On the Selective Opening Security of Practical Public-Key Encryption Schemes

Felix Heuer Tibor Jager Eike Kiltz Sven Schäge

Horst Görtz Institute for IT-Security, Ruhr-University Bochum, Germany
{felix.heuer,tibor.jager,eike.kiltz,sven.schaege}@rub.de

Abstract

We show that two well-known and widely employed public-key encryption schemes – RSA Optimal Asymmetric Encryption Padding (RSA-OAEP) and the Diffie-Hellman Integrated Encryption Scheme (DHIES), instantiated with a one-time pad – are selective opening secure (under the strong, simulation-based notion) against chosen-ciphertext attacks in the random oracle model.

Both schemes are obtained via known generic transformations that transform relatively weak primitives (with security in the sense of one-wayness) to IND-CCA secure encryption schemes. We also show a similar result for the well-known Fujisaki-Okamoto transformation that can generically turn a one-way secure public key encryption system into a IND-CCA-secure public-key encryption system.

We prove that selective opening security comes for free in these transformations. Both DHIES and RSA-OAEP are important building blocks in several standards for public key encryption and key exchange protocols. The Fujisaki-Okamoto transformation is very versatile and has successfully been utilised to build efficient lattice-based cryptosystems. The considered schemes are the first practical cryptosystems that meet the strong notion of simulation-based selective opening (SIM-SO-CCA) security.

Keywords: public key encryption, selective opening security, SIM-SO-CCA, OAEP, DHIES, Fujisaki–Okamoto

1 Introduction

Consider a set of clients $A_1, \ldots, A_n$ connecting to a server S. To encrypt a message m_i, each client A_i draws fresh randomness r_i and transmits ciphertext $c_i = \mathsf{Enc}_{pk_S}(m_i; r_i)$ to S. Here pk denotes the public key of S, m_i is the transmitted message, and r_i is the randomness used for encryption. Assume an adversary observes these ciphertexts, and is then able to 'corrupt' a subset of clients $\{A_i\}_{i\in\mathcal{I}}$, $\mathcal{I} \subseteq \{1, \ldots, n\}$, for instance by installing a malware on their computers. Then, for all $i \in \mathcal{I}$, the adversary learns not only the message m_i, but also the randomness r_i that A_i has used to encrypt m_i. Attacks of this type are called *selective-opening* (SO) attacks (under sender corruptions) and a central question in cryptography is whether the unopened ciphertexts remain secure.

At a first glance, one may be tempted to believe that security of the non-corrupted ciphertexts follows immediately, if the encryption scheme meets some standard security notion, like indistinguishability under chosen-plaintext (IND-CPA) or chosen-ciphertext (IND-CCA) attacks, due to the fact that each user A_i samples the randomness r_i independently from the other users. However, it has been observed [BH92, CFGN96, CDNO97, Bea97, CHK05] that this is not true in general, see e.g. [HR14] for an overview.

Basically, it is a combination of two features of the notion of selective opening security that make this security definition essentially stronger than traditional forms of security. The first one is randomness. Observe that the previous security experiment explicitly models that the attacker may also obtain the random coins that are used to produce ciphertexts when corrupting a sender. In practice this is motivated by the fact that erasing (cryptographic) information is technically very difficult and expensive. However, very importantly from a provable-security point of view, the randomness may act as a proof of well-formedness of a ciphertext *for the attacker*. In particular it is now much harder for the simulator

to create ciphertexts that look 'normal' but which are actually computed via some other mechanism (possibly involving some secret trapdoor information). This is because whenever the simulator produces a ciphertext, it must also be able to produce the corresponding message and randomness that map to this ciphertext via the public encryption routine. In traditional security games, even when modelling chosen-ciphertext security, the attacker may not obtain access to such proofs. We also emphasize that the probability to simply guess which of the ciphertexts the attacker is going to corrupt is negligible $1/\binom{n}{n/2}$. As a consequence it is not a viable strategy for the simulator to prepare two size $n/2$ subsets of ciphertexts, one where ciphertexts are produced in the usual way and one where they are produced differently.

The second feature is that the notion of selective opening security considers *message distributions* over all encrypted messages. In particular revealing on plaintext may narrow down the number of possible plaintexts for the remaining ciphertexts. This is very realistic in many practical scenarios as usually queries of many clients to a single server are often very similar (like providing a file request in some special format). In traditional notions of security the simulator has to only guarantee that a single ciphertext that it produced (usually termed the challenge ciphertext) is secure.

