J. Cryptology (2007) 20: 203–235
DOI: 10.1007/s00145-007-0211-0

Journal of
CRYPTOLOGY

Formal Proofs for the Security of Signcryption*

Joonsang Baek
Institute for Infocomm Research,
21 Heng Mui Keng Terrace, Singapore 119613
jsbaek@i2r.a-star.edu.sg

Ron Steinfeld
Department of Computing, Macquarie University,
North Ryde, NSW 2109, Australia
rons@ics.mq.edu.au

Yuliang Zheng
Department of Software and Information Systems,
University of North Carolina at Charlotte,
Charlotte, NC 28223, U.S.A.
yzheng@uncc.edu

Communicated by Dan Boneh

Received 10 May 2002 and revised 22 December 2006
Online publication 22 March 2007

Abstract. Signcryption is an asymmetric cryptographic method that provides simultaneously both message confidentiality and unforgeability at a low computational and communication overhead. In this paper we propose realistic security models for signcryption, which give the attacker power to choose both messages/signcryptexts as well as recipient/sender public keys when accessing the signcryption/unsigncryption oracles of attacked entities. We then show that Zheng's original signcryption scheme is secure in our confidentiality model relative to the Gap Diffie–Hellman problem and is secure in our unforgeability model relative to a Gap version of the discrete logarithm problem. All these results are shown in the random oracle model.

Keywords. Signcryption, Flexible signcryption/unsigncryption oracle models, Gap Diffie–Hellman problem, Gap discrete log problem.

* A part of this work was done while the first author was with the School of Network Computing, Monash University (Australia)/the School of Information Technology and Computer Science, the University of Wollongong (Australia) and the second author was with the School of Network Computing, Monash University.

1. Introduction

1.1. *Motivation*

Message confidentiality is one of the most important goals of cryptography, both in the symmetric and asymmetric settings. Over the last decade, in the asymmetric setting, a number of encryption schemes meeting strong confidentiality requirements, such as security against adaptive chosen ciphertext attacks [19], [24], have emerged. Early constructions of such schemes include Zheng and Seberry's [36] public key encryption schemes, which are efficient but were not proven to be secure against chosen ciphertext attacks "in the reductionist way" (namely, in such a way that presents a reduction from attacking cryptographic schemes to solving well-known computationally hard problems). Shortly after Zheng and Seberry's proposals, several other schemes [8], [14], [22] were proposed, whose security against chosen ciphertext attacks can be analyzed in the reductionist way under an additional heuristic assumption known as the random oracle model [7]. The first practical scheme that does not depend on the random oracle model was given by Cramer and Shoup [10] and received great attention from the cryptographic community.

Along with message confidentiality, message authenticity is another important goal of cryptography. In the asymmetric setting, this goal was realized by the advent of digital signatures. The essential security requirement for digital signatures is the (existential) unforgeability against adaptive chosen message attacks [16], where an attacker is allowed to query a number of messages of his choice to the signing oracle. Note that slight modifications of the classical ElGamal signature [12] and the Schnorr signature [26] schemes were proved [23], [20] to be secure in this sense, that is, (existentially) unforgeable against adaptive chosen message attack [16] (in the random oracle model).

A natural question one can now ask is how to integrate encryption and signature schemes in an efficient way without sacrificing each scheme's security, in other words, how to provide efficiently communicating messages with confidentiality and authenticity *simultaneously* as one cryptographic function. In 1997 Zheng [33] gave a positive answer to the question: He proposed a cryptographic scheme called "signcryption" which integrates the functionality of discrete log based public key encryption and digital signature schemes in a very efficient way.

Although Zheng's signcryption scheme has been the focus of a number of research works, no reductionist-style security analysis of Zheng's signcryption, as far as we know, has ever been given. In this paper we propose precise and realistic security models for generic signcryption schemes and provide rigorous proofs, based on these proposed models, that Zheng's original signcryption scheme meets strong security requirements with respect to message confidentiality and unforgeability under known cryptographic assumptions.

1.2. *Related Work*

Compared with the asymmetric setting, research on the integration of message confidentiality and authenticity was relatively more active in the symmetric setting. A series of research works appeared on using modes of block ciphers to give both message confidentiality and integrity [17], [25]. Also, security issues related to the composition of symmetric key encryption and message authentication code (MAC) were considered by Bellare and Namprepre [6]. They concluded that only "Encrypt-then-MAC (EtM)" composition is *generically* secure against chosen ciphertext attack and existentially unforgeable against chosen message attack. Krawczyk [18] also considered the same problem when building a secure channel over insecure networks. Interestingly, his conclusion was that the "MAC-then-Encrypt (MtE)" composition is also secure, under the assumption that the encryption method is either a secure CBC mode or a stream cipher that XORs the data with a random pad.

In the asymmetric setting, Tsiounis and Yung [32] proposed a variant of the ElGamal encryption scheme where Schnorr's signature is used to provide non-malleability. However, the security goal of their scheme is to provide confidentiality; consequently, the strong origin authentication is not supported in their scheme. (Note that this scheme was analyzed again by Schnorr and Jakobsson [27] under the generic model plus the random oracle model. Note also that the security proof of Tsiounis and Yung's scheme given in [32] was later found to be flawed [29]: The Schnorr signature scheme that was used as a "proof of knowledge" in their public key encryption scheme, makes it impossible to simulate efficiently the responses to the chosen ciphertext attacker's decryption queries. We refer readers to [29] for more details.)

The first attempt to provide formal security analysis of signcryption schemes was made by Steinfeld and Zheng [31], who proposed a signcryption scheme based on the integer factorization problem and provided a formal security model and proof for the unforgeability of the proposed scheme. However, a formal security model and proof for the confidentiality of their scheme was not provided. (We note, however, that following an earlier version of this work, the analysis of the factoring-based signcryption scheme has been extended to cover both confidentiality and unforgeability in the strong sense that is presented in the following sections [30]. Interestingly, although the result in [30] for confidentiality is analogous to ours in its reliance on a variant of the Gap Diffie–Hellman assumption in a subgroup of $\mathbb{Z}_N^*$ for N an RSA modulus, the unforgeability result in [30], for a suitable choice of scheme parameters, does not rely on a "gap" assumption, but only on the hardness of factoring an RSA modulus N, given a generator for the utilized subgroup of $\mathbb{Z}_N^*$.)

Independently of our work, security models for signcryption similar to ours were proposed by An et al. [1], who analyzed the security of *generic* compositions of blackbox signature and encryption schemes. Our unforgeability notion FSO-UF-CMA, which is defined precisely in Section 3.3, corresponds to unforgeability in the "Multi-User Insider" setting defined by An et al. in [1], whereas our confidentiality notion FSO/FUO-IND-CCA2, which is defined precisely in Section 3.2, corresponds to confidentiality in the weaker "Multi-User Outsider" setting of An et al. In Section 1.3.2 we discuss our models and their relationship to those defined by An et al. in great detail.

1.3. *Differences between Our Security Model and Other Models*

1.3.1. *Differences between Symmetric and Asymmetric Models*

To address the significant difference between security implication of the compositions of encryption and authentication in the symmetric setting and that in the asymmetric setting, we consider confidentiality of the "Encrypt-then-MAC (EtM)" and "Encrypt-and-MAC

(EaM)" compositions in the symmetric setting, and the security of the directly corresponding simple asymmetric versions "Encrypt-then-Sign (EtS)" and "Encrypt-and-Sign (EaS)" (defined in the natural way, with the signer's public key appended). We point out that while the symmetric composition EtM is secure against chosen ciphertext attack (indeed, EtM is generically secure as shown in [6]), the simple asymmetric version EtS is *completely insecure against adaptive chosen ciphertext attack*, even if the underlying encryption scheme is secure against adaptive chosen ciphertext attack. The reason is that in the asymmetric version, a ciphertext in the composed scheme contains an additional component (not present in the symmetric versions), namely, the *sender's signature public key*. The fact that this component is easily malleable implies the insecurity of the asymmetric version EtS under adaptive chosen ciphertext attack.

As an example, assume that a sender Alice encrypts and signs her message m using the EtS composition. That is, she encrypts the message m using a public key encryption algorithm $\mathsf{E}_{pk_B}(\cdot)$ and computes $c = \mathsf{E}_{pk_B}(m)$. Then she signs on c using her digital signature algorithm $\mathsf{S}_{sk_A}(\cdot)$ to produce $\sigma = \mathsf{S}_{sk_A}(c)$. Now the ciphertext C is (c, σ). However, an adversary Marvin now generates his own public and private key pair (pk_M, sk_M) and signs on c obtained by eavesdropping the ciphertext C en route from Alice to Bob. Namely, he can produce $C' = (c, \mathsf{S}_{sk_M}(c))$ where $\mathsf{S}_{sk_M}(\cdot)$ is Marvin's digital signature algorithm. Then he hands in his public key pk_M (which may be contained in Marvin's digital certificate) to Bob. Now notice that C' which is different from C is completely verified as a valid ciphertext using Marvin's public key pk_M and Bob decrypts it into m. Hence Marvin succeeds in his chosen ciphertext attack on the EtS scheme even if the underlying asymmetric encryption scheme is strong, say, secure against adaptive chosen ciphertext attack. (For completeness, we remark that a secure *generic* EtS variant which fixes the above problem of the simple EtS was given by An et al. [1].)

1.3.2. *Discussion of Our Models in the Context of Other Asymmetric Models*

The discussion in this section focuses on explaining the relationship between security models for signcryption schemes defined by An et al. [1] and our security notions as defined in Section 3. First we review the classification of security models for signcryption schemes defined by An et al. [1].

Two-User versus Multi-User Setting. The first classification of security models for signcryption schemes depends on the assumed application setting. In the "Two-User" setting, it is assumed that there are only two users of the scheme: a single sender Alice with key pair (sk_A, pk_A) and a single receiver Bob with key pair (sk_B, pk_B). Hence, in this setting, the receiver's public key for all messages signcrypted by Alice is fixed to Bob's public key pk_B. Similarly, the sender's public key for all signcryptexts unsigncrypted by Bob is fixed to Alice's public key pk_A. In contrast, the "Multi-User" setting assumes that there are many users of the scheme besides the attacked users Alice and Bob. Thus, in this setting the receiver's public key for messages signcrypted by Alice can be *any* receiving user's public key pk_R (not necessarily Bob's pk_B). Similarly, the sender's public key for signcryptexts unsigncrypted by Bob can be any sending user's public key pk_S (not necessarily Alice's pk_A). In particular, in this setting the attacker is given the power to choose his own reciever/sender public keys when accessing Alice/Bob's signcryption/unsigncryption oracles. This power does not exist in the Two-User setting.

Insider versus Outsider Setting. The second classification of security models for signcryption schemes depends on the identity of the attacker. In the "Outsider" setting, the attacker is assumed to be a *third party*, distinct from both the attacked users Alice and Bob. To break confidentiality in this setting, the goal of the attacker is to recover some information on a message signcrypted by Alice to Bob, assuming the signcryptext has not been unsigncrypted by Bob. To break unforgeability in this setting, the goal of the attacker is to forge a signcryptext from Alice to Bob on a message which has not been signcrypted by Alice. Note that in the outsider setting, since the attacker is a third party, he only knows the *public* keys of Alice and Bob. In contrast, in the "Insider" setting, the attacker is assumed to be a *second party*, meaning that the attacker is either Alice (attacking Bob's confidentiality) or Bob (attacking Alice's unforgeability). To break Bob's confidentiality in this setting, Alice's goal is to recover any partial information on a message signcrypted to Bob with Alice's public key as the sender's public key, assuming the signcryptext has not been unsigncrypted by Bob with Alice's public key as the sender's public key (note that in this setting, the attacker Alice may know the sender's private key). To break Alice's unforgeability in this setting, Bob's goal is to forge a valid signcryptext from Alice to Bob on a message which has never been signcrypted by Alice to Bob (note that in this setting the attacker Bob may know the receiver's private key).

Our Confidentiality Notion. The strongest confidentiality notion for signcryption schemes is obtained by requiring confidentiality in the "Multi-User Insider" setting. It is easy to verify that Zheng's signcryption scheme is completely insecure in this setting because the Diffie–Hellman key $g^{x_A x_B}$ (which is easily recoverable by the sender Alice) defined by Alice and Bob's public keys g^{x_A} and g^{x_B} suffices to unsigncrypt *any* signcryptext from Alice to Bob. However, we make the following observations. First, we emphasize, as also acknowledged in [1], that this model is under normal circumstances not of significant importance because it effectively assumes that the sender Alice is trying to unsigncrypt a signcryptext which was sent by herself. Thus this model appears only useful in providing "forward security" under special circumstances in which an attacker who breaks into Alice's system obtains her secret key in order to unsigncrypt a message previously signcrypted by Alice to Bob. Second, as pointed out by Zheng in the full version of the original paper [34], this insecurity can be considered a positive feature, called "Past Message Recovery", since it allows Alice to store signcryptexts and unsigncrypt them in the future when desired.

In view of the above discussion, we believe that for most applications it suffices for a signcryption scheme to achieve confidentiality in the "Multi-User Outsider" setting. Our independently defined confidentiality notion "FSO/FUO-IND-CCA2" for this setting matches the corresponding definition by An et al. [1].

1.3.2.1. *Our Unforgeability Notion*. The strongest unforgeability notion for signcryption schemes corresponds to unforgeability in the "Multi-User Insider" setting. Our independently defined unforgeability notion "FSO-UF-CMA" for this setting matches the corresponding definition by An et al. [1].

Like the model proposed by An et al. [1], our model also does not explicitly include support for *non-repudiation*, that is, the ability of a receiver of a valid signcryptext to convince a third party that a given sender has sent this signcryptext. However, as also pointed out in [1], unforgeability in the sense of FSO-UF-CMA guarantees that the receiver cannot forge any valid signcryptext by the sender, so non-repudiation can always be achieved using a protocol run between the receiver and the third party, which convinces the third party of the *validity* of a signcryptext with respect to a given message and sender and receiver public keys. A generic solution which does not compromise the receiver's secret key to the third party, is to use a zero-knowledge proof of signcryptext validity. Specific protocols for Zheng's scheme are presented by Zheng in [34].

On the Power of Attackers in the Multi-User Setting. The extra power given to the attacker in the Multi-User setting is the ability to access "flexible" signcryption/unsigncryption oracles which allow the attacker to specify a receiver/sender's public key in addition to a message/signcryptext. In a practical application, such an attack might be mounted by the attacker Marvin by requesting a new public key certificate from the Certificate Authority (CA) each time he wants to query Alice's signcryption oracle with a new public key of his choice. A scheme meeting our security notion must be secure even if Marvin can get as many public key certificates issued as he wishes for arbitrary public keys of his choice. In some applications it may be possible to place significant constraints on the public keys that Marvin can use, for example through additional checks by the CA that users "know" the secret key associated to their public key. However, we believe that for the sake of wide applicability one should be conservative and avoid such assumptions if possible.

Security of signcryption in the Two-User setting does not imply security in the Multi-User setting. Furthermore, there is no known *efficient* (in particular, not using encryption/signature primitives) generic conversion of a "Two-User secure" scheme into a "Multi-User secure" scheme. The "semi-generic" efficient conversion given by An et al. [1] only works for the schemes they considered, which are built from separate signature and encryption primitives (the incorrect claim in the conference paper [1] that the conversion is generic was subsequently corrected in an updated version of the paper [2]).

To get a feeling for the issues involved, consider the following example (which can be used as a counterexample to prove that the semi-generic conversion in [1] and [2] is not generic). Given a signcryption scheme secure in the Two-User setting, we construct a new signcryption scheme which is identical except that the signcryption algorithm appends in the signcryptext one bit of the sender's secret key, where the secret bit position is determined as a function of the receiver's public key. In the Two-User setting a forging attacker can only query the sender's signcryption oracle with one receiver public key fixed for the whole attack and hence in this setting the forger can only get a single bit of the secret key. Consequently, the new scheme is still unforgeable in the Two-User setting. On the other hand, in the Multi-User setting the attacker can quickly get all the bits of the sender's secret key by querying the signcryption oracle with many different receiver public keys, so the scheme is easily forgeable in the Multi-User setting (and it remains forgeable in the Multi-User setting, for the same reason, even after applying the semi-generic conversion in [1] and [2]). This example is not entirely artificial—indeed, it is because of an interaction between the receiver's public key and the sender's secret key in Zheng's signcryption scheme that we need in this paper, for instance, the "*Gap* Discrete Log" assumption to prove unforgeability in the Multi-User setting, whereas just the weaker "Discrete Log" assumption suffices for unforgeability in the Two-User setting [3].

