D. Nuñez, I. Agudo, and J. Lopez, "On the Application of Generic CCA-Secure Transformations to Proxy Re-Encryption", *Security and Communication Networks*, vol. 9, pp. 1769-1785, 2016.
http://doi.org/10.1002/sec.1434
NICS Lab. Publications: https://www.nics.uma.es/publications

On the Application of Generic CCA-Secure Transformations to Proxy Re-Encryption

David Nuñez, Isaac Agudo, and Javier Lopez

Abstract

Several generic methods exist for achieving CCA-secure public-key encryption schemes from weakly secure cryptosystems, such as the Fujisaki-Okamoto and REACT transformations. In the context of Proxy Re-Encryption (PRE), it would be desirable to count on analogous constructions that allow PRE schemes to achieve better security notions. In this paper, we study the adaptation of these transformations to proxy re-encryption and find both negative and positive results. On the one hand, we show why it is not possible to directly integrate these transformations with weakly-secure PRE schemes due to general obstacles coming from both the constructions themselves and the security models, and we identify twelve PRE schemes that exhibit these problems. On the other hand, we propose an extension of the Fujisaki-Okamoto transformation for PRE, which achieves a weak form of CCA-security in the random oracle model, and we describe the sufficient conditions for applying it.

1 Introduction

A coveted goal for any cryptosystem is to satisfy a strong notion of security, relevant to its security objectives. In the case of Public-Key Encryption (PKE), indistinguishability against chosen-ciphertext attacks (IND-CCA) is widely regarded as the right notion of security [1]. Informally, the IND-CCA notion describes a security model where any adversary, even with access to a decryption oracle, is not able to distinguish messages for a given ciphertext.

The same desire to achieve right security notions drives the design of other kinds of cryptosystems. This paper is devoted to Proxy Re-Encryption (PRE), a type of Public-Key Encryption (PKE) where, in addition to the usual encryption and decryption capabilities, a "re-encryption" functionality enables a proxy entity to transform ciphertexts under the public key of Alice into ciphertexts decryptable by Bob. For doing so, the proxy must have a re-encryption key that makes this transformation possible. In addition, the proxy cannot learn any information about the encrypted messages, under any of the keys.

Similarly to the PKE case, the IND-CCA notion is also considered as the target security notion for PRE schemes. In addition to the decryption capabilities, this notion has to consider the ability of the adversary to re-encrypt chosen ciphertexts by means of a re-encryption oracle. However, this added capability poses an interesting challenge since the possibility of re-encrypting ciphertexts conflicts with the traditional view of CCA security, which implies non-malleability of ciphertexts, as pointed out by Canetti and Hohenberger, the

authors of one of the first CCA-secure PRE scheme [2]. Given the peculiarities of IND-CCA security in PRE, it is often not easy to devise schemes that achieve strong security notions. Most CCA-secure PRE schemes usually resort to additional constructions such as one-time signatures [2, 3], Schnorr signatures [4] and non-interactive zero-knowledge proofs [5], which have to be carefully integrated in an ad-hoc manner. Apart from this difficulty, there are also other factors, such as efficiency and design simplicity, which are usually negatively impacted in CCA-secure PRE schemes.

In the context of PKE, several generic methods exist to achieve CCA-secure schemes from weakly secure cryptosystems. Fujisaki and Okamoto proposed in 1999 [6], and revisited recently in [7], a generic conversion to achieve CCA-security in the random oracle model from a weakly secure asymmetric cryptosystem and a symmetric encryption scheme. Similar transformations, such as REACT [8] and GEM [9], have been proposed since then.

Hence, it is natural to wonder whether an analogous transformation can be constructed for proxy re-encryption schemes. A direct and naive application of the aforementioned transformations leads, in general, to problems in the security proofs, and in this paper we show several examples from the literature.

On a positive note, we also describe sufficient conditions that allow to apply the Fujisaki-Okamoto directly. These conditions include the satisfaction of a new property of PRE called "*perfect key-switching*", which characterizes schemes where the re-encryption process preserves the original randomness of ciphertexts, and a weak notion of security called $\mathsf{IND\text{-}CCA}_{0,1}$, due to Nuñez et al. [10], where the adversary only has access to the re-encryption oracle before the challenge. The schemes resulting from this transformation achieve $\mathsf{IND\text{-}CCA}_{2,1}$ security in the random oracle model, a notion slightly weaker than full CCA-security. As an illustration, we present an example of a scheme that satisfies the application conditions and we show the resulting scheme after the transformation. Finally, in addition to the Fujisaki-Okamoto proposal, we also outline how the REACT [8] and GEM [9] transformations could be extended for proxy re-encryption.

Our results have a direct impact on the security of PRE schemes. There are multitude of applications of proxy re-encryption, ranging from data sharing in the cloud to key management in sensor networks. Most of these applications would benefit from more secure schemes, that are, at the same time, easy to understand and implement.

1.1 Our Contribution

In this paper we present the following results:

- We describe an extension of the Fujisaki-Okamoto transformation to PRE, and formulate sufficient conditions that allow to use it. These conditions include a new proxy re-encryption property called *perfect key-switching*, which characterizes PRE schemes whose re-encryption procedure does not affect the original randomness, so the original public key is cleanly switched to a new one.

- We sketch similar transformations based on REACT and GEM, which, potentially, could be applicable to a wider class of PRE schemes than the Fujisaki-Okamoto extension.

- We describe flaws in twelve PRE schemes that are allegedly "CCA-secure" caused by a direct use of the Fujisaki-Okamoto transformation and similar constructions, which are generally not applicable due to the peculiarities of re-encryption.

1.2 Related work

The Fujisaki-Okamoto transformation, originally proposed in [6] and recently revisited in [7], was the first generic transformation from weakly secure encryption schemes to a CCA-secure public key cryptosystem. Later, Okamoto and Pointcheval described a more efficient construction, called REACT [8], which in turn inspired Coron et al. to propose GEM [9], a more complex construction, but that achieves shorter ciphertexts. Although both REACT and GEM are more efficient than the Fujisaki-Okamoto transformation, they require stronger assumptions on the underlying public-key scheme (see Section 5).

There are several examples of the application and modification of such transformations for other kinds of cryptosystems. Kitagawa et al. study in [11] the adaptation of both Fujisaki-Okamoto and REACT to Identity-Based Encryption, and estimate their reduction costs. Similar works exist in the context of certificateless public-key encryption [12], and certificated-based encryption [13]. Although these extensions are relevant to our work, the inherent difficulties that arise from the re-encryption capabilities of PRE pose a challenging problem.

To the best of our knowledge, there is no prior work on generic transformations for PRE, in the sense of constructions that increase the security of existing proxy re-encryption schemes. There are, however, a couple of works that propose generic methods for constructing CCA-secure PRE schemes out of other kinds of cryptosystems. Shao et al. propose in [14] a generic method based on a CCA-secure threshold encryption scheme; additionaly they describe variants to achieve collusion resistance and identity-based PRE. In a posterior work, Hanaoka et al. present in [15] another generic method to construct CCA-secure PRE schemes, also based on the use of threshold encryption, together with a CCA-secure PKE scheme and a strongly unforgeable signature scheme. However, this latter method requires long keys and ciphertexts. For instance, a re-encryption key is made of two PKE public keys, one PKE ciphertext, a threshold encryption share and a signature.

1.3 Organization

The rest of this paper is organized as follows: In Section 2 we formalize proxy re-encryption syntax and definitions of security. In Section 3 we introduce the original Fujisaki-Okamoto transformation and present the conditions that allow to directly apply it to PRE. In Section 4 we show an example of scheme that permits the application of the transformation and present its result. In Section 5 we discuss other possible generic transformations, based on REACT and GEM. Finally, Section 6 concludes the paper and future work is outlined.

2 Preliminaries

In this section we introduce the syntax and security definitions associated to proxy re-encryption, as well as the original Fujisaki-Okamoto transformation for public-key encryption.

2.1 Proxy re-encryption syntax and properties

We define the syntax of a proxy re-encryption scheme as follows, based on the definitions by Canetti and Hohenberger [2] and Ateniese et al. [16]:

Definition 1 (PRE scheme)**.** *A proxy re-encryption scheme is a tuple of algorithms (*Setup*,* KeyGen*,* ReKeyGen*,* Enc*,* ReEnc*,* Dec*):*

- $\mathsf{Setup}(\lambda) \to params$. *On input the security parameter λ, the setup algorithm produces a set of global parameters, publicly known by all parties.*
- $\mathsf{KeyGen}(\lambda) \to (pk_i, sk_i)$. *The key generation algorithm outputs a pair of public and secret keys (pk_i, sk_i) for user i.*
- $\mathsf{ReKeyGen}(pk_i, sk_i, pk_j, sk_j) \to rk_{i \to j}$. *On input the pair of public and secret keys (pk_i, sk_i) for user i and the pair of public and secret keys (pk_j, sk_j) for user j, the re-encryption key generation algorithm outputs a re-encryption key $rk_{i \to j}$.*
- $\mathsf{Enc}(pk_i, m) \to c_i$. *On input the public key pk_i and a message $m \in \mathcal{M}$, the encryption algorithm outputs a ciphertext $c_i \in \mathcal{C}$.*
- $\mathsf{ReEnc}(rk_{i \to j}, c_i) \to c_j$. *On input a re-encryption key $rk_{i \to j}$ and a ciphertext $c_i \in \mathcal{C}$, the re-encryption algorithm outputs a second ciphertext $c_j \in \mathcal{C}$ or the symbol $\perp$ indicating c_i is invalid.*
- $\mathsf{Dec}(sk_i, c_i) \to m$. *On input the secret key sk_i and a ciphertext $c_i \in \mathcal{C}$, the decryption algorithm outputs a message $m \in \mathcal{M}$ or the symbol $\perp$ indicating c_i is invalid.*

The message and ciphertext spaces are denoted by $\mathcal{M}$ and $\mathcal{C}$, respectively.

A more general definition of the syntax of PRE schemes was initially proposed by Ateniese et al. [16]. This definition considers sets of algorithms $\overrightarrow{\mathsf{Enc}}$ and $\overrightarrow{\mathsf{Dec}}$, instead of single encryption and decryption algorithms. This is a usual choice for several PRE schemes in the literature [16, 3]. Definition 1 can be considered as a special case of this one.

A PRE scheme is *unidirectional* if the re-encryption keys only allow the transformation of ciphertexts in one direction, from delegator to delegatee, while it is *bidirectional* otherwise. We say a PRE scheme is *single-hop* if the ciphertext resulting from a re-encryption cannot be re-encrypted again, while it is *multi-hop* otherwise. Finally, if the secret key sk_j of the user j is not needed by the re-encryption key generation algorithm ReKeyGen, then the scheme is *non interactive*, since re-encryption keys for user j can be produced without her involvement.

2.2 Definitions of security for PRE

Security of PRE schemes mostly follows a game-based approach, where a challenger and an adversary interact with each other. Normally, the goal of the latter is to correctly distinguish which message, from two possible options of her choice, is encrypted by the challenge ciphertext; this is called the indistinguishability (IND) game. The capabilities of the adversary are specified by means of an attack model, which in PRE is usually either the Chosen Plaintext Attack (CPA) or the Chosen Ciphertext Attack (CCA) models.

For reasons we will see later, in this work we need a finer-grained definition of the attack models, so we will make use of the parametric family of attack models for PRE proposed in [10] by Nuñez et al., which is defined in terms of the availability of the decryption and re-encryption oracles during the security game. These attack models are of the form $\mathsf{CCA}_{i,j}$, where indices $i, j \in \{0, 1, 2\}$ mark the last phase of the security game where the decryption and re-encryption oracles, respectively, are available to the adversary. For example, $\mathsf{CCA}_{2,1}$ represents the attack model where the decryption oracle is available until phase 2 (i.e., in both phases of the security game) and the re-encryption oracle only in phase 1, while $\mathsf{CCA}_{0,0}$ is an attack model where both oracles are not available (in fact, this latter attack model is equivalent to CPA). From this family of attack models, we can derive a parametric set of indistinguishability-based notions of the form $\mathsf{IND\text{-}CCA}_{i,j}$, which represents the instantiation of the indistinguishability game with the $\mathsf{CCA}_{i,j}$ attack model. Figure 1 depicts the resulting lattice-shaped hierarchy of indistinguishability notions. See [10] for more details about the definition of the parametric families of attack models and security notions for PRE.