Results on SO Security. Defining the right notion of security against selective opening attacks has proven highly non-trivial. There are three notions of security that are not polynomial-time equivalent to each other, two indistinguishability-based notions usually denoted as weak IND-SO and (full) IND-SO security, and a simulation-based notion of selective opening security referred to as SIM-SO security. Previous results showed that SIM-SO-CCA and full IND-SO-CCA security are the strongest notions of security [BHK12, BDWY12, HR14]. However, only SIM-SO-CCA has been realised so far [FHKW10, HLOV11, Hof12]. Unfortunately, the existing constructions are very inefficient and rather constitute theoretical contributions. Intuitively, SIM-SO security says that for every adversary in the above scenario there exists a simulator which can produce the same output as the adversary without ever seeing any ciphertext, randomness, or the public key. It is noteworthy that unlike weak IND-SO security, which requires message distributions that support 'efficient conditional re-sampling' (cf. [BHY09]), SIM-SO is independent of the concrete distribution of the messages.

1.1 Our Contributions

In this paper we show that three important public key encryption systems are secure under the strong notion of SIM-SO-CCA security. Previous results only established IND-CCA security of the resulting schemes. Most notably, our results cover the well-known DHIES scheme, instantiated with a one-time pad, RSA-OAEP, and the Fujisaki-Okamoto (FO) transform instantiated with a one-time pad. Our results show that SIM-SO security essentially comes for free in the random oracle model. This yields the first practical public key encryption schemes that meet the strong notion of SIM-SO-CCA security.

First Construction: DHIES. The first construction we consider is a generalisation of the well-known 'Diffie-Hellman integrated encryption scheme' (DHIES) [ABR01]. (DHIES or 'Hashed ElGamal Encryption' uses a MAC to make plain ElGamal encryption IND-CCA secure.) This generic idea behind DHIES was formalised by Steinfeld, Baek, and Zheng [SBZ02] who showed how to build an IND-CCA secure public key encryption system from a key encapsulation mechanism (KEM) that is one-way under plaintext checking attacks (OW-PCA). OW-PCA is a comparatively weak notion of security in which the adversary's main task is to decapsulate a given encapsulation of some symmetric key. In addition to the public key, the adversary has only access to an oracle which checks, given a KEM key and a ciphertext, whether the ciphertext indeed constitutes an encapsulation of the KEM key under the public key. This construction is IND-CCA secure in the random oracle model [SBZ02]. We show that it is furthermore SIM-SO-CCA secure in the random oracle model. We stress that our result generically holds for the entire construction and therefore for any concrete instantiation of it. Most importantly, it covers the well-known DHIES scheme (when instantiated with a one-time pad) that is contained in several public-key encryption standards like IEEE P1363a, SECG, and ISO 18033-2. DHIES is the de-facto standard for elliptic-curve encryption.

Second Construction: OAEP. The second construction of public key encryption schemes that we consider is the well-known Optimal Asymmetric Encryption Padding (OAEP) transformation [BR94]. OAEP is a generic transformation for constructing public-key encryption schemes from trapdoor permutations that was proposed by Bellare and Rogaway. Since then, it has become an important ingredient in many

security protocols and security standards like TLS [DR08, Res02], SSH [Har06], S/MIME [RT10, Hou03], EAP [CA06], and Kerberos [NSF05, Rae05].

We show that OAEP is SIM-SO-CCA secure when instantiated with a *partial-domain trapdoor permutation* (cf. Section 4.1). Since it is known [FOPS01] that the RSA permutation is partial-domain one-way under the RSA assumption, this implies that RSA-OAEP is SIM-SO-CCA secure under the RSA assumption. In fact, our result holds not only for trapdoor permutations, but for *injective* trapdoor functions as well.

Since SIM-SO-CCA security implies IND-CCA security, our proof also provides an alternative to the IND-CCA security proof of [FOPS01]. Interestingly, despite that we are analysing security in a stronger security model, our proof seems to be somewhat simpler than the proof of [FOPS01], giving a more direct insight into which properties of the OAEP construction and the underlying trapdoor permutation make OAEP secure. This might be due to the fact that our proof is organised as a *sequence of games* [BR06].

