A Brief History of Provably-Secure Public-Key Encryption

Alexander W. Dent

Royal Holloway, University of London
Egham, Surrey, TW20 0EX, U.K.
a.dent@rhul.ac.uk

Abstract. Public-key encryption schemes are a useful and interesting field of cryptographic study. The ultimate goal for the cryptographer in the field of public-key encryption would be the production of a very efficient encryption scheme with a proof of security in a strong security model using a weak and reasonable computational assumption. This ultimate goal has yet to be reached. In this invited paper, we survey the major results that have been achieved in the quest to find such a scheme.

1 Introduction

The most popular field of study within public-key cryptography is that of public-key encryption, and the ultimate goal of public-key encryption is the production of a simple and efficient encryption scheme that is provably secure in a strong security model under a weak and reasonable computational assumption. The cryptographic community has had a lot of successes in this area, but these successes tend to fall into two categories: the production of very efficient encryption schemes with security proofs in idealised models, and the production of less-efficient encryption schemes with full proofs of security in strong models. The ultimate prize has yet to be claimed.

However, we are getting closer to that important break-through. Schemes with full security proofs are getting more efficient and the efficient schemes are getting stronger security guarantees. This paper aims to briefly discuss some of the history behind the production of standard-model-secure encryption schemes and to give a personal interpretation of some of the major results.

The first attempt to prove the security of a public-key encryption scheme was by Rabin [27] in 1979, who described an encryption scheme for which recovering the message was as intractable as factoring an RSA modulus. Later, Goldwasser and Micali [21] described a scheme which they could prove hid all information about the plaintext. However, it wasn't until the early 1990s that researchers began to establish reliable and easy to use formal models for the security of an encryption scheme and that the cryptographic community began to think about constructing practical and efficient provably-secure public-key encryption schemes.

1.1 Notation

We will use standard notation. For a natural number $k \in \mathbb{N}$, we let $\{0,1\}^k$ denote the set of k-bit strings and $\{0,1\}^*$ denote the set of bit strings of finite length. We let 1^k denote a string of k ones.

We let $\leftarrow$ denote assignment; hence, $y \leftarrow x$ denotes the assignment to y of the value x. For a set S, we let $x \xleftarrow{\$} S$ denote the assignment to x of a uniformly random element of S. If $\mathcal{A}$ is a randomised algorithm, then $y \xleftarrow{\$} \mathcal{A}(x)$ denotes the assignment to y of the output of $\mathcal{A}$ when run on input x with a fresh set of random coins. If we wish to execute $\mathcal{A}$ using a particular set of random coins R, then we write $y \leftarrow \mathcal{A}(x; R)$, and if $\mathcal{A}$ is deterministic, then we write $y \leftarrow \mathcal{A}(x)$.

1.2 The IND-CCA2 Security Model

A public-key encryption scheme is formally defined as a triple of probabilistic, polynomial-time algorithms $(\mathcal{G}, \mathcal{E}, \mathcal{D})$. The key generation algorithm $\mathcal{G}$ takes as input a security parameter 1^k and outputs a public/private key pair (pk, sk). The public key implicitly defines a message space $\mathcal{M}$ and a ciphertext space $\mathcal{C}$. The encryption algorithm takes as input the public key pk and a message $m \in \mathcal{M}$, and outputs a ciphertext $C \in \mathcal{C}$. The decryption algorithm takes as input the private key sk and a ciphertext $C \in \mathcal{C}$, and outputs either a message $m \in \mathcal{M}$ or the error symbol $\perp$. We demand that the encryption scheme is sound in the sense that if $C \xleftarrow{\$} \mathcal{E}(pk, m)$, then $m \leftarrow \mathcal{D}(sk, C)$, for all keys $(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$ and $m \in \mathcal{M}$.

If we are going to prove that an encryption scheme is secure, then we need to have some formal notion of confidentiality. The commonly accepted "correct" definition is that of indistinguishability under adaptive chosen ciphertext attack (IND-CCA2) was proposed by Rackoff and Simon [28]. It built on the weaker notion of IND-CCA1 security proposed by Naor and Yung [26].

Definition 1. *An attacker $\mathcal{A}$ against the IND-CCA2 security of an encryption scheme $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ is a pair of probabilistic polynomial-time algorithms $(\mathcal{A}_1, \mathcal{A}_2)$. The success of the attacker is defined via the IND-CCA2 game:*

$$\begin{aligned}
&(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)\\
&(m_0, m_1, state) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{D}}(pk)\\
&b \xleftarrow{\$} \{0,1\}\\
&C^* \xleftarrow{\$} \mathcal{E}(pk, m_b)\\
&b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{D}}(C^*, state)
\end{aligned}$$

The attacker may query a decryption oracle with a ciphertext C at any point during its execution, with the exception that $\mathcal{A}_2$ may not query the decryption oracle on C^*. The decryption oracle returns $m \leftarrow \mathcal{D}(sk, C)$. The attacker wins the game if $b = b'$. An attacker's advantage is defined to be

$$Adv_{\mathcal{A}}^{\text{IND}}(k) = |Pr[b = b'] - 1/2| \,. \tag{1}$$

We require that a "reasonable" attacker's advantage is "small". This can either be phrased by saying that every polynomial-time attacker must have negligible advantage under the assumption that it is hard to solve some underlying problem (asymptotic security) or by relating the advantage ϵ of an attacker that runs in time t to the success probability ϵ' that an algorithm that runs in time t' has in breaking some underlying hard problem (concrete security). Much is often made of the difference in these two approaches, but in practice they are very similar – they both require that the proof demonstrate a tight reduction from the encryption scheme to the underlying problem. This issue is discussed in more detail in a previous paper [16].

It is sometimes convenient to work with a slightly different definition for advantage. If the IND-CCA2 game encrypts a message defined by the bit b and the attacker outputs the bit b', then

$$Adv_{\mathcal{A}}^{\texttt{IND*}}(k) = |Pr[b' = 0 | b = 0] - Pr[b' = 0 | b = 1]| \,. \tag{2}$$

It can easily be shown that

$$Adv_{\mathcal{A}}^{\texttt{IND*}}(k) = 2 \cdot Adv_{\mathcal{A}}^{\texttt{IND}}(k) \,. \tag{3}$$

Hence, it is sufficient to bound $Adv_{\mathcal{A}}^{\texttt{IND*}}$ in order to prove security.