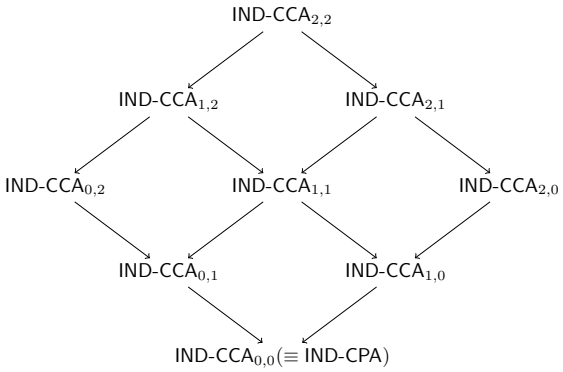

Figure 1: Hierarchy of indistinguishability notions for PRE.

The definition of the indistinguishability game for PRE, parametrized by the $\mathsf{CCA}_{i,j}$ attack model, is as follows.

Definition 2 ($\mathsf{IND\text{-}CCA}_{i,j}$ security)**.** *Let* Π*=(KeyGen, ReKeyGen, Enc, Dec, ReEnc) be a proxy re-encryption scheme,* $A = (A_1, A_2)$ *a polynomial-time adversary, and* Ω_1 *and* Ω_2 *be the set of available oracles for* A_1 *and* A_2*, respectively. For* $i, j \in \{0, 1, 2\}$*,* $\delta \in \{0, 1\}$*, and* $\lambda \in \mathbb{N}$*, the indistinguishability game*

is defined by the following experiment

$$\begin{array}{l}\text{Experiment } \mathbf{Exp}^{\mathit{IND\text{-}CCA}_{i,j}}_{\Pi,A,\delta}(\lambda)\\ \quad (pk^*, sk^*) \xleftarrow{R} \mathsf{KeyGen}(\lambda); \qquad (m_0, m_1, s) \leftarrow A_1(pk^*);\\ \quad c^* \leftarrow \mathsf{Enc}(pk^*, m_\delta); \qquad d \leftarrow A_2(m_0, m_1, s, c^*)\\ \quad \text{return } d\end{array}$$

The set of available oracles is defined by indices i, j in the attack model $\mathsf{CCA}_{i,j}$. Additionally, it is required that adversarial oracle queries satisfy the usual restrictions about derivatives of the challenge ciphertext c^*. The advantage of A is given by

$$\mathbf{Adv}^{\mathit{IND\text{-}CCA}_{i,j}}_{\Pi,A}(\lambda) = |\Pr[\mathbf{Exp}^{\mathit{IND\text{-}CCA}_{i,j}}_{\Pi,A,1}(\lambda) = 1] - \Pr[\mathbf{Exp}^{\mathit{IND\text{-}CCA}_{i,j}}_{\Pi,A,0}(\lambda) = 1]|$$

We say that Π is $\mathit{IND\text{-}CCA}_{i,j}$ secure if this advantage is negligible.

Note that, for space reasons, we omit here the definition of the PRE oracles and the concept of derivatives of the challenge. Informally, this concept allows to filter out those pairs of (pk, c) that are linked to the challenge (pk^*, c^*) through queries to the re-encryption and re-encryption key generation oracles, and which would allow trivial attacks from the adversary. See [10] for a detailed description of the PRE oracles and the concept of derivatives of the challenge.

3 Adapting the Fujisaki-Okamoto Transformation to PRE

In this section we describe the original Fujisaki-Okamoto transformation and describe the conditions that allow to apply it correctly to proxy re-encryption.

3.1 The Fujisaki-Okamoto Transformation

Fujisaki and Okamoto proposed in 1999 a generic transformation to achieve IND-CCA security in the random oracle model from a public key encryption scheme with one-way security under chosen plaintext attacks (OW-CPA) [6]. In order to do so, the generic transformation integrates the PKE scheme with an IND-CPA-secure symmetric scheme and a pair of hash functions. Recently, the authors presented a revised version [7], which will be the one we consider in this paper.

The hybrid transformation, which we denote as Hyb, is as follows. Let PKE be a public-key encryption scheme, Sym a symmetric encryption scheme, and H and G hash functions. PKE is non-deterministic, so in addition to the public key and the message, its encryption function takes an additional parameter that introduces randomness in the ciphertext.

In order to encrypt a message m, the hybrid transformation first samples randomly a term σ from the message space of PKE. The message m is then encrypted with Sym using $G(\sigma)$ as key, which produces the term $c = \mathsf{Sym.Enc}(G(\sigma), m)$. Next, the σ term is encrypted with PKE, taking $H(\sigma, c)$ as

the random coins. The hybrid transformation produces the following tuple as the encryption of message m:

$$\mathsf{Hyb.Enc}(pk, m) = (\mathsf{PKE.Enc}(pk, \sigma; H(\sigma, c)), c)$$

When decrypting a ciphertext, which we will denote by the tuple (e, c), the hybrid transformation performs the inverse procedures in reverse order: it decrypts σ from e, and computes $G(\sigma)$ in order to extract the decryption key for c, thus obtaining the original message m. However, an additional validation step is performed during the decryption. This step involves re-computing the term $e = \mathsf{PKE.Enc}(pk, \sigma; H(\sigma, c))$ of the ciphertext, ensuring this way that the ciphertext is valid. If this check does not succeed, the ciphertext is rejected.

This very last step is the reason why the Fujisaki-Okamoto transformation fails if applied as is in PRE: the re-encryption function may change the ciphertexts in such a way that the validation checking that takes place during the decryption inevitably fails. In particular, if the alteration produced by the re-encryption affects the original random coins introduced in the encryption, the validation check will fail once a ciphertext is re-encrypted. Appendix B shows how the CCA scheme from Aono et al. [17] is not correct as a consequence of this issue.

As a solution to this problem, it is interesting to think of PRE schemes where the re-encryption does not alter the original randomness. In this section we characterize a property of PRE schemes that captures this notion, called *perfect key-switching*, and use it as one of the conditions to successfully apply the Fujisaki-Okamoto transformation to PRE.

3.2 A New Property of PRE: Perfect Key-Switching

In the previous section we identified that the alteration of the original randomness prevents Fujisaki-Okamoto's decryption procedure from validating a re-encrypted ciphertext. We are therefore interested on those schemes where the original randomness is preserved.

Informally, a PRE scheme satisfies the *perfect key-switching* property if for all pairs of public keys pk_i and pk_j, all messages m, and all randomness r then the re-encrypted ciphertext produced by $\mathsf{ReEnc}(rk_{i\to j}, \mathsf{Enc}(pk_i, m; r))$ is exactly the same than the ciphertext generated by $\mathsf{Enc}(pk_j, m; r)$. This informal definition assumes than in both cases one is using the same encryption function Enc; however, several schemes are defined with multiple encryption and decryption functions, as discussed in Section 2.1. The following is a more generic definition, which takes this latter issue into consideration.

Definition 3 (Perfect Key-Switching)**.** *Let* $\Pi = (\mathsf{KeyGen}, \mathsf{ReKeyGen}, \overrightarrow{\mathsf{Enc}}, \overrightarrow{\mathsf{Dec}}, \mathsf{ReEnc})$ *be a PRE scheme, with message space* $\mathcal{M}$ *and random coins space* $\mathcal{R}$*, and* λ *the security parameter. Let us assume that the re-encryption function* ReEnc *transforms ciphertexts encrypted by* $\mathsf{Enc}_k \in \overrightarrow{\mathsf{Enc}}$ *into ciphertexts decryptable by* $\mathsf{Dec}_{k'} \in \overrightarrow{\mathsf{Dec}}$*. We say that* Π *satisfies the* perfect key-switching *property if for all keypairs* $(pk_i, sk_i), (pk_j, sk_j)$ *generated by* $\mathsf{KeyGen}(\lambda)$*, all* $m \in \mathcal{M}$*, all* $r \in \mathcal{R}$*, and all* $rk_{i\to j} = \mathsf{ReKeyGen}(pk_i, sk_i, pk_j, sk_j)$*, then*

$$\mathsf{ReEnc}(rk_{i\to j}, \mathsf{Enc}_k(pk_i, m; r)) = \mathsf{Enc}_{k'}(pk_j, m; r)$$

It can be seen that the key-switching procedure that takes places during re-encryption is "perfect", in the sense that it does not affect the random coins used. Informally, the re-encryption simply "switches" one public key for another. Examples of this type of scheme are the BBS [18] and CH [2] schemes. It is easy to check that the former exhibits this property. The BBS scheme, based on the ElGamal cryptosystem, is constructed over a group $\mathbb{G}$ of prime order q, with generator g. Secret keys are of the form $sk = a \in \mathbb{Z}_q$, whereas public keys are $pk = g^a \in \mathbb{G}$. There is only one type of ciphertexts, which are of the form $(pk^r, g^r \cdot m)$, for a random exponent r. Re-encryption keys are computed as $rk_{i \to j} = \frac{sk_j}{sk_i}$, so the re-encryption process simply consists on raising the pk^r component of the ciphertext to the re-encryption key. This scheme fulfills the perfect key-switching property since the re-encryption process gracefully removes the original public key and substitutes it with the new one, preserving the original randomness:

$$\begin{aligned}
&\mathsf{ReEnc}(rk_{i \to j}, \mathsf{Enc}(pk_i, m; r)) \\
&\quad = \mathsf{ReEnc}(\frac{b}{a}, ((g^a)^r, g^r \cdot m)) = ((g^{ar})^{\frac{b}{a}}, g^r \cdot m) \\
&\quad = (g^{br}, g^r \cdot m) = \mathsf{Enc}(pk_j, m; r)
\end{aligned}$$

Therefore, the original ciphertext $(g^{ar}, g^r \cdot m)$ is perfectly transformed after re-encryption into $(g^{br}, g^r \cdot m)$.

An immediate consequence of the perfect key-switching property is that it implies *proxy invisibility*. A PRE scheme is said to be proxy-invisible if a delegatee is unable to distinguish a ciphertext computed under her public key from a re-encrypted ciphertext, originally encrypted under another public key [16]. That is, the proxy is "invisible", in the sense that the delegatee cannot discern whether the proxy has transformed the ciphertexts. From this description, it is clear that perfect key-switching implies proxy invisibility: a re-encrypted ciphertext has exactly the same form as a ciphertext originally encrypted with the delegatee's public key. However, the converse implication (i.e., proxy invisibility implies perfect key-switching) is not necessarily true. For example, the third proposal from Ateniese et al. [16] is proxy invisible, but does not satisfy the perfect key-switching property, since the original randomness is altered after the re-encryption; in particular, the original random component r is polluted during the re-encryption with the original secret key a, so the new randomness becomes $r' = a \cdot r$. This does not affect security of the scheme in any way because the random elements are safely conveyed as exponents (i.e., exploiting this would imply solving the discrete logarithm problem), but it is sufficient for disallowing the reconstruction of a re-encrypted ciphertext from the original message and randomness, hence breaking the perfect key-switching property.

Another interesting result is that the perfect key-switching property also implies that the PRE scheme cannot be *key-private*, a property of PRE schemes where the proxy cannot learn the identity of the involved users from the re-encryption key. This property was first defined in [19], where the authors formulated some necessary conditions to achieve key privacy. One of this conditions is that the re-encryption function must be probabilistic. However, it can be seen that our formulation of perfect key-switching requires deterministic re-encryption. Therefore, if a PRE scheme satisfies the perfect key-switching

property, it cannot be key-private.

3.3 Extension of the Fujisaki-Okamoto transformation to PRE

The intuition behind how the Fujisaki-Okamoto transformation can be extended to PRE is simple: assuming the underlying PRE scheme satisfies the perfect key-switching property, then a re-encrypted ciphertext is equivalent to an original ciphertext created with the same randomness. Therefore, the re-encryption does not affect the transformation.