Other Assumptions. We point out two implicit assumptions we have made in the current work. The first is that our Multi-User models apply to "static" attackers because the attacked public keys are fixed at the beginning of the attack game. The second is that our scheme assumes the standard practice that each user generates two independent private/public key-pairs for sending and receiving, respectively. However, we remark that our security proofs for Zheng's scheme under the Gap Diffie–Hellman (GDH) assumption can be extended to the "key-reuse" setting where a single key-pair is used for both signcryption and unsigncryption (this involves simulating the additional oracles present in this setting in the same way as the oracle simulations performed in the current proofs).

1.4. *Our Main Results*

The most attractive feature of Zheng's signcryption scheme is its efficiency. Namely, the dominant computational cost in both signcryption and unsigncryption algorithms is approximately only a *single* exponentiation in the underlying subgroup. This high efficiency is achieved by *sharing* the exponentiation for both the encryption and signature "portions" of the computation, and is therefore at least twice as efficient as a generic composition (using one of the generic compositions presented in [1]) of discrete log based signature and encryption schemes, each of which would presumably perform (at least) one separate exponentiation.

Our results demonstrate that despite its high efficiency, Zheng's scheme still achieves strong security notions in the Multi-User setting with respect to known cryptographic assumptions and the random-oracle model for the underlying hash functions. In particular, our main results can be summarized as follows. First, we prove, in the random-oracle model, that Zheng's scheme achieves confidentiality in the *Multi-User Outsider* setting (or equivalently "FSO/FUO-IND-CCA2", which is formally defined in the next section) under the GDH assumption [21] in a prime-order subgroup of $\mathbb{Z}_p^*$, where p is prime, and the assumption that the underlying one-time symmetric encryption scheme is secure. Second, we prove, in the random oracle model, that Zheng's scheme achieves unforgeability in the *Multi-User Insider* setting (or equivalently "FSO-UF-CMA", which is formally defined in the next section) assuming the Gap Discrete Log (GDL) assumption in the underlying subgroup, which is implied by, but is possibly a weaker assumption than, the GDH assumption in the same subgroup.

We note that Zheng's scheme relies for its security on specific number-theoretic computational complexity assumptions, and on the random oracle model. These assumptions may be avoided, at the cost of efficiency, by using a generic encryption/signature composition scheme and applying the results in [1].

2. Preliminaries

We use the notation $\mathsf{A}(\cdot, \cdot)$ to denote an algorithm, with input arguments separated by commas (our underlying computational model is a Turing Machine). If algorithm A makes calls to oracles, we list the oracles separated from the algorithm inputs by the symbol "|". For a probabilistic algorithm $\mathsf{A}(\cdot)$, we use $\mathsf{A}(x; r)$ to denote the output of A on input x with a randomness input r. If we do not specify r explicitly we do so with the understanding that r is chosen statistically independent of all other variables. We denote by $\{\mathsf{A}(x)\}$ the set of outputs of A on input x as we sweep the randomness input for A through all possible strings.

We denote by $\langle g \rangle$ a subgroup generated by a group element g.

We denote $|\cdot|$ as the number of bits in the binary representation of an input.

Given a set SP_{sk} we denote by $sk \stackrel{R}{\leftarrow} SP_{sk}$ the assignment of a uniformly and independently distributed random element from the set SP_{sk} to the variable sk.

Let $\mathbb{Z}_n^* = \{x \in \mathbb{Z}_n \mid \gcd(x, n) = 1\}$. (Note that if q is prime, $\mathbb{Z}_q^* = \mathbb{Z}_q \backslash \{0\}$.)

For integers g and p, we let $\mathrm{Ord}_p(g)$ denote the order of g in the multiplicative group $\mathbb{Z}_p^*$.

We say a probability function $f: \mathbb{N} \to \mathbb{R}_{[0,1]}$ is negligible in k if, for all $c > 0$, there exists $k_0 \in \mathbb{N}$ such that $f(k) \leq 1/k^c$ whenever $k \geq k_0$. Here, $\mathbb{R}_{[0,1]} = \{x \in \mathbb{R} \mid 0 \leq x \leq 1\}$.

3. Our Security Notions for Signcryption Schemes

3.1. *Description of Generic Signcryption Scheme*

First we formally define a "signcryption" scheme in a general way as follows.

Definition 1 (Generic Signcryption Scheme). A signcryption scheme $\mathcal{SCR} = (\mathsf{GC}, \mathsf{GK}_A, \mathsf{GK}_B, \mathsf{SC}, \mathsf{USC})$ consists of the following algorithms:

1. A probabilistic common parameter/oracle generation algorithm GC that takes a security parameter k as input, and returns a sequence of common parameters cp containing the security parameter k and other system-wide parameters such as the description of computational groups and hash functions.
2. A probabilistic sender key-pair generation algorithm GK_A that takes a common parameter sequence cp as input and returns a sender's secret/public key-pair (sk_A, pk_A).
3. A probabilistic receiver key-pair generation algorithm GK_B that takes a common parameter sequence cp as input and returns a receiver's secret/public key-pair (sk_B, pk_B).
4. A probabilistic signcryption algorithm SC that takes a common parameter sequence cp, a sender's secret key sk_A, a receiver's public key pk_B, and a message $m \in SP_m$ (SP_m is the message space) as input, and returns a signcryptext C.
5. An unsigncryption algorithm USC that takes a common parameter sequence cp, a receiver's secret key sk_B, a sender's public key pk_A, and a signcryptext C as input, and returns either a message m or a "*Rej* (reject)" symbol.

3.2. *Confidentiality Notion for Signcryption Schemes in the FSO/FUO Model*

Following the discussions in Section 1.3, we provide an attack model for confidentiality of the generic signcryption scheme $\mathcal{SCR}$, which we call the "Flexible Signcryption Oracle/Flexible Unsigncryption Oracle (FSO/FUO)" model. In this model the adversary Marvin's goal is to break the confidentiality of messages between the sender Alice and the receiver Bob. Marvin is given Alice's public key pk_A^* and Bob's public key pk_B^*, and has access to a "flexible" signcryption oracle, as well as a "flexible" unsigncryption oracle: on receiving (pk_R, m) where pk_R denotes a receiver's public key generated by Marvin at will (Marvin may choose the receiver's public key as Bob's public key pk_B^*, say, $pk_R = pk_B^*$.) and m denotes a plaintext, the flexible signcryption oracle returns a signcryptext after performing signcryption under Alice's private key sk_A^*. We denote the flexible signcryption oracle by $\mathsf{SC}(cp, sk_A^*, \cdot, \cdot)$ where no specified receiver's public key is presented as the input argument. On the other hand, the flexible unsigncryption oracle, on receiving (pk_S, C) where pk_S denotes a sender's public key generated by Marvin at will (similarly to the flexible signcryption oracle, Marvin may choose the sender's public key as Alice's public key pk_A^*, say, $pk_S = pk_A^*$) and C denotes a signcryptext, returns a plaintext or a "*Rej*" (Reject) symbol after performing unsigncryption under Bob's private key sk_B^*. Note that the unsigncryption oracle is denoted by $\mathsf{USC}(cp, sk_B^*, \cdot, \cdot)$, where no specified sender's public key is presented as the input argument.

In other words, the flexible signcryption and unsigncryption oracles are not constrained to be executed only under pk_B^* and pk_A^*, respectively—Bob and Alice's public key can be replaced by the public keys generated by Marvin. Accordingly, the FSO/FUO model gives Marvin the full chosen-plaintext/ciphertext power with the ability to choose the sender and receiver's public keys, the message as well as the signcryptext.

Using the notion of indistinguishability of encryption [5], [15], we formalize the confidentiality of signcryption against the above-described (adaptive) chosen ciphertext attack under the FSO/FUO model. We say a signcryption scheme is secure in the sense of indistinguishability (abbreviated by "IND"), there is no polynomial-time adversary that can learn any information about the plaintext from the signcryptext except for its length. Following the style of [5], we call this confidentiality notion of signcryption "FSO/FUO-IND-CCA2." Below, we formally define FSO/FUO-IND-CCA2.

Definition 2 (FSO/FUO-IND-CCA2). Let $\mathcal{SCR} = (\mathsf{GC}, \mathsf{GK}_A, \mathsf{GK}_B, \mathsf{SC}, \mathsf{USC})$ be a generic signcryption scheme. Let $\mathsf{A}^{\mathsf{CCA}}$ be an attack algorithm (attacker) against the indistinguishability of the scheme $\mathcal{SCR}$. Consider the following attack game:

SCRINDGame$(k, \mathsf{A}^{\mathsf{CCA}}, \mathcal{SCR})$
$cp \leftarrow \mathsf{GC}(k)$
$(sk_A^*, pk_A^*) \xleftarrow{R} \mathsf{GK}_A(cp)$
$(sk_B^*, pk_B^*) \xleftarrow{R} \mathsf{GK}_B(cp)$
$(m_0, m_1) \leftarrow \mathsf{A}^{\mathsf{CCA}}(k, cp, \mathsf{find}, pk_A^*, pk_B^* \mid \mathsf{SC}(cp, sk_A^*, \cdot, \cdot), \mathsf{USC}(cp, sk_B^*, \cdot, \cdot))$
$\beta \xleftarrow{R} \{0, 1\}$; $C^* \leftarrow \mathsf{SC}(cp, sk_A^*, pk_B^*, m_\beta)$
$\beta' \leftarrow \mathsf{A}^{\mathsf{CCA}}(k, cp, \mathsf{guess}, pk_A^*, pk_B^*, C^* \mid \mathsf{SC}(cp, sk_A^*, \cdot, \cdot), \mathsf{USC}(cp, sk_B^*, \cdot, \cdot))$
`If` $\beta' = \beta$ and (pk_A^*, C^*) was never queried to $\mathsf{USC}(cp, sk_B^*, \cdot, \cdot)$
`Return` 1 `Else Return` 0

Note that two messages m_0 and m_1 output by the attacker satisfy $|m_0| = |m_1|$. Note also that $\mathsf{A}^{\mathsf{CCA}}$ is *allowed* to query (pk_S, C^*) to the unsigncryption oracle $\mathsf{USC}(cp, sk_B^*, \cdot, \cdot)$ where unsigncryption is performed under the public key pk_S which is arbitrarily chosen by $\mathsf{A}^{\mathsf{CCA}}$ and is different from pk_A^*.

We quantify $\mathsf{A}^{\mathsf{CCA}}$'s success by the probability

$$\mathbf{Succ}^{\text{FSO/FUO-IND-CCA2}}_{\mathsf{A}^{\mathsf{CCA}},\mathcal{SCR}}(k) \stackrel{\text{def}}{=} 2\Pr[\mathbf{SCRINDGame}(k, \mathsf{A}^{\mathsf{CCA}}, \mathcal{SCR}) = 1] - 1.$$

We also quantify the insecurity of scheme $\mathcal{SCR}$ in the sense of FSO/FUO-IND-CCA2 against arbitrary attackers with resource parameters $RP = (t, q_{\mathsf{SC}}, q_{\mathsf{USC}})$ by the advantage

$$\mathbf{InSec}^{\text{FSO/FUO-IND-CCA2}}_{\mathcal{SCR}}(t, q_{\mathsf{SC}}, q_{\mathsf{USC}}) \stackrel{\text{def}}{=} \max_{\mathsf{A}^{\mathsf{CCA}} \in AS_{RP}} \{\mathbf{Succ}^{\text{FSO/FUO-IND-CCA2}}_{\mathsf{A}^{\mathsf{CCA}},\mathcal{SCR}}(k)\}.$$

The attacker set AS_{RP} contains all attackers with resource parameters RP, meaning running time + program size at most t, and at most q_{SC} and q_{USC} queries to the signcryption and unsigncryption oracles, respectively.

We say $\mathcal{SCR}$ is FSO/FUO-IND-CCA2 secure if $\mathbf{InSec}^{\text{FSO/FUO-IND-CCA2}}_{\mathcal{SCR}}(t, q_{\mathsf{SC}}, q_{\mathsf{USC}})$ is a negligible function in k for any polynomials t, q_{SC}, and q_{USC} in k.

3.3. *Unforgeability Notion for Signcryption Schemes in the FSO Model*

We now present our unforgeability notion which we call "FSO-UF-CMA", meaning unforgeability of signcryption against adaptive chosen message attack with respect to the FSO model. Recall that this notion corresponds to An et al.'s Multi-User Insider model.

The model is as follows. The forger Marvin's goal is to forge a valid signcryptext from Alice to some other user. Marvin is given Alice's (random) public key pk_A^*. In addition, Marvin is given access to Alice's *flexible* signcryption oracle (FSO), namely $\mathsf{SC}(cp, sk_A^*, \cdot, \cdot)$. Marvin can choose any receiver public key pk_R and message m and query the flexible signcryption oracle to get a signcryptext by Alice on message m to the specified receiver's public key pk_R. At the end of the attack, Marvin is considered successful in his forgery if he produces a forgery signcryptext C^* and a forgery receiver public key pk_R^* such that: (1) C^* is a valid signcryptext from Alice to the receiver who holds a public key pk_R^* (this means that $\mathsf{USC}(cp, sk_R^*, pk_A^*, C^*)$ does not reject, where sk_R^* is the private key corresponding to the forgery recipient public key pk_R^*), and (2) Marvin did not query (pk_R^*, m^*) to Alice's flexible signcryption oracle, where $m^* = \mathsf{USC}(cp, sk_R^*, pk_A^*, C^*)$ is the forgery message.

We remark that, because it applies to the Multi-User setting, our new unforgeability model is stronger than those which appeared in our earlier works [31], [3] in two ways. First, earlier models allowed Marvin only chosen message access to Alice's signcryption oracle with a fixed receiver public key, whereas we allow Marvin full flexibility in choosing the receiver public key pk_M. Second, in earlier models Marvin's goal was to forge a signcryptext from Alice to a specified receiver (who possesses a fixed receiver public key). However, in our new model, we allow Marvin full flexibility in choosing a receiver whose receiver public key is denoted by pk_R^*. Note that our new forgery goal is very weak: we do not even require Marvin to demonstrate "knowledge" of the secret key sk_R^* corresponding to pk_R^*, and we allow either (i) conventional forgeries, where the message m^* is "new" (as in previous models) or (ii) "Recipient Transfer" forgery, where the forgery message m^* was previously queried to Alice's signcryption oracle but it was never signcrypted under the recipient key pk_R^*. We remark that a "Recipient Transfer" forgery was called a "Double Spending" attack in [34], due to its implication in e-commerce payment applications.

Finally, one may wonder why we do not give the attacker access to the sender's *unsigncryption* oracle. The reason is that we assume the well-established practice that users generate independent key-pairs for sending and receiving. In this setting it is clear that the sender's unsigncryption oracle cannot help a forger because the forger can simulate such an oracle by himself.

We now give the precise definition of our new unforgeability notion FSO-UF-CMA.