A scheme that is secure against attackers that can only make decryption oracle queries before receiving the challenge ciphertext C^* is said to be IND-CCA1 secure. A scheme that is secure against attackers that do not make any decryption oracle queries at all is said to be IND-CPA or passively secure.

2 The Random Oracle Methodology

No paper on the history of secure encryption schemes would be complete without a mention of the random oracle methodology. In the early 1990s, after the development of the IND-CCA2 security model, researchers turned to the random oracle methodology [4] in order to provide proofs of security for practical public key encryption schemes. The intuition is simple: secure hash functions would share many properties with random functions. Hence, it made sense to model a secure hash function as a completely random function in a security analysis.

This greatly simplifies the process of proving the security of a cryptographic scheme. By modelling the hash function as a random function, we know that the hash function will output completely random and independently generated values on different inputs. Knowledge of the hash values for several different inputs gives absolutely no information about the hash value for any other input and therefore the only way that an attacker can compute the hash value for a given input is to query the hash function oracle on that input. This means that the attacker's behaviour is no longer completely black-box – we may now observe the attacker's behaviour during the attack process (in some limited way). We may even construct the responses that the hash function oracle gives in ways that help prove the security of the cryptosystem (subject to the restriction that they appear to the attacker to be chosen at random).

Of course, schemes proven secure using the random oracle methodology are not necessarily secure when the hash function is instantiated with a given fixed hash function. There is always the possibility that the particular hash function will interact badly with the mathematics of the encryption scheme, and that the resulting system will be insecure. It was, however, hoped that the number of hash functions that "interacted badly" would be small and that a scheme proven secure using the random oracle methodology would be secure when the random oracle was replaced with almost any hash function.

This turned out not to be true. In an amazing paper by Canetti, Goldreich and Halevi [11], it was shown that it was possible to construct an encryption scheme that was provably secure using the random oracle methodology, but was insecure when the random oracle was instantiated with *any* hash function. The paper notes that in the standard model (i.e. when we are not using the random oracle methodology) the attacker has an extra piece of information not available to the attacker in the random oracle model: the attacker has a description of the hash function. The paper gives a scheme for which an attacker can use this description like a password – the attacker submits the description of the hash function to the decryption oracle as a ciphertext and the decryption oracle helpfully returns the private key of the encryption scheme.

It is clear that the encryption scheme of Canetti, Goldreich and Halevi is completely artificial – no real encryption scheme would make use of a decryption algorithm that would output the private key if it were given a ciphertext of a particular (checkable) form. However, it does act as a proof of concept: it is possible to construct a scheme that is secure in the random oracle model, but insecure in the standard model. We therefore cannot completely trust schemes that are only proven secure in the random oracle model. A lot of effort has been expended by cryptographers attempting to find a non-artificial scheme which is secure in the random oracle model, but insecure in practice, but so far no such scheme has been found.

Personally, I still think the random oracle model is a useful tool in cryptography. I believe that it provides trustworthy security guarantees for the vast majority of practical cryptosystems. Furthermore, I don't think I know of a single industrial company or standardisation body that would reject an efficient cryptosystem because it "only" had a proof of security in the random oracle model.

3 Double-and-Add Schemes

We now turn our attention to schemes that can be proven secure in the standard model. The approaches to constructing encryption schemes secure in the standard model tend to fall several categories. The first approach is to use a "double-and-add" technique, in which a message is encrypted twice (using two weak encryption schemes) and a checksum value is added to the ciphertext.

3.1 The NIZK Schemes

The first attempt to prove the security of a scheme against chosen ciphertext attacks was given by Naor and Yung [26]. Their approach was to encrypt a message twice using two independent IND-CPA secure encryption schemes, and then to provide a non-interactive zero-knowledge (NIZK) proof that the two ciphertexts were encryptions of the same message. The Naor-Yung result only produced an encryption scheme that was IND-CCA1 secure. Their approach was extended by Sahai [29] to cover IND-CCA2 attacks by using a slightly more powerful NIZK proof system.

It is not going to be possible, due to space constraints, to fully explain the technical details of this scheme. However, we will give an overview of the scheme. Suppose $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ is an IND-CPA secure encryption scheme. The Sahai encryption scheme works as follows:

- **Key generation**. Generate two independent key pairs $(pk_1, sk_1) \xleftarrow{\$} \mathcal{G}(1^k)$ and $(pk_2, sk_2) \xleftarrow{\$} \mathcal{G}(1^k)$, and a random string σ (for use by the NIZK proof). The public key is $pk = (pk_1, pk_2, \sigma)$ and the private key is $sk = (sk_1, sk_2)$.
- **Encryption**. To encrypt a message m, compute $C_1 \xleftarrow{\$} \mathcal{E}(pk_1, m)$ and $C_2 \xleftarrow{\$} \mathcal{E}(pk_2, m)$, and give a NIZK proof π that C_1 and C_2 are encryptions of the same message (using the random string σ). The ciphertext is (C_1, C_2, π).
- **Decryption**. To decrypt a message, first check the proof π. If the proof fails, then output $\perp$. Otherwise, output $m \leftarrow \mathcal{D}(sk_1, C_1)$.

Of course, as the NIZK proof π proves that C_1 and C_2 are the encryption of the same message, we could have equivalently computed $m \leftarrow \mathcal{D}(sk_2, C_2)$ in the decryption algorithm.

