{sy} \to \mathcal{R}^{asy}$ |
| Revised version [FO13] | $G : \{0,1\}^* \to \mathcal{K}^{sy}$<br>$H : \{0,1\}^* \times \{0,1\}^* \to \mathcal{R}^{asy}$ |
| Thesis version | $G : \mathcal{M}^{asy} \to \mathcal{K}^{sy}$<br>$H : \mathcal{M}^{asy} \times \mathcal{C}^{sy} \to \mathcal{R}^{asy}$ |

Another difference is found in the definition of the Random Oracle $H$: Instead of hashing the one-time key with the plaintext message, the key is hashed with the encrypted message. Note, that the difference between [FO13] and the definition used in this thesis is just of technical nature.

The decryption algorithm of the revised FOT had to be adapted to the encryption algorithm. The oracles were now defined for sets like $\{0,1\}^*$. Therefore, we now need to check that $Dec_{sk}^{asy}(c_r) \in \mathcal{M}^{asy}$ – the oracle $H$ alone would not reject this invalid value. In our version, we kept this check although it is implicitly given in the definition of $H$.

**Definition of encryption schemes** Both [FO99, 540] and [FO13, 85f] do not specify how the decryption algorithms of the underlying schemes react to invalid ciphertexts. In this thesis however we stated that the decryption algorithms may output a symbol $\perp \notin \mathcal{M}$ to denote failure (see Definitions 2.5 and 2.6).

Listing 3.14: Original Encryption

$Enc^{fo}_{pk}(m)\{$
$\quad r \leftarrow \mathcal{M}^{asy}$
$\quad k := G(r)$
$\quad c_m \leftarrow Enc^{asy}_{k}(m)$
$\quad h := H(r, m)$
$\quad c_r := Enc^{asy}_{pk}(r; h)$
$\quad \texttt{return}\ \ c := c_m \parallel c_r$
$\}$

Listing 3.15: Original Decryption

$Dec^{fo}_{sk}(c_m \parallel c_r)\{$
$\quad r' := Dec^{asy}_{sk}(c_r)$
$\quad k' := G(r')$
$\quad m' := Dec^{sy}_{k'}(c_m)$
$\quad h' := H(r', m')$
$\quad c_r' := Enc^{asy}_{pk}(r'; h')$
$\quad \text{if } (c_r \neq c_r'), \ \texttt{return}\ \perp$
$\quad \texttt{return}\ \ m'$
$\}$

Listing 3.16: Revised Encryption

$Enc^{fo}_{pk}(m)\{$
$\quad r \leftarrow \mathcal{M}^{asy}$
$\quad k := G(r)$
$\quad c_m \leftarrow Enc^{sy}_{k}(m)$
$\quad h := H(r, c_m)$
$\quad c_r := Enc^{asy}_{pk}(r; h)$
$\quad \texttt{return}\ \ c := c_m \parallel c_r$
$\}$

Listing 3.17: Revised Decryption

$Dec^{fo}_{sk}(c_m \parallel c_r)\{$
$\quad r' := Dec^{asy}_{sk}(c_r)$
$\quad \text{if } (r' \notin \mathcal{M}^{asy}), \ \texttt{return}\ \perp$
$\quad h' := H(r, c_m)$
$\quad c_r{}' := Enc^{asy}_{pk}(r'; h')$
$\quad \text{if } (c_r \neq c_r{}'), \ \texttt{return}\ \perp$
$\quad k' := G(r')$
$\quad \texttt{return}\ \ Dec^{sy}_{k'}(c_m)$
$\}$

**Security properties** It is mentioned in [FO13, 84] that the original FOT from [FO99] was only designed for a deterministic and bijective symmetric encryption scheme. After removing this restriction in [FO13], the FOT can now be applied to arbitrary eavesdropping secure symmetric encryption schemes. The notion of Plaintext Awareness (PA) is also mentioned. Informally this security notion means that "an adversary cannot create ciphertexts without knowing the corresponding plaintext"[BR94, 101]. Plaintext Awareness was first introduced in [BR94] and later refined in [BDPR98] to fix some inaccuracies. It is stated in [FO13, 84] that the FO scheme obtained by the original FOT is PA-secure whereas the revised FO scheme is not.