Definition 3 (FSO-UF-CMA). Let $\mathcal{SCR} = (\mathsf{GC}, \mathsf{GK}_A, \mathsf{GK}_B, \mathsf{SC}, \mathsf{USC})$ be a signcryption scheme. Let A^{UF} be an attack algorithm (attacker) against the unforgeability of the scheme $\mathcal{SCR}$. Consider the following attack game:

SCRUFGame$(k, \mathcal{SCR}, \mathsf{A}^{\mathsf{UF}})$
$\quad cp \leftarrow \mathsf{GC}(k)$
$\quad (sk_A^*, pk_A^*) \leftarrow \mathsf{GK}_A(cp)$
$\quad (C^*, pk_R^*) \leftarrow \mathsf{A}^{\mathsf{UF}}(k, cp, pk_A^* \mid \mathsf{SC}(cp, sk_A^*, \cdot, \cdot))$
$\quad$ Find some sk_R^* such that $(sk_R^*, pk_R^*) \in \{\mathsf{GK}_B(k, cp)\}$
$\quad$ `If` such sk_R^* does not exist, `Return` 0
$\quad m^* \leftarrow \mathsf{USC}(cp, sk_R^*, pk_A^*, C^*)$
$\quad$ `If` $m^* \neq Rej$ and (pk_R^*, m^*) has not been queried by A^{UF} to $\mathsf{SC}(cp, sk_A^*, \cdot, \cdot)$
$\quad\quad$ `Return` 1
$\quad$ `Else Return` 0

We quantify A^{UF}'s success in breaking the FSO-UF-CMA security notion of scheme $\mathcal{SCR}$ by the probability

$$\mathbf{Succ}_{\mathsf{A}^{\mathsf{UF}}, \mathcal{SCR}}^{\text{FSO-UF-CMA}}(k) \stackrel{\text{def}}{=} \Pr[\mathbf{SCRUFGame}(k, \mathcal{SCR}, \mathsf{A}^{\mathsf{UF}}) = 1].$$

We quantify the insecurity of scheme $\mathcal{SCR}$ in the sense of FSO-UF-CMA against arbitrary attackers with resource parameters $RP = (t, q_{\mathsf{SC}})$ by the advantage

$$\mathbf{InSec}_{\mathcal{SCR}}^{\text{FSO-UF-CMA}}(t, q_{\mathsf{SC}}) \stackrel{\text{def}}{=} \max_{\mathsf{A}^{\mathsf{UF}} \in AS_{RP}} \mathbf{Succ}_{\mathsf{A}^{\mathsf{UF}}, \mathcal{SCR}}^{\text{FSO-UF-CMA}}(k).$$

The attacker set AS_{RP} contains all attackers with resource parameters RP, meaning running time + program size at most t and at most q_{SC} queries to the signcryption oracle.

We say $\mathcal{SCR}$ is FSO-UF-CMA secure if $\mathbf{InSec}_{\mathcal{SCR}}^{\text{FSO-UF-CMA}}(t, q_{\mathsf{SC}})$ is a negligible function in k for any polynomials t and q_{SC} in k.

4. Zheng's Original Signcryption Scheme

In this section we give a full description of Zheng's original signcryption scheme [33].

4.1. *One-Time Symmetric Key Encryption Scheme*

As a preliminary, we review the definition of the "one-time symmetric key encryption" [11] which serves as a building block for Zheng's original signcryption scheme. In fact, one-time symmetric key encryption schemes are usually used to build hybrid public key encryption schemes as discussed in [11]. The one-time symmetric key encryption scheme defined here plays the same role as the one used in hybrid public key encryption schemes: the symmetric key is used only once to encrypt a single message.

Definition 4 (One-Time Symmetric Key Encryption). A one-time symmetric key encryption scheme $\mathcal{SKE}$ consists of the following algorithms:

1. A deterministic encryption algorithm E that takes a security parameter k, a symmetric key $\tau \in SP_\tau$, and a message $m \in SP_m$ as input, and returns a ciphertext $c \in SP_c$. (Note that SP_m, SP_τ, and SP_c denote, respectively, the message, key, and ciphertext spaces whose size varies as the security parameter k).
2. A deterministic decryption algorithm D that takes a security parameter k, a symmetric key $\tau \in SP_\tau$, and a ciphertext $c \in SP_c$ as input, and returns a message $m \in SP_m$. The function defined by D is one-to-one on SP_c and onto SP_m.

Note that we do not need the security against chosen plaintext attacks [4] for the one-time symmetric key encryption scheme to prove the confidentiality of Zheng's scheme. An appropriate security notion for the one-time symmetric key encryption scheme is given in Section 5.2. On the other hand, we do need in our security proof of security that this scheme is *bijective*, meaning in particular the decryption function is one-to-one on the ciphertext space SP_c (and hence encryption is also deterministic).

We remark that the one-time pad is a computationally efficient and unconditionally secure bijective one-time symmetric encryption scheme suitable for our application. The key size can be reduced by generating it from a short key using a pseudorandom generator, resulting in a computationally secure one-time symmetric encryption scheme.

4.2. *Description of Zheng's Original Signcryption Scheme*

Zheng's signcryption scheme described in this section is based on the shorthand digital signature scheme (SDSS1) [33] which is a variant of ElGamal based signature schemes. Another signcryption scheme SDSS2 can be described and analyzed in a very similar manner presented in this paper so that we only consider the SDSS1-type signcryption scheme.

To simplify the security analysis, we have slightly modified SDSS1. In particular, in our modified scheme the "Diffie–Hellman Key" K is directly provided as input to the hash function H without first being hashed by the other hash function G.

Definition 5 (Zheng's Original Signcryption Scheme). Each subalgorithm of Zheng's original signcryption scheme $\mathcal{ZSCR}$ works as follows:

Zheng's Original Signcryption $\mathcal{ZSCR}$

Common parameter/oracle generation $\mathsf{GC}(k)$
  Choose at random primes p and q such that
  $|p| = k$, $q > 2^{l_q(k)}$, and $q \mid (p-1)$
  (l_q: $\mathbb{N} \to \mathbb{N}$ is a function determining the length of q)
  Choose a random $g \in \mathbb{Z}_p^*$ such that $\mathrm{Ord}_p(g) = q$
  Choose a hash function G: $\{0,1\}^* \to \{0,1\}^{l_{\mathsf{G}}(k)}$
  (l_{G}: $\mathbb{N} \to \mathbb{N}$ is a function determining the length of the output of G)
  Choose a hash function H: $\{0,1\}^* \to \mathbb{Z}_q$
  Choose a bijective one-time symmetric key encryption
  scheme $\mathcal{SKE} = (\mathsf{E}, \mathsf{D})$
  with message/key/ciphertext spaces $SP_m/\{0,1\}^{l_G}/SP_c$
  $cp \leftarrow (k, p, q, g, \mathsf{G}, \mathsf{H}, \mathcal{SKE})$
  `Return` cp

Sender key-pair generation $\mathsf{GK}_A(cp)$
  $x_A \xleftarrow{R} \mathbb{Z}_q^*$; $y_A \leftarrow g^{x_A}$
  $sk_A \leftarrow (x_A, y_A)$; $pk_A \leftarrow y_A$
  `Return` (sk_A, pk_A)

Receiver key-pair generation $\mathsf{GK}_B(cp)$
  $x_B \xleftarrow{R} \mathbb{Z}_q^*$; $y_B \leftarrow g^{x_B}$
  $sk_B \leftarrow (x, y)$; $pk_B \leftarrow y_B$
  `Return` (sk_B, pk_B)

Signcryption $\mathsf{SC}(cp, sk_A, pk_B, m)$
  Parse sk_A as (x_A, y_A); Parse pk_B as y_B
  `If` $y_B \notin \langle g \rangle \backslash \{1\}$ `Return` *Rej*
  $x \xleftarrow{R} \mathbb{Z}_q^*$; $K \leftarrow y_B^x$; $\tau \leftarrow \mathsf{G}(K)$
  $c \leftarrow \mathsf{E}_\tau(m)$; $r \leftarrow \mathsf{H}(m, y_A, y_B, K)$;
  `If` $r + x_A = 0$ `Return` *Rej*
  `Else` $s \leftarrow x/(r + x_A)$
  $C \leftarrow (c, r, s)$
  `Return` C

Unsigncryption $\mathsf{USC}(cp, sk_B, pk_A, C)$
  Parse sk_B as (x_B, y_B); Parse pk_A as y_A
  `If` $y_A \notin \langle g \rangle \backslash \{1\}$ `Return` *Rej*
  Parse C as (c, r, s)
  `If` $r \notin \mathbb{Z}_q$ or $s \notin \mathbb{Z}_q^*$ or $c \notin SP_c$
    `Return` *Rej*
  `Else`
    $\omega \leftarrow (y_A g^r)^s$; $K \leftarrow \omega^{x_B}$; $\tau \leftarrow \mathsf{G}(K)$
    $m \leftarrow \mathsf{D}_\tau(c)$
    `If` $\mathsf{H}(m, y_A, y_B, K) = r$ `Return` m
    `Else Return` *Rej*

Note that the hash functions G: $\{0,1\}^* \to \{0,1\}^{l_G(k)}$ and H: $\{0,1\}^* \to \mathbb{Z}_q$ are modelled as the random oracles [7] in the security analysis. Note also that the key length of the symmetric encryption is actually $l_{\mathsf{G}}(k)$.

5. Security Analysis of Zheng's Signcryption Scheme

In this section we prove the confidentiality and unforgeability of Zheng's signcryption by providing reductions from known cryptographic assumptions. Although we provide a concrete analysis of our reductions, our main goal is to demonstrate the security of signcryption against polynomial-time attackers. Hence we did not attempt to optimize the insecurity bounds for our reductions.

First we recall the definition of the Gap Diffie–Hellman problem given in [21] and define a Gap Discrete Log problem.

5.1. *Computational Primitives*

5.1.1. *Gap Diffie–Hellman* (*GDH*)

At PKC 2001, Okamoto and Pointcheval [21] proposed a new computational problem called a "Gap problem" in which an attacker tries to solve an inverting problem with the help of an oracle that solves a related decisional problem. Namely, the Gap problem is the dual of inverting and decisional problems.

For our proof of confidentiality of Zheng's original signcryption in the security model proposed in this paper, we will need the "Gap Diffie–Hellman (GDH)" problem [21] in which the attacker is given, in addition to the group element g^a and g^b for random $a, b \in \mathbb{Z}_q^*$, access to a Decisional Diffie–Hellman (DDH) oracle $\mathsf{O}^{\mathsf{DDH}}$ that given $(\bar{g}, \bar{g}^u, \bar{g}^v, z) \in \langle g\rangle \times \langle g\rangle \times \langle g\rangle \times \langle g\rangle$ checks whether $z = \bar{g}^{uv}$ or not (it is possible that $\bar{g} = g$, $\bar{g}^u = g^a$, and $\bar{g}^v = g^b$), and tries to find the Diffie–Hellman key $K = g^{ab}$ corresponding to the given pair (g^a, g^b). The GDH assumption says that the GDH problem is computationally intractable. A precise definition follows.

Definition 6 (GDH Assumption). Let $\mathsf{GC}(k)$ be the common parameter generation algorithm that outputs (g, p, q), where p and q are primes such that $|p| = k$, $q \mid (p-1)$, and $q > 2^{l_q(k)}$ where l_q: $\mathbb{N} \to \mathbb{N}$ denotes a function determining the length of q; $g \in \mathbb{Z}_p^*$ satisfies $\mathrm{Ord}_p(g) = q$. Let $\mathsf{A}^{\mathsf{GDH}}$ be an attacker. Define the following game:

GDHGame$(k, \mathsf{A}^{\mathsf{GDH}})$
  $(g, p, q) \leftarrow \mathsf{GC}(k)$
  $a \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$; $b \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$
  $K \leftarrow \mathsf{A}^{\mathsf{GDH}}((g, p, q), g^a, g^b \mid \mathsf{O}^{\mathsf{DDH}}(\cdot,\cdot,\cdot,\cdot))$
  `If` $K = g^{ab}$ then `Return` 1 `Else Return` 0

Here, $\mathsf{O}^{\mathsf{DDH}}(\cdot,\cdot,\cdot,\cdot)$ is a DDH oracle, which, on input $(\bar{g}, \bar{g}^u, \bar{g}^v, z)$, outputs 1 if $z = \bar{g}^{uv}$ and 0 otherwise.

We quantify $\mathsf{A}^{\mathsf{GDH}}$'s success in solving the GDH problem by the probability

$$\mathbf{Succ}^{\mathrm{GDH}}_{\mathsf{A}^{\mathsf{GDH}},\mathbb{Z}_p^*}(k) \stackrel{\mathrm{def}}{=} \Pr[\mathbf{GDHGame}(k, \mathsf{A}^{\mathsf{GDH}}) = 1].$$

Also we quantify the insecurity of the GDH problem against arbitrary attackers with resource parameters $RP = (t, q_{\mathsf{O}^{\mathsf{DDH}}})$ by the probability

$$\mathbf{InSec}^{\mathrm{GDH}}_{\mathbb{Z}_p^*}(t, q_{\mathsf{O}^{\mathsf{DDH}}}) \stackrel{\mathrm{def}}{=} \max_{\mathsf{A}^{\mathsf{GDH}} \in AS_{RP}} \mathbf{Succ}^{\mathrm{GDH}}_{\mathsf{A}^{\mathsf{GDH}},\mathbb{Z}_p^*}(k).$$

The attacker set AS_{RP} contains all attackers with resource parameters RP, meaning running time + program size at most t, and at most $q_{\mathsf{O}^{\mathsf{DDH}}}$ queries to oracle $\mathsf{O}^{\mathsf{DDH}}$.

We say the GDH assumption holds if $\mathbf{InSec}^{\mathrm{GDH}}_{\mathbb{Z}_p^*}(t, q_{\mathsf{O}^{\mathsf{DDH}}})$ is a negligible function in k for any polynomials t and $q_{\mathsf{O}^{\mathsf{DDH}}}$ in k.

5.1.2. *Gap Discrete Log (GDL)*

For our proof of unforgeability of Zheng's original signcryption scheme, we will need the following "Gap Discrete Log (GDL)" problem. The GDL problem is the discrete log analogue of the GDH problem defined above. The GDL problem is possibly easier than the classical discrete log problem because here the attacker is given, in addition to the group element g^a whose discrete log a with respect to a given base g is desired, access to a restricted DDH oracle $\mathsf{O}^{\mathsf{rDDH}}$ that given $(g, g^a, \bar{g}^v, z) \in \langle g \rangle \times \langle g \rangle \times \langle g \rangle \times \langle g \rangle$ checks whether $z = (\bar{g}^v)^a$ or not. Notice that compared with the DDH oracle $\mathsf{O}^{\mathsf{DDH}}$ used in the GDH problem, the first two inputs g and g^a are fixed in the restricted DDH oracle $\mathsf{O}^{\mathsf{rDDH}}$. The GDL assumption says that the GDL problem is computationally intractable. A precise definition now follows.

Definition 7 (GDL Assumption). Let $\mathsf{GC}(k)$ be the common parameter generation algorithm that outputs (g, p, q), where p and q are primes such that $|p| = k$, $q \mid (p-1)$, and $q > 2^{l_q(k)}$ where $l_q\colon \mathbb{N} \to \mathbb{N}$ denotes a function determining the length of q; $g \in \mathbb{Z}_p^*$ satisfies $\mathrm{Ord}_p(g) = q$. Let $\mathsf{A}^{\mathsf{GDL}}$ be an attacker. Define the game

GDLGame$(k, \mathsf{A}^{\mathsf{GDL}})$
  $(g, p, q) \leftarrow \mathsf{GC}(k)$
  $a \stackrel{\mathrm{R}}{\leftarrow} \mathbb{Z}_q^*$
  $a' \leftarrow \mathsf{A}^{\mathsf{GDL}}((g, p, q), g^a \mid \mathsf{O}^{\mathsf{rDDH}}(g, g^a, \cdot, \cdot))$
  `If` $a' = a$ then `Return` 1 `Else Return` 0

Here, $\mathsf{O}^{\mathsf{rDDH}}(g, g^a, \cdot, \cdot)$ is a restricted DDH oracle, which, on input $(g, g^a, \bar{g}^v, z)$, outputs 1 if $z = (\bar{g}^v)^a$ and 0 otherwise.