The key to understanding the security of this scheme is in understanding the security properties of the NIZK proof system. We require two properties from the NIZK proof system:

- **Zero knowledge**. It should be possible to choose the random string σ in such a way that the NIZK proof system has a trapdoor τ that allows an entity in possession of the trapdoor to produce false proofs – i.e. it should be possible to "prove" that any pair of ciphertexts (C_1, C_2) are the encryption of the same message using the trapdoor τ, even if (C_1, C_2) are encryptions of different messages. Furthermore, it should be impossible for the attacker (who only knows the string σ and not the trapdoor τ) to be able to distinguish false proofs from real ones.
- **Simulation Sound**. It should be impossible for the attacker to produce a proof π that two ciphertexts (C_1, C_2) are encryptions of the same message unless the ciphertexts actually are the encryptions of the same message. Furthermore, this property should hold even if the attacker is given a false proof π which is computed using the trapdoor τ.

The ideas behind the proof become very simple to understand if one considers bounding $Adv_{\mathcal{A}}^{\texttt{IND*}}$ rather than $Adv_{\mathcal{A}}^{\texttt{IND}}$. In the IND* security model, we observe the

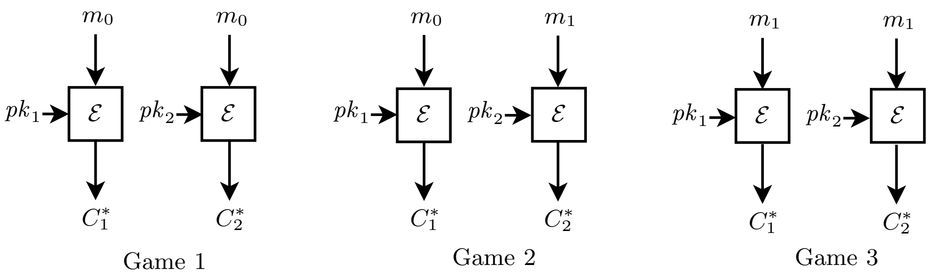

Fig. 1. The games used in security proof of the Sahai construction (with the NIZK proof omitted).

difference in the attacker's behaviour when the challenge encryption C^* is an encryption of m_0 and when the challenge encryption C^* is an encryption of m_1. Recall that this C^* is of the form (C_1^*, C_2^*, π^*) where $C_i^* \xleftarrow{\$} \mathcal{E}(pk_i, m_b)$. First, since the NIZK proof system is zero knowledge, we may assume that the challenger has chosen a random string with a trapdoor, and that the NIZK proof π^* is produced using the trapdoor τ, rather than by using the normal proof algorithm.

We use a simple game-hopping argument (as illustrated in Figure 1). Let Game 1 be the game in which the challenge ciphertext C^* is computed as an encryption of m_0. In other words,

$$C_1^* \xleftarrow{\$} \mathcal{E}(pk_1, m_0) \qquad C_2^* \xleftarrow{\$} \mathcal{E}(pk_2, m_0)$$

and π^* is a proof that (C_1^*, C_2^*) are encryptions of the same message. Let Game 2 be the game in which the challenge ciphertext C^* is computed as

$$C_1^* \xleftarrow{\$} \mathcal{E}(pk_1, m_0) \qquad C_2^* \xleftarrow{\$} \mathcal{E}(pk_2, m_1)$$

and π^* is a false proof that (C_1^*, C_2^*) are encryptions of the same message computed using the trapdoor τ. We claim that any attacker that can distinguish between Game 1 and Game 2 can also break the IND-CPA security of the second encryption scheme. The reduction makes use of the fact that we may decrypt a valid ciphertext using the secret key for the first encryption scheme – this allows us to simulate the decryption oracle.

Similarly, let Game 3 be the game in which the challenge ciphertext C^* is computed as

$$C_1^* \xleftarrow{\$} \mathcal{E}(pk_1, m_1) \qquad C_2^* \xleftarrow{\$} \mathcal{E}(pk_2, m_1)$$

and π^* is a proof that (C_1^*, C_2^*) are encryptions of the same message. If the attacker can distinguish between Game 2 and Game 3, then the attacker can break the IND-CPA security of the first scheme. This time the reduction makes use of the fact that we may decrypt a valid ciphertext using the secret key for the second encryption scheme.

The beauty of this construction is that it allows us to prove that secure public-key encryption schemes exist assuming only the existence of trapdoor one-way permutations. Sahai [29] notes that passively secure encryption schemes exist under the assumption that trapdoor one-way functions exist [20] and builds suitable NIZK proof systems using the results of Feige, Lapidot and Shamir [19] and Bellare and Yung [6]. This is a wonderful theoretical result, but, due to the theoretical nature of the NIZK proof system used in the construction, the construction is not practical.

3.2 The Cramer-Shoup Encryption Scheme

The first practical public-key encryption scheme that was proven secure in the standard model was the Cramer-Shoup scheme [13]. Although not explicitly presented as an extension of the Sahai construction, it can be thought of as building on these ideas. Suppose $\mathbb{G}$ is a cyclic group of prime order p that is generated by g and that $Hash : \mathbb{G}^3 \to \mathbb{Z}_p$ is a (target collision resistant) hash function. The Cramer-Shoup encryption scheme can be written as[1]:

$\mathcal{G}(1^k)$	$\mathcal{E}(pk, m)$	$\mathcal{D}(sk, C)$
$\quad \hat{g} \xleftarrow{\$} \mathbb{G}$	$\quad r \xleftarrow{\$} \mathbb{Z}_p$	$\quad$ Parse C as $(a, \hat{a}, c, d)$
$\quad x_1, x_2, y_1, y_2, z \xleftarrow{\$} \mathbb{Z}_p$	$\quad a \leftarrow g^r$	$\quad v \leftarrow Hash(a, \hat{a}, c)$
$\quad h \leftarrow g^z$	$\quad \hat{a} \leftarrow \hat{g}^r$	$\quad$ If $d \neq a^{x_1+y_1 v}\hat{a}^{x_2+y_2 v}$
$\quad e \leftarrow g^{x_1}\hat{g}^{x_2}$	$\quad c \leftarrow h^r m$	$\qquad$ Output $\perp$
$\quad f \leftarrow g^{y_1}\hat{g}^{y_2}$	$\quad v \leftarrow Hash(a, \hat{a}, c)$	$\quad m \leftarrow c/a^z$
$\quad pk \leftarrow (g, \hat{g}, h, e, f)$	$\quad d \leftarrow e^r f^{rv}$	$\quad$ Output m
$\quad sk \leftarrow (x_1, x_2, y_1, y_2, z)$	$\quad$ Output $(a, \hat{a}, c, d)$	

This scheme is proven secure under the assumption that the DDH problem is hard to solve in $\mathbb{G}$ and the hash function is target collision resistant.