## 3.4 Comparison of Security Proofs

There are some differences between the proof in [FO13] and the proof presented in this paper. The first difference is caused by different definitions of the security properties (see Section 2.3). While [FO13] uses

$$2Pr[\mathcal{A} \text{ wins a given experiment } Exp] - 1 \leq \varepsilon(n),$$

this thesis uses

$$Pr[\mathcal{A} \text{ wins a given experiment } Exp] \leq \frac{1}{2} + \varepsilon(n)'.$$

A major change was done to the order of games. Our first game is the same as in [FO13]. We then switch to simulate the oracles $G$ and $H$ by the challenger. This transition is made last in [FO13] and it is not stated if the ROs are simulated by the challenger. The motivation behind this change was that the failure events can be explained in a better way.

After switching to simulated ROs we constructed a decryption oracle that works without a secret key and identified the failure event *Bad*. This event was also used in [FO13] but in a slightly different way. [FO13] first needed to introduce the notion of almost-valid ciphertexts because all ROs used $\{0,1\}^*$ as domain and the result of invoking the decryption algorithm on invalid ciphertexts was not specified [FO13, 93].

**Definition 3.13** (Almost-valid ciphertext [FO13, 92])**.** *A correctly formed ciphertext $c_m \parallel c_r$ is called almost-valid with respect to $(pk, sk) \in \left[Gen^{fo}(1^n)\right]$ on $\Pi^{hy}$, if and only if we have*

$$c_r = Enc^{asy}_{pk}(r; H(r, c_m)), \text{ where } r := Dec^{asy}_{sk}(c_r).$$

The decryption of a invalid $c_r$ could result in a valid $r'$ in [FO13] and needed to be further investigated. In this thesis however, the decryption algorithms of the underlying asymmetric scheme may output a failure symbol $\perp \notin \mathcal{M}^{asy}$. As decryption algorithms are deterministic, $\perp$ means that the given ciphertext $c_r$ is invalid under the assumption that $sk$ is correct. We therefore do not need the notion of an almost-valid ciphertext in this thesis and could simplify the definition of *Bad*.

In our last game hop, we created the challenge ciphertext without querying the oracles and identified the failure event *Ask*. This transition was divided into two game hops in [FO13]. In the first of these steps, the oracles $G$ and $H$ are modified by replacing the values $G(r^*)$ with $\hat{k}$ and $H(r^*, c^*_m)$ with $\hat{h}$. The challenge ciphertext is then created such that

$$c^*_m = Enc^{sy}_{\hat{k}}(m_b) \text{ and } c^*_r = Enc^{asy}_{pk}(m_b; \hat{h}). \tag{3.14}$$

Also, the oracles are modified to answer with the replaced values, i.e. $G$ returns $\hat{k}$ if $r^*$ is queried and $H$ returns $\hat{h}$ if $(r^*, c^*_m)$ is queried. According to [FO13, 93], the oracle $G$ is replaced with $G'$ which is identical to $G$ except for the query $(r^*)$. Analogous, $H$ is replaced with $H'$. However it is not stated when exactly the oracles are replaced. The first possibility would be that the oracles are replaced in the initialization phase. This could explain the fact that $\hat{k}$ and $\hat{h}$ were called "imaginary variables" – $c_m$ cannot be known before the Challenge Phase and therefore it would be impossible to replace $H(r^*, c^*_m)$.

Another possibility would be that $G$ and $H$ are replaced right after the challenge has been created. The variables $\hat{h}$ and $\hat{k}$ are then set to the respective values

$$\hat{k} := G(r^*) \text{ and } \hat{h} := H(r^*, c_m^*).$$

In [FO13]'s final game hop, $G'$ and $H'$ then were replaced again with $G$ and $H$ but the challenge is created as in Equation 3.14. It is not specified how $\hat{h}$ and $\hat{k}$ are picked in this game. One possibility would be that they are picked uniformly at random from their corresponding domains.