3.3.1 The extended transformation

Let PRE be a proxy re-encryption scheme and Sym a symmetric encryption scheme. Let $\mathcal{M}^{pre}, \mathcal{C}^{pre}$ and $\mathcal{R}^{pre}$ denote the message, ciphertext and randomness spaces of PRE, respectively, while $\mathcal{M}^{sym}, \mathcal{C}^{sym}$ and $\mathcal{K}^{sym}$ denote the message, ciphertext and key spaces of Sym. The encryption function in PRE is non-deterministic, so besides the public key and the message, it takes a parameter from $\mathcal{R}^{pre}$ that introduces randomness in the ciphertext. Let H and G be hash functions, where $H : \mathcal{M}^{pre} \times \mathcal{C}^{sym} \to \mathcal{R}^{pre}$ and $G : \mathcal{M}^{pre} \to \mathcal{K}^{sym}$. We denote as Hyb to the hybrid scheme that results from applying the extended Fujisaki-Okamoto transformation, which is generically defined as follows:

- $\mathsf{Hyb.Setup}(\lambda) \to params'$. On input the security parameter λ, the setup algorithm first computes $params \leftarrow \mathsf{PRE.Setup}(\lambda)$ and outputs the set of global parameters $params' = params \cup \{H(\cdot), G(\cdot)\}$.

- $\mathsf{Hyb.KeyGen}(\lambda) \to (pk_i, sk_i)$. On input the security parameter λ, the key generation algorithm outputs $(pk_i, sk_i) \leftarrow \mathsf{PRE.KeyGen}(\lambda)$, which is the pair of public and secret keys for user i.

- $\mathsf{Hyb.ReKeyGen}(pk_i, sk_i, pk_j, sk_j) \to rk_{i \to j}$. On input the pair of public and secret keys (pk_i, sk_i) for user i and the pair of public and secret keys (pk_j, sk_j) for user j, the re-encryption key generation algorithm outputs the re-encryption key $rk_{i \to j} \leftarrow \mathsf{PRE.ReKeyGen}(pk_i, sk_i, pk_j, sk_j)$.

- $\mathsf{Hyb.Enc}(pk_i, m) \to (e_i, c_i)$. On input a public key pk_i and a message $m \in \mathcal{M}^{sym}$, the encryption algorithm first samples a random $\sigma \in \mathcal{M}^{pre}$, and computes $c_i \leftarrow \mathsf{Sym.Enc}(G(\sigma), m)$. Next, it computes $e_i \leftarrow \mathsf{PRE.Enc}(pk_i, \sigma; H(\sigma, c_i))$. Finally, it outputs ciphertext (e_i, c_i).

- $\mathsf{Hyb.ReEnc}(rk_{i \to j}, (e_i, c_i)) \to (e_j, c_j)$. On input a re-encryption key $rk_{i \to j}$ and a ciphertext (e_i, c_i), the re-encryption algorithm outputs the ciphertext $(\mathsf{PRE.ReEnc}(rk_{i \to j}, e_i), c_i)$.

- $\mathsf{Hyb.Dec}(sk_i, (e_i, c_i)) \to m$. On input the secret key sk_i and a ciphertext (e_i, c_i), the decryption algorithm first computes $\sigma \leftarrow \mathsf{PRE.Dec}(sk_i, e_i)$, and verifies that $\sigma \in \mathcal{M}^{pre}$; otherwise, it outputs $\perp$. Next, it verifies that $e_i = \mathsf{PRE.Enc}(pk_i, \sigma; H(\sigma, c_i))$; otherwise, it outputs $\perp$. Finally, it outputs the result of the symmetric decryption algorithm $\mathsf{Sym.Dec}(G(\sigma), c_i)$.

Note that the resulting hybrid PRE scheme requires the public key during the decryption, since it needs to reconstruct the input ciphertext for validation purposes. Nevertheless, in order to preserve the generic syntax of PRE schemes, one could compute the public key from the secret key (if the underlying PRE scheme allows this possibility) or simply consider the public key as part of the secret key. This extension could also implement countermeasures against reject timing attacks, such as those proposed by Galindo et al. [20]; we will, however, consider them out of the scope of this paper.

3.3.2 Correctness of the transformation

It is necessary to prove that the extended hybrid transformation produces a correct PRE scheme. We assume that the underlying PRE and symmetric schemes are correct. Recall that we additionally require that the PRE scheme satisfies the perfect key-switching property. In PRE, besides the usual PKE condition for correctness, schemes must verify that re-encrypted ciphertexts are correctly decrypted, for any message m:

$$\mathsf{Dec}(sk_j, \mathsf{ReEnc}(rk_{i\to j}, \mathsf{Enc}(pk_i, m)) = m$$

Therefore, for the scheme that results from applying our hybrid transformation, we must prove that the following equation holds:

$$\mathsf{Hyb.Dec}(sk_j, \mathsf{Hyb.ReEnc}(rk_{i\to j}, \mathsf{Hyb.Enc}(pk_i, m)) = m \tag{1}$$

First, by definition of the encryption function of the hybrid transformation, $\mathsf{Hyb.Enc}(pk_i, m)) = (e_i, c_i)$, for $c_i = \mathsf{Sym.Enc}(G(\sigma), m)$ and $e_i = \mathsf{PRE.Enc}(pk_i, \sigma; H(\sigma, c_i)))$. Then, the left side of Equation 1 can be written as:

$$\mathsf{Hyb.Dec}(sk_j, \mathsf{Hyb.ReEnc}(rk_{i\to j}, (e_i, c_i))$$

Next, by definition of the re-encryption function of the hybrid transformation, we have that:

$$\mathsf{Hyb.ReEnc}(rk_{i\to j}, (e_i, c_i)) = (\mathsf{PRE.ReEnc}(rk_{i\to j}, e_i), c_i)$$

By assumption, the underlying PRE scheme satisfies the property of perfect key-switching. Then:

$$\begin{aligned}&\mathsf{PRE.ReEnc}(rk_{i\to j}, e_i)\\ &\quad= \mathsf{PRE.ReEnc}(rk_{i\to j}, \mathsf{PRE.Enc}(pk_i, \sigma; H(\sigma, c_i)))\\ &\quad= \mathsf{PRE.Enc}(pk_j, \sigma; H(\sigma, c_i)) = e_j\end{aligned} \tag{2}$$

Therefore, Equation 1 is equivalent to:

$$\mathsf{Hyb.Dec}(sk_j, (e_j, c_i)) = m$$

Finally, we must check that the decryption is correct. By definition of the decryption algorithm of the hybrid transformation, we first compute $\sigma' \leftarrow \mathsf{PRE.Dec}(sk_j, e_j)$, and verify that $\sigma' \in \mathcal{M}^{pre}$. By correctness of the PRE scheme, we have that $\sigma' = \sigma$ and $\sigma \in \mathcal{M}^{pre}$, so the verification succeeds. Next, we must verify that $e_j = \mathsf{PRE.Enc}(pk_j, \sigma'; H(\sigma', c_i))$, which is true since $\sigma' = \sigma$ and the result of Equation 2. The last step outputs the result of the symmetric decryption algorithm $\mathsf{Sym.Dec}(G(\sigma), c_i)$, which is equal to m, since $\sigma' = \sigma$ and the symmetric encryption scheme is correct, by assumption. Therefore, Equation 1 holds, and the hybrid transformation is correct.

3.3.3 Security of the transformation

Although the transformation seems to be rather straightforward, proving its security is a more elusive matter. The original Fujisaki-Okamoto transformation for PKE achieves IND-CCA2 security, requiring the underlying PKE scheme to be only one-way secure under chosen-plaintext attacks (OW-CPA). This is possible because, in the security proof, the decryption oracle can be constructed without knowledge of the secret key, using only the random oracle tables.

For PRE, it is also necessary to define a re-encryption oracle. A tempting strategy, used in the proofs of several PRE schemes, is to do it similarly to the decryption oracle: on input a re-encryption query $(pk_i, pk_j, (e_i, c_i))$, the simulator searches in the random oracle tables for a tuple (σ, c_i, h), such that $e_i = \mathsf{PRE.Enc}(pk_i, \sigma; h)$. If such tuple is found, then the re-encrypted ciphertext is $(\mathsf{PRE.Enc}(pk_j, \sigma; h), c_i)$; otherwise, the oracle cannot respond to the query and returns $\perp$.

This solution seems elegant and simple, but is flawed. If the adversary inputs an ill-formed ciphertext where the randomness does not come from the random oracle H, the oracle rejects the ciphertext, outputting $\perp$. However, in a real execution of the scheme, the re-encryption function is unable to verify whether the input ciphertext was created using H or not, since σ must be kept secret during re-encryption. Therefore, the security proof differs at this point from the real execution. More worryingly still, the adversary can potentially make an arbitrary number of such queries. If the simulator opts to abort at this point, then it cannot use the output from the adversary since the probability of aborting the simulation could be correlated with the adversary's queries. Unfortunately, these proofs cannot be "patched" using techniques like the artificial aborts from Waters' IBE proof [21]. The identified problem seems to be common and appears in several PRE schemes that make use of constructions inspired in the Fujisaki-Okamoto transformation, rendering their security proofs invalid [22, 5, 4, 23, 24, 25, 26, 27, 28, 29, 30]. Appendix A analyzes this problem in detail.

Therefore, since the re-encryption function of our extended transformation cannot check whether the hash function H was used or not, it is necessary to construct the re-encryption oracle without relying on the random oracle tables. Furthermore, since the simulator does not have access to the target secret key sk^*, it is not possible in general to compute re-encryption keys of the form $rk_{pk^* \to pk_j}$, for any user j.

In order to bypass these problems, we strengthen the requirements on the underlying PRE scheme: instead of OW-CPA security, we now require $\mathsf{IND\text{-}CCA}_{0,1}$ security. We then construct the security proof of the transformation as a reduction from the security of the underlying scheme, making use of its definition of the re-encryption oracle. This also implies that the transformation cannot achieve $\mathsf{IND\text{-}CCA}_{2,2}$ security, but $\mathsf{IND\text{-}CCA}_{2,1}$, which is a slightly weaker notion.

The following theorem conveys our main security result. For simplicity in the proofs, we have opted for using the one-time pad rather than a generic symmetric encryption scheme, so the second component of ciphertexts is computed as $c = G(\sigma) \oplus m$, instead of $c = \mathsf{Sym.Enc}(G(\sigma), m)$.

Theorem 1 (Security of the transformation)**.** *Let* Π *be a PRE scheme that is* γ*-spread [7], fulfills the perfect key-switching property and is* $\mathsf{IND\text{-}CCA}_{0,1}$*-secure.*

Let Π' be the resulting scheme after applying the Fujisaki-Okamoto transformation for proxy re-encryption. Then, Π' is IND-CCA$_{2,1}$-secure in the random oracle model.

The proof for Theorem 1 is presented below. It basically consists on a reduction from the IND-CCA$_{0,1}$ adversary to the IND-CCA$_{2,1}$ adversary: if the underlying scheme is IND-CCA$_{0,1}$-secure, then the transformed scheme must be IND-CCA$_{2,1}$-secure.

Proof. Here we prove that if proxy re-encryption scheme Π is secure in the IND-CCA$_{0,1}$ sense, then the scheme Π^{hyb}, which results from applying the Fujisaki-Okamoto transformation to Π as described in Section 3.3.1, is IND-CCA$_{2,1}$-secure in the random oracle model. In order to do so, we show how to use an IND-CCA$_{2,1}$ adversary that breaks Π^{hyb} to create an IND-CCA$_{0,1}$ adversary that breaks Π.

Let $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ be an IND-CCA$_{2,1}$ adversary attacking Π^{hyb}; that is, it wins the IND-CCA$_{2,1}$ game with non-negligible advantage ε^{hyb}. We want to show that it is possible to construct an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ attacking Π that wins the IND-CCA$_{0,1}$ security game with also non-negligible advantage. To this end, we follow a strategy similar to Shoup and Gennaro's proof for the TDH1 cryptosystem [31]: if the IND-CCA$_{2,1}$ adversary wins, then he must have queried any of the random oracles with the same input used to create the challenge; the simulation is maintained up to this point, but after that, it is no longer necessary since we already are able of producing the response for the IND-CCA$_{0,1}$ adversary.