On first glance, this scheme does not appear to have much in common with Sahai's double-and-add scheme. However, consider a variant of the ElGamal encryption scheme [18] in the group $\mathbb{G}$ generated by an element h:

$\mathcal{G}(1^k)$	$\mathcal{E}(pk, m)$	$\mathcal{D}(sk, C)$
$\quad z \xleftarrow{\$} \mathbb{Z}_p^*$	$\quad r \xleftarrow{\$} \mathbb{Z}_p$	$\quad$ Parse C as (a, c)
$\quad g \leftarrow h^{1/z}$	$\quad a \leftarrow g^r$	$\quad m \leftarrow c/a^z$
$\quad pk \leftarrow g$	$\quad c \leftarrow h^r m$	$\quad$ Output m
$\quad sk \leftarrow z$	$\quad$ Output (a, c)	

This scheme is known to be IND-CPA secure under the DDH assumption. In order to use this scheme with Sahai's construction, we would need to encrypt the same message twice using separate random values for each encryption. However, Bellare, Boldyreva and Staddon [1] show that the ElGamal scheme remains secure when it is used to encrypt the same message under multiple public keys *even if the same random value r is used in all the encryptions*. Hence, we may

[1] Technically, this is the CS1a scheme.

think of $(a, \hat{a}, c)$ as a double encryption of the same message under two separate public keys.

To complete the analogy, we must show that d acts in a manner similar to the NIZK proof π in the Sahai construction. Therefore, d would have to have properties similar to simulation soundness and zero knowledge. In the Cramer-Shoup scheme $(a, \hat{a}, c)$ is a valid double encryption of the same message providing that there exists a value r such that $a = g^r$ and $\hat{a} = \hat{g}^r$ – i.e. providing that $(g, \hat{g}, a, \hat{a})$ form a DDH triple. An examination of the security proof for the Cramer-Shoup scheme shows that a large portion of that proof is devoted to showing that we can reject ciphertexts submitted to the decryption oracle for which $(g, \hat{g}, a, \hat{a})$ is not a DDH triple. This is analogous to simulation soundness. A further examination of the proof shows that it constructs the challenge ciphertext as $a \leftarrow g^r$ and $\hat{a} \leftarrow g^{r'}$ for $r \neq r'$. The "proof" d is falsely constructed from $(a, \hat{a})$ using knowledge of (x_1, x_2, y_1, y_2). This is clearly analogous to the zero knowledge property.

We note that the analogy is not entirely correct. In order to verify the correctness of the "proof" d, it is necessary to know the secret values (x_1, x_2, y_1, y_2). In the analogy, this would be the equivalent to requiring the trapdoor τ to verify the NIZK proof and Sahai's construction does not appear to work if the trapdoor is required to verify proofs. However, the similarities between the Cramer-Shoup encryption scheme and the Sahai construction are striking. Other variants of the Cramer-Shoup scheme, such as the Kurosawa-Desmedt scheme [24], can be viewed similarly, albeit with more complex analyses.

4 Signatures and Identities

The security of the "double-and-add" schemes of the preceding section can be proven because there are two equivalent ways in which a ciphertext can be decrypted. Therefore, if part of the security proof prevents us from using one decryption method, then we may still decrypt ciphertexts correctly using the other decryption method. In this section, we look at a technique which handles decryption in another way.

4.1 The Canetti-Halevi-Katz transform

The elegant technique we will look at was proposed by Canetti, Halevi and Katz [12] and converts a passively secure identity-based encryption scheme into a fully secure public-key encryption scheme using a one-time signature scheme. We will examine a more generalised version proposed by MacKenzie, Reiter and Yang [25] and Kiltz [23] based on the concept of *tag-based encryption*.

A tag-based encryption scheme is an encryption scheme in which the encryption and decryption algorithm take an extra input called a *tag*. In order for decryption to work, the tag used at encryption must also be presented at decryption. In other words, a tag-based encryption scheme is a triple of algorithms $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ where $\mathcal{G}$ has the same syntax as in a traditional public-key encryption algorithm. The encryption algorithm takes as input a public key pk, a message $m \in \mathcal{M}$, and a tag $t \in \{0,1\}^*$, and outputs a ciphertext $C \in \mathcal{C}$. The decryption algorithm takes as input a private key sk, a ciphertext $C \in \mathcal{C}$, and a tag $t \in \{0,1\}^*$, and outputs either a message $m \in \mathcal{M}$ or the error symbol $\perp$. We require that if $C \xleftarrow{\$} \mathcal{E}(pk, m, t)$, then $m \leftarrow \mathcal{D}(sk, C, t)$, for all key-pairs $(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$, messages $m \in \mathcal{M}$ and tags $t \in \{0,1\}^*$.

Definition 2. *An attacker $\mathcal{A}$ against the selective-tag weak chosen-ciphertext attacks of a tag-based encryption scheme $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ is a triple of probabilistic polynomial-time algorithms $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$. The success of the attacker is defined via the following game:*

$$\begin{array}{l}
(t^*, state) \xleftarrow{\$} \mathcal{A}_0(1^k) \\
(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k) \\
(m_0, m_1, state) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{D}}(pk, state) \\
b \xleftarrow{\$} \{0,1\} \\
C^* \xleftarrow{\$} \mathcal{E}(pk, m_b, t^*) \\
b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{D}}(C^*, state)
\end{array}$$

The attacker may query a decryption orale with any *ciphertext C and any tag $t \neq t^*$. The decryption oracle $m \leftarrow \mathcal{D}(sk, C, t)$. The attacker wins the game if $b = b'$. The attacker's advantage is defined to be*

$$Adv_{\mathcal{A}}^{\mathit{Tag}}(k) = |Pr[b = b'] - 1/2| \,. \tag{4}$$

This notion of security basically states that the ability to decrypt ciphertexts using one tag does not help an attacker decrypt ciphertexts using another tag.