We quantify $\mathsf{A}^{\mathsf{GDL}}$'s success in solving the GDL problem by the probability

$$\mathbf{Succ}^{\mathrm{GDL}}_{\mathsf{A}^{\mathsf{GDL}},\mathbb{Z}_p^*}(k) \stackrel{\mathrm{def}}{=} \Pr[\mathbf{GDLGame}(k, \mathsf{A}^{\mathsf{GDL}}) = 1].$$

We quantify the insecurity of GDL against arbitrary attackers with resource parameters $RP = (t, q_{\mathsf{O}^{\mathsf{rDDH}}})$ by the probability

$$\mathbf{InSec}^{\mathrm{GDL}}_{\mathbb{Z}_p^*}(t, q_{\mathsf{O}^{\mathsf{rDDH}}}) \stackrel{\mathrm{def}}{=} \max_{\mathsf{A}^{\mathsf{GDL}} \in AS_{RP}} \mathbf{Succ}^{\mathsf{GDL}}_{\mathsf{A}^{\mathsf{GDL}}, \mathbb{Z}_p^*}(k).$$

The attacker set AS_{RP} contains all attackers with resource parameters RP, meaning running time + program size at most t, and at most $q_{\mathsf{O}^{\mathsf{rDDH}}}$ queries to oracle $\mathsf{O}^{\mathsf{rDDH}}$.

We say the GDL assumption holds if $\mathbf{InSec}^{\mathrm{GDL}}_{\mathbb{Z}_p^*}(t, q_{\mathsf{O}^{\mathsf{rDDH}}})$ is a negligible function in k for any polynomials in t and $q_{\mathsf{O}^{\mathsf{rDDH}}}$ in k.

We remark that in the GDH problem, the attacker's goal is weaker, namely to find the Diffie–Hellman key $K = g^{ab}$ (to given base g) corresponding to the given pair (g^a, g^b). Since the discrete log of g^a allows the attacker to compute the Diffie–Hellman key $K = g^{ab}$ easily, it follows that if the attacker can solve the GDL problem, then he can also solve the GDH problem. This means that the GDH assumption implies the GDL assumption. However, the converse may not hold, and the GDL assumption may actually be a weaker assumption than the GDH assumption.

5.2. *Security Notion for One-Time Symmetric Encryption*

We now define a security notion for the one-time symmetric key encryption scheme $\mathcal{SKE}$ presented in Section 4.1. As mentioned earlier, we do not need the security against chosen plaintext attacks for $\mathcal{SKE}$. We merely need the security against a passive attack called "passive indistinguishability of symmetric key encryption (PI-SKE)" [11]. A formal definition follows.

Definition 8 (PI-SKE for One-Time Symmetric Key Encryption). Let $\mathcal{SKE} = (\mathsf{E}, \mathsf{D})$ be a bijective one-time symmetric key encryption scheme. Let A^{PI} be an attacker that defeats the security of $\mathcal{SKE}$ in the sense of PI-SKE. Let $k \in \mathbb{N}$ be a security parameter. A specification for the attack game is as follows:

SKECFGame$(k, \mathsf{A}^{\mathsf{PI}}, \mathcal{SKE})$
$\tau \stackrel{R}{\leftarrow} SP_\tau$
$(m_0, m_1) \leftarrow \mathsf{A}^{\mathsf{PI}}(k, \mathsf{find})$
$\beta \stackrel{R}{\leftarrow} \{0, 1\}$; $c \leftarrow \mathsf{E}_\tau(m_\beta)$
$\beta' \leftarrow \mathsf{A}^{\mathsf{PI}}(k, \mathsf{guess}, m_0, m_1, c)$
`If` $\beta' = \beta$ `Return` 1 `Else Return` 0

We quantify A^{PI}'s success by the probability

$$\mathbf{Succ}^{\text{PI-SKE}}_{\mathsf{A}^{\mathsf{PI}}, \mathcal{SKE}}(k) \stackrel{\mathrm{def}}{=} 2\Pr[\mathbf{SKECFGame}(k, \mathsf{A}^{\mathsf{PI}}, \mathcal{SYM}) = 1] - 1.$$

We quantify the insecurity of scheme $\mathcal{SKE}$ in the sense of PI-SKE against arbitrary attackers with resource parameters $RP = t$ by the advantage

$$\mathbf{InSec}^{\text{PI-SKE}}_{\mathcal{SKE}}(t) \stackrel{\mathrm{def}}{=} \max_{\mathsf{A}^{\mathsf{PI}} \in AS_{RP}} \{\mathbf{Succ}^{\text{PI}-\text{SKE}}_{\mathsf{A}^{\mathsf{PI}}, \mathcal{SKE}}(k)\}.$$

The attacker set AS_{RP} contains all attackers with resource parameters RP, meaning running time + program size at most t.

We say $\mathcal{SKE}$ is PI-SKE secure if $\mathbf{InSec}^{\text{PI-SKE}}_{\mathcal{SKE}}(t)$ is a negligible function in k for any polynomial in t in k.

5.3. *Confidentiality of Zheng's Signcryption Scheme*

For the confidentiality proof of Zheng's original signcryption scheme $\mathcal{ZSCR}$, we adopt the proof methodology recently appearing in the literature. (Readers are referred to the surveys on this technique such as [28] or [9].) We start with the real attack game where the attacker $\mathsf{A}^{\mathsf{CCA}}$ tries to defeat the security of the $\mathcal{ZSCR}$ scheme in the sense of FSO/FUO-IND-CCA defined in Section 3.2. We then modify this game by changing its rules and obtain a new game. Note here that the rules of each game are to describe how variables in the view of $\mathsf{A}^{\mathsf{CCA}}$ are computed. We repeat the modification until we obtain games related to the ability of the attackers A^{PI} and $\mathsf{A}^{\mathsf{GDH}}$ to defeat the security of the one-time symmetric key encryption scheme $\mathcal{SKE}$ and to solve the GDH problem, respectively. When a new game is derived from a previous one, a difference of the views of the attacker in each game might occur. This difference is measured by the technique presented in the following lemma.

Lemma 1. *Let* A_1, A_2, B_1 *and* B_2 *be events defined over some probability space. If* $\Pr[\mathsf{A}_1 \wedge \neg\mathsf{B}_1] = \Pr[\mathsf{A}_2 \wedge \neg\mathsf{B}_2]$, $\Pr[\mathsf{B}_1] \leq \varepsilon$ *and* $\Pr[\mathsf{B}_2] \leq \varepsilon$ *then we have* $|\Pr[\mathsf{A}_1] - \Pr[\mathsf{A}_2]| \leq \varepsilon$.

The proof is a straightforward calculation and can be found in [28] and [9]. We now state and prove the following theorem.

Theorem 1. *If the GDH assumption holds and the bijective one-time symmetric key encryption scheme* $\mathcal{SKE}$ *is PI-SKE secure then Zheng's original signcryption scheme* $\mathcal{ZSCR}$ *is secure in the FSO/FUO-IND-CCA2 sense. Concretely, the following bound holds*:

$$\begin{aligned}
&\mathbf{InSec}^{\text{FSO/FUO-IND-CCA2}}_{\mathcal{ZSCR}}(t, q_{\mathsf{SC}}, q_{\mathsf{USC}}, q_{\mathsf{G}}, q_{\mathsf{H}}) \\
&\quad \leq 2\mathbf{InSec}^{\text{GDH}}_{\mathbb{Z}_p^*}(t', q_{\mathsf{ODDH}}) + \mathbf{InSec}^{\text{PI-SKE}}_{\mathcal{SKE}}(t'') \\
&\qquad + q_{\mathsf{SC}}\left(\frac{q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{SC}} + q_{\mathsf{USC}} + 2}{2^{l_q(k)-1}}\right) + \frac{q_{\mathsf{H}} + 2q_{\mathsf{USC}}}{2^{l_q(k)-1}},
\end{aligned}$$

where $t' = t + O((q_{\mathsf{G}})^2 + 1) + O((q_{\mathsf{H}})^2 + 1) + O(k^3 q_{\mathsf{SC}}) + O((k^3 + q_{\mathsf{G}} + q_{\mathsf{H}})q_{\mathsf{USC}}) + t''(q_{\mathsf{SC}} + q_{\mathsf{USC}})$ *and* $q_{\mathsf{ODDH}} = (q_{\mathsf{SC}} + q_{\mathsf{USC}})(q_{\mathsf{G}} + q_{\mathsf{H}})$.

Proof. Our aim is to keep modifying the real attack game **SCRINDGame** presented in Definition 2 until we get to the stage where we obtain **SKECFGame** in Definition 8 and **GDHGame** in Definition 6.

We use "$\mathsf{A}^{\mathsf{CCA}}$" to refer to the FSO/FUO-IND-CCA2 attacker and use "$\mathsf{A}^{\mathsf{GDH}}$" to refer to the attacker for the GDH problem. Given (k, p, q, g, g^a, g^b) for random $a, b \in \mathbb{Z}_q^*$,

$\mathsf{A}^{\mathsf{GDH}}$'s goal is to compute the Diffie–Hellman key g^{ab} with the help of the DDH oracle $\mathsf{O}^{\mathsf{ODH}}(\cdot, \cdot, \cdot, \cdot)$.

We start with the following game.

• *Game* G_0. This game is the same as the real attack game **SCRINDGame** in Definition 2.

First we run the common parameter/oracle generation algorithm GC of $\mathcal{ZSCR}$ on input a security parameter k and obtain a common parameter $cp = (p, q, g, \mathsf{G}, \mathsf{H}, \mathcal{SKE})$, where p and q are primes such that $|p| = k$, $q > 2^{l_q(k)}$, and $q \mid (p-1)$; g is an element in $\mathbb{Z}_p^*$ such that $\mathrm{Ord}_p(g) = q$; G: $\{0,1\}^* \to \{0,1\}^{l_G(k)}$ and H: $\{0,1\}^* \to \mathbb{Z}_q$ are hash functions modelled as the random oracles [7]; $\mathcal{SKE}$ is the bijective one-time symmetric key encryption scheme that consists of the encryption function E and the decryption function D. We then run the sender/receiver key generation algorithms GK_A and GK_B respectively on input cp and k, and obtain Alice (sender) and Bob (receiver)'s fixed private/public key pairs. Here, Alice's private key consists of (x_A^*, y_A^*) where $y_A^* = g^{x_A^*}$, and her public key is y_A^* itself. Similarly, Bob's private key consists of (x_B^*, y_B^*) where $y_B^* = g^{x_B^*}$, and y_B^* itself is his public key.

We give the public key-pair (y_A^*, y_B^*) to $\mathsf{A}^{\mathsf{CCA}}$. Once $\mathsf{A}^{\mathsf{CCA}}$ submits a pair of plaintexts (m_0, m_1) where $|m_0| = |m_1|$ at the find stage, we pick $\beta \in \{0,1\}$ uniformly at random and create a target signcryptext $C^* = (c^*, r^*, s^*)$ as follows:

$$c^* = \mathsf{E}_{\tau^*}(m_\beta), \qquad r^* = \mathsf{H}(m_\beta, y_A^*, y_B^*, K^*), \qquad s^* = x^*/(r^* + x_A^*),$$

where

$$K^* = {y_B^*}^{x^*}, \qquad \tau^* = \mathsf{G}(K^*)$$

for x^* picked uniformly at random from $\mathbb{Z}_q^*$. On input C^*, $\mathsf{A}^{\mathsf{CCA}}$ outputs $\beta' \in \{0,1\}$ at the guess stage. We denote by S_0 the event $\beta' = \beta$ and use a similar notation S_i for all games G_i.

Since this game is the same as the real attack game, we have

$$\Pr[S_0] = \tfrac{1}{2} + \tfrac{1}{2}\mathbf{Succ}^{\text{FSO/FUO-IND-CCA2}}_{\mathsf{A}^{\mathsf{CCA}}, \mathcal{ZSCR}}(k).$$

• *Game* G_1. In this game we modify the target signcrytext C^* presented in the previous game. The modification obeys the following rules:

R1-1 First we choose $\tau^+ \in \{0,1\}^{l_G(k)}$, $r^+ \in \mathbb{Z}_q$, and $s^+ \in \mathbb{Z}_q^*$ uniformly at random. We then compute $c^+ = \mathsf{E}_{\tau^+}(m_\beta)$ for random $\beta \in \{0,1\}$ and replace c^*, r^*, s^*, and $\mathsf{G}(K^*)$ in the target signcryptext C^* by c^+, r^+, s^+, and τ^+, respectively. A new target signcryptext is now (c^+, r^+, s^+) and is denoted by C_+^*.

R1-2 Whenever the random oracle G is queried at $K^* = (y_B^*)^{s^+(r^+ + x_A^*)}$ (as defined by r^+ and $s+$), we respond with τ^+.

R1-3 Whenever the random oracle H is queried at $(m_\beta, y_A^*, y_B^*, K^*)$, where $K^* = (y_B^*)^{s^+(r^+ + x_A^*)}$, we respond with r^+.

R1-4 We assume that the signcryption and unsigncryption oracles are perfect. That is, on receiving $\mathsf{A}^{\mathsf{CCA}}$'s signcryption query (y_R, m) or unsigncryption query

$(y_S, C) \neq (y_A^*, C^*)$, where y_S and y_R respectively denote the sender and receiver's public keys arbitrarily selected by $\mathsf{A}^{\mathsf{CCA}}$, and m and C denote a message and a signcryptext, respectively, we signcrypt (y_R, m) using the private key x_A^* or unsigncrypt (y_S, m) using the private key x_B^* in the same way as we do in the real attack game.

Since we have replaced one set of random variables by another set of random variables which is different, yet has the same distribution, the attacker $\mathsf{A}^{\mathsf{CCA}}$'s view has the same distribution in both Game G_0 and Game G_1 except for the event that $(m_\beta, y_A^*, y_B^*, K^*)$ is queried to H at the find stage because we only know m_β at the end of the find stage. However, the error probability is small and is at most $(q_{\mathsf{H}} + q_{\mathsf{SC}} + q_{\mathsf{USC}})/2^{l_q(k)}$ because K^* is independent of the attacker's view in the find stage.

Accordingly, we have

$$|\Pr[S_1] - \Pr[S_0]| \leq \frac{q_{\mathsf{H}} + q_{\mathsf{SC}} + q_{\mathsf{USC}}}{2^{l_q(k)}}.$$

- *Game G_2.* In this game we retain rules **R1-1** and **R1-4**, renaming them "**R2-1**" and "**R2-4**", respectively. However, we drop rules **R1-2** and **R1-3** meaning that τ^+ and s^+ are used only in producing the target signcryptext C_+^* while, in other cases, when the signcryption or unsigncryption oracle queries to the random oracles G and H, or $\mathsf{A}^{\mathsf{CCA}}$ directly queries to them, answers from G or H are taken. We refer to these rules regarding the random oracles G and H as "**R2-2**" and "**R2-3**", respectively.

Since we have dropped rule **R1-2**, τ^+ is not used anywhere in Game G_2 except in computing c^*. Hence, if $\beta' = \beta$ then $\mathsf{A}^{\mathsf{CCA}}$ has broken the PI-SKE security of the bijective one-time symmetric encryption scheme. Hence, we have

$$\Pr[S_2] = \tfrac{1}{2} + \tfrac{1}{2}\mathbf{Succ}^{\text{PI-SKE}}_{\mathsf{A}^{\mathsf{PI}},\mathcal{SKE}}(k).$$

Now let AskKey_2 denote an event that, in Game G_2, G is queried at K^* by $\mathsf{A}^{\mathsf{CCA}}$ (rather than by the signcryption or unsigncryption oracles) or H is queried at (m, y', y'', K^*) for some (m, y', y'') by $\mathsf{A}^{\mathsf{CCA}}$ (again, rather than by the signcryption or unsigncryption oracles). We use an identical notation AskKey_i for all the remaining games.

Notice that Game G_1 and Game G_2 may differ if G is queried at K^*, where $K^* = (y_B^*)^{s^+(r^+ + x_A^*)}$, or H is queried at $(m_\beta, y_A^*, y_B^*, K^*)$. Therefore, besides AskKey_2, we need to consider the following events defined in Game G_2 (we define them to be disjoint by terminating the game as soon as one of them occurs):

- SCBad_2: G is queried at K^* or H is queried at (m, y', y'', K^*) by the signcryption oracle.
- USCBad_2: For some unsigncryption query (y_S, c, r, s), the unsigncryption oracle queries G at K^* and the unsigncryption oracle accepts (y_S, c, r, s) (i.e. does not reject).