Adversaries $\mathcal{A}$ and $\mathcal{B}$ have access to the oracles that correspond to their respective attack models. In short, $\mathcal{B}$ has access to all oracles, except to the re-encryption oracle in phase 2 (which corresponds to the CCA$_{2,1}$ attack model), whereas $\mathcal{A}$ only has access to the key generation oracles and the re-encryption oracle in phase 1 (which corresponds to the CCA$_{0,1}$ attack model). Note also that $\mathcal{B}$ has access to random oracles H and G. We assume that $\mathcal{B}$ makes at most q_{dec} to the decryption oracle.

Algorithm 1 shows how adversary $\mathcal{A}$ can be constructed using $\mathcal{B}$. Basically, $\mathcal{A}$ simulates the view of the IND-CCA$_{2,1}$ game to $\mathcal{B}$, using his own challenge to construct $\mathcal{B}$'s challenge. Since he does not know m_δ, he randomly chooses m_β for the rest of the challenge ciphertext. Therefore, half of the times, $\mathcal{B}$'s view is correct and $\mathcal{A}$ uses it for deciding δ.

Consequently, $\mathcal{A}$ should simulate the view of $\mathcal{B}$, that is, it should answer $\mathcal{B}$'s oracle queries, including the random oracles. The latter are simulated by $\mathcal{A}$ so that their output is generated on-demand. This corresponds to the typical intuition of the random oracle: a function that returns a randomly assigned output for each possible input, sampled uniformly from its output domain. $\mathcal{A}$ maintains a list of tuples for each random oracle, where each tuple contains the input and output of a query: $\mathcal{L}_H$ is the list for oracle H and $\mathcal{L}_G$ the list for oracle G. Key generation oracle queries (i.e., $\mathcal{O}_{corrupt}, \mathcal{O}_{honest}$ and $\mathcal{O}_{rkgen}$) are trivially answered by relaying them to $\mathcal{A}$'s oracles.

All that remains is to tackle with decryption and re-encryption queries. Algorithm 2 shows the algorithms that simulate $\mathcal{B}$'s view of the decryption and re-encryption oracles. Recall that, according to the CCA$_{2,1}$ attack model, the decryption oracle is available in both phases, whilst the re-encryption oracle is

`Algorithm` $A_1(pk^*)$
  $(x_0, x_1, s_B) = \mathcal{B}_1(pk^*)$
  Sample random $m_0, m_1 \in \mathcal{M}^{pre}$
  $s_A = (x_0, x_1, s_B)$
  Return (m_0, m_1, s_A)

`Algorithm` $A_2(m_0, m_1, s_A, e^*)$
  Parse s_A as (x_0, x_1, s_B)
  Sample random $\beta, \mu \in \{0, 1\}$
  $c^* = x_\beta \oplus G(m_\beta)$
  Execute $\mathcal{B}_2(x_0, x_1, s_B, (e^*, c^*))$
  If $\mathcal{B}_2$ aborted, then return β
  Else, return μ

Algorithm 1: Adversary $\mathcal{A}$

`Algorithm` $\mathcal{O}^B_{dec}(pk_i, (e, c))$
  Search $(\sigma, c, h) \in \mathcal{L}_H$,
    such that $e = \mathsf{PRE.Enc}(pk_i, \sigma, h)$
  If such tuple does not exist, then return $\perp$
  Return $c \oplus G(\sigma)$

`Algorithm` $\mathcal{O}^B_{reenc}(pk_i, pk_j, (e, c))$
  $e' = \mathcal{O}^A_{reenc}(pk_i, pk_j, e)$
  If $e' = \perp$, then return $\perp$
  Else, return (e', c)

Algorithm 2: Decryption and re-encryption oracles for $\mathcal{B}$

only available in phase 1. This definition of $\mathcal{B}$' decryption oracle is essentially the same than the one found in Fujisaki and Okamoto's proof [7] and does not need any secret keys. The perfect key-switching property of the underlying PRE scheme guarantees that the simulation of this oracle is correct, even when the input ciphertext is a re-encryption. Note also that it is possible that $\mathcal{B}$ submits a valid ciphertext without having used the random oracles, so it gets wrongfully rejected. This event is called Bad in Fujisaki and Okamoto's proof; assuming that the underlying PRE scheme is γ-spread, the probability of this event is bound by $q_{dec} \cdot 2^{-\gamma}$.

Similarly, $\mathcal{B}$' re-encryption oracle does not require re-encryption keys, but relies on $\mathcal{A}$'s re-encryption oracle, which is provided in the $\mathsf{IND\text{-}CCA}_{0,1}$ security game. The simulation of this oracle is perfect. Key generation oracles are also perfectly simulated, since they depend exclusively on $\mathcal{A}$'s oracles.

The simulation of the random oracles is perfect, until $\mathcal{B}$ queries $G(m_\beta)$ or $H(m_\beta, \cdot)$. If this event occurs, $\mathcal{A}$ aborts the simulation and outputs β, following the strategy mentioned at the beginning. Otherwise, the simulation continues until $\mathcal{B}$ halts, and $\mathcal{A}$ outputs a random bit μ.

Assuming that Bad does not occur, it can be seen that when $\beta = \delta$ in $\mathcal{A}_2$, which happens with probability $\frac{1}{2}$, the challenge ciphertext is identically distributed as the one defined by the security game; in this case, $\mathcal{B}$ should win the game with probability $\frac{1}{2} + \varepsilon^{hyb}$, and therefore, should query $G(m_\beta)$ or $H(m_\beta, \cdot)$ with the same probability. Therefore, $\mathcal{A}$ wins the game with probability $\frac{1}{2} + \varepsilon^{hyb}$ too. On the contrary, when $\beta \neq \delta$, then $\mathcal{B}$ does not have any advantage for distinguishing the challenge, since x_β is information-theoretically hidden, so $\mathcal{A}$ wins the game with probability $\frac{1}{2}$. Therefore, the success probability of $\mathcal{A}$ under this assumption is:

$$Pr[\delta = \delta' | \neg \mathsf{Bad}] = \frac{1}{2} \cdot (\frac{1}{2} + \varepsilon^{hyb}) + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} + \frac{\varepsilon^{hyb}}{2}$$

Finally, taking into consideration the probability that Bad occurs, the overall success probability and advantage of adversary $\mathcal{A}$ are, respectively:

$$\begin{aligned} Pr[\delta = \delta'] &\geq Pr[\delta = \delta' | \neg \mathsf{Bad}] Pr[\neg \mathsf{Bad}] \\ &\geq (\frac{1}{2} + \frac{\varepsilon^{hyb}}{2}) \cdot (1 - q_{dec} \cdot 2^{-\gamma}) \end{aligned}$$

$$\varepsilon^{pre} \geq \frac{\varepsilon^{hyb}}{2} - \frac{(1 + \varepsilon^{hyb}) \cdot q_{dec}}{2^{\gamma+1}}$$

□

An interesting question is why not aim for $\mathsf{IND\text{-}CCA}_{2,2}$ security. The answer is that the extended transformation, as is, is vulnerable to certain type of attacks that use the re-encryption oracle after the challenge. Since the re-encryption function simply re-encrypts the asymmetric part, it cannot verify whether the symmetric part is valid or not. For example, suppose that the adversary takes the challenge ciphertext (e^*, c^*), produces $(e^*, c^* \oplus K)$ for some value $K \in \mathcal{C}^{sym}$, and asks for the re-encryption of the resulting ciphertext from the target user to a corrupt one. This query is legal because $(e^*, c^* \oplus K)$ is, technically, not a

Table 1: Performance of the transformation

Scheme	Original	Extended
Encryption	t_{enc}	$t_{enc} + 2 \cdot t_H$
Re-encryption	t_{reenc}	t_{reenc}
Decryption	t_{dec}	$t_{dec} + t_{enc} + 2 \cdot t_H$

derivative of the challenge ciphertext. Now the adversary only has to decrypt the response, once he removes the mask K.

As we have discussed before, the difficulty of constructing the re-encryption oracle in the security proof constrains both the security requirements and expectations of this transformation. An interesting possibility is to modify the transformation so as to be able to construct the security proof without these constraints. To this end, an option is to use some kind of non-interactive zero-knowledge (NIZK) proof in order to extract the randomness used during encryption, as well as to sign the rest of the ciphertext. However, because of the rewinding problems that appear in the security proof, which make it run in exponential time, this is not possible in general. To solve this, Canard et al. propose in [32] the use of a NIZK proof with online extractor (NIZKOE) as a way to patch the flawed scheme from Chow et al. [23]. In this paper, however, we focus only on what it can be achieved without extending the transformation with new primitives.

3.3.4 Performance of the transformation

Table 1 shows a theoretical comparison of the computational costs of a obtained scheme after the transformation with respect to the original one. Let t_{enc}, t_{reenc}, and t_{dec} denote the computational cost of the original scheme for the encryption, re-encryption and decryption operation, respectively. Note that we are still assuming that the one-time pad is used as the symmetric encryption, in accordance to the proved transformation. It should be observed that only the decryption procedure has a perceptible overhead, which depends exclusively on the cost of the original encryption operation (represented by t_{enc}), since the cost associated to the hash functions is negligible in practice.

4 Applying the proposed transformation

For illustration purposes, in this section we present a PRE scheme that fulfills the conditions of our transformation. That is, this scheme satisfies the perfect key-switching property and is secure in the sense of $\mathsf{IND\text{-}CCA}_{0,1}$, assuming the 3-wDBDHI problem [3] is hard. We later apply our extended Fujisaki-Okamoto transformation to obtain a $\mathsf{IND\text{-}CCA}_{2,1}$-secure scheme.

4.1 The original scheme

This scheme is based on a scheme from Ateniese et al. [16] that lacked a security proof. In order to prove that this scheme is $\mathsf{IND\text{-}CCA}_{0,1}$ secure, we borrowed

some ideas from the RCCA secure scheme from Libert and Vergnaud [3]. The scheme is as follows:

- $\mathsf{Setup}(\lambda)$: The setup algorithm first determines the cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of order q, with q a prime of λ bits, and a bilinear pairing e, so that $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. A set of generators $g, u, v \in \mathbb{G}$ is chosen randomly, and $Z = e(g, g)$. A function $F : \mathbb{Z}_q \to \mathbb{G}$ is defined as $F(t) = u^t \cdot v$. The global parameters are represented by the tuple:

$$params = (\mathbb{G}, \mathbb{G}_T, e, g, Z, u, v, F(\cdot))$$

- $\mathsf{KeyGen}(\lambda)$: Sample a random $x_i \in \mathbb{Z}_q$, and compute the public and private key of user i as $pk_i = g^{x_i}$ and $sk_i = x_i$.

- $\mathsf{ReKeyGen}(sk_i, pk_j)$: The re-encryption key from user i to user j is computed as

$$rk_{i \to j} = (pk_j)^{1/sk_i} = g^{x_j/x_i}$$

- $\mathsf{Enc}_2(pk_i, m)$: Sample random $r_1, r_2 \in \mathbb{Z}_q$. The second-level encryption of m under pk_i is the tuple $CT_i = (r_1, C_0, C_1, C_2)$, where

$$C_0 = F(r_1)^{r_2} \qquad C_1 = Z^{r_2} \cdot m$$
$$C_2 = (pk_i)^{r_2} = g^{x_i r_2}$$

- $\mathsf{Enc}_1(pk_i, m)$: The first-level encryption of m under pk_i is exactly as the second-level, except that $C_2 = e(g, pk_i)^{r_2} = Z^{x_i r_2}$.