### 3.4.1 Final thoughts

**Theorem 3.7** *Let $\Pi^{asy}$ be a $\gamma$-spread $(t^{asy}, \varepsilon^{asy})$-OWE secure asymmetric encryption scheme and let $\Pi^{sy}$ be a $(t^{sy}, \varepsilon^{sy})$-EAV secure symmetric encryption scheme. Denote by $T^{asy}(n)$ the worst case of the running time of $Enc_{pk}^{asy}(m)$ for every $m \in \mathcal{M}^{asy}$ and every $(pk, sk) \in [Gen^{asy}(1^n)]$. Let $\Pi^{fo}$ be the FO scheme and let $\mathcal{A}$ be a $(t^{fo}, q_{Hash}, q_{Dec})$-adversary.*

*$\Pi^{fo}$ is $(t^{fo}, q_{Hash}, q_{Dec}, \varepsilon^{fo})$-CCA secure in the Random Oracle Model where*

$$t^{fo}(n) = \min(t^{asy}(n), t^{sy}(n)) - q_{Hash}O(n) - (q_{Dec} + 1)T^{asy}(n) \text{ and} \tag{3.15}$$

$$\varepsilon^{fo}(n) = \varepsilon^{sy}(n) + q_{Hash}\varepsilon^{asy}(n) + 2^{-\gamma}q_{Dec}. \tag{3.16}$$

One can use the Theorem repeated above and Lemmas 3.11 and 3.12 to estimate what schemes need to be used in the FOT to get a specific level of security. To do so, one fixes some values for $q_{Hash}$ and $q_{Dec}$, and chooses two functions $t^{fo}$ and $\varepsilon^{fo}$. We will give Example 3.14 to show how this could be done.

**Example 3.14.** *Assume that we want to construct a $(t^{fo}, q_{Hash}, q_{Dec}, \varepsilon^{fo})$-CCA-RO secure FO scheme, where*

$$\begin{aligned} t^{fo}(n) &= 2^{60} \cdot n^4 \text{ cycles,} \\ q_{Hash} &= 2^{60}, \\ q_{Dec} &= 2^{30}, \text{ and} \\ \varepsilon^{fo}(n) &= 2^{61-n}. \end{aligned}$$

*We assume that sampling random oracle answer takes $n$ cycles, regardless of what is sampled and that $T^{asy}(n) = n^2$ cycles. We can use these values to estimate how secure the asymmetric and the symmetric scheme need to be. We will first estimate $t^{asy}$ and $t^{sy}$ and then proceed to find possible solutions for $\varepsilon^{asy}$ and $\varepsilon^{sy}$.*

*Equation 3.15 can be rephrased as*

$$\min(t^{asy}(n), t^{sy}(n)) = t^{fo}(n) + q_{Hash}O(n) + (q_{Dec} + 1)T^{asy}(n).$$

*We then define a new function* $t^{min}$ *as the lower bound of both* $t^{asy}$ *and* $t^{sy}$*:*

$$t^{min}(n) := 2^{60} \cdot n^4 + 2^{60} \cdot n + (2^{30} + 1) \cdot n^2.$$

*By Theorem 3.7 we can estimate some values to fulfill our security requirement* $\varepsilon^{fo}(n) \leq 2^{61-n}$*. Possible solutions would be* $\varepsilon^{sy}(n) = 2^{59-n}$*,* $\varepsilon^{asy}(n) = 2^{-n}$*, and* $\gamma = n$ *which would result in*

$$\varepsilon^{fo}(n) = 2^{59-n} + 2^{60} \cdot 2^{-n} + 2^{30} \cdot 2^{-n} \approx 2^{60.58-n} \leq 2^{61-n}.$$

*Therefore a possible solution in this example would be to use a* $n$*-spread* $(t^{min}, 2^{-n})$*-OWE secure asymmetric encryption scheme and a* $(t^{min}, 2^{40-n})$*-EAV secure symmetric encryption scheme. In Table 3.2 some example values for these schemes and the resulting FO scheme are listed.*

Table 3.2: Example values for an asymmetric, a symmetric, and the resulting Fujisaki-Okamoto scheme

| $n$ | $t^{min}(n)$ | $\varepsilon^{asy}(n)$ | $t^{min}(n)$ | $\varepsilon^{sy}(n)$ | $t^{fo}(n)$ | $\varepsilon^{fo}(n)$ |
|---|---|---|---|---|---|---|
| 61 | $2^{83.72}$ | $2^{-61}$ | $2^{83.72}$ | $2^{-21}$ | $2^{83.72}$ | 1 |
| 100 | $2^{86.58}$ | $2^{-100}$ | $2^{86.58}$ | $2^{-60}$ | $2^{86.58}$ | $2^{-39}$ |
| 121 | $2^{87.68}$ | $2^{-121}$ | $2^{87.68}$ | $2^{-81}$ | $2^{87.68}$ | $2^{-60}$ |