Let $B_2 = \mathsf{AskKey}_2 \vee \mathsf{SCBad}_2 \vee \mathsf{USCBad}_2$. We claim that if $\neg\mathsf{B}_2$ occurs, the view of $\mathsf{A}^{\mathsf{CCA}}$ is identical in Games G_1 and G_2, so $\Pr[S_1 \wedge \neg B_2] = \Pr[S_2 \wedge \neg B_2]$. To show this, note first that if $\neg B_2$ occurs then K^* does not appear in G and H queries of $\mathsf{A}^{\mathsf{CCA}}$ and the signcryption oracle, so these queries are answered identically in Games G_1 and G_2.

We now show by induction that if $\neg B_2$ occurs then unsigncryption queries of $\mathsf{A}^{\mathsf{CCA}}$ are also answered identically in Games G_1 and G_2.

In the following analysis we assume that in both games the random oracle H is implemented in the following standard way: at the start of the game $q_{\mathsf{H}}+q_{\mathsf{SC}}+q_{\mathsf{USC}}$ uniformly random values $h_{\mathsf{H}}[1], \ldots, h_{\mathsf{H}}[q_{\mathsf{H}}]$,$h_{\mathsf{SC}}[1], \ldots, h_{\mathsf{SC}}[q_{\mathsf{SC}}]$, $h_{\mathsf{USC}}[1], \ldots, h_{\mathsf{USC}}[q_{\mathsf{USC}}]$ in $\{0, 1\}^{l_q(k)}$ are chosen. These values are identical in Games G_1 and G_2. The value $h_{\mathsf{H}}[i]$ is used to answer $\mathsf{A}^{\mathsf{CCA}}$'s ith H-query if it is "new" (otherwise the value is answered consistently with previous queries), and, similarly, $h_{\mathsf{SC}}[i]$ and $h_{\mathsf{USC}}[i]$ are used to answer "new" queries of the signcryption and unsigncryption oracles to H during the processing of $\mathsf{A}^{\mathsf{CCA}}$'s ith signcryption and unsigncryption queries, respectively. The only exception is that in Game G_1, the $(m_\beta, y_A^*, y_B^*, K^*)$ queries during the guess stage are answered with r^+.

Consider an outcome of event $\neg \mathsf{B}_2$ in which the view of $\mathsf{A}^{\mathsf{CCA}}$ is identical in Games G_1 and G_2 up to the ith unsigncryption oracle query (y_S, c, r, s) of $\mathsf{A}^{\mathsf{CCA}}$. We show that this query is answered identically by the unsigncryption oracle in both Game G_1 and Game G_2. Let $K = (g^r y_S)^{s x_B^*}$ denote the key queried to G (in both Game G_1 and Game G_2) by the unsigncryption oracle. If $K \neq K^*$ then the unsigncryption oracle proceeds identically in Games G_1 and G_2, so we assume that $K = K^*$. Also, we assume that $r \in \mathbb{Z}_q$, $s \in \mathbb{Z}_q^*$, $c \in SP_c$, and $y_S \in \langle g \rangle$ since otherwise the query is rejected in both Game G_1 and Game G_2.

- In Game G_2 the unsigncryption oracle obtains $\tau = \mathsf{G}(K^*)$ and queries H at $(\mathsf{D}_\tau(c), y_S, y_B^*, K^*)$, obtaining response $h_{\mathsf{USC}}[j]$ where $j \leq i$ is the index of the earliest unsigncryption query where the unsigncryption oracle queried H at $(\mathsf{D}_\tau(c), y_S, y_B^*, K^*)$. Thanks to the one-to-one property of the decryption function D, j is the index of the first unsigncryption query $(y_{S,j}, c_j, r_j, s_j)$ satisfying $(y_{S,j}, c_j) = (y_S, c)$ and $(g^{r_j} y_S)^{s_j x_B^*} = K^*$. By definition of $\neg \mathsf{B}_2$ we know that the unsigncryption oracle rejects the query (y_S, c, r, s), i.e. we have $h_{\mathsf{USC}}[j] \neq r$.
- In Game G_1 the unsigncryption oracle queries K^* to G and obtains response τ^+. It then queries H at $(\mathsf{D}_{\tau^+}(c), y_S, y_B^*, K^*)$. We consider two possible cases:
  * *Case* 1: $(c, y_S) = (c^+, y_A^*)$ in the guess stage. Because $\mathsf{D}_{\tau^+}(c^+) = m_\beta$, in this case the unsigncryption oracle queries H at $(m_\beta, y_A^*, y_B^*, K^*)$ and obtains response r^+. We claim that $r \neq r^+$ so the unsigncryption oracle rejects the query (y_S, c, r, s): if the query (y_S, c, r, s) was accepted then it would have to be equal to the challenge (y_A^*, c^+, r^+, s^+), which is disallowed from being queried in the guess stage. To show this, suppose towards a contradiction that $r = r^+$. Then using $K = K^*$ we have $(y_B^*)^{s(r^+ + x_A^*)} = (y_B^*)^{s^+(r^+ + x_A^*)}$. Since y_B^* has order q we have $s(r^+ + x_A^*) = s^+(r^+ + x_A^*)$ and using $r^+ + x_A^* \neq 0$ we get $s = s^+$, a contradiction. Hence, the unsigncryption oracle rejects in this case.
  * *Case* 2: $(c, y_S) = (c^+, y_A^*)$ in the find stage OR $(c, y_S) \neq (c^+, y_A^*)$. In this case the unsigncryption oracle queries H at $(\mathsf{D}_{\tau^+}(c), y_S, y_B^*, K^*)$. Note that thanks to the one-to-one property of the decryption function D, we have from $(c, y_S) \neq (c^+, y_A^*)$ that $(\mathsf{D}_{\tau^+}(c), y_S) \neq (m_\beta, y_A^*)$ in the guess stage. Hence, the unsigncryption oracle obtains response $h_{\mathsf{USC}}[\ell]$ from the H oracle, where $\ell \leq i$ is the index of the earliest unsigncryption query where the unsigncryp-

tion oracle queried H at $(\mathsf{D}_{\tau^+}(c), y_S, y_B^*, K^*)$. Thanks to the one-to-one property of the decryption function D, ℓ is the index of the first unsigncryption query $(y_{S,\ell}, c_\ell, r_\ell, s_\ell)$ satisfying $(y_{S,\ell}, c_\ell) = (y_S, c)$ and $(g^{r_\ell} y_S)^{s_\ell x_B^*} = K^*$. However, by the induction hypothesis the ℓth unsigncryption query is identical in Games G_1 and G_2 for all $\ell \le i$. Hence, we must have $\ell = j$, where $j \le i$ is the index of the first unsigncryption query $(y_{S,j}, c_j, r_j, s_j)$ satisfying $(y_{S,j}, c_j) = (y_S, c)$ and $(g^{r_j} y_S)^{s_j x_B^*} = K^*$ in Game G_2 (see analysis of Game G_2 above). So in Game G_1, the unsigncryption oracle obtains the same response $h_{\mathsf{USC}}[j] \neq r$ to its H query as in Game G_2 and rejects.

Therefore, the unsigncryption oracle responds identically in Games G_1 and G_2 when $\neg B_2$ occurs, as claimed.

However, event SCBad_2 has a negligible probability. Namely due to the uniform distribution of K computed by the signcryption in $\langle g\rangle\backslash\{1\}$, the probability that K hits K^* is less than $1/2^{l_q(k)}$ per each signcryption query. Consequently we have $\Pr[\mathsf{SCBad}_2] \le q_{\mathsf{SC}}/2^{l_q(k)}$.

Also, event USCBad_2 has a negligible probability. Namely, let $\mathsf{USCBad}_2[i]$ denote the event in G_2 that i is the index of the *earliest* unsigncryption query (y_S, c, r, s) such that the unsigncryption oracle queries G at K^* and the unsigncryption oracle accepts (y_S, c, r, s). Note that for any outcome in $\mathsf{USCBad}_2[i]$, the unsigncryption oracle queries H at $(\mathsf{D}_\tau(c), y_S, y_B^*, K^*)$ and receives response $h_{\mathsf{USC}}[j]$, where j is the index of the earliest unsigncryption query where the unsigncryption oracle queried H at $(\mathsf{D}_\tau(c), y_S, y_B^*, K^*)$ (we know that $(\mathsf{D}_\tau(c), y_S, y_B^*, K^*)$ was not queried to H by $\mathsf{A}^{\mathsf{CCA}}$ or the signcryption oracle since otherwise AskKey_2 or SCBad occur). Fixing the values of all random variables in Game G_2 except $h_{\mathsf{USC}}[j]$ and varying the value of $h_{\mathsf{USC}}[j]$ we see that for all values of $h_{\mathsf{USC}}[j]$ different from r, either the view of $\mathsf{A}^{\mathsf{CCA}}$ remains unchanged up to the ith unsigncryption query so that the unsigncryption oracle rejects, or the view of $\mathsf{A}^{\mathsf{CCA}}$ is unchanged until $\mathsf{A}^{\mathsf{CCA}}$'s ℓth unsigncryption query for some $\ell < i$ is accepted, so event $\mathsf{USCBad}_2[\ell]$ occurs for some $\ell < i$. So, thanks to the uniformly random choice of $h_{\mathsf{USC}}[j]$ in $\{0,1\}^{l_q}$, it follows that $\Pr[\mathsf{USCBad}_2[i]] \le 1/2^{l_q(k)}$ for $i = 1, \ldots, q_{\mathsf{USC}}$ and hence at most $q_{\mathsf{USC}}/2^{l_q(k)}$ over all unsigncryption queries.

Thus, finally we get

$$|\Pr[S_2] - \Pr[S_1]| \le \Pr[\mathsf{AskKey}_2] + \frac{q_{\mathsf{SC}} + q_{\mathsf{USC}}}{2^{l_q(k)}}.$$

- *Game G_3.* In this game we modify rule **R2-4** and obtain a new rule **R3-4**. However, we retain rules **R2-1**, **R2-2**, and **R2-3** in Game G_2, renaming them "**R3-1**", "**R3-2**", and "**R3-3**", respectively.

R3-4 In this rule we replace the random oracles G and H by the random oracle simulators GSim and HSim. Note that two types of "query-answer" lists $\mathsf{GList1}$ and $\mathsf{GList2}$ are maintained for the simulation of the random oracle G. $\mathsf{GList1}$ consists of simple "input–output" entries for G of the form (K, τ). $\mathsf{GList2}$ (whose new entries are added by either the signcryption oracle simulator or the unsigncryption oracle simulator) consists of the special input–output entries for G which are of the form $y_R\|\omega\|(?, \tau)$. This implicitly represents the input–output relation $\tau = \mathsf{G}(\omega^{\log_g y_R})$, although the input $\omega^{\log_g y_R}$ is not explicitly stored and hence is denoted by "?". Similarly to GSim, the simulator HSim also maintains two input–output lists HList1 and HList2. HList1 consists of simple input–output entries for H, which are of the form (μ, r). HList2 (whose new entries are added by either the signcryption or unsigncryption oracle simulators in later games) consists of the special input–output entries for H which are of the form $y_R \| \omega \| ((m, y_S, y_R, ?), r)$ and implicitly represents the input–output relation $\mathsf{H}(m, y_S, y_R, K) = r$, where $K = \omega^{\log_g y_R}$ is not explicitly stored and hence is denoted by "?". Complete specifications for GSim and HSim are as follows:

Random Oracle Simulators GSim and HSim

```
GSim(K)
  If O(g, ω, y_R, K) = 1
  for some y_R‖ω‖(?, τ) ∈ GList2
    Return τ
  Else if (K, τ) exists in GList1
    Return τ
  Else τ ←R {0, 1}^{l_G(k)}
    Return τ
  Add (K, τ) to GList1
```

```
HSim(m, y_S, y_R, K)
  If O(g, ω, y_R, K) = 1 and
  y_R‖ω‖(m, y_S, y_R, ?), r) ∈ HList2
    Return r
  Else if ((m, y_S, y_R, K), r)
  exists in HList1 Return r
  Else r ←R Z_q Return r
  Add ((m, y_S, y_R, K), r) to HList1
```

We note that GList2 and HList2 are actually empty throughout this game because we still have the original signcryption and unsigncryption oracles, so no entries are ever added to them—GList1 and HList1 are used in this game.

Finally, notice that the above simulation for the random oracles G and H are perfect. Hence, we have

$$\Pr[\mathsf{AskKey}_3] = \Pr[\mathsf{AskKey}_2].$$

• *Game G_4*. We retain all the rules **R3-1**, **R3-2** and **R3-3**, renaming them "**R4-1**", "**R4-2**", and "**R4-3**", respectively. However, we further modify **R3-4** and obtain a new rule:

R4-4 In this rule we replace the signcryption oracle by the signcryption oracle simulator SCSim. On the other hand, we assume that the unsigncryption oracle is perfect.

Signcryption Oracle Simulator SCSim

```
SC Sim(y*_A, (y_R, m))
  If y_R ∉ ⟨g⟩\{1} Return Rej
  τ ←R {0, 1}^{l_G(k)}; r ←R Z_q; c ← E_τ(m); s ←R Z*_q
  If g^r y*_A = 1 Return Rej
  ω ← (y*_A g^r)^s
  Add y_R‖ω‖(?, τ) to GList2
  Add y_R‖ω‖((m, y*_A, y_R, ?), r) to HList2
  C ← (c, r, s)
  Return C
```

Let $K = (y_A^* g^r)^{sx_B^*}$ denote the query of signcryption oracle to G in Game G_4. Note that if neither (K, τ) nor $((m, y_A^*, y_R, K), r)$ exists in GList1 and HList1, respectively, the simulated signcryptext in Game G_4 is distributed the same as the signcryptext in Game G_3 and a simulation error occurs otherwise.

However, in Game G_3, thanks to the uniform distribution of K in $\langle g\rangle\backslash\{1\}$, and since GList1 and HList1 contain all the queries to G and H both by the attacker, and the signcryption and unsigncryption oracles, we have $\Pr[(K, \tau) \in \mathsf{GList1} \vee ((m, y_A^*, y_R, K), r) \in \mathsf{HList1}] \leq (q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{SC}} + q_{\mathsf{USC}})/2^{l_q(k)}$.

Since there are up to q_{SC} signcryption queries, the total probability of outcomes leading to signcryption oracle simulation error is upper-bounded by

$$q_{\mathsf{SC}}\left(\frac{q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{SC}} + q_{\mathsf{USC}}}{2^{l_q(k)}}\right).$$

Summing up all decryption queries, we have

$$|\Pr[\mathsf{AskKey}_4] - \Pr[\mathsf{AskKey}_3]| \leq q_{\mathsf{SC}}\left(\frac{q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{SC}} + q_{\mathsf{USC}}}{2^{l_q(k)}}\right).$$

- *Game* G_5. We retain rules **R4-1**, **R4-2**, and **R4-3**, renaming them "**R5-1**", "**R5-2**", and "**R5-3**", respectively. However, we modify **R4-4** and obtain the following new rule:

R5-4 We replace the unsigncryption oracle by an unsigncryption oracle simulator USCSim which can unsigncrypt a submitted unsigncryption query (y_S, C) where $C = (c, r, s)$, without knowing the private key. Notice that the unsigncryption oracle simulator makes use of $\mathsf{A}^{\mathsf{GDH}}$'s DDH oracle $\mathsf{O}^{\mathsf{DDH}}(\cdot, \cdot, \cdot, \cdot)$ to check whether a given tuple is a Diffie–Hellman one.