Complementing the work of [FOPS01, BF06, BDU08], our result gives new evidence towards the belief that the OAEP construction is sound, and that OAEP-type encryption schemes can be used securely in various practical scenarios.

Third Construction: Fujisaki-Okamoto. Like the previous transformations, the Fujisaki-Okamoto [FO13] transform takes as input a one-way secure public-key encryption system and turns it into a IND-CCA secure one. The transformation excels through its generality. In [Pei14], it has successfully been applied to construct an efficient lattice-based cryptosystem. Remarkably, all other major transformations to IND-CCA cryptosystems where either deemed inefficient, insecure, or inapplicable in this setting. (For more details we refer to [Pei14]).

1.2 Related Work

The problem of selective-opening attacks is well-known, and has already been observed twenty years ago [BH92, CFGN96, CDNO97, Bea97, CHK05]. The problem of constructing encryption schemes that are provably secure against this class of adversaries without random oracles has only been solved recently by Bellare, Hofheinz, and Yilek [BHY09]. In [BHY09], the authors show that *lossy* encryption [PW08] implies security against selective openings under chosen-plaintext attacks (SO-CPA). This line of research is continued in [HLOV11] by Hemenway *et al.*, who show that re-randomisable encryption and statistically hiding two-round oblivious transfer imply lossy encryption. From a cryptographic point of view, the above works solve the problem of finding SO-CPA secure encryption schemes, as there are several constructions of efficient lossy or re-randomizable encryption schemes, e.g. [PW08, BHY09, HLOV11]. When it comes to selective openings under chosen-ciphertext attacks, the situation is somewhat different. Hemenway *et al.* [HLOV11], Fehr *et al.* [FHKW10], Hofheinz [Hof12], and Fujisaki [Fuj12] describe SIM-SO-CCA secure encryption schemes which are all too inefficient for practical applications. More recently, an identity-based encryption scheme with selective-opening security was proposed [BWY11]. It is noteworthy, that the most efficient public key encryption systems proven to be weak IND-SO secure do not meet the stronger notion of SIM-SO security. Lately, SIM-SO-CCA security for IBE has been achieved [LDL+14].

State-of-the-Art of the Provable Security of OAEP. The OAEP construction was proved IND-CCA secure if the underlying trapdoor permutation is partial-domain one-way [BR94, Sho02, FOPS01]. Since the RSA trapdoor permutation is a partial-domain one-way function, this yields the IND-CCA security of RSA-OAEP as well. Fischlin and Boldyreva [BF06] studied the security of OAEP when only one of the two hash functions is modelled as a random oracle, and furthermore showed that OAEP is non-malleable under chosen plaintext attacks for random messages without random oracles. The latter result was strengthened by Kiltz *et al.* [KOS10], who proved the IND-CPAsecurity of OAEP without random oracles, when the underlying trapdoor permutation is *lossy* [PW08]. Since lossy encryption implies IND-SO-CPA security [BHY09], this immediately shows that OAEP is IND-SO-CPA secure in the standard model. However, we stress that prior to our work it was not clear if OAEP meets the stronger notion of SIM-SO security, neither in the standard model nor in the random oracle. Backes *et al.* [BDU08] showed that OAEP is secure under so-called *key-dependent message* attacks in the random oracle model.

There also exist a number of negative results [Bro06, KP09] showing the impossibility of instantiating OAEP without random oracles.

State-of-the-Art of the Provable Security of DHIES. The IND-CCA security of DHIES in the random oracle model is equivalent to the Strong Diffie-Hellman (sDH) assumption [ABR01, SBZ02].

State-of-the-Art of the Provable Security of FO. In our analysis we consider a slightly modified transformation that was given in the journal version [FO13]. This variant is less restrictive than the original version in [FO99]. In particular, the input symmetric encryption system does not need to be deterministic and bijective anymore. The version in [FO13] also clarifies that the two conditions on which the decryption algorithm may abort should trigger the output of the same error symbol. Joye, Quisquater, and Yung [JQY01] have shown that such a behaviour is crucial for security. In fact, Sakurai and Takagi [ST02] have practically exploited the sole fact that in some implementations the error symbol is output slightly earlier when its generated by the first abort condition (timing side-channel). This lead to a successful attack on the EPOC cryptosystem, that applies the Fujisaki-Okamoto transform from [FO99] to the Okamoto-Uchiyama encryption system [OU98].