*One can easily see in this table that* $n = 61$ *may be a secure choice for the asymmetric scheme but is insecure for the resulting FO scheme: The adversary can gain an advantage of* $\varepsilon^{fo} = 1$ *for* $n \leq 61$*. A better choice would be* $n = 121$ *which reduces the adversary's advantage against the Fujisaki-Okamoto scheme to* $2^{-60}$*.*

**One-Time Padded Fujisaki-Okamoto Scheme** Fujisaki and Okamoto introduced a variant of the FO scheme that utilizes a *one-time pad* based encryption scheme. Assume that the underlying symmetric scheme is a One-Time Pad (OTP) encryption scheme defined over $\{0,1\}^{l(n)}$ as follows:

$Enc_k(m)$ On input $m \in \{0,1\}^{l(n)}$ and $k \in \{0,1\}^{l(n)}$, output $c = k \oplus m$.

$Dec_k(c)$ On input $c \in \{0,1\}^{l(n)}$ and $k \in \{0,1\}^{l(n)}$, output $m = k \oplus c$.

The One-Time Pad encryption scheme as described above is perfectly secure which means that it is a $(\infty, 0)$-EAV secure encryption scheme [KL07, 35][FO13, 90].

**Corollary 3.15.** *Let $\Pi^{asy}$ be a $\gamma$-spread $(t^{asy}, \varepsilon^{asy})$-OWE secure asymmetric encryption and let $\Pi^{sy}$ a OTP encryption scheme. Let $G : \mathcal{M}^{asy} \to \{0,1\}^{l(n)}$ and $H : \mathcal{M}^{asy} \times \{0,1\}^{l(n)} \to \mathcal{R}^{asy}$. The FO scheme is $(t^{fo}, q_{Hash}, q_{Dec}, \varepsilon^{fo})$-CCA secure in the ROM where*

$$t^{fo}(n) = t^{asy}(n) - q_{Hash}O(n) - (q_{Dec}+1)T^{asy}(n) \text{ and}$$
$$\varepsilon^{fo}(n) = q_{Hash}\varepsilon^{asy}(n) + 2^{-\gamma}q_{Dec}$$

*Proof of Corollary 3.15 ([FO13, 90]).* As mentioned, the One-Time Pad encryption scheme is $(\infty, 0)$-EAV secure. Therefore Equation 3.15 can be simplified to

$$t^{fo}(n) = t^{asy}(n) - q_{Hash}O(n) - (q_{Dec}+1)T^{asy}(n),$$

and Equation 3.16 can be simplified to

$$\varepsilon^{fo}(n) = q_{Hash}\varepsilon^{asy}(n) + 2^{-\gamma}q_{Dec}.$$

□

One can easily see that the usage of a OTP simplifies the equations for $t^{fo}(n)$ and $\varepsilon^{fo}(n)$. Nonetheless, using the OTP encryption scheme is impractical for big messages. Assume that we want to sent 1 GigaByte messages, i.e. $\mathcal{M}^{sy} = 2^{30}$. To encrypt such a message one needs a one-time key $k \in \{0,1\}^{2^{30}}$ – which further requires that there is a hash function $G$ mapping values onto $\{0,1\}^{2^{30}}$. Constructing such a hash function may be possible but also seems very impractical.

# Bibliography


[BDPR98] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations Among Notions of Security for Public-Key Encryption Schemes. In *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–45. Springer, 1998.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-Based Encryption from the Weil Pairing. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993*, pages 62–73. ACM, 1993.

[BR94] Mihir Bellare and Phillip Rogaway. Optimal Asymmetric Encryption. In *Advances in Cryptology - EUROCRYPT '94, Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9-12, 1994, Proceedings*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. Springer, 1994.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The Random Oracle Methodology, Revisited. In *J. ACM*, volume 51, pages 557–594. ACM, 2004.

[CS03] Ronald Cramer and Victor Shoup. Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack. In *SIAM J. Comput.*, volume 33, pages 167–226, 2003.

[Den06] Alexander W. Dent. A Note On Game-Hopping Proofs. In *IACR Cryptology ePrint Archive*, volume 2006, pages 1–4, 2006.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference,*

*Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554. Springer, 1999.

[FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In *J. Cryptology*, volume 26, pages 80–101, 2013.