A tag-based encryption scheme can be constructed from a selective-identity IND-ID-CPA secure identity based encryption scheme, where the identity in the identity-based encryption scheme plays the same role as the tag in a tag-based encryption scheme. Since an identity-based encryption scheme allows the attacker to derive a private key for any identity not equal to the challenge identity, the attacker can decrypt any ciphertext for any identity except the challenge identity.

The CHK transform converts a selective-tag weak chosen-ciphertext secure tag-based encryption scheme into an IND-CCA2 secure public-key encryption scheme via a one-time signature scheme. A one-time signature scheme is a triple of probabilistic, polynomial-time algorithms (*Gen*, *Sign*, *Verify*). The *Gen* algorithm takes as input the security parameter 1^k and outputs a public/private key pair (vrk, snk). The signing algorithm *Sign* takes as input a private signing key *snk* and a message m, and outputs a signature σ. The verification algorithm *Verify* takes as input a public verification key *vrk*, a message m and a signature σ, and outputs either *true* or *false*. The verification algorithm should verify all signature created using the signing algorithm. Furthermore, the attacker should not be able to forge a new signature on any message after having seen a single message/signature pair.

The complete public-key encryption scheme $(\mathcal{G}', \mathcal{E}', \mathcal{D}')$ is as follows:

$$
\begin{array}{l}
\mathcal{G}'(1^k) \\
\quad (pk, sk) \xleftarrow{\$} \mathcal{G}(1^k) \\
\quad \text{Output } (pk, sk)
\end{array}
\qquad
\begin{array}{l}
\mathcal{E}'(pk, m) \\
\quad (vrk, snk) \\
\quad \qquad \xleftarrow{\$} Gen(1^k) \\
\quad t \leftarrow vrk \\
\quad c \xleftarrow{\$} \mathcal{E}(pk, m, t) \\
\quad \sigma \xleftarrow{\$} Sign(snk, c) \\
\quad \text{Output } (c, vrk, \sigma)
\end{array}
\qquad
\begin{array}{l}
\mathcal{D}'(sk, C) \\
\quad \text{Parse } C \text{ as } (c, vrk, \sigma) \\
\quad \text{If } Verify(vrk, c, \sigma) \neq true \\
\quad \quad \text{Output } \perp \\
\quad t \leftarrow vrk \\
\quad m \leftarrow \mathcal{D}(sk, c, t) \\
\quad \text{Output } m
\end{array}
$$

The principle behind the security proof for this elegant construction couldn't be simpler. Suppose the challenge ciphertext is (c^*, vrk^*, σ^*) and consider a ciphertext (c, vrk, σ) submitted to a decryption oracle. We reduce the security of the public-key encryption scheme to the security of the tag-based encryption scheme. We know that the tag-based encryption scheme is passively secure; hence, it suffices to explain how we handle decryption oracle queries. If $vrk \neq vrk^*$ then we may request the decryption of c using the decryption oracle for the tag-based encryption scheme. If $vrk = vrk^*$ then either the signature σ is invalid or the attacker has broken the unforgeability of the one-time signature scheme. Hence, with overwhelming probability, we may return $\perp$ as the decryption oracle's response.

There are a number of other schemes that prove their security using similar principles [9, 10]. In many ways, it is ironic that it was the development of standard-model-secure identity-based encryption schemes (a harder primitive to construct) that produced the next chapter in the development of public-key encryption schemes. However, these schemes are similar to the "double-and-add" schemes in that they convert a passively secure scheme into a fully secure scheme using a cryptographic checksum. This two-stage process is never going to be as efficient as other constructions might be.

4.2 The Dolev-Dwork-Naor Scheme

The first IND-CCA2 secure public-key encryption scheme was proposed by Dolev, Dwork and Naor [17] in 1991. For many years, this complex and elegant scheme remained the only IND-CCA2 public-key encryption scheme proven secure in the standard model. It is ironic that the easiest way to understand this early scheme is via the "double-and-add" schemes and "signature-and-identity" schemes that we have discussed. Essentially, the authors use a Naor-Yung technique to construct a tag-based encryption scheme and then apply a CHK transform. This, of course, is particularly impressive since the concept of a tag-based encryption scheme was more than a decade away.

The Dolev-Dwork-Naor (DDN) scheme makes use of a passively secure encryption scheme $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ and a NIZK proof system to prove that ℓ ciphertexts all contain encryptions of the same message. It also makes use of a one-time signature scheme $(Gen, Sign, Verify)$ which produces ℓ-bit verification keys. It runs as follows:

- **Key Generation.** Generate 2ℓ independent key-pairs $(pk_j^t, sk_j^t) \xleftarrow{\$} \mathcal{G}(1^k)$ for $1 \leq j \leq \ell$ and $t \in \{0, 1\}$, and a random string s (for use with the NIZK

proof). The public key consists of $(pk_1^0, pk_1^1, \ldots, pk_\ell^0, pk_\ell^1, s)$ and the private key is $(sk_1^0, sk_1^1, \ldots, sk_\ell^0, sk_\ell^1)$.
- **Encryption**. To encrypt a message m, generate a one-time signature key-pair $(vrk, snk) \xleftarrow{\$} Gen(1^k)$ and parse vrk as bits $t_1 \ldots t_\ell$. Parse the message as bits $m_1 \ldots m_n$. For each message bit m_i compute ℓ encryptions $c_{ij} \xleftarrow{\$} \mathcal{E}(pk_j^{t_j}, m_i)$ and a proof π_i that each of these ciphertexts encrypts the same message bit. At this stage, the ciphertext consists of n sets of encryptions $c_i \leftarrow (c_{i1}, \ldots, c_{i\ell})$ and n proofs π_i. Let $c \leftarrow ((c_1, \pi_1), \ldots, (c_n, \pi_n))$. Compute the signature $\sigma \xleftarrow{\$} Sign(snk, c)$. The ciphertext is (c, vrk, σ).
- **Decryption.** To decrypt a ciphertext (c, vrk, σ), first verify that the signature is correct and return $\perp$ if the signature is invalid. Then check each proof π_i is correct and return $\perp$ if any proof is invalid. Lastly, if both checks are correct, then parse vrk as bits $t_1 \ldots t_\ell$, compute the message bits $m_i \leftarrow \mathcal{D}(sk_1^{t_1}, c_{i1})$ and return $m \leftarrow m_1 \ldots m_n$.