- $\mathsf{ReEnc}(rk_{i \to j}, CT_i = (r_1, C_0, C_1, C_2))$: Check that the condition $e(C_0, pk_i) = e(C_2, F(r_1))$ holds; otherwise, output $\perp$. The re-encryption of the second-level ciphertext CT_i is the first-level ciphertext $CT_j = (r_1, C_0, C_1, C_2')$, where $C_2' = e(C_2, rk_{i \to j}) = Z^{x_j r_2}$.

- $\mathsf{Dec}_1(sk_i, CT_i = (r_1, C_0, C_1, C_2))$: Given a first-level ciphertext CT_i, the original message is computed as

$$m = \frac{C_1}{C_2^{1/sk_i}}$$

- $\mathsf{Dec}_2(sk_i, CT_i = (r_1, C_0, C_1, C_2))$: Given a second-level ciphertext CT_i, the original message is computed as

$$m = \frac{C_1}{e(g, C_2)^{1/sk_i}}$$

The first important characteristic of this scheme is that it fulfills the perfect key-switching property, since the original randomness used in second-level ciphertexts, namely r_1 and r_2, is preserved after re-encryption, so a re-encrypted ciphertext is equal to a first-level ciphertext encrypted with the same randomness. Let us express this formally, according to Definition 3. First, note that in this case the message space $\mathcal{M}$ is the cyclic group $\mathbb{G}_T$, and the random coins space $\mathcal{R}$ is $\mathbb{Z}_q \times \mathbb{Z}_q$, since the encryption function samples two independent random values from $\mathbb{Z}_q$; For the sake of clarity, let us define $r = (r_1, r_2)$ so as to

represent the two random values by a single element. Then, for all messages $m \in \mathbb{G}_T$, randomness $r \in \mathbb{Z}_q \times \mathbb{Z}_q$ and public keys pk_i, pk_j generated by KeyGen, it holds that:

$$\mathsf{ReEnc}(rk_{i\to j}, \mathsf{Enc}_2(pk_i, m; r)) = \mathsf{Enc}_1(pk_j, m; r)$$

It can be seen that this is true since the re-encrypted ciphertext is computed with the very same operations than the first-level ciphertext, except for element C_2. The C_2 component of the first-level encryption under pk_j is computed as $C_2 = e(g, pk_j)^{r_2} = Z^{x_j r_2}$, while the corresponding component in the re-encrypted ciphertext is $C_2' = e((pk_i)^{r_2}, rk_{i\to j}) = e(g^{x_i r_2}, g^{x_j/x_i}) = Z^{x_j r_2}$, which yields the same result. Therefore, this scheme satisfies the perfect key-switching property.

In addition, it can be seen that it is also well-spread, since for any message m there can be $q^2 \approx 2^{2\lambda}$ different ciphertexts: component r_1 is sampled randomly from $\mathbb{Z}_q$, and the rest of the components resemble an ElGamal ciphertext, for random $r_2 \in \mathbb{Z}_q$.

4.2 Proving IND-CCA$_{0,1}$ security

In this section we prove that the scheme is secure under the IND-CCA$_{0,1}$ notion, assuming the hardness of the 3-wDBDHI problem. We use an alternative definition of this hard problem, due to Libert and Vergnaud [3].

Definition 4 (3-wDBDHI problem)**.** *Given a tuple $(g, g^a, g^{a^2}, g^{1/a}, g^b, e(g,g)^d)$, the 3-weak Decisional Bilinear DH Inversion problem (3-wDBDHI) in $(\mathbb{G}, \mathbb{G}_T)$ is to decide whether $d = b/a^2$.*

The proof consists on a reduction from the 3-wDBDHI problem to the IND-CCA$_{0,1}$ security of the scheme. Suppose that the scheme is not IND-CCA$_{0,1}$ secure, then there is an adversary $\mathcal{B}$ that wins the IND-CCA$_{0,1}$ game with non-negligible advantage ε, so its success probability is $\frac{1}{2}+\varepsilon$. From this adversary, we can construct an algorithm $\mathcal{A}$ that solves the 3-wDBDHI problem with probability $\frac{1}{2} + \frac{\varepsilon}{2} - q_{reenc} \cdot 2^{-\lambda-1}$, where q_{reenc} is the number of queries to the re-encryption oracle made by $\mathcal{B}$.

$\mathcal{A}$ receives as input a tuple $(g, g^a, g^{a^2}, g^{1/a}, g^b, Z^d)$ from the 3-wDBDHI problem, and uses it to simulate the environment for the adversary $\mathcal{B}$. First, $\mathcal{A}$ samples random $r_1^*, \alpha_1, \alpha_2 \in \mathbb{Z}_q$, and sets $u = (g^a)^{\alpha_1}$ and $v = (g^a)^{-r_1^*\alpha_1}(g^{a^2})^{\alpha_2}$, so $F(t) = g^{a\alpha_1(t-r_1^*)}g^{a^2\alpha_2}$. Note that $F(r_1^*) = g^{a^2\alpha_2}$; we will use this later in the proof.

The public key of the target user is set as $pk^* = g^{a^2}$. For honest users, $\mathcal{A}$ samples random $w_i \in \mathbb{Z}_q$ and sets $pk_i = g^{aw_i}$. For corrupt users, $\mathcal{A}$ simply runs the key generation algorithm and returns the resulting public and private key pair $(pk_i = g^{x_i}, sk_i = x_i)$.

Re-encryption keys $rk_{i\to j}$ are generated as follows:

- Honest user to target user: $rk_{i\to *} = (g^a)^{1/w_i}$
- Target user to honest user: $rk_{*\to j} = (g^a)^{w_j}$
- Honest user to honest user: $rk_{i\to j} = g^{w_j/w_i}$

- Honest user to corrupt user: $rk_{i\to j} = (g^{1/a})^{x_j/w_i}$
- Corrupt user to another user: $rk_{i\to j} = (pk_j)^{1/x_i}$

Since the attack model is $\mathsf{CCA}_{0,1}$, it is only necessary to simulate the re-encryption oracle in Phase 1 (i.e., before the challenge). It is possible to simulate correctly all possible re-encryption queries $\mathcal{O}_{reenc}(pk_i, pk_j, CT_i)$ using the re-encryption keys described, except for the case when $pk_i = pk^*$ and user j is corrupt. These queries are solved as below, without requiring the corresponding re-encryption key. Since $pk_i = pk^*$, the ciphertext CT_i is a tuple $(r_1, C_0, C_1, C_2) = (r_1, F(r_1)^{r_2}, m \cdot Z^{r_2}, (pk^*)^{r_2}) = (r_1, (g^{a\alpha_1(r_1 - r_1^*)} g^{a^2\alpha_2})^{r_2}, m \cdot Z^{r_2}, g^{a^2 r_2})$. In order to compute the re-encryption, the challenger produces the auxiliar value g^{ar_2} in the following way:

$$g^{ar_2} = \frac{C_0}{C_2^{\alpha_2}}^{\frac{1}{(r_1 - r_1^*)\alpha_1}} = \left(\frac{F(r_1)}{g^{a^2\alpha_2}}\right)^{\frac{r_2}{(r_1 - r_1^*)\alpha_1}}$$

Once $\mathcal{A}$ computes g^{ar_2}, the re-encryption of C_2 (which, as mentioned before, is the only component of the ciphertext that changes after re-encryption) is as follows:

$$C_2' = e((g^{1/a})^{x_j}, g^{ar_2}) = Z^{x_j r_2}$$

Note that this procedure is correct except for the case then $r_1 = r_1^*$. Since r_1^* is not disclosed by $\mathcal{A}$ in Phase 1, and r_1 is sampled randomly, the event that $r_1 = r_1^*$ can happen only with negligible probability $1/q \approx 2^{-\lambda}$, for each re-encryption query. We assume that the adversary $\mathcal{B}$ can make up to q_{reenc} queries, so the overall probability of this type of events is $q_{reenc} \cdot 2^{-\lambda}$.

The challenge ciphertext is constructed as:

$$CT^* = (r_1^*, (g^b)^{\alpha_2}, m_\delta \cdot Z^d, g^b)$$

$\mathcal{A}$ runs adversary $\mathcal{B}$ to obtain the guess δ' and decides that $d = b/a^2$ when $\delta = \delta'$. Note that when $d = b/a^2$, the challenge ciphertext is a valid encryption of m_δ under pk^*, where the random exponent r_2 is defined implicitly as $r_2 = b/a^2$.

$$CT^* = (r_1^*, F(r_1^*)^{b/a^2}, m_\delta \cdot Z^{b/a^2}, (g^{a^2})^{b/a^2})$$

Therefore, in this case $\mathcal{B}$ guesses δ correctly with probability $\frac{1}{2} + \varepsilon - q_{reenc} \cdot 2^{-\lambda}$ and $\mathcal{A}$ solves the 3-wDBDHI problem with the same probability. On the contrary, when d is random, m_δ is information-theoretically hidden, so the probability of $\mathcal{B}$ guessing δ correctly is $\frac{1}{2}$. The overall success probability of $\mathcal{A}$ is then $\frac{1}{2} + \frac{\varepsilon}{2} - q_{reenc} \cdot 2^{-\lambda - 1}$.

4.3 A IND-CCA$_{2,1}$-secure PRE scheme

Since the previous scheme is IND-CCA$_{0,1}$-secure, satisfies the perfect key-switching property and is well-spread, then it can be extended for IND-CCA$_{2,1}$ security using our transformation. The resulting scheme is very similar, except that now r_1 and r_2 are not chosen randomly, but using a hash function H. Recall that we use the one-time pad as the symmetric encryption scheme. The scheme is as follows:

- $\mathsf{Setup}(\lambda)$: The setup algorithm first determines the cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of order q, with q a prime of λ bits, and a bilinear pairing e, so that $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. A set of generators $g, u, v \in \mathbb{G}$ is chosen randomly, and $Z = e(g, g)$. Let ℓ be the length of the one-time pad, which depends on the security parameter λ. Let us also define hash functions $H : \mathbb{G}_T \times \{0,1\}^\ell \to \mathbb{Z}_q \times \mathbb{Z}_q$ and $G : \mathbb{G}_T \to \{0,1\}^\ell$. A function $F : \mathbb{Z}_q \to \mathbb{G}$ is defined as $F(t) = u^t \cdot v$.

  The global parameters are represented by the tuple:

$$params = (\mathbb{G}, \mathbb{G}_T, e, g, Z, u, v, \ell, F(\cdot), H(\cdot), G(\cdot))$$

- $\mathsf{KeyGen}(\lambda)$: Sample a random $x_i \in \mathbb{Z}_q$, and compute the public and private key of user i as $pk_i = g^{x_i}$ and $sk_i = x_i$.

- $\mathsf{ReKeyGen}(sk_i, pk_j)$: The re-encryption key from user i to user j is computed as

$$rk_{i \to j} = (pk_j)^{1/sk_i} = g^{x_j/x_i}$$

- $\mathsf{Enc}_2(pk_i, m)$: Sample random $\sigma \in \mathbb{G}_T$ and compute $C_3 = G(\sigma) \oplus m$. Next, the randomness used for encryption is produced as $(r_1, r_2) = H(\sigma, C_3)$. The second-level encryption of m under pk_i is the tuple $CT_i = (r_1, C_0, C_1, C_2, C_3)$, where

$$C_0 = F(r_1)^{r_2} \qquad C_1 = Z^{r_2} \cdot \sigma$$
$$C_2 = (pk_i)^{r_2} = g^{x_i r_2}$$

- $\mathsf{Enc}_1(pk_i, m)$: The first-level encryption of m under pk_i is exactly as the second-level, except that $C_2 = e(g, pk_i)^{r_2} = Z^{x_i r_2}$.

- $\mathsf{ReEnc}(rk_{i \to j}, CT_i = (r_1, C_0, C_1, C_2, C_3))$: Check that the condition $e(C_0, pk_i) = e(C_2, F(r_1))$ holds; otherwise, output the error symbol $\perp$. The re-encryption of the second-level ciphertext CT_i is the first-level ciphertext $CT_j = (r_1, C_0, C_1, C_2', C_3)$, where $C_2' = e(C_2, rk_{i \to j}) = Z^{x_j r_2}$.