[HK07] Dennis Hofheinz and Eike Kiltz. Secure Hybrid Encryption from Weakened Key Encapsulation. In *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, volume 4622 of *Lecture Notes in Computer Science*, pages 553–571. Springer, 2007.

[HK09] Dennis Hofheinz and Eike Kiltz. Practical Chosen Ciphertext Secure Encryption from Factoring. In *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, volume 5479 of *Lecture Notes in Computer Science*, pages 313–332. Springer, 2009.

[KL07] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography*. Chapman and Hall/CRC Press, 2007.

[OP01] Tatsuaki Okamoto and David Pointcheval. REACT: Rapid Enhanced-Security Asymmetric Cryptosystem Transform. In *Topics in Cryptology - CT-RSA 2001, The Cryptographer's Track at RSA Conference 2001, San Francisco, CA, USA, April 8-12, 2001, Proceedings*, volume 2020 of *Lecture Notes in Computer Science*, pages 159–175. Springer, 2001.

[Sho04] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. In *IACR Cryptology ePrint Archive*, volume 2004, pages 1–33, 2004.

# Acronyms

**FO** Fujisaki-Okamoto

**FOT** Fujisaki-Okamoto Transformation

**PPT** probabilistic polynomial-time

**RO** Random Oracle

**ROM** Random Oracle Model

**EAV** eavesdropping indistinguishability

**OWE** one-way attack

**CPA** chosen-plaintext attack

**CCA** chosen-ciphertext attack

**CCA-RO** chosen-ciphertext attack in the random oracle model

**PA** Plaintext Awareness

**OTP** One-Time Pad

# Proposal: The Fujisaki-Okamoto-Transformation

Jan Lippert

## 1 Introduction

One important goal of cryptography is to ensure confidentiality by providing secure communications via encryption. To prove the security of different encryption schemes, a formalized model is used: adversaries play "games" that model specific attacks against a challenger. In the main part of these games, the adversary chooses two messages of equal length; one of these is encrypted by the challenger and the ciphertext is given to the adversary. The adversary then has to guess which of his messages has been encrypted. Different attack types are modeled in the strength of the adversary – the stronger the attack, the more possibilities the adversary has. Furthermore, we only consider probabilistic polynomial-time adversaries.

The weakest attack is the so called eavesdropping attack in which the adversary only knows his two messages and the encryption of one message – and in the case of public key encryption the corresponding public key. If the adversary can distinguish between the encryptions of his messages with a probability that is only negligible better than random guessing, the encryption scheme is called to have indistinguishable encryptions in the presence of an eavesdropper. Informally, negligible probability means that this event is so unlikely to occur that it can be ignored for all practical purposes [6, p. 57]. In a chosen plaintext attack, the adversary gains access to an encryption oracle compared to an eavesdropping attack. This means that he can submit messages of his choice to that encryption oracle and is given the encryptions back [6, p. 338ff].

In a chosen ciphertext attack (CCA), the attacker gains access to both an encryption and a decryption oracle. The adversary can submit ciphertexts of his choice to the decryption oracle and is given the corresponding decryptions. The challenge ciphertext itself may not be submitted to the oracle though – otherwise the attacker would always win. An encryption scheme that is secure in this case has indistinguishable encryptions in the presence of a chosen ciphertext attack [6, p. 103]. CCA-indistinguishability implies that an adversary is unable to "logically manipulate" a given ciphertext, i.e. every result of that manipulation is either an invalid ciphertext or has no relation to the original message [6, p. 104]. One example for a CCA in reality would be the so called "lunchtime attack" in which the adversary can temporarily access the hardware needed for decryption [5, p. 90].

Because public key encryption schemes are relatively slow when used for large messages, hybrid encryption schemes are used in practice. In a "straightforward" hybrid scheme the message is encrypted using a symmetric scheme and a randomly chosen one-time key k. Then, k is encrypted using the public key encryption scheme [6, p. 347]. One drawback of this straightforward approach is

that in general these hybrid encryption schemes are CCA-indistinguishable only when the underlying asymmetric and symmetric encryption schemes both are CCA-indistinguishable [3]. Fujisaki and Okamoto introduced a transformation that removes this restriction[4, 5]: CCA-indistinguishable hybrid encryption schemes can be obtained from encryption schemes that have weaker security notions than CCA-indistinguishability: the symmetric scheme needs to be one-time secure which means that the scheme has secure encryption in the presence of an eavesdropper when every key is used once [5, p. 89]. The asymmetric scheme on the other hand only has to be a one-way encryption scheme. This means that an adversary $\mathcal{A}$ has only a negligible probability to find any correct plaintext to a given ciphertext $c$, i.e. $\mathcal{A}$ outputs an $m$ that - when encrypted - can result in $c$ [5, p. 87f].