This can easily be seen to be the result of applying the CHK transform to the following tag-based encryption scheme:

- **Key Generation.** Generate 2ℓ independent key-pairs $(pk_j^t, sk_j^t) \xleftarrow{\$} \mathcal{G}(1^k)$ for $1 \leq j \leq \ell$ and $t \in \{0, 1\}$, and a random string s (for use with the NIZK proof). The public key consists of $(pk_1^0, pk_1^1, \ldots, pk_\ell^0, pk_\ell^1, s)$ and the private key is $(sk_1^0, sk_1^1, \ldots, sk_\ell^0, sk_\ell^1)$.
- **Encryption**. To encrypt a message m using a tag t, parse t as bits $t_1 \ldots t_\ell$. Parse the message as bits $m_1 \ldots m_n$. For each message bit m_i compute ℓ encryptions $c_{ij} \xleftarrow{\$} \mathcal{E}(pk_j^{t_j}, m_i)$ and a proof π_i that each of these ciphertexts encrypts the same message bit. Let $c_i \leftarrow (c_{i1}, \ldots, c_{i\ell})$ and return the ciphertext $c \leftarrow ((c_1, \pi_1), \ldots, (c_n, \pi_n))$.
- **Decryption.** To decrypt a ciphertext $c = ((c_1, \pi_1), \ldots, (c_n, \pi_n))$ using a tag t, first check each proof π_i is correct and return $\perp$ if any proof is invalid. If each proof is correct, then parse t as bits $t_1 \ldots t_\ell$, compute the message bits $m_i \leftarrow \mathcal{D}(sk_1^{t_1}, c_{i1})$ and return $m \leftarrow m_1 \ldots m_n$.

This scheme can be seen to be selective-tag weakly chosen ciphertext secure by observing that as the attacker in the tag-based encryption security model has to output the challenge tag t^* at the beginning of the game, we can arrange for ℓ "real" instances of the public-key encryption scheme to be used to encrypt all messages using the challenge tag t^*. The private key corresponding to these keys are unknown. However, we may generate the ℓ remaining "false" public/private key pairs ourselves. The challenge ciphertext is computed using the "real" public keys and so the value of the challenge message is unknown. However, we may still answer decryption oracle queries. If the attacker submits a ciphertext $c = (c_1, \pi_1), \ldots, (c_n, \pi_n))$ and tag $t \neq t^*$ to the decryption oracle, then we may be sure that each ciphertext component c_i contains ℓ encryptions of the same message bit. Furthermore, since $t \neq t^*$, we must have that one ciphertext c_{ij} was computed using a "false" public key, for which we know the corresponding private key. Hence, we may recover the message bit by decrypting this component and so answer the decryption oracle query correctly. This demonstrates that the security of the tag-based encryption scheme can be reduced to the security of the underlying passively-secure encryption scheme.

Of course, this scheme is highly inefficient. The use of Naor-Yung "double-and-add" technique means that we have to encrypt every message bit multiple times, and use an arbitrary NIZK proof system. Furthermore, the use of the CHK transform implies the need for an inexpensive signing operation. Hence, this scheme can only be considered to be of theoretical interest.

5 Extracting Plaintext Awareness

Plaintext awareness is a simple idea with a complicated explanation. An encryption scheme is plaintext aware if it is impossible for a user to create a valid ciphertext without knowing the underlying message. This effectively makes a decryption oracle useless to the attacker – any valid ciphertext he submits to the decryption oracle will return a message that he already knows. If he submits a ciphertext to the decryption oracle for which he does not know the underlying message, then the decryption oracle will return $\perp$. This leads to the central theorem of plaintext awareness: that a scheme that is IND-CPA secure and plaintext aware is IND-CCA2 secure.

The difficulty with this idea is formalising what it means to say that a user "knows" an underlying message. The first attempt to produce a formal definition for plaintext awareness was given in the random oracle model [2, 5] but had the disadvantage that it could *only* be realised in the random oracle model. It took several years before a definition compatible with the standard model was found.

5.1 Plaintext Awareness via Key Registration

The first attempt to provide a standard-model definition of plaintext awareness was given by Herzog, Liskov and Micali [22]. In their model, if a sender wishes to send a message to a receiver, then both the sender and the receiver must have a public key. Furthermore, the sender must register their public key with some trusted registration authority in a process that includes a zero-knowledge proof of knowledge for the private key. Now, whenever the sender wants to send a message, it forms two ciphertexts – an encryption of the message using the receiver's public key and an encryption of the message using the sender's own public key – and provides a NIZK proof that the ciphertexts are the encryption of the same message. The receiver decrypts the ciphertext by checking the validity of the NIZK proof and decrypting the component that was encrypted using their public key.

The plaintext awareness of the scheme can be easily shown: since the NIZK proves that the encryptions are identical, we know that both ciphertexts are the encryption of the same message. Furthermore, since the sender has proven knowledge of the private key, we know that the sender can decrypt the component of the ciphertext encrypted using the sender's public key and recover the message. Hence, we can conclude that the sender "knows" the message.

This is an interesting idea, and clearly related to the security of the Sahai construction, but it is never really been adopted to prove the security of practical schemes. The requirement that the sender must have a registered public key creates the need for a huge public-key infrastructure which is unlikely to exist in practice. Furthermore, the scheme still makes use of arbitrary zero-knowledge proofs of knowledge and NIZK proof systems, which are impractical.