- $\mathsf{Dec}_1(sk_i, CT_i = (r_1, C_0, C_1, C_2))$: Given a first-level ciphertext CT_i, first decrypt σ as $\sigma = \frac{C_1}{C_2^{1/sk_i}}$, and use it to extract the original randomness $(r_1, r_2) = H(\sigma, C_3)$. Next, check that equations $F(r_1)^{r_2} = C_0$, $Z^{r_2} \cdot \sigma = C_1$, and $e(g, pk_i)^{r_2} = C_2$ hold and return $m = G(\sigma) \oplus C_3$; otherwise, return $\perp$.

- $\mathsf{Dec}_2(sk_i, CT_i = (r_1, C_0, C_1, C_2))$: Given a second-level ciphertext CT_i, first decrypt σ as $\sigma = \frac{C_1}{e(g, C_2)^{1/sk_i}}$, and use it to extract the original randomness $(r_1, r_2) = H(\sigma, C_3)$. Next, check that $F(r_1)^{r_2} = C_0$, $Z^{r_2} \cdot \sigma = C_1$, and $(pk_i)^{r_2} = C_2$ hold and return $m = G(\sigma) \oplus C_3$; otherwise, return $\perp$.

According to Theorem 1, this scheme is $\mathsf{IND\text{-}CCA}_{2,1}$-secure in the random oracle model.

5 Towards Other Generic Transformations for Proxy Re-Encryption

It is worthwhile thinking about other possible generic transformations for achieving strong security notions in the context of proxy re-encryption. In Section 3, we have shown how to directly apply the Fujisaki-Okamoto generic transformation. However, this transformation has two main drawbacks:

- During the decryption procedure it is necessary to encrypt again the ciphertext with the PRE scheme to check if it is the same as the one received. This produces a computational overhead in the decryption, as discussed in Section 3.3.4.
- The underlying PRE scheme has to satisfy the perfect key-switching property. This is not always the case, since often the re-encryption function leaves "remnants" of the previous public key, which would make the decryption check to fail inevitably.

It is desirable to come up with other transformations that achieve a strong security notion without requiring the perfect key-switching property and the overhead produced by re-computing the ciphertext during decryption. The latter problem is solved in the PKE context by some transformations such as REACT [8] and GEM [9]. The security of these transformations is based on including a hash of the ciphertext that acts as a checksum. For example, in the REACT transformation, the output of the encryption $\mathsf{Hyb.Enc}(pk, m)$ is a ciphertext of the form:

$$(\underbrace{\mathsf{PKE.Enc}(pk, \sigma)}_{e}, \underbrace{\mathsf{Sym.Enc}(G(\sigma), m)}_{c}, \underbrace{H(\sigma, m, e, c)}_{h})$$

The decryption process is simple. It first deciphers σ from e; next, it extracts m from c using the key $G(\sigma)$; and, finally, it verifies that $H(\sigma, m, e, c) = h$.

As in the case of the Fujisaki-Okamoto transformation, directly applying REACT to a PRE scheme would not work in general, since the re-encryption process invalidates the validity check. This check is based on a hash of rest of ciphertext, so a re-encrypted ciphertext will not produce the same hash value h. However, we could modify the transformation so the hash does not include the component e, which is altered during the re-encryption. Then, ciphertexts would be of the form:

$$(\underbrace{\mathsf{PRE.Enc}(pk, \sigma)}_{e}, \underbrace{\mathsf{Sym.Enc}(G(\sigma), m)}_{c}, \underbrace{H(\sigma, m, c)}_{h})$$

It can be seen that the only difference with respect the original transformation is in the component h. The re-encryption process would be similar to our Fujisaki-Okamoto extension, re-encrypting the component e with the underlying PRE scheme:

$$\mathsf{Hyb.ReEnc}(rk, (e, c, h)) = (\underbrace{\mathsf{PRE.ReEnc}(rk, e)}_{e'}, c, h)$$

Assuming the PRE scheme is correct, the decryption process will work too. Decrypting e' will output σ, so the process remains the same, except for the checksum h, which now does not consider the e' term.

The modification we introduced in this transformation can be seen as a relaxation of the validity check that takes place during the decryption, since we change it from $H(\sigma, m, e, c)$ to $H(\sigma, m, c)$. Dropping the term e from the checksum implies that we are not concerned anymore with possible alterations on this part; however, the checksum still contains the original message m, its encryption c, and the term σ. This prompt us to analyze what alterations are possible in the term e so that the decryption is still correct.

It is clear that σ cannot be altered, since this would imply not being able to recover the key for extracting m from c, so the only options are changing the public key (which is precisely the goal of PRE) or the original random coins used (which is what happens in PRE schemes that do not satisfy the perfect key-switching property). Therefore, any ciphertext that decrypts to the original message would be deemed valid, regardless of being altered (e.g., by re-encryption).

This idea corresponds precisely to the notion of *Replayable CCA* (RCCA) from [33], a relaxation of CCA that can be considered sufficiently secure for many existing applications, and indeed, is the target security notion for some PRE schemes, such as the one from Libert and Vergnaud [3]. Note that RCCA, as opposed to CCA, allows any possible kind of alterations, even those not related to the re-encryption of ciphertexts, as long as it decrypts to the original message.

It would appear, then, that a modified REACT transformation could be defined for PRE, achieving a security notion similar to RCCA. However, some of the problems mentioned in Section 3.3.3 also arise here: it is still not possible to perform the validation step during re-encryption, since the hash input contains information that is hidden during this process, and the construction of the re-encryption oracle in the security proof cannot be based on the random oracle tables either. This poses similar challenges to the Fujisaki-Okamoto case when it comes to proving the security of the transformation.

Moreover, REACT puts additional restrictions on the underlying asymmetric scheme. In particular, it requires the scheme to be *one-way secure under plaintext-checking attacks* (OW-PCA), in order to enable the construction of the decryption oracle in the security proof. Informally, this notion means that the scheme must not allow the adversary to recover the plaintext from a given ciphertext, even if she has access to a plaintext-checking oracle, that is able to tell whether, for input (m, c), the ciphertext c is an encryption of the plaintext m. An implication of this requirement is that, when considering a PCA adversary, the hardness assumption may vary with respect to traditional attack models, such as CPA. For instance, the ElGamal encryption scheme, which is OW-CPA secure under the Computational Diffie-Hellman (CDH) assumption, is OW-PCA secure under the Gap Diffie-Hellman (gap-DH) assumption [8]. The explanation of this change is that the PCA oracle usually is equivalent to a solver of a hard decisional problem (e.g., the PCA oracle for ElGamal is equivalent to a DDH oracle). Therefore, the security of the scheme under OW-PCA must be based on a hardness assumption that still holds when the adversary has access to an oracle of some hard decisional problem (i.e., the PCA oracle). Some computational problems are still hard when one has a solver for the decisional version,

and this is precisely the essence behind the concept of "gap problems", defined by Okamoto and Pointcheval in [34].

In the same way, the extension of REACT to PRE would require the underlying PRE schemes to be OW-PCA. This is the case of several PRE schemes. For instance, it can be shown that the BBS scheme [18] is OW-PCA-secure under the gap-DH assumption, as ElGamal. However, certain kinds of schemes, such as those based on the Learning With Errors (LWE) assumption [17, 35, 36, 37], cannot be proven OW-PCA-secure since the decisional and computational version of LWE are equivalent, as pointed out by Peikert in [38].

It is also worth mentioning that, in order to be able to define the re-encryption oracle, the security proof for this modification of REACT seems to require to take the form of a reduction to the security of the underlying scheme under a notion that provides a re-encryption oracle. For the Fujisaki-Okamoto case, we required IND-CCA$_{0,1}$ so as to simplify the security proof, but in this case OW-CCA$_{0,1}$ is a weaker requirement. Therefore, the underlying scheme should satisfy the notion of one-wayness under an attack model that combines PCA and CCA$_{0,1}$.

The resulting transformation would achieve an intermediate notion between RCCA and IND-CCA$_{2,1}$, and may be applicable to a wider class of PRE schemes than the Fujisaki-Okamoto extension, since it does not require the schemes to satisfy the perfect key-switching property. However, it is an open issue to analyze these ad-hoc security definitions in detail, in order to provide a complete proof of the security of this potential generic transformation for PRE.

Finally, it is worth mentioning that there is at least another generic transformation, to which the same argumentation seems to apply. The GEM transformation [9] is very similar to REACT, but with a more involved construction that achieves shorter ciphertexts. In GEM, ciphertexts are of the form:

$$(\underbrace{\mathsf{PKE.Enc}(pk, \sigma)}_{e}, \underbrace{\mathsf{Sym.Enc}(G(\sigma, e), m)}_{c})$$

where r is a random term, $s = F(m, r)$, and $\sigma = s||(r \oplus H(s))$. In order to decrypt a ciphertext (e, c), it first deciphers σ from e, and extracts m from c using the key $G(\sigma, e)$; next, it parses $s||t$ from σ and computes $r = t \oplus H(s)$; and, finally, it verifies whether $F(m, r) = s$.

Being similar to REACT, the strategy for modifying GEM for proxy re-encryption would be similar too. The part of the ciphertext produced by the underlying PRE scheme would not be used as input to the hash function G, so the re-encryption procedure does not break the validation. Thus, the encryption is modified for producing ciphertexts of the form:

$$(\underbrace{\mathsf{PRE.Enc}(pk, \sigma)}_{e}, \underbrace{\mathsf{Sym.Enc}(G(\sigma), m)}_{c})$$

Therefore, re-encryption is identical to that of the Fujisaki-Okamoto extension:

$$\mathsf{Hyb.ReEnc}(rk, (e, c)) = (\underbrace{\mathsf{PRE.ReEnc}(rk, e)}_{e'}, c)$$

The arguments regarding the security of this extension of GEM are very similar to those for the extension of REACT.

6 Conclusions

In this paper we analyze the integration of generic transformations to proxy re-encryption and find both negative and positive results. On the one hand, we first describe why it is not possible to directly integrate known transformations, such as Fujisaki-Okamoto and REACT, with weakly-secure PRE schemes due to general obstacles coming from both the constructions themselves and the security models, and we show twelve PRE schemes that exhibit these problems. These transformations are artifacts conceived for securing public-key encryption schemes, and cannot be used as is for proxy re-encryption due to the special nature of the re-encryption capability. On the other hand, we also show that, under some conditions that include the satisfaction of a new property of PRE called "*perfect key-switching*", the Fujisaki-Okamoto transformation can be used to generically bootstrap a weak notion of security ($\mathsf{IND\text{-}CCA}_{0,1}$) into a much stronger notion ($\mathsf{IND\text{-}CCA}_{2,1}$), in the random oracle model. However, to achieve full CCA-security (i.e., $\mathsf{IND\text{-}CCA}_{2,2}$), it appears to be necessary to apply ad-hoc modifications. For illustrating our proposal we present a PRE scheme that satisfies the conditions for applying the Fujisaki-Okamoto extension and show the resulting scheme after the transformation.

Other generic transformations for public-key encryption are also discussed for its application in proxy re-encryption. We show how the REACT and GEM transformations [8, 9] can be modified to support re-encryptions. Since the perfect key-switching property is no longer required, these proposals are potentially applicable to a wider class of schemes, which makes them very attractive. In addition, they are more efficient than the Fujisaki-Okamoto transformation, since they do not require to reconstruct the ciphertext during decryption. Note, however, that these transformations seem to require the original scheme to satisfy a combination of $\mathsf{OW\text{-}PCA}$ and $\mathsf{OW\text{-}CCA}_{0,1}$, while achieve an intermediate notion between Replayable CCA (RCCA) and $\mathsf{IND\text{-}CCA}_{2,1}$. It is an open issue to analyze these security definitions in detail, in order to provide a complete proof of the security of these constructions. This is left as future work.

Other future lines of research include working towards concrete estimations of the obtained security level of the extended Fujisaki-Okamoto transformation. Finally, all the transformations discussed here are defined for the random oracle model. It is an open problem to devise generic transformations that are valid in the standard model.