Unsigncryption Oracle Simulator USCSim

USC Sim(y_B^*, y_S, C)
 If $y_S \notin \langle g\rangle\backslash\{1\}$ Return *Rej*
 Parse C as (c, r, s)
 If $r \notin \mathbb{Z}_q$ or $s \notin \mathbb{Z}_q^*$ or $c \notin SP_c$ Return *Rej*
 $\omega \leftarrow (y_S g^r)^s$
 If there exists $(K, \tau) \in \mathsf{GList1}$ such that $\mathsf{O}^{\mathsf{DDH}}(g, \omega, y_B^*, K) = 1$ or
 there exists $y_R\|\omega'\|(?, \tau) \in \mathsf{GList2}$ such that $\mathsf{O}^{\mathsf{DDH}}(\omega, \omega', y_R, y_B^*) = 1$
  $m \leftarrow \mathsf{D}_\tau(c)$
 Else $\tau \stackrel{R}{\leftarrow} \{0, 1\}^{l_G(k)}$; Add $y_B^*\|\omega\|(?, \tau)$ to Glist2
  $m \leftarrow \mathsf{D}_\tau(c)$
 If there exists $((m, y_S, y_B^*, K), h) \in \mathsf{HList1}$ such that $\mathsf{O}^{\mathsf{DDH}}(g, \omega, y_B^*, K) = 1$ or
 there exists $y_R\|\omega'\|((m, y_S, y_R, ?), h) \in \mathsf{HList2}$ such that $\mathsf{O}^{\mathsf{DDH}}(\omega, \omega', y_R, y_B^*) = 1$
  If $h = r$ Return m Else Return *Rej*
 Else $h \stackrel{R}{\leftarrow} \mathbb{Z}_q$
  Add $(y_B^*\|w\|(m, y_S, y_B^*, ?), h)$ to HList2
  If $h = r$ Return m Else Return *Rej*

Observe that the full contents of GList1 $\vee$ GList2 and HList1 $\vee$ HList2 are updated identically in Games G_4 and G_5, where full means that before comparing the lists in the two games, we convert the implicit $\mathsf{GList2}$ and $\mathsf{HList2}$ entries into the explicit entries that they represent (with the appropriate K values). This is because the only difference is that in Game G_5 the unsigncryption oracle adds implicit entries to $\mathsf{GList2}$ and $\mathsf{HList2}$, while in Game G_4 they are added explicitly to $\mathsf{GList1}$ and $\mathsf{HList1}$. Thanks to the DDH oracle used by GSim, HSim, and USCSim, the oracles respond in a way which depends only on the full contents of GList and HList, and hence the view of $\mathsf{A}^{\mathsf{CCA}}$ is identical in Games G_4 and G_5 so

$$\Pr[\mathsf{AskKey}_5] = \Pr[\mathsf{AskKey}_4].$$

Since Game G_2, AskKey_i for $i = 2, 3, 4, 5$ has implied that when AskKey_i occurs the GDH problem can be solved. More precisely, the event AskKey_i for $i \geq 2$ means that $K^* = (y_B^*)^{s^+(r^+ + x_A^*)} = (y_B^*)^{s^+ x_A^*}(y_B^*)^{s^+ r^+}$ has been queried to G or H and hence one can compute $g^{ab} = (y_B^*)^{x_A^*} = (K^*/(y_B^*)^{s^+ r^+})^{1/s^+}$. Furthermore, at this stage, one can check which one of the queries to the random oracles G and H is a Diffie–Hellman key of g^{ab} using $\mathsf{A}^{\mathsf{GDH}}$'s DDH oracle $\mathsf{O}^{\mathsf{DDH}}(\cdot, \cdot, \cdot, \cdot)$. Consequently we have

$$\Pr[\mathsf{AskKey}_5] \leq \mathbf{Succ}^{\mathrm{GDH}}_{\mathbb{Z}_p^*, \mathsf{A}^{\mathsf{GDH}}}(k).$$

Putting all the bounds we have obtained in each game together, we obtain

$$\begin{aligned}
\tfrac{1}{2}\mathbf{Succ}^{\mathrm{FSO/FUO\text{-}IND\text{-}CCA2}}_{\mathsf{A}^{\mathsf{CCA}}, \mathcal{ZSCR}}(k) &= |\Pr[S_0] - \tfrac{1}{2}| \\
&\leq \frac{q_{\mathsf{H}} + q_{\mathsf{SC}} + q_{\mathsf{USC}}}{2^{l_q(k)}} + \tfrac{1}{2}\mathbf{Succ}^{\mathrm{PI\text{-}SKE}}_{\mathsf{A}^{\mathsf{PI}}, \mathcal{SKE}}(l) + \frac{q_{\mathsf{SC}} + q_{\mathsf{USC}}}{2^{l_q(k)}} \\
&\quad + q_{\mathsf{SC}}\left(\frac{q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{SC}} + q_{\mathsf{USC}}}{2^{l_q(k)}}\right) + \Pr[\mathsf{AskKey}_5] \\
&\leq \tfrac{1}{2}\mathbf{Succ}^{\mathrm{PI\text{-}SKE}}_{\mathsf{A}^{\mathsf{PI}}, \mathcal{SKE}}(l) + q_{\mathsf{SC}}\left(\frac{q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{SC}} + q_{\mathsf{USC}} + 2}{2^{l_q(k)}}\right) \\
&\quad + \frac{q_{\mathsf{H}} + 2q_{\mathsf{USC}}}{2^{l_q(k)}} + \mathbf{Succ}^{\mathrm{GDH}}_{\mathbb{Z}_p^*, \mathsf{A}^{\mathsf{GDH}}}(k).
\end{aligned}$$

The advantage bound claim of the theorem follows upon taking maximums over all adversaries with the appropriate resource parameters. The running time counts can be readily checked. □

5.4. *Unforgeability of Zheng's Signcryption Scheme*

In this section we prove that the GDL assumption is sufficient for the signcryption scheme $\mathcal{ZSCR}$ to achieve the strong unforgeability in the sense of FSO-UF-CMA in the random oracle model.

Theorem 2. *If the GDL assumption holds then Zheng's original signcryption scheme $\mathcal{ZSCR}$ is unforgeable in the FSO-UF-CMA sense. Concretely, the following bound holds:*

$$\begin{aligned}
&\mathbf{InSec}^{\mathrm{FSO\text{-}UF\text{-}CMA}}_{\mathcal{ZSCR}}(t, q_{\mathsf{SC}}, q_{\mathsf{G}}, q_{\mathsf{H}}) \\
&\qquad \leq 2\sqrt{q_{\mathsf{H}} \cdot \mathbf{InSec}_{\mathsf{GDL}}(t', q_{\mathsf{O}^{\mathsf{rDDH}}})} + \frac{q_{\mathsf{SC}}(q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{SC}}) + q_{\mathsf{H}} + 1}{2^{l_q(k)-1}},
\end{aligned}$$

where $t' = 2t + O(q_{\mathsf{G}}^2 + 1) + O(q_{\mathsf{H}}^2 + 1) + O((t'' + k^3)q_{\mathsf{SC}})$ *and* $q_{\mathsf{O}^{\mathsf{rDDH}}} = 2q_{\mathsf{SC}}(q_{\mathsf{G}} + q_{\mathsf{H}}) + 2q_{\mathsf{H}}$. *Here* t'' *denotes the running time of the one-time symmetric key encryption scheme.*

Proof. We show how to use any efficient FSO-UF-CMA attacker A^{UF} to construct an efficient attacker $\mathsf{A}^{\mathsf{GDL}}$ against the GDL problem, thus contradicting the GDL assumption. We do this in two stages. In stage 1 we use A^{UF} to construct an efficient algorithm $\mathsf{A}^{\mathsf{GDL}'}$ for a variant of the GDL problem that we define below and call GDL$'$. Then in Stage 2 we show how to transform any efficient algorithm for GDL$'$ into an efficient algorithm for GDL.

Stage 1. In this stage our aim is to keep modifying the real attack game **SCRUFGame** presented in Definition 3 until we get to the stage where we obtain a game which constitutes an algorithm for the GDL$'$ problem, which is defined as follows:

Problem GDL$'$. Given (g, p, q, g^a), where $(g, p, q) = \mathsf{GC}(k)$ and $a \stackrel{\mathrm{R}}{\leftarrow} \mathbb{Z}_q^*$, up to q_{R} queries $(y[i], K[i]) \in \langle g\rangle\backslash\{1\} \times \langle g\rangle$ to a random beacon R which returns uniformly random independent integers $r[i] \in \mathbb{Z}_q$ (for $i = 1, \ldots, q_{\mathsf{R}}$), and up to $q_{\mathsf{O}^{\mathsf{rDDH}}}$ queries to a restricted DDH oracle $\mathsf{O}^{\mathsf{rDDH}}(g, g^a, \cdot, \cdot)$, compute $s^* \in \mathbb{Z}_q^*$ and $i^* \in \{1, \ldots, q_{\mathsf{R}}\}$ such that $K[i^*] = y[i^*]^{s^*(r[i^*]+a)}$.

We stress that the random beacon R above differs from a random oracle because R *always* returns independent random integers, even for repeated queries.

We start with the following game.

• *Game* G_0. This game is the same as the real attack game **SCRUFGame** in Definition 3.

First we run the common parameter/oracle generation algorithm GC of $\mathcal{ZSCR}$ on input a security parameter k and obtain a common parameter $cp = (p, q, g, \mathsf{G}, \mathsf{H}, \mathcal{SKE})$, where p and q are primes such that $q \mid (p-1)$, g is an element in $\mathbb{Z}_p^*$ such that $\mathrm{Ord}_p(g) = q$, G: $\{0,1\}^* \to \{0,1\}^{l_G(k)}$ and H: $\{0,1\}^* \to \mathbb{Z}_q$ are hash functions modelled as the random oracles [7], and $\mathcal{SKE}$ is the one-time symmetric key encryption scheme that consists of the encryption function E and the decryption function D. We then run the sender key generation algorithm GK_A on input (k, cp) to obtain Alice (sender)'s fixed private/public key pair. Here, Alice's private key consists of (x_A^*, y_A^*) where $y_A^* = g^{x_A^*}$, and her public key is y_A^* itself.

We then run A^{UF} on input the public key y_A^*. We answer A^{UF}'s queries using the real SC, G and H oracles. Eventually, A^{UF} returns a forgery (C^*, y_R^*), where $C^* = (c^*, r^*, s^*)$ is the forged signcryptext and y_R^* is the forgery recipient's public key. We then apply the unsigncryption algorithm to compute $K^* = (y_R^*)^{(r^*+x_A^*)s^*}$, $\tau^* = \mathsf{G}(K^*)$, and $m^* = \mathsf{D}_\tau(c^*)$ and perform the following verification checks:

1. Check whether y_R^* is in $\langle g\rangle\backslash\{1\}$.
2. $(c^*, r^*, s^*) \in SP_c \times \mathbb{Z}_q \times \mathbb{Z}_q^*$.
3. $\mathsf{H}(m^*, y_A^*, y_R^*, K^*) = r^*$.
4. (y_R^*, m^*) was not queried by A^{UF} to SC oracle.

We denote by S_0 the event that A^{UF} succeeds in the sense of FSO-UF-CMA so all verification checks above are passed, and use a similar notation S_i for all games G_i.

Since this game is the same as the real attack game, we have

$$\Pr[S_0] = \mathbf{Succ}^{\text{FSO-UF-CMA}}_{\mathsf{A}^{\mathsf{UF}},\mathcal{ZSCR}}(k).$$

• *Game* G_1: In this game we modify one of the verification checks for A^{UF}'s output forgery. The modification obeys the following rules (note that rules **R1-2**, **R1-3**, and **R1-4** are also satisfied in Game G_0):

R1-1 Instead of the verification check 3, we check that $\mathsf{H}'(m^*, y_A^*, y_R^*, K^*) = r^*$, where H' is a new random oracle independent of H.
R1-2 We use the random oracle G to answer all queries to G in the game.
R1-3 We use the random oracle H to answer all queries to H in the game.
R1-4 We use the signcryption algorithm SC to answer all signcryption queries in the game.

Since the new random oracle H' is never queried during the game until the verification query (m^*, y_A^*, y_R^*, K^*) is made to it at the end of the game, we know that $\mathsf{H}'(m^*, y_A^*, y_R^*, K^*)$ is uniformly random in $\mathbb{Z}_q$ and independent of r^*. Hence we have

$$\Pr[S_1] \leq \frac{1}{2^{l_q(k)}}.$$

Note that in Game G_0 we also have that $\mathsf{H}(m^*, y_A^*, y_R^*, K^*)$ is uniformly random in $\mathbb{Z}_q$ and independent of r^*, unless H was queried at (m^*, y_A^*, y_R^*, K^*) by either A^{UF} or SC. However, if event S_0 occurred then H could not have been queried at (m^*, y_A^*, y_R^*, K^*) by SC because this would imply that A^{UF} queried (m^*, y_R^*) to SC, contradicting verification check 4. So let AskH_0 denote the event in Game G_0 that A^{UF} queried (m^*, y_A^*, y_R^*, K^*) to H and (y_R^*, m^*) was not queried by A^{UF} to the SC oracle. We use an identical notation $\mathsf{AskH_i}$ for all the remaining games. We therefore have

$$|\Pr[\mathsf{S}_1] - \Pr[\mathsf{S}_0]| \leq \Pr[\mathsf{AskH}_0] = \Pr[\mathsf{AskH}_1].$$

• *Game* G_2. In this game we modify rules **R1-2** and **R1-3** to obtain new rules **R2-2** and **R2-3**. However, we retain rules **R1-1** and **R1-4**, renaming them **R2-1** and **R2-4**, respectively.

R2-2 We use a random oracle simulator GSim to answer all queries to G in the game.
R2-3 We use a random oracle simulator HSim to answer all queries to H in the game.

Note that two types of "query–answer" lists $\mathsf{GList1}$ and $\mathsf{GList2}$ are used by Gsim for the simulation of the random oracle G. These lists are initialized as empty and updated as the game runs. Note that list $\mathsf{GList2}$ is never updated in this game but will be updated by the signcryption oracle simulator in Game G_3. The list $\mathsf{GList1}$ consists of simple "input–output" entries for G of the form (K, τ). The list $\mathsf{GList2}$ consists of the special input–output entries for G which are of the form $y_R\|r\|s\|(?, \tau)$. This implicitly represents the input–output relation $\tau = \mathsf{G}(y_R^{(r+x_A^*)s})$, although the input $y_R^{(r+x_A^*)s}$ is not explicitly stored and hence is denoted by "?". Similarly to GSim, the simulator HSim also uses two input–output lists, HList1 and HList2 (once again, list HList2 is never updated in this game but will be in Game G_3). HList1 consists of simple input–output entries for H, which are of the form (μ, r). HList2 consists of the special input–output entries for H which are of the form $y_R\|r\|s\|((m, y_S, y_R, ?), r)$ and implicitly represents the input–output relation $\mathsf{H}(m, y_S, y_R, K) = r$, where $K = y_R^{(r+x_A^*)s}$ is not explicitly stored and hence is denoted by "?". Note that HSim also uses a random beacon R to compute independent and uniformly random values $r = \mathsf{R}(y_R, K)$ in $\mathbb{Z}_q$ (the beacon R differs from a random oracle because it *always* returns independent random strings, even for repeated queries). Complete specifications for GSim and HSim are as follows:

Random Oracle Simulators GSim and HSim

GSim(K)
  If $\mathsf{O}^{\mathsf{rDDH}}(g, y_A^*, y_R^s, K/y_R^{rs}) = 1$
  for some $y_R\|r\|s\|(?, \tau) \in$ GList2
    Return τ
  Else if (K, τ) exists in GList1
    Return τ
  Else $\tau \xleftarrow{R} \{0,1\}^{l_G(k)}$
    Return τ
  Add (K, τ) to GList1

Hsim(m, y_S, y_R, K)
  If $\mathsf{O}^{\mathsf{rDDH}}(g, y_A^*, y_R^s, K/y_R^{rs}) = 1$
  for some $y_R\|r\|s\|((m, y_A^*, y_R, ?), r) \in$ HList2
    Return r
  Else if $((m, y_S, y_R, K), r)$
  exists in HList1 Return r
  Else $r = \mathsf{R}(y_R, K)$ Return r
  Add $((m, y_S, y_R, K), r)$ to HList1

Note that the above simulation for the random oracles G and H is perfect. Hence, we have

$$\Pr[\mathsf{AskH}_2] = \Pr[\mathsf{AskH}_1].$$

• *Game G_3*. We retain rules **R2-1**, **R2-2** and **R2-3**, renaming them "**R3-1**", "**R3-2**", and "**R3-3**", respectively. However, we modify rule **R2-4** and obtain a new rule:

R3-4 We use a signcryption oracle simulator SCSim to answer all signcryption queries in the game.