5.2 Using Extractors

In 2004, Bellare and Palacio [3] introduced a new standard-model definition for plaintext awareness. Their definition has several advantages over the definition of Herzog, Liskov and Micali. In particular, Bellare and Palacio's definition doesn't require a sender to register a key. It is also compatible with earlier definitions in the random oracle model, in the sense that a scheme proven plaintext aware using the random-oracle-based definition of plaintext awareness is also plaintext aware using the standard-model-based definition of plaintext awareness (although the proof of this fact uses the random oracle model).

The Bellare and Palacio definition of plaintext awareness uses a definition of "knowledge" that is similar to the definition used in zero knowledge. An attacker $\mathcal{A}$ is deemed to "know" a value x if it is possible to alter $\mathcal{A}$ to give a new algorithm $\mathcal{A}^*$ that outputs x.

Let (pk, sk) be a randomly generated key pair for a public-key encryption scheme $(\mathcal{G}, \mathcal{E}, \mathcal{D})$. We consider an attacker $\mathcal{A}$ that takes as input a public key pk and a set of random coins R, and interacts with an "oracle" to which it can submit ciphertexts. The form of the oracle depends upon the game that the attacker is playing. In the Real game, the oracle is instantiated using the decryption algorithm $\mathcal{D}(sk, \cdot)$. In the Fake game, the oracle is instantiated by an algorithm $\mathcal{A}^*$ which we call the *plaintext extractor*. This plaintext extractor $\mathcal{A}^*$ is a stateful, probabilistic, polynomial-time algorithm that depends upon $\mathcal{A}$ and initially takes as input the public key pk and the random coins R used by $\mathcal{A}$. Since $\mathcal{A}^*$ has all the inputs of $\mathcal{A}$, one can think of $\mathcal{A}^*$ as observing $\mathcal{A}$'s behaviour as it creates ciphertexts. If the attacker $\mathcal{A}$ submits a ciphertext C to the plaintext extractor $\mathcal{A}^*$, then it is $\mathcal{A}^*$'s task to determine the underlying message from $\mathcal{A}$'s behaviour.

It would be nice if we were done here, but we also need to consider the possibility that the attacker $\mathcal{A}$ can obtain some ciphertexts for which he does not know the underlying message. In the real world, this corresponds to the idea that the attacker might be able to observe ciphertexts created by other people. In the IND security model, this allows for the fact that the attacker is given the challenge ciphertext C^* (for which he does not know the underlying encryption). This possibility is allowed for in the security model for plaintext awareness by giving the attacker access to an encryption oracle that, when queried with some auxiliary information aux, generates a message $m \xleftarrow{\$} \mathcal{P}(aux)$ (using some arbitrary, stateful, probabilistic polynomial-time algorithm $\mathcal{P}$) and returns the ciphertext $C \xleftarrow{\$} \mathcal{E}(pk, m)$. We are forced to give C to the plaintext extractor $\mathcal{A}^*$

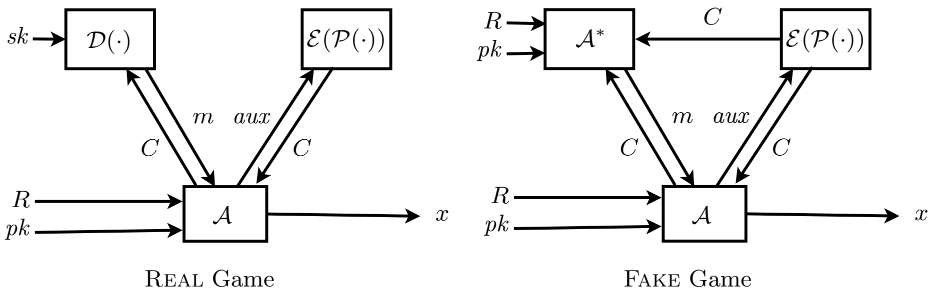

Fig. 2. The Real and Fake games for plaintext awareness

so that it may continue to observe $\mathcal{A}$'s behaviour. We forbid $\mathcal{A}$ from asking for the decryption of C. We show the differences between the Real and Fake game graphically in Fig. 2.

We say that a scheme is plaintext aware if, for any attacker $\mathcal{A}$, there exists a plaintext extractor $\mathcal{A}^*$ such that, for *any* plaintext creating algorithm $\mathcal{P}$, the output x of $\mathcal{A}$ in the Real game is indistinguishable from the output x of $\mathcal{A}$ in the Fake game.

Bellare and Palacio [3] prove that any scheme that is IND-CPA secure and plaintext aware in this model is necessarily IND-CCA2 secure. In an extraordinary paper, Teranishi and Ogata [30] prove that a scheme that is one-way and plaintext aware in this model is necessarily IND-CCA2 secure. There are weaker models for plaintext awareness that are similar to this model, and their relationships to the full security model have been well explored by Bellare and Palacio [3] and by Birkett and Dent [8].

The first scheme that was proven fully plaintext aware in the standard model was the Cramer-Shoup encryption scheme [15]. This proof relies heavily on the *Diffie-Hellman Knowledge* assumption first introduced by Damgård [14]. This assumption is meant to capture the intuition that the only way the attacker can compute a Diffie-Hellman tuple (g, h, g^r, h^r) from the pair (g, h) is by generating r and computing (g^r, h^r) directly. The definition states that for every attacker $\mathcal{A}$ that outputs (g^r, h^r), there exists an algorithm $\mathcal{A}^*$ that can output r given the random coins of $\mathcal{A}$. This is known as an extractor assumption, as the algorithm $\mathcal{A}^*$ extracts the random value r by observing the execution of $\mathcal{A}$. Birkett and Dent [7] have shown that other schemes with a similar structures to the Cramer-Shoup [13] and Kurosawa-Desmedt [24] schemes are plaintext aware under similar extractor assumptions.

This highlights the most significant problem with the plaintext awareness approach to proving security: no-one has yet managed to prove the plaintext awareness of an encryption scheme without the use of an extractor assumption. These extractor assumption are poor things on which to base the security of an encryption scheme as it is very difficult to gain any evidence about whether the assumption is true or not. It can be as difficult to prove the assumption is false as it is to prove the assumption is true.