2 Preliminaries

For $n \in \mathbb{N}$ let $[n] := \{1, \ldots, n\}$. For two strings μ, ν, we denote with $\mu||\nu$ the string obtained by concatenating μ with ν. If L is a set, then $|L|$ denotes the cardinality of L. Let λ denote the security parameter. We assume implicitly that any algorithm receives the unary representation 1^λ of the security parameter as input as its first argument. We say that an algorithm is a PPT algorithm, if it runs in probabilistic polynomial time (in λ). For a finite set A we denote the sampling of a uniform random element a by $a \xleftarrow{\$} A$, while we denote the sampling according to some distribution $\mathfrak{D}$ by $a \leftarrow \mathfrak{D}$.

2.1 Games

We present definitions of security and encryption schemes in terms of games and make use of sequences of games to proof our results. A game G is a collection of procedures/oracles $\{\textsc{Initialise}, \mathrm{P}_1, \mathrm{P}_2, \ldots, \mathrm{P}_t, \textsc{Finalise}\}$ for $t \geq 0$, where P_1 to P_t and Finalise might require some input parameters, while Initialise is run on the security parameter 1^λ. We implicitly assume that boolean flags are initialised to false, numerical types are initialised to 0, sets are initialised to $\emptyset$, while strings are initialised to the empty string ϵ.

An adversary $\mathcal{A}$ is *run in game* G, if $\mathcal{A}$ calls Initialise. During the game $\mathcal{A}$ may run the procedures P_i as often as allowed by the game. If a procedure P was called by $\mathcal{A}$, the output of P is returned to $\mathcal{A}$, except for the Finalise procedure. On $\mathcal{A}$'s call of Finalise the game ends and outputs whatever Finalise returns. Let $\mathsf{G}^{\mathcal{A}} \Rightarrow \mathit{out}$ denote the event that G outputs *out* after running $\mathcal{A}$. If a game's output is either 0 or 1, $\mathcal{A}$ *wins* G if $\mathsf{G}^{\mathcal{A}} \Rightarrow 1$ happens. Further, the *advantage* $\mathbf{Adv}(\mathsf{G}^{\mathcal{A}}, \mathsf{H}^{\mathcal{A}})$ of $\mathcal{A}$ in *distinguishing* games G and H is defined as $|\Pr[\mathsf{G}^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{H}^{\mathcal{A}} \Rightarrow 1]|$. For $\mathcal{A}$ run in G and $\mathcal{S}$ run in game H the *advantage* of $\mathcal{A}$ is defined as $|\Pr[\mathsf{G}^{\mathcal{A}} \Rightarrow 1] - \Pr[\mathsf{H}^{\mathcal{S}} \Rightarrow 1]|$. Setting a boolean flag "Abort..." to *true* implicitly aborts the adversary.

2.2 Public Key Encryption Schemes

Let $\mathcal{M}$, $\mathcal{R}$, $\mathcal{C}$ be sets, let $\mathcal{R}$ be finite. We say that $\mathcal{M}$ is the *message space*, $\mathcal{R}$ is the *randomness space*, and $\mathcal{C}$ is the *cipertext space*. A public key encryption scheme $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ consists of three polynomial-time algorithms.

- Gen generates, given the unary representation of the security parameter 1^λ, a key pair $(sk, pk) \leftarrow \mathsf{Gen}(1^\lambda)$, where pk defines $\mathcal{M}$, $\mathcal{R}$, and $\mathcal{C}$.
- Given pk, and a message $m \in \mathcal{M}$, Enc outputs an encryption $c \leftarrow \mathsf{Enc}_{pk}(m) \in \mathcal{C}$ of m under the public key pk.
- The decryption algorithm Dec takes a secret key sk and a ciphertext $c \in \mathcal{C}$ as input, and outputs a message $m = \mathsf{Dec}_{sk}(c) \in \mathcal{M}$, or a special symbol $\perp \notin \mathcal{M}$ indicating that c is not a valid ciphertext.

Notice, that Enc is a probabilistic algorithm; we make the used randomness only explicit when needed. In that case we write $c = \mathsf{Enc}(m; r)$ for $r \xleftarrow{\$} \mathcal{R}$. We require correctness of PKE, that is for all security parameters 1^λ, for all $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$, and for all $m \in \mathcal{M}$ we have $\Pr[\mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(m)) = m] = 1$.