The specification for signcryption oracle SCSim follows:

Signcryption Oracle Simulator SCSim

SC Sim($y_A^*, (y_R, m)$)
  If $y_R \notin \langle g \rangle \backslash \{1\}$ Return *Rej*
  $\tau \xleftarrow{R} \{0,1\}^{l_G(k)}$; $r \xleftarrow{R} \mathbb{Z}_q$; $c \leftarrow \mathsf{E}_\tau(m)$; $s \xleftarrow{R} \mathbb{Z}_q^*$
  If $g^r y_A^* = 1$ Return *Rej*
  Add $y_R\|r\|s\|(?, \tau)$ to GList2
  Add $y_R\|r\|s\|((m, y_A^*, y_R, ?), r)$ to HList2
  $C \leftarrow (c, r, s)$
  Return C

In Game G_2 define the event B_2 that for some signcryption query we have $(K, \tau) \in \mathsf{GList1}$ or $((m, y_A^*, y_R, K), r) \in \mathsf{HList1}$. Note that, in Game G_2, if B_2 does not occur then (τ, r, s) are independent and uniformly distributed in $\{0, 1\}^{l_G} \times \mathbb{Z}_q \times \mathbb{Z}_q^*$, exactly as in Game G_3. Hence, $\Pr[\mathsf{AskH}_3]$ and $\Pr[\mathsf{AskH}_2]$ can differ by at most $\Pr[\mathsf{B}_2]$. However, for each signcryption query in Game G_2, thanks to the uniform distribution of K in $\langle g\rangle\backslash\{1\}$, we have $\Pr[(K, \tau) \in \mathsf{GList1} \vee ((m, y_A^*, y_R, K), r) \in \mathsf{HList1}] \le (q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{SC}})/2^{l_q(k)}$. Finally, since there are up to q_{SC} signcryption queries we add up these bounds to obtain

$$\Pr[\mathsf{B}_2] \le q_{\mathsf{SC}} \left(\frac{q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{SC}}}{2^{l_q(k)}} \right)$$

and, therefore,

$$|\Pr[\mathsf{AskH}_3] - \Pr[\mathsf{AskH}_2]| \le q_{\mathsf{SC}} \left(\frac{q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{SC}}}{2^{l_q(k)}} \right).$$

Now we observe that Game G_3 constitutes an algorithm $\mathsf{A}^{\mathsf{GDL}'}$ for breaking the GDL′ problem with success probability $\Pr[\mathsf{AskH}_3]$. Namely, $\mathsf{A}^{\mathsf{GDL}'}$ is given input $(g, p, q, y_A^* = g^{x_A^*})$ and runs A^{UF} on this input using the rules of Game G_3. $\mathsf{A}^{\mathsf{GDL}'}$ uses its random beacon R (q_H queries) and restricted DDH oracle $\mathsf{O}^{\mathsf{rDDH}}$ in running GSim and HSim simulators. If AskH_3 occurs then we know that A^{UF} returns (y_R^*, c^*, r^*, s^*) such that HSim was queried at (m^*, y_A^*, y_R^*, K^*) (with $K^* = (y_R^*)^{(r^*+x_A^*)s^*}$) by A^{UF} and hence (since (y_R^*, m^*) was not queried to SC) R was queried with (y_R^*, K^*) by HSim, returning the answer r^*. At the end of the game, $\mathsf{A}^{\mathsf{GDL}'}$ checks which query to R was equal to (y_R^*, K^*) (using at most q_H^{UF} additional queries to the DDH oracle $\mathsf{O}^{\mathsf{rDDH}}$), and outputs s^* and the index i^* of the matching R query as the solution to the GDL′ problem instance.

This completes the "Stage 1" reduction. The algorithm $\mathsf{A}^{\mathsf{GDL}'}$ has run time $t^{\mathrm{GDL}'} = t + O(q_{\mathsf{G}}^2+1) + O(q_{\mathsf{H}}^2+1) + O((t^{\mathrm{SKE}}+k^3)q_{\mathsf{SC}})$, makes $q_{\mathsf{OrDDH}}^{\mathrm{GDL}'} = q_{\mathsf{SC}}(q_{\mathsf{G}}+q_{\mathsf{H}}) + q_H$ DDH queries and $q_{\mathsf{R}}^{\mathrm{GDL}'} = q_H$ R queries, and has success probability

$$\mathbf{Succ}^{\mathrm{GDL}'}_{\mathbb{Z}_p^*,\mathsf{A}^{\mathsf{GDL}'}}(k) \ge \Pr[\mathsf{AskH}_3] \ge \mathbf{Succ}^{\text{FSO-UF-CMA}}_{\mathsf{A}^{\mathsf{UF}},\mathcal{ZSCR}}(k) - \frac{q_{\mathsf{SC}}(q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{SC}}) + 1}{2^{l_q(k)}}.$$

Stage 2. We use the "forking technique" [13], [26], [23] to perform the "Stage 2" reduction between GDL′ and GDL. In the analysis of this stage we will use the following two lemmas.

Lemma 2 (Splitting Lemma [23]). *Let a and b denote independent random variables over finite sets A and B, respectively, with probability distribution functions $P_A(\cdot)$ and $P_B(\cdot)$, respectively. Let $S \subseteq A \times B$ be a set with* $\Pr[(a, b) \in S] \ge \varepsilon$. *For each $a \in A$, let $S(a) \subseteq B$ denote the set of $b \in B$ such that $(a, b) \in S$. Then there exists a "good" subset G of S such that*

$$\Pr_{(a,b)\in A\times B}[(a, b) \in G] \ge \varepsilon/2$$

and, for all $(a', b') \in G$,

$$\Pr_{b\in B}[b \in S(a')] \ge \varepsilon/2.$$

Proof. Let us *define* the good set G to be the set of all $(a', b') \in S$ such that $\Pr[b \in S(a')] \geq \varepsilon/2$. Then it is enough to show that $\Pr[(a, b) \in G] \geq \varepsilon/2$.

Suppose, towards a contradiction, that $\Pr[(a, b) \in G] < \varepsilon/2$. Then $\Pr[(a, b) \in S] = \Pr[(a, b) \in G] + \Pr[(a, b) \in (S \wedge \neg G)] < \varepsilon/2 + \Pr[(a, b) \in (S \wedge \neg G)]$. However, $(a, b) \in (S \wedge \neg G)$ means that $a \in W_A$, where $W_A \subseteq A$ is the set of $a' \in A$ such that $\Pr[b \in S(a')] < \varepsilon/2$. So $\Pr[(a, b) \in (S \wedge \neg G)] = \sum_{a' \in W_A} \sum_{b' \in S(a)} P_A(a') P_B(b') = \sum_{a' \in W_A} P_A(a') \cdot \Pr[b \in S(a')] < \varepsilon/2$ since $\Pr[b \in S(a')] < \varepsilon/2$ for all $a' \in W_A$. It follows that $\Pr[(a, b) \in S] < \varepsilon/2 + \varepsilon/2 = \varepsilon$, a contradiction. This shows that $\Pr[(a, b) \in G] \geq \varepsilon/2$, which completes the proof. $\square$

We will also use the following inequality.

Lemma 3. *Let $p = \sum_{j=1}^{q} p_j$ for some q real numbers $p_1, \ldots, p_q$ and let $\delta > 0$ be given. If $p \geq q \cdot \delta$ then the following inequality holds*:

$$\sum_{j=1}^{q} p_j \cdot (p_j - \delta) \geq (1/q) \cdot (p - q \cdot \delta)^2.$$

Proof. We have $\sum_{j=1}^{q} p_j \cdot (p_j - \delta) = \sum_{j=1}^{q} p_j^2 - p \cdot \delta$. Using the Cauchy–Schwartz inequality we have $\sum_{j=1}^{q} \geq (1/q) \cdot (\sum_{j=1}^{q} p_j)^2 = (1/q) \cdot p^2$, so $\sum_{j=1}^{q} p_j \cdot (p_j - \delta) \geq (1/q)(p^2 - q \cdot \delta \cdot p)$. However, from the assumption that $p \geq q \cdot \delta$, we have $(q \cdot \delta) p \geq (q \cdot \delta)^2$ and hence $p^2 - q \cdot \delta \cdot p \geq p^2 - 2(q \cdot \delta) p + (q \cdot \delta)^2 = (p - q \cdot \delta)^2$, which gives the claimed inequality. $\square$

Now we present the "Stage 2" reduction as the following lemma:

Lemma 4 (Stage 2). *Any algorithm $\mathsf{A}^{\mathsf{GDL}'}$ for problem GDL′ with run-time $t^{\mathrm{GDL}'}$, $q^{\mathrm{GDL}'}_{\mathsf{O}^{\mathsf{rDDH}}}$ DDH queries and $q^{\mathrm{GDL}'}_{\mathsf{R}}$ R queries, and success probability $\mathbf{Succ}^{\mathrm{GDL}'}_{\mathbb{Z}_p^*, \mathsf{A}^{\mathsf{GDL}'}}(k) \geq 2q^{\mathrm{GDL}'}_{\mathsf{R}}/2^{l_q}$ can be converted into an algorithm $\mathsf{A}^{\mathsf{GDL}}$ for problem GDL with run-time $t^{\mathrm{GDL}} = 2t^{\mathrm{GDL}'} + O(l_q^2)$ which makes $q^{\mathrm{GDL}}_{\mathsf{O}^{\mathsf{rDDH}}} = 2q^{\mathrm{GDL}'}_{\mathsf{O}^{\mathsf{rDDH}}}$ DDH queries, and has success probability*

$$\mathbf{Succ}^{\mathrm{GDL}}_{\mathbb{Z}_p^*, \mathsf{A}^{\mathsf{GDL}}}(k) \geq (1/q^{\mathrm{GDL}'}_{\mathsf{R}}) \cdot \left(\mathbf{Succ}^{\mathrm{GDL}'}_{\mathbb{Z}_p^*, \mathsf{A}^{\mathsf{GDL}'}}(k)/2 - q^{\mathrm{GDL}'}_{\mathsf{R}}/2^{l_q} \right)^2.$$

Proof. On input (g, p, q, g^a), our GDL algorithm $\mathsf{A}^{\mathsf{GDL}}$ works as follows.

Setup. $\mathsf{A}^{\mathsf{GDL}}$ first sets up two random vectors $\overrightarrow{r} = (r[1], \ldots, r[q^{\mathrm{GDL}'}_{\mathsf{R}}])$ and $\overrightarrow{\widehat{r}} = (\widehat{r}[1], \ldots, \widehat{r}[q^{\mathrm{GDL}'}_{\mathsf{R}}])$ with $r[i]$'s and $\widehat{r}[i]$'s chosen uniformly and independently at random from $\mathbb{Z}_q$ (these vectors will be used to answer $\mathsf{A}^{\mathsf{GDL}'}$'s R queries).

First Run. $\mathsf{A}^{\mathsf{GDL}}$ runs $\mathsf{A}^{\mathsf{GDL}'}$ on input $(g, p, q, g^a; \omega)$, where ω denotes the random coins input of $\mathsf{A}^{\mathsf{GDL}'}$, and answers $\mathsf{A}^{\mathsf{GDL}'}$'s oracle queries as follows:

(1) R**-Query simulator**. When $\mathsf{A}^{\mathsf{GDL}'}$ makes its ith R query $(y[i], K[i])$, $\mathsf{A}^{\mathsf{GDL}}$ responds with $r[i]$.
(2) $\mathsf{O}^{\mathsf{rDDH}}$**-Query simulator.** $\mathsf{A}^{\mathsf{GDL}}$ simply forwards $\mathsf{A}^{\mathsf{GDL}'}$'s query to $\mathsf{O}^{\mathsf{rDDH}}(g, g^a, \cdot, \cdot)$ and sends the response back.

First Run Output. At the end of first run, $\mathsf{A}^{\mathsf{GDL}'}$ outputs (s^*, i^*). Note that if this run is successful then $K[i^*] = y[i^*]^{(r[i^*]+a)s^*}$.

Second Run. $\mathsf{A}^{\mathsf{GDL}}$ now runs $\mathsf{A}^{\mathsf{GDL}'}$ again on the *same* input $(g, p, q, g^a; \omega)$ as used in first run, but answers its oracle queries differently as follows:

(1) R**-Query simulator**. When $\mathsf{A}^{\mathsf{GDL}'}$ makes its ith R query $(\widehat{y}[i], \widehat{K}[i])$, $\mathsf{A}^{\mathsf{GDL}}$ responds with $r[i]$ if $i < i^*$ and with $\widehat{r}[i]$ if $i \geq i^*$.
(2) $\mathsf{O}^{\mathsf{rDDH}}$**-Query simulator.** As in first run.

Second Run Output. At the end of the second run, $\mathsf{A}^{\mathsf{GDL}'}$ outputs $(\widehat{s}^*, \widehat{i}^*)$. Note that if this run is successful then $\widehat{K}[\widehat{i}^*] = \widehat{y}[\widehat{i}^*]^{\widehat{(r[\widehat{i}^*]+a)s^*}}$.

$\mathsf{A}^{\mathsf{GDL}}$*'s output*. If $\widehat{i}^* = i^*$ and $\widehat{r}[i^*] \neq r[i^*]$ then $\mathsf{A}^{\mathsf{GDL}}$ returns $\widehat{a} = (s^* r[i^*] - \widehat{s}^* \widehat{r}[i^*])/(\widehat{s}^* - s^*) \in \mathbb{Z}_q$. Otherwise, $\mathsf{A}^{\mathsf{GDL}}$ fails.

This completes the description of $\mathsf{A}^{\mathsf{GDL}}$. The running-time of $\mathsf{A}^{\mathsf{GDL}}$ is twice the run-time of $\mathsf{A}^{\mathsf{GDL}'}$ plus the time $O(k^2)$ to compute $\widehat{a}$ at the end. The number of $\mathsf{O}^{\mathsf{rDDH}}$ queries made by $\mathsf{A}^{\mathsf{GDL}}$ is up to twice the number of queries made by $\mathsf{A}^{\mathsf{GDL}'}$. This establishes the claimed resources of $\mathsf{A}^{\mathsf{GDL}}$.

We now lower bound the success probability of $\mathsf{A}^{\mathsf{GDL}}$. For $i \in \{1, \ldots, q_{\mathsf{R}}^{\mathrm{GDL}'}\}$, we call a run of $\mathsf{A}^{\mathsf{GDL}'}$ i-successful if $\mathsf{A}^{\mathsf{GDL}}$ succeeds and $i^* = i$. Note that in the above algorithm, if both first and second runs of $\mathsf{A}^{\mathsf{GDL}'}$ are i-successful for some i and $r[i] \neq \widehat{r}[i]$, then we have $i^* = \widehat{i^*} = i$, $K[i^*] = y[i^*]^{(r[i^*]+a)s^*}$ and $K[i^*] = y[i^*]^{\widehat{(r[i^*]+a)s^*}}$ (note that the first ith R queries are the same in both runs because the view of $\mathsf{A}^{\mathsf{GDL}'}$ is the same up to ith query response). Since $y[i^*] \in \langle g \rangle \backslash 1$ has order q this implies that $(\widehat{s}^* - s^*)a + \widehat{s}^* \widehat{r}[i^*] - s^* r[i^*] = 0$ and hence (noting that $\widehat{r}[i^*] \neq r[i^*]$ implies $\widehat{s}^* \neq s^*$) $\mathsf{A}^{\mathsf{GDL}}$'s output $\widehat{a} = (s^* r[i^*] - \widehat{s}^* \widehat{r}[i^*])/(\widehat{s}^* - s^*)$ is equal to a, so $\mathsf{A}^{\mathsf{GDL}}$ succeeds.