The hybrid scheme obtained by applying the FOT uses some hash functions which are modeled as random oracles in the security proof. A random oracle is an idealized model of cryptographic hash functions. Informally, a random oracle can be described as a public, randomly-chosen function that can be evaluated by "querying" values to it; for every query the oracle returns a value which is picked uniformly at random from its range. The random oracle model is called a "middle ground between a fully-rigorous proof of security [...] and no proof"[6, p. 460f].

The idea of the random oracle model is to proof the security of an encryption scheme in an idealized world. The only problem could be the real-life instantiation of the hash function – if this instantiation is not "good enough" the encryption scheme becomes insecure [6, p. 459]. Canetti et. al have shown that there are theoretical *contrived* constructions that are secure in the random oracle model but are insecure in the standard model - no matter how the random oracle is instantiated [2]. However, there have been few successful attacks on *practical* schemes that were proven secure in the random oracle model [6, p. 468].

# 2 Purpose

The purpose of this thesis is to provide a detailed understanding of the Fujisaki-Okamoto transformation from the viewpoint of modern cryptography.

The first goal of this work is to convert the notation used in [5] to a more standard notation like the notation used in [6]. The definitions of the different security properties like one-time secure symmetric encryption and $\gamma$-spread will be also converted into a more standard notation. Since these properties are not used very often and can be labeled as "non-standard", they will be compared to the standard definitions like CPA and CCA and explained in detail using examples.

The generic conversion of the FOT and the security results will be reviewed. The proof for the main theorems will be revisited, completed, and presented in detail. First these proofs will be analyzed. Parts that need clarification or further explanations will be identified. In a second step, the proofs will be completed, and clarified as necessary. In particular, the details of game-hopping technique will be worked out. This technique is used to approximate the adversary's unknown probability to win in a specific game. To do so, the game is altered step by step until the adversary's probability to win can be computed[3].

In addition to the general explanation of game-hopping technique, the different games used in [5] will be explained in detail. The reductions between the games will be reviewed and clarified. In the final step, the proofs for the main theorems will be rewritten in the notation of [6]. In particular we will use the modern notation of random oracles.

**Optional: FOT and identity-based encryption systems** Although the FOT was designed to be used for standard encryption schemes, it has nonetheless been suggested to apply the FOT in identiy-based (e.g. [1]) and attribute based encryption schemes (e.g. [7]). Unfortunately, no general way to apply the FOT to these schemes is known.

After completing the proof, different applications of this transformation for identity-based encryption schemes will be investigated. Then an attempt is made to derive a general method of applying the FOT in identity-based encryption schemes.

# 3 Schedule

| **Task** | | **Weeks** |
|---|---|---|
| Basic Research | | 1-5 |
| Security proof of FOT: | Analysis | 6-8 |
| | Completion | 9-15 |
| | Rewrite | 16-17 |
| (Optional) FOT and identity-based encryption schemes | | 18-19 |
| Thesis Review | | 20-21 |

# 4 Preliminary bibliography


[1] Dan Boneh and Matthew K. Franklin. Identity-Based Encryption from the Weil Pairing. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.

[2] Ran Canetti, Oded Goldreich, and Shai Halevi. The Random Oracle Methodology, Revisited. In *J. ACM*, volume 51, pages 557–594. ACM, 2004.

[3] Ronald Cramer and Victor Shoup. Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack. In *SIAM J. Comput.*, volume 33, pages 167–226, 2003.

[4] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554. Springer, 1999.

[5] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In *J. Cryptology*, volume 26, pages 80–101, 2013.

[6] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography.* Chapman and Hall/CRC Press, 2007.

[7] Amit Sahai and Brent Waters. Fuzzy Identity-Based Encryption. In *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. ProceedingsAdvances in Cryptology - EUROCRYPT 2005, 24th Annual International Co*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer, 2005.

Paderborn, den _____

Prof. Dr. Johannes Blömer

Jan Lippert