Procedure INITIALISE	**Procedure** ENC($\mathfrak{D}$)	**Procedure** DEC(c)
$(pk, sk) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$ Return pk	$(m_i)_{i\in[n]} \leftarrow \mathfrak{D}$ $(r_i)_{i\in[n]} \xleftarrow{\$} \mathcal{R}$ $(c_i)_{i\in[n]} := \mathsf{Enc}_{pk}(m_i; r_i)$ Return $(c_i)_{i\in[n]}$	Return $\mathsf{Dec}_{sk}(c)$
Procedure FINALISE(out)		**Procedure** OPEN(i)
Return $\mathsf{Rel}((m_i)_{i\in[n]}, \mathfrak{D}, \mathcal{I}, out)$		$\mathcal{I} := \mathcal{I} \cup \{i\}$ Return (m_i, r_i)

Figure 1: REAL-SIM-SO-CCA$_{\mathsf{PKE}}$ game. Remember that $\mathfrak{D}$ is a distribution over $\mathcal{M}^n$.

Procedure INITIALISE	**Procedure** ENC($\mathfrak{D}$)	**Procedure** FINALISE(out)
Return ϵ	$(m_i)_{i\in[n]} \leftarrow \mathfrak{D}$ Return ϵ	Return $\mathsf{Rel}((m_i)_{i\in[n]}, \mathfrak{D}, \mathcal{I}, out)$

Figure 2: IDEAL-SIM-SO-CCA$_{\mathsf{PKE}}$ game.

2.3 SIM-SO-CCA Security Definition

Definition 2.1. *Let $\mathsf{PKE} := (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a public-key encryption scheme with message space $\mathcal{M}$, randomness space $\mathcal{R}$ and ciphertext space $\mathcal{C}$. Let $n = n(\lambda) > 0$ be a polynomially bounded function. Further, let $\mathfrak{D}$ be a distribution over $\mathcal{M}^n$ and Rel a relation. We consider the following games, whereby an adversary $\mathcal{A}$ is run in the REAL-SIM-SO-CCA$_{\mathsf{PKE}}$ game (Figure 1), while a simulator $\mathcal{S}$ is run in the IDEAL-SIM-SO-CCA$_{\mathsf{PKE}}$ game (Figure 2). We demand that $\mathcal{A}$ and $\mathcal{S}$ call* ENC *exactly one time before calling* OPEN *or* FINALISE*. Further, $\mathcal{A}$ is not allowed to call* DEC *on any c_i. To an adversary $\mathcal{A}$, a simulator $\mathcal{S}$, a relation Rel and n we associate the advantage function*

$$\mathbf{Adv}^{\text{SIM-SO-CCA}}_{\mathsf{PKE}}(\mathcal{A}, \mathcal{S}, \mathsf{Rel}, n, \lambda) := \left|\Pr[\text{REAL-SIM-SO-CCA}^{\mathcal{A}}_{\mathsf{PKE}} \Rightarrow 1] - \Pr[\text{IDEAL-SIM-SO-CCA}^{\mathcal{S}}_{\mathsf{PKE}} \Rightarrow 1]\right| .$$

PKE is SIM-SO-CCA secure if for every PPT adversary $\mathcal{A}$ and every PPT relation Rel there exists a PPT simulator $\mathcal{S}$ such that $\mathbf{Adv}^{\text{SIM-SO-CCA}}_{\mathsf{PKE}}(\mathcal{A}, \mathcal{S}, \mathsf{Rel}, n, \lambda) \leq \mathsf{negl}(\lambda)$.

3 Transformation from any OW-PCA secure KEM

3.1 Key Encapsulation Mechanisms and Message Authentication Codes

Definition 3.1. *Let $\mathcal{K}$ a key space, let $\mathcal{R}$ denote a finite randomness space, and $\mathcal{C}$ a ciphertext space.*

A Key Encapsulation Mechanism (KEM) KEM consists of three PPT algorithms KEM=(KEMGen, Encap, Decap) defined to have the following syntax.