Acknowledgements

This work was partly supported by the Junta de Andalucía through the project FISICCO (P11-TIC-07223) and by the Spanish Ministry of Economy and Competitiveness through the PERSIST project (TIN2013-41739-R). The first author has been funded by a FPI fellowship from the Junta de Andalucía as part of the project PISCIS (P10-TIC-06334).

References

[1] Victor Shoup. *Why chosen ciphertext security matters.* IBM TJ Watson Research Center, 1998.

[2] R. Canetti and S. Hohenberger. Chosen-ciphertext secure proxy re-encryption. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 185–194. ACM, 2007.

[3] B. Libert and D. Vergnaud. Unidirectional chosen-ciphertext secure proxy re-encryption. *Information Theory, IEEE Transactions on*, 57(3):1786–1802, 2011.

[4] Jian Weng, Robert H Deng, Shengli Liu, and Kefei Chen. Chosen-ciphertext secure bidirectional proxy re-encryption schemes without pairings. *Information Sciences*, 180(24):5077–5089, 2010.

[5] J. Shao and Z. Cao. CCA-secure proxy re-encryption without pairings. In *Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography: PKC'09*, pages 357–376. Springer-Verlag, 2009.

[6] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Advances in Cryptology—CRYPTO'99*, pages 537–554. Springer, 1999.

[7] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of cryptology*, 26(1):80–101, 2013.

[8] Tatsuaki Okamoto and David Pointcheval. REACT: Rapid enhanced-security asymmetric cryptosystem transform. In *Topics in Cryptology—CT-RSA 2001*, pages 159–174. Springer, 2001.

[9] Jean-Sébastien Coron, Helena Handschuh, Marc Joye, Pascal Paillier, David Pointcheval, and Christophe Tymen. GEM: A generic chosen-ciphertext secure encryption method. In *Topics in Cryptology—CT-RSA 2002*, pages 263–276. Springer, 2002.

[10] David Nuñez, Isaac Agudo, and Javier Lopez. A parametric family of attack models for proxy re-encryption. In *Proceedings of the 28th IEEE Computer Security Foundations Symposium*, CSF'15, pages 290–301. IEEE Computer Society, 2015.

[11] Takashi Kitagawa, Peng Yang, Goichiro Hanaoka, Rui Zhang, Hajime Watanabe, Kanta Matsuura, and Hideki Imai. Generic transforms to acquire CCA-security for identity based encryption: The cases of fopkc and react. In *Information Security and Privacy*, pages 348–359. Springer, 2006.

[12] Zhaohui Cheng, Liqun Chen, Li Ling, and Richard Comley. General and efficient certificateless public key encryption constructions. In *Pairing-Based Cryptography–Pairing 2007*, pages 83–107. Springer, 2007.

[13] Yang Lu, Jiguo Li, and Junmo Xiao. Applying the Fujisaki-Okamoto conversion to certificate-based encryption. In *Electronic Commerce and Security, 2008 International Symposium on*, pages 296–300. IEEE, 2008.

[14] Jun Shao, Zhenfu Cao, and Peng Liu. SCCR: a generic approach to simultaneously achieve cca security and collusion-resistance in proxy re-encryption. *Security and Communication Networks*, 4(2):122–135, 2011.

[15] Goichiro Hanaoka, Yutaka Kawai, Noboru Kunihiro, Takahiro Matsuda, Jian Weng, Rui Zhang, and Yunlei Zhao. Generic construction of chosen ciphertext secure proxy re-encryption. In *Topics in Cryptology–CT-RSA 2012*, pages 349–364. Springer, 2012.

[16] G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Transactions on Information and System Security*, 9(1):1–30, 2006.

[17] Yoshinori Aono, Xavier Boyen, Le Trieu Phong, and Lihua Wang. Key-private proxy re-encryption under LWE. In *Progress in Cryptology–INDOCRYPT 2013*, pages 1–18. Springer, 2013.

[18] M. Blaze, G. Bleumer, and M. Strauss. Divertible protocols and atomic proxy cryptography. *Advances in Cryptology—EUROCRYPT'98*, pages 127–144, 1998.

[19] G. Ateniese, K. Benson, and S. Hohenberger. Key-private proxy re-encryption. *Topics in Cryptology–CT-RSA 2009*, pages 279–294, 2009.

[20] David Galindo, Sebastià Martín, Paz Morillo, and Jorge L Villar. Fujisaki-Okamoto hybrid encryption revisited. *International Journal of Information Security*, 4(4):228–241, 2005.

[21] Brent Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology–EUROCRYPT 2005*, pages 114–127. Springer, 2005.

[22] Jian Weng, Robert H Deng, Xuhua Ding, Cheng-Kang Chu, and Junzuo Lai. Conditional proxy re-encryption secure against chosen-ciphertext attack. In *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security*, pages 322–332. ACM, 2009.

[23] Sherman SM Chow, Jian Weng, Yanjiang Yang, and Robert H Deng. Efficient unidirectional proxy re-encryption. In *Progress in Cryptology–AFRICACRYPT 2010*, pages 316–332. Springer, 2010.

[24] Jian Weng, Yanjiang Yang, Qiang Tang, Robert H Deng, and Feng Bao. Efficient conditional proxy re-encryption with chosen-ciphertext security. In *Proceedings of the 12th International Conference on Information Security*, pages 151–166. Springer-Verlag, 2009.

[25] Chul Sur, Chae Duk Jung, Youngho Park, and Kyung Hyune Rhee. Chosen-ciphertext secure certificateless proxy re-encryption. In *Communications and Multimedia Security*, pages 214–232. Springer, 2010.

[26] Xiaoqi Jia, Jun Shao, Jiwu Jing, and Peng Liu. CCA-secure type-based proxy re-encryption with invisible proxy. In *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*, pages 1299–1305. IEEE, 2010.

[27] Liming Fang, Willy Susilo, Chunpeng Ge, and Jiandong Wang. Interactive conditional proxy re-encryption with fine grain policy. *Journal of Systems and Software*, 84(12):2293–2302, 2011.

[28] Jun Shao, Guiyi Wei, Yun Ling, and Mande Xie. Identity-based conditional proxy re-encryption. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1–5. IEEE, 2011.

[29] Liming Fang, Willy Susilo, Chunpeng Ge, and Jiandong Wang. Hierarchical conditional proxy re-encryption. *Computer Standards & Interfaces*, 34(4):380–389, 2012.

[30] Jun Shao. Anonymous ID-based proxy re-encryption. In *Information Security and Privacy*, pages 364–375. Springer, 2012.

[31] Victor Shoup and Rosario Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. *Journal of Cryptology*, 15(2):75–96, 2002.

[32] Sébastien Canard, Julien Devigne, and Fabien Laguillaumie. Improving the security of an efficient unidirectional proxy re-encryption scheme. *Jounal of Internet Services and Information Security*, pages pp140–160, 2011.

[33] Ran Canetti, Hugo Krawczyk, and Jesper B Nielsen. Relaxing chosen-ciphertext security. In *Advances in Cryptology-CRYPTO 2003*, pages 565–582. Springer, 2003.

[34] Tatsuaki Okamoto and David Pointcheval. The gap-problems: A new class of problems for the security of cryptographic schemes. In *Public Key Cryptography*, pages 104–118. Springer, 2001.

[35] Keita Xagawa and Keisuke Tanaka. Proxy re-encryption based on learning with errors. In *Proceedings of the 2010 Symposium on Cryptography and Information Security (SCIS 2010)*, 2010.

[36] Elena Kirshanova. Proxy re-encryption from lattices. In *Public-Key Cryptography–PKC 2014*, pages 77–94. Springer, 2014.

[37] David Nuñez, Isaac Agudo, and Javier Lopez. NTRUReEncrypt: An efficient proxy re-encryption scheme based on NTRU. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, ASIA CCS '15, pages 179–189, New York, NY, USA, 2015. ACM.

[38] Chris Peikert. Lattice cryptography for the internet. In *Post-Quantum Cryptography*, pages 197–219. Springer, 2014.

[39] R.H. Deng, J. Weng, S. Liu, and K. Chen. Chosen-ciphertext secure proxy re-encryption without pairings. In *Proceedings of the 7th International Conference on Cryptology and Network Security*, pages 1–17. Springer-Verlag, 2008.

[40] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In *Topics in Cryptology–CT-RSA 2011*, pages 319–339. Springer, 2011.

[41] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.

Appendices

A A Frequent Error in CCA-security Proofs in the Random Oracle model

In Section 3.3.3, we discussed the problems that arise in the security proof when generic transformations are directly applied in PRE, in particular when the re-encryption oracle is based on the random oracle tables. This solution is incorrect, as the security proof differs from the real execution for certain re-encryption queries, and this behavior can be forced arbitrarily by the adversary.

We surveyed the literature of CCA-secure PRE schemes in the random oracle model, looking for instances of this problem. Our study yielded eleven schemes [22, 5, 4, 23, 24, 25, 26, 27, 28, 29, 30], although this list might not be exhaustive.

Although these PRE schemes have different characteristics, such as being bidirectional [4], conditional [22] or identity-based [29], all of them share the same intrinsic problem that arises from a wrong integration of the Fujisaki-Okamoto transformation or other related techniques. It is important to note that this problem was first identified by Canard et al. in [32], but restricted to the analysis of scheme from Chow et al. [23].

The problem is based on two different issues. First, in the encryption function, secret values are used as input to a hash function to produce the randomness needed for encryption. For instance, in the original Fujisaki-Okamoto transformation [6], the session key σ and the original message m are used as this input: it is clear that m should not be disclosed during re-encryption; in the same manner, neither σ should be revealed, since this could be used to decrypt the message. However, this also means that in a real execution of the re-encryption function, it is not possible to verify that the hash function has been used correctly (i.e., (σ, m) was used as input to H).

Second, in the CCA attack model in PRE, queries of the form $(pk^*, pk_j, \hat{c})$, where user j is corrupt and $\hat{c}$ is not a derivative of the challenge, are legitimate and must be answered correctly. This requirement can be inferred from usual security models for CCA in proxy re-encryption, such as [2, 10], and in fact, represents a critical, yet often overlooked, point in most CCA security proofs. In particular, the simulation of these queries in the security proofs of the identified schemes relies on examining the random oracle tables, which contain the input and output of previous random oracle queries, as generally the challenger does not have access to the target secret key sk^*, which is necessary for generating the corresponding re-encryption key.

Taking both issues into consideration, let us suppose that the adversary creates an ill-formed ciphertext $\hat{c}$ under the target public key pk^*, where the randomness r is not derived from $H(\sigma, m)$, but chosen randomly. Next, the adversary calls the re-encryption oracle with the input $(pk^*, pk_j, \hat{c})$, for some corrupt user j. Note that this query is legal, since it is not related to the challenge ciphertext (in fact, it can occur in phase 1), and the simulator must answer it correctly since the adversary can check whether its decryption results in the same original message. The re-encryption strategy based on the random oracle tables does not work in this case, since the adversary did not use the random oracle, so the challenger cannot respond to this type of queries. The security proofs in the identified schemes opt to return $\perp$, indicating that the

ciphertext is invalid. However, this approach differs from the real execution, where the re-encryption function is unable to check whether the random oracle was used or not (i.e., that the ciphertext is valid or not), since the input to the random oracle should be hidden from the proxy; hence, the re-encryption will work normally. At this point, security proofs differ from the real execution with an arbitrary probability, since the adversary can force this behavior trivially. Consequently, these security proofs are invalid.