So it remains to lower bound the probability of the event S^* that both runs of $\mathsf{A}^{\mathsf{GDL}'}$ are i-successful for some $i \in \{1, \ldots, q_{\mathsf{R}}^{\mathrm{GDL}'}\}$ and $\widehat{r}[i] \neq r[i]$. To do this, we split S^* into $q_{\mathsf{R}}^{\mathrm{GDL}'}$ disjoint subevents $\mathsf{S}_{\mathsf{i}}^*$ according the value of i and bound each one. For each i, let A_i denote the outcome space for the random variable $a_i = (g, p, q, g^a, \omega, r[1], \ldots, r[i-1])$ consisting of the view of $\mathsf{A}^{\mathsf{GDL}'}$ up to the ith R-query, and let B_i denote the outcome space for the independent random variable $b_i = (r[i], \ldots, r[q_{\mathsf{R}}^{\mathrm{GDL}'}])$ consisting of the view of $\mathsf{A}^{\mathsf{GDL}'}$ after the ith R-query (including the response $r[i]$ to the ith query). Note that the event S_{i} that a run of $\mathsf{A}^{\mathsf{GDL}'}$ is i-successful is a subset of $A_i \times B_i$ with probability $p_i \stackrel{\text{def}}{=} \Pr[(a_i, b_i) \in \mathsf{S}_{\mathsf{i}}]$. Applying the Splitting Lemma 2, we know that there exists a subevent G_i of S_{i} such that $\Pr[(a_i, b_i) \in \mathsf{G}_{\mathsf{i}}] \geq p_i/2$, and for each $(a, b) \in \mathsf{G}_{\mathsf{i}}$, the probability that $(a, \widehat{b}) \in \mathsf{S}_{\mathsf{i}}$ over a random choice of $\widehat{b}$ in B_i is also at least $p_i/2$. Hence, the probability that the outcome (a, b) of the first run of $\mathsf{A}^{\mathsf{GDL}'}$ in our algorithm is in G_{i} is at least $p_i/2$, and then for each of those outcomes, the probability over the random choice of $\widehat{b} = (\widehat{r}[i], \ldots, \widehat{r}[q_{\mathsf{R}}^{\mathrm{GDL}'}])$ that the second run outcome $(a, \widehat{b})$ is in S_{i} is at least $p_i/2$. Since $\widehat{r}[i]$ is uniformly chosen in $\mathbb{Z}_q$, the chance that it collides with $r[i]$ is less than $1/2^{l_q(k)}$, so with probability at least $p_i/2 - 1/2^{l_q(k)}$ over $\widehat{b}$ we know that $(a, \widehat{b}) \in \mathsf{S}_{\mathsf{i}}$ and also $\widehat{r}[i] \neq r[i]$. Summarizing, we have that the probability that (1) $(a, b) \in \mathsf{G}_{\mathsf{i}}$ and (2) $(a, \widehat{b}) \in \mathsf{S}_{\mathsf{i}}$ and (3) $\widehat{r}[i] \neq r[i]$ all occur is at least $p_i/2(p_i/2 - 1/2^{l_q(k)})$. This latter event implies that both runs are i-successful and $\widehat{r}[i] \neq r[i]$, i.e. that event $\mathsf{S}_{\mathsf{i}}^*$ occurs. Hence,

$$\Pr[\mathsf{S}_{\mathsf{i}}^*] \geq p_i/2(p_i/2 - 1/2^{l_q(k)}) \qquad \text{for all} \quad i \in \{1, \ldots, q_{\mathsf{R}}^{\mathrm{GDL}'}\},$$

and since p_i is the probability that a run of $\mathsf{A}^{\mathsf{GDL}'}$ is i-successful, we know that $\sum_{i=1}^{q_o} p_i = \mathbf{Succ}^{\mathrm{GDL}'}_{\mathbb{Z}_p^*,\mathsf{A}^{\mathsf{GDL}'}}(k)$. Assuming that $\mathbf{Succ}^{\mathrm{GDL}'}_{\mathbb{Z}_p^*,\mathsf{A}^{\mathsf{GDL}'}}(k) \geq 2q_{\mathsf{R}}/2^{l_q(k)}$, we apply Lemma 3 to get the desired lower bound on $\mathsf{A}^{\mathsf{GDL}}$'s success probability:

$$\Pr[\mathsf{S}^*] = \sum_{i=1}^{q_{\mathsf{R}}^{\mathrm{GDL}'}} \Pr[\mathsf{S}_\mathsf{i}^*] \geq (1/q_{\mathsf{R}}^{\mathrm{GDL}'})\cdot(\mathbf{Succ}^{\mathrm{GDL}'}_{\mathbb{Z}_p^*,\mathsf{A}^{\mathsf{GDL}'}}(k)/2 - q_{\mathsf{R}}^{\mathrm{GDL}'}/2^{l_q(k)})^2. \qquad \square$$

To complete the proof of the theorem, we compose the reductions from "Stage 1" and "Stage 2" and convert A^{UF} into the desired algorithm $\mathsf{A}^{\mathsf{GDL}}$ with the resources claimed in the theorem statement and success probability satisfying the bound

$$\mathbf{Succ}^{\mathrm{FSO\text{-}UF\text{-}CMA}}_{\mathsf{A}^{\mathsf{UF}},\mathcal{ZSCR}}(k) \leq 2\sqrt{q_{\mathsf{H}}\cdot\mathbf{Succ}^{\mathrm{GDL}}_{\mathbb{Z}_p^*,\mathsf{A}^{\mathsf{GDL}}}(k)} + \frac{q_{\mathsf{SC}}(q_{\mathsf{G}}+q_{\mathsf{H}}+q_{\mathsf{SC}})+q_{\mathsf{H}}+1}{2^{l_q(k)-1}}.$$

The claimed insecurity bound of the theorem now follows by taking a maximum over all adversaries with the appropriate resource parameters. $\square$

We remark here that the above result is optimal in the sense that the GDL assumption is also *necessary* for the scheme $\mathcal{ZSCR}$ to achieve FSO-UF-CMA. If an efficient algorithm for the GDL problem is available, it can be used to compute efficiently the sender's secret key from his public key, by using the sender's flexible signcryption oracle to simulate the answers to the GDL algorithm's DDH queries. We refer the reader to [30] for the details of this attack.

6. Concluding Remarks

We have proved the confidentiality of Zheng's original signcryption scheme with respect to a strong and well-defined security notion that we introduced and called "FSO/FUO-IND-CCA2". Although this notion bears some similarities to the well-known "IND-CCA2" notion defined for standard public-key encryption schemes, it is stronger than the direct adaptation of "IND-CCA2" to the setting of signcryption, since we allow an attacker to query both the signcryption oracle and the unsigncryption oracle in a flexible way. We have also introduced a strong unforgeability notion called "FSO-UF-CMA" which allows a forger to query the signcryption oracle in a flexible way. We have successfully proved the unforgeability of Zheng's original signcryption with respect to this notion.

We emphasize here that our security models for signcryption are applicable not only to Zheng's original scheme but also to other various signcryption schemes: As mentioned earlier in this paper, Zheng's SDSS2-type signcryption scheme described in [33] and [34] can be proven to be secure relative to the same computational assumptions for the SDSS1-type signcryption scheme using the same proof techniques presented in this paper. Another immediate consequence of the results of this work is the provable confidentiality and unforgeability of the elliptic curve variants of Zheng's original signcryption scheme described in [35]. The only difference is that we need to rely on elliptic curve equivalent GDH and GDL assumptions in proving the security of the elliptic curve variants. It should also be noted that all these schemes require the use of two separate hash oracles, one in generating τ which acts as an encryption key for a message m and the other in obtaining r which participates in the generation of a signature.

Acknowledgements

The authors thank anonymous referees of PKC 2002 and *Journal of Cryptology* for their fruitful comments. The first two authors also thank Dr. Jan Newmarch at Monash University for his support on their research.

References

[1] J. An, Y. Dodis and T. Rabin: On the Security of Joint Signature and Encryption, *Advances in Cryptology—Proceedings of EUROCRYPT* 2002, Vol. 2332 of LNCS, Springer-Verlag, Berlin, 2002, pages 83–107.

[2] J. An, Y. Dodis and T. Rabin: On the Security of Joint Signature and Encryption, Report 2002/046, Cryptology ePrint Archive, 2002.

[3] J. Baek, R. Steinfeld and Y. Zheng: Formal Proofs for the Security of Signcryption, *Proceedings of Public Key Cryptography* 2002 (*PKC* 2002), Vol. 2274 of LNCS, Springer-Verlag, Berlin, 2002, pages 80–98.

[4] M. Bellare, A. Desai, E. Jokipii and P. Rogaway: A Concrete Security Treament of Symmetric Encryption, *Proceedings of FOCS* '97, IEEE Computer Society Press, Los Alamitos, CA, 1997, pages 394–403.

[5] M. Bellare, A. Desai, D. Pointcheval and P. Rogaway: Relations Among Notions of Security for Public-Key Encryption Schemes, *Advances in Cryptology—Proceedings of CRYPTO* '98, Vol. 1462 of LNCS, Springer-Verlag, Berlin, 1998, pages 26–45.

[6] M. Bellare and C. Namprepre: *Authenticated Encryption*: *Relations Among Notions and Analysis of the Generic Composition Paradigm*, *Advances in Cryptology—Proceedings of ASIACRYPT* 2000, Vol. 1976 of LNCS, Springer-Verlag, Berlin, 2000, pages 531–545.

[7] M. Bellare and P. Rogaway: Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols, *Proceedings of the First ACM Conference on Computer and Communications Security*, 1993, pages 62–73.

[8] M. Bellare and P. Rogaway: Optimal Asymmetric Encryption, *Advances in Cryptology—Proceedings of Eurocrypt* '94, Vol. 950 of LNCS, Springer-Verlag, Berlin, 1994, pages 92–111.

[9] M. Bellare and P. Rogaway: The Game-Playing Technique, Report 2004/331, International Association for Cryptographic Research (IACR) ePrint Archive, 2004.

[10] R. Cramer and V. Shoup: A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack, *Advances in Cryptology—Proceedings of CRYPTO* '98, Vol. 1462 of LNCS, Springer-Verlag, Berlin, 1998, pages 13–25.

[11] R. Cramer and V. Shoup: Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack, Report 2001/108, International Association for Cryptographic Research (IACR) ePrint Archive, 2001.

[12] T. ElGamal: A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms, *IEEE Transactions on Information Theory*, Vol. 31, 1985, pages 469–472.

[13] A. Fiat and A. Shamir: How to Prove Yourself: Practical Solutions of Identification and Signature Problems, *Proceedings of CRYPTO* '86, Vol. 263 of LNCS, Springer-Verlag, Berlin, 1987, pages 186–194.

[14] E. Fujisaki and T. Okamoto: How to Enhance the Security of Public-Key Encryption at Minimum Cost, *Proceedings of Public Key Cryptography* '99 (*PKC* '99), Vol. 1666 of LNCS, Springer-Verlag, Berlin, 1999, pages 53–68.

[15] S. Goldwasser and S. Micali: Probabilistic Encryption, *Journal of Computer and System Sciences*, Vol. 28, 1984, pages 270–299.

[16] S. Goldwasser, S. Micali and R. Rivest: A Digital Signature Scheme Secure against Adaptive Chosen-Message Attacks, *SIAM Journal on Computing*, Vol. 17, No. 2, 1988, pages 281–308.

[17] C. Jutla: Encryption Modes with Almost Free Message Integrity, *Advances in Cryptology—Proceedings of EUROCRYPT* 2001, Vol. 2045 of LNCS, Springer-Verlag, Berlin, 2001, pages 529–544.

[18] H. Krawczyk: The Order of Encryption and Authentication for Protecting Communications (Or: How Secure Is SSL?), *Advances in Cryptology—Proceedings of CRYPTO* 2001, Vol. 2139 of LNCS, Springer-Verlag, Berlin, 2001, pages 310–331.

[19] M. Naor and M. Yung: Public-Key Cryptosystems Secure against Chosen Ciphertext Attacks, *Proceedings of the* 22*nd ACM Sysmposium on Theory of Computing*, 1990, pages 427–437.

[20] K. Ohta and T. Okamoto: On Concrete Security Treatment of Signatures Derived from Identification, *Advances in Cryptology—Proceedings of CRYPTO* '98, Vol. 1462 of LNCS, Springer-Verlag, Berlin, 1998, pages 354–369.

[21] T. Okamoto and D. Pointcheval: The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes, *Proceedings of Public Key Cryptography* 2001 (*PKC* 2001), Vol. 1992 of LNCS, Springer-Verlag, Berlin, 2001, pages 104–118.

[22] D. Pointcheval: Chosen-Ciphertext Security for Any One-Way Cryptosystem, *Proceedings of Public Key Cryptography* 2000 (*PKC* 2000), Vol. 1751 of LNCS, Springer-Verlag, Berlin, 2000, pages 129–146.

[23] D. Pointcheval and J. Stern, *Security Arguments for Digital Signatures and Blind Signatures*, *Journal of Cryptology*, Vol. 13, No. 3, Springer-Verlag, Berlin, 2000, pages 361–396.

[24] C. Rackoff and D. Simon: Non-Interacitve Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack, *Advances in Cryptology—Proceedings of CRYPTO* '91, Vol. 576 of LNCS, Springer-Verlag, Berlin, 1992, pages 433–444.

[25] P. Rogaway, M. Bellare, J. Black and T. Krovetz: OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption, *Proceedings of the* 8*th ACM Conference on Computer and Communications Security*, 2001, pages 196–205.

[26] C. P. Schnorr: Efficient Identification and Signatures for Smart Cards, *Advances in Cryptology—Proceedings of CRYPTO* '89, Vol. 435 of LNCS, Springer-Verlag, Berlin, 1990, pages 235–251.

[27] C. P. Schnorr and M. Jakobsson: Security of Signed ElGamal Encryption, *Advances in Cryptology—Proceedings of ASIACRYPT* 2000, Vol. 1976 of LNCS, Springer-Verlag, Berlin, 2000, pages 73–89.

[28] V. Shoup: Sequences of Games: A Tool for Taming Complexity in Security Proofs, Report 2004/332, International Association for Cryptographic Research (IACR) ePrint Archive, 2004.

[29] V. Shoup and R. Gennaro: Securing Threshold Cryptosystems against Chosen Ciphertext Attack, *Advances in Cryptology—Proceedings of EUROCRYPT* '98, Vol. 1403 of LNCS, Springer-Verlag, Berlin, 1998, pages 1–16.

[30] R. Steinfeld: Analysis and Design of Public-Key Cryptographic Schemes, Ph.D. Thesis, Monash University, January 2003.

[31] R. Steinfeld and Y. Zheng: A Signcryption Scheme Based on Integer Factorization, *Proceedings of Information Security Workshop* 2000 (*ISW* 2000), Vol. 1975 of LNCS, Springer-Verlag, Berlin, 2000, pages 308–322.

[32] Y. Tsiounis and M. Yung: On the Security of ElGamal-Based Encryption, *Proceedings of Public Key Cryptography* '98 (*PKC* '98), Vol. 1431 of LNCS, Springer-Verlag, Berlin, 1998, pages 117–134.

[33] Y. Zheng: Digital Signcryption or How to Achieve Cost (Signature & Encryption) $\ll$ Cost (Signature) $+$ Cost (Encryption), *Advances in Cryptology—Proceedings CRYPTO* '97, Vol. 1294 of LNCS, Springer-Verlag, Berlin, 1997, pages 165–179.

[34] Y. Zheng: Digital Signcryption or How to Achieve Cost (Signature & Encryption) $\ll$ Cost (Signature) $+$ Cost (Encryption), full version, available at `http://www.sis.uncc.edu/~yzheng/papers/`.

[35] Y. Zheng and H. Imai: Efficient Signcryption Schemes on Elliptic Curves, *Proceedings of the IFIP* 14*th International Information Security Conference* (*IFIP/SEC* '98), Chapman and Hall, New York, 1998, pages 75–84.

[36] Y. Zheng and J. Seberry: Immunizing Public Key Cryptosystems against Chosen Ciphertext Attacks, *IEEE Journal on Selected Areas in Communications*, Vol. 11, No. 5, 1993, pages 715–724 (Special Issue on Secure Communications).