- *KEMGen generates a key pair (pk, sk) on input 1^λ: $(pk, sk) \leftarrow \mathsf{KEMGen}(1^\lambda)$, where pk specifies $\mathcal{K}$, $\mathcal{R}$ and $\mathcal{C}$.*
- *Encap is given pk and outputs a key $k \in \mathcal{K}$ and an encapsulation $c \in \mathcal{C}$ of k: $(c, k) \leftarrow \mathsf{Encap}_{pk}$.*
- *Given sk, Decap decapsulates $c \in \mathcal{C}$: $k \leftarrow \mathsf{Decap}_{sk}(c)$, where $k \in \mathcal{K}$ or outputs some $\perp \notin \mathcal{K}$.*

We require correctness: for all $\lambda \in \mathbb{N}$, for all (pk, sk) generated by $\mathsf{KEMGen}(1^\lambda)$, and for all (c, k) output by Encap_{pk} we have $\Pr[\mathsf{Decap}_{sk}(c) = k] = 1$. We make the randomness used in Encap only explicit when needed. Without loss of generality we assume Encap to sample $k \xleftarrow{\$} \mathcal{K}$. Further, let $\mathcal{K}$, $\mathcal{C}$ be exponentially large in the security parameter: $|\mathcal{K}| \geq 2^\lambda$, $|\mathcal{C}| \geq 2^\lambda$.

KEM has unique encapsulations if for every $\lambda \in \mathbb{N}$ and every (pk, sk) output by $\mathsf{KEMGen}(1^\lambda)$ and all $c, c' \in \mathcal{C}$ we have $\mathsf{Decap}_{sk}(c) = \mathsf{Decap}_{sk}(c') \Rightarrow c = c'$.

We introduce a security notion for KEMs that appeared in [OP01], namely *one-way* security in the presence of a *plaintext-checking oracle* (OW-PCA) amounting an adversary to test if some c is a *valid* encapsulation of a key k. That is, on input (c, k) and given the sk the oracle returns $\text{CHECK}_{sk}(c, k) := (\mathsf{Decap}_{sk}(c) \stackrel{?}{=} k) \in \{0, 1\}$.

Procedure INITIALISE(1^λ)	**Procedure** CHALLENGE	**Procedure** CHECK(k, c)
$(pk, sk) \xleftarrow{\$} \mathsf{KEMGen}(1^\lambda)$ Return pk	$(k^*, c^*) \xleftarrow{\$} \mathsf{Encap}_{pk}$ Return c^*	Return $(\mathsf{Decap}(c) \stackrel{?}{=} k)$ **Procedure** FINALISE(k) Return $(k \stackrel{?}{=} k^*)$

Figure 3: $\mathsf{OW\text{-}PCA}_{\mathsf{KEM}}$ game

Procedure INITIALISE(1^λ)	**Procedure** TAG(m)	**Procedure** VRFY($\tilde{m}, \tilde{t}$)
$k \xleftarrow{\$} \mathsf{MACGen}(1^\lambda)$ Return ϵ	$t \leftarrow \mathsf{Tag}_k(m)$ Return t	Return $\mathsf{Vrfy}_k(\tilde{m}, \tilde{t})$ **Procedure** FINALISE(m^*, t^*) Return $(\mathsf{Vrfy}_k(m^*, t^*) \wedge (m^*, t^*) \neq (m, t))$

Figure 4: $\mathsf{sUF\text{-}OT\text{-}CMA}_{\mathsf{MAC}}$ game

Definition 3.2. *Let $\mathsf{KEM} = (\mathsf{KEMGen}, \mathsf{Encap}, \mathsf{Decap})$ be a Key Encapsulation Mechanism and $\mathcal{A}$ an adversary run in the $\mathsf{OW\text{-}PCA}_{\mathsf{KEM}}$ game stated in Figure 3. We restrict the adversary to call* CHALLENGE *exactly one time and define $\mathcal{A}$'s advantage in winning the $\mathsf{OW\text{-}PCA}_{\mathsf{KEM}}$ game as*

$$\mathbf{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{KEM}}(\mathcal{A}, \lambda) := \Pr[\mathsf{OW\text{-}PCA}^{\mathcal{A}}_{\mathsf{KEM}} \Rightarrow 1] \ .$$

KEM is OW-PCA secure, if $\mathbf{Adv}^{OW\text{-}PCA}_{\mathsf{KEM}}(\mathcal{A}, \lambda)$ is negligible for all PPT $\mathcal{A}$.

Definition 3.3. *Let $\mathcal{M}$ be a message space, $\mathcal{