6 Conclusion

The cryptographic community have come a long way in proving the security of public-key encryption schemes. However, the ultimate prize is still yet to be claimed: a proof of security for an ultra-efficient encryption scheme in the standard model. The approaches we have discussed in this paper do make significant advantages in improving the efficiency of schemes with full security proofs. However, none of the approaches seem likely to break the final efficiency barrier. Both the "double-and-add" schemes and the identity-based schemes require separate encryption and checksum operations. Hence, the resulting encryption schemes require two "expensive" calculations. On the other hand, the plaintext awareness approach relies on extractor-based assumptions, which do not engender confidence in the security of the scheme, and still do not seem to be able to prove the security of a scheme that uses less than two "expensive" calculations. It seems as if a new technique has to be developed before this barrier can be broken.

Acknowledgements I'd like to thank Prof. Serge Vaudenay and Prof. Abdelhak Azhari for extending the invitation to me to give an invited talk at Africacrypt 2008, and I would like to stress that this paper has not been refereed by a peer review process. Thus, any mistakes or inaccuracies in the paper should be considered mine alone and no blame should be attached to the programme committee or any external reviewers. I'd also like to thank James Birkett, Gaven Watson and Jonathan Katz for their comments on the paper.

References

1. M. Bellare, A. Boldyreva, and J. Staddon. Multi-recipient encryption schemes: Security notions and randomness re-use. In Y. G. Desmedt, editor, *Public Key Cryptography – PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 85–99. Springer-Verlag, 2003.
2. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In H. Krawczyk, editor, *Advances in Cryptology – Crypto '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–45. Springer-Verlag, 1998.
3. M. Bellare and A. Palacio. Towards plaintext-aware public-key encryption without random oracles. In P. J. Lee, editor, *Advances in Cryptology – Asiacrypt 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 48–62. Springer-Verlag, 2004.
4. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proc. of the First ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

5. M. Bellare and P. Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *Advances in Cryptology – Eurocrypt '94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. Springer-Verlag, 1994.
6. M. Bellare and M. Yung. Certifying permutations: Non-interactive zero-knowledge based on any trapdoor permutation. *Journal of Cryptology*, 9(1):149–166, 1996.
7. J. Birkett and A. W. Dent. The generalised Cramer-Shoup and Kurosawa-Desmedt schemes are plaintext aware. Unpublished Manuscript, 2008.
8. J. Birkett and A. W. Dent. Relations among notions of plaintext awareness. In R. Cramer, editor, *Public Key Cryptography – PKC 2008*, volume 4939 of *Lecture Notes in Computer Science*, pages 47–64. Springer-Verlag, 2008.
9. D. Boneh and J. Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In A. Menezes, editor, *Topics in Cryptology - CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 87–103. Springer-Verlag, 2005.
10. X. Boyen, Q. Mei, and B. Waters. Direct chosen ciphertext security from identity-based techniques. In *Proc. of the 12th ACM Conference on Computer and Communications Security*, pages 320–329, 2005.
11. R. Canetti, O. Goldreich, and S. Halvei. The random oracle model, revisited. In *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing*, pages 209–218, 1998.
12. R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology – Eurocrypt 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222. Springer-Verlag, 2004.
13. R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2004.
14. I. B. Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In J. Feigenbaum, editor, *Advances in Cryptology – Crypto '91*, volume 576 of *Lecture Notes in Computer Science*, pages 445–456. Springer-Verlag, 1991.
15. A. W. Dent. The Cramer-Shoup encryption scheme is plaintext aware in the standard model. In S. Vaudenay, editor, *Advances in Cryptology – Eurocrypt 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 289–307. Springer-Verlag, 2006.
16. A. W. Dent. Fundamental problems in provable security and cryptography. *Phil. Trans. R. Soc. A*, 364(1849):3215–3230, Dec 2006.
17. D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. In *Proc. 23rd Symposium on the Theory of Computing – STOC 1991*, pages 542–552. ACM, 1991.
18. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.
19. U. Feige, D. Lapidot, and A. Shamir. Multiple noninteractive zero knowledge proofs under general assumptions. *SAIM Journal on Computing*, 29(1):1–28, 1999.
20. O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proc. 21st Symposium on Theory of Computer Science – STOC 1989*, pages 25–32. ACM, 1989.
21. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Science*, 28:270–299, 1984.
22. J. Herzog, M. Liskov, and S. Micali. Plaintext awareness via key registration. In D. Boneh, editor, *Advances in Cryptology – Crypto 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 548–564. Springer-Verlag, 2003.

23. E. Kiltz. Chosen-ciphertext security from tag-based encryption. In S. Halevi and T. Rabin, editors, *Theory of Cryptography – TCC 2006*, volume 3876 of *Lecture Notes in Computer Science*, pages 581–600. Springer-Verlag, 2006.
24. K. Kurosawa and Y. Desmedt. A new paradigm of hybrid encryption scheme. In M. Franklin, editor, *Advances in Cryptology – Crypto 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 426–442. Springer-Verlag, 2004.
25. P. MacKenzie, M. K. Reiter, and K. Yang. Alternatives to non-malleability: Definitions, constructions and applications. In M. Naor, editor, *Theory of Cryptography – TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 171–190. Springer-Verlag, 2004.
26. M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proc. 22nd Symposium on the Theory of Computing – STOC 1990*, pages 427–437. ACM, 1990.
27. M. O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.
28. C. Rackoff and D. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In J. Feigenbaum, editor, *Advances in Cryptology – Crypto '91*, volume 576 of *Lecture Notes in Computer Science*, pages 434–444. Springer-Verlag, 1991.
29. A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *Proc. 40th Annual Symposium on Foundations of Computer Science – FOCS '99*, pages 543–553. IEEE Computer Society, 1999.
30. I. Teranishi and W. Ogata. Relationship between standard model plaintext awareness and message hiding. In X. Lai and K. Chen, editors, *Advances in Cryptology – Asiacrypt 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 226–240. Springer-Verlag, 2006.