Let us illustrate this problem by analyzing one of the identified schemes. The PRE scheme from Weng et al. [4, 39] is bidirectional, single-hop, interactive and is allegedly CCA-secure in the random oracle model. Although this scheme does not use the Fujisaki-Okamoto transformation, it integrates a variation of the Hashed ElGamal scheme. Still, the same error in the security proof applies, since the randomness is generated from a value that must be kept hidden from the proxy (i.e., the original message m), which implies it can only be correctly verified during decryption, and not during re-encryption. The following is a slightly simplified version of the scheme:

- $\mathsf{Setup}(\lambda)$: The setup algorithm first determines the cyclic group $\mathbb{G}$ of order q, with q a prime of λ bits. A generator $g \in \mathbb{G}$ is chosen randomly. The message space is $\mathcal{M} = \{0,1\}^n$, where n is polynomial in the security parameter λ. It is also required a set of hash functions H_1, H_2 and H_3, where $H_1 : \{0,1\}^n \times \mathbb{G} \to \mathbb{Z}_q, H_2 : \mathbb{G} \to \{0,1\}^n$ and $H_3 : \mathbb{G}^2 \times \{0,1\}^n \to \mathbb{Z}_q$. The global parameters are represented by the tuple:

$$params = (q, \mathbb{G}, g, H_1, H_2, H_3)$$

- $\mathsf{KeyGen}(\lambda)$: Sample a random $x_i \in \mathbb{Z}_q$, and compute the public and private key of user i as $pk_i = g^{x_i}$ and $sk_i = x_i$.

- $\mathsf{ReKeyGen}(sk_i, sk_j)$: The re-encryption key from user i to user j is computed as $rk_{i \to j} = sk_j / sk_i$.

- $\mathsf{Enc}(pk_i, m)$: First, compute $r = H_1(m, pk_i)$. Next, sample random $u \in \mathbb{Z}_q$, and compute $D = (pk_i)^u$, $E = (pk_i)^r$, $F = H_2(g^r) \oplus m$, and $s = u + r \cdot H_3(D, E, F) \mod q$. The ciphertext is the tuple $CT_i = (D, E, F, s)$.

- $\mathsf{ReEnc}(rk_{i \to j}, CT_i = (D, E, F, s))$: First, check that the condition $(pk_i)^s = D \cdot E^{H_3(D,E,F)}$ holds; otherwise, output $\perp$. The re-encryption of ciphertext CT_i is the tuple $CT_j = (pk_i, E', F)$, where $E' = E^{rk_{i \to j}} = (pk_j)^r$.

- $\mathsf{Dec}_1(sk, CT = (pk_i, E', F))$: Given a first-level ciphertext CT, the original message is computed as $m = F \oplus H_2((E')^{1/sk})$. Finally, check that $E' = pk^{H_1(m, pk_i)}$ holds and return m; otherwise, return $\perp$.

- $\mathsf{Dec}_2(sk, CT = (D, E, F, s))$: Given a second-level ciphertext CT, first check that the condition $pk^s = D \cdot E^{H_3(D,E,F)}$ holds; otherwise, output $\perp$. The original message is computed as $m = F \oplus H_2(E^{1/sk})$. Finally, check that $E = pk^{H_1(m, pk)}$ holds and return m; otherwise, return $\perp$.

Note that the check that is made during re-encryption is unable to verify whether $r = H_1(m, pk_i)$. For instance, if the value r is chosen randomly, the re-encryption still works normally. Therefore, the security proof should also behave

in the same way, but we will see below that this is not the case. Algorithm 3 shows the definition of the re-encryption oracle given in the security proof of this scheme.

```
Algorithm O_reenc(pk_i, pk_j, CT_i = (D, E, F, s))
  If (pk_i)^s ≠ D · E^{H_3(D,E,F)} return ⊥
  If i and j are both honest or corrupt
    Compute E' = E^{x_j / x_i}
  Else
    Search tuple (m, pk_i, r) ∈ L_{H_1}, such that (pk_i)^r = E
    If such tuple exists, then compute E' = (pk_j)^r
    Else return ⊥
  Return CT_j = (pk_i, E', F)
```

Algorithm 3: Re-encryption oracle from Weng et al.'s security proof [4]

It can be seen that when users i and j are both honest or corrupt, the re-encryption oracle simply computes the re-encryption keys; otherwise, it resorts to examining $\mathcal{L}_{H_1}$, the random oracle table for H_1.

However, as pointed out previously, this oracle is unable to correctly answer queries of the form $(pk^*, pk_j, \hat{c})$, where $\hat{c}$ is an ill-formed ciphertext under the target public key pk^* whose randomness r is not derived from $H_1(m, pk^*)$. In this case, the target user is honest, by definition, whereas user j is corrupt, so the re-encryption oracle resorts to the random oracle table $\mathcal{L}_{H_1}$, which does not contain any tuple of the form (m, pk^*, r), and returns $\perp$. This behavior, which can occur arbitrarily, deviates from the real-world execution of the re-encryption function, where no ciphertext is rejected when $r \neq H_1(m, pk^*)$.

B Another example of wrong usage of the transformation

In addition to the problems in the security proofs presented in the previous Appendix, we describe here a scheme where the Fujisaki-Okamoto transformation is incorrectly applied. In [17], Aono et al. proposed a lattice-based encryption scheme which is proven CPA-secure in the standard model. Then, on top of this scheme, they construct a "CCA-secure" version in the random oracle model, using the generic conversion from Fujisaki and Okamoto [6]. However, this version is flawed because decryption will always fail for re-encrypted ciphertexts, breaking the correctness of the scheme.

B.1 The PRE scheme from Aono et al.

In this subsection we describe the scheme from Aono et al. This scheme is based upon a lattice cryptosystem from Lindner and Peikert [40], whose hardness relies on the Learning With Errors (LWE) problem [41]. Some details are omitted for clarity; we refer the interested reader to [17] and [40] for a complete description of the scheme.

Let n be the security parameter, q a prime number, and l a fixed message length. Set $k = \log q$. Let the randomly generated matrix $A \in \mathbb{Z}_q^{n \times l}$ be publicly

known. Let $D^{n_1 \times n_2}$ be a gaussian noise distribution over $\mathbb{Z}^{n_1 \times n_2}$. Auxiliary function $Power2 : \mathbb{Z}_q^{n \times l} \to \mathbb{Z}_q^{nk \times l}$ is defined as

$$Power2(X) = \begin{bmatrix} X \\ 2X \\ \vdots \\ 2^{k-1}X \end{bmatrix}$$

while function $Bits : \mathbb{Z}_q^{1 \times n} \to \{0,1\}^{1 \times nk}$ produces a bit representation of the elements of input vector v, such that $Bits(v) \cdot Power2(X) = v \cdot X \in \mathbb{Z}_q^{1 \times l}$. Details about noise distribution D and functions $Power2(\cdot)$ and $Bits(\cdot)$ are omitted here, but can be found on [17, 40].

The scheme is as follows:

- $\mathsf{KeyGen}(n)$: Sample $R, S \leftarrow D^{n \times l}$. The public key is $pk = P = R - AS \in \mathbb{Z}_q^{n \times l}$, while the secret key is $sk = S \in \mathbb{Z}^{n \times l}$.

- $\mathsf{Enc}(pk = P, m \in \{0,1\}^l)$: Sample gaussian noise vectors f_1, f_2 from $D^{1 \times n}$, and f_3 from $D^{1 \times l}$. Encode the message $m \in \{0,1\}^l$ to $m \cdot \lfloor \frac{q}{2} \rfloor \in \mathbb{Z}_q^{1 \times l}$. Output the ciphertext $e = (e_1, e_2) \in \mathbb{Z}_q^{1 \times (n+l)}$, where

$$\begin{aligned} e_1 &= f_1 A + f_2 \\ e_2 &= f_1 P + f_3 + m \cdot \lfloor \frac{q}{2} \rfloor \end{aligned}$$

- $\mathsf{Dec}(sk = S, e = (e_1, e_2))$: Compute $\bar{m} = e_1 S + e_2 \in \mathbb{Z}_q^{1 \times l}$. Let $(\bar{m}_1, ..., \bar{m}_l)$ be the individual elements of $\bar{m}$. Output the decrypted message $m = (m_1, ..., m_l)$, where $m_i = 0$ if $\bar{m}_i \in [-\lfloor \frac{q}{4} \rfloor, \lfloor \frac{q}{4} \rfloor) \subset \mathbb{Z}_q$, and $m_i = 1$ otherwise.

- $\mathsf{ReKeyGen}(sk_i = S_i, pk_j = P_j, sk_j = S_j)$: Sample $X \in \mathbb{Z}_q^{nk \times n}$ from the uniform distribution and E from $D^{nk \times l}$. Output the re-encryption key $rk_{i \to j} = (P_j, Q)$, where

$$Q = \begin{bmatrix} X & -XS_j + E + Power2(S_i) \\ 0 & I \end{bmatrix}$$

- $\mathsf{ReEnc}(rk_{i \to j} = (P_j, Q), e = (e_1, e_2))$: Sample gaussian noise vectors g_1, g_2 from $D^{1 \times n}$, and g_3 from $D^{1 \times l}$. Compute

$$g_1[A|P_j] + [g_2|g_3] + [Bits(e_1)|e_2] \cdot Q \in \mathbb{Z}_q^{1 \times (n+l)}$$

  and parse the result as $[e_1'|e_2']$, where $e_1' \in \mathbb{Z}_q^{1 \times n}$ and $e_2' \in \mathbb{Z}_q^{1 \times l}$. Output the re-encrypted ciphertext $e' = (e_1', e_2')$. It can be seen that

$$\begin{aligned} e_1' &= g_1 A + g_2 + Bits(e_1) X \\ e_2' &= g_1 P_j + g_3 + Bits(e_1)(E - XS_j) + e_1 S_i + e_2 \end{aligned}$$

B.2 Description of the flaw

The flaw is based on the fact that a direct application of the Fujisaki-Okamoto conversion does not work in general for proxy re-encryption, since the validity check during the decryption fails for re-encrypted ciphertexts when the randomness is altered. In the definition of their CCA scheme, they only consider the validity of decryption in the case of original ciphertexts. However, when one considers re-encrypted ciphertexts, it can be seen that the validity check during the decryption *always* fails.

Let us consider the CCA-secure version of the scheme, after applying directly the Fujisaki-Okamoto transformation, and let us define a scenario with two users i and j, with public keys P_i and P_j, respectively. A ciphertext originally encrypted for user i is a tuple (e_1, e_2, c) where:

$$\begin{aligned} e_1 &= f_1 A + f_2 \\ e_2 &= f_1 P_i + f_3 + \sigma \cdot \lfloor \frac{q}{2} \rfloor \\ c &= \mathsf{Sym.Enc}(G(\sigma), m) \end{aligned}$$

The terms (e_1, e_2) are the encryption of σ with the PRE scheme, where the term c is the symmetric encryption of the original message, with key $G(\sigma)$. Thus, the original random coins in this case are the noise vectors (f_1, f_2, f_3), which by the definition of the transformation, are computed from $H(\sigma, c)$. In the random oracle model, if this query to H has not been done previously, H produces a random output and records the tuple $(\sigma, c, f_1, f_2, f_3)$ for future queries.

Now lets consider that this ciphertext, originally intended for user i, is re-encrypted for some user j. Let g_1, g_2, g_3 be the random vectors introduced by the re-encryption function, X, E random matrices defined by user j during the interactive process for generating re-encryption keys and S_i, S_j the secret keys of users i and j. The transformation only re-encrypts the terms (e_1, e_2), so the result is a tuple (e'_1, e'_2, c) of the form:

$$\begin{aligned} e'_1 &= g_1 A + g_2 + Bits(e_1) X \\ e'_2 &= g_1 P_j + g_3 + Bits(e_1)(E - X S_j) + e_1 S_i + e_2 \\ c &= \mathsf{Sym.Enc}(G(\sigma), m) \end{aligned}$$

Now user j decrypts this ciphertext. Recall that the Fujisaki-Okamoto conversion dictates that the output of the asymmetric encryption must be reconstructed, using the same inputs. By the correctness of the original PRE scheme, she correctly receives σ. However, the output of $H(\sigma, c)$ must be again (f_1, f_2, f_3), by the definition of the random oracle. When trying to reconstruct (e'_1, e'_2) she would obtain $\hat{e}'_1 = f_1 A + f_2$, $\hat{e}'_2 = f_1 P_j + f_3 + \sigma \cdot \lfloor \frac{q}{2} \rfloor$, and the validation would fail since $(\hat{e}'_1, \hat{e}'_2) \neq (e'_1, e'_2)$. Therefore, the decryption of re-encrypted ciphertexts will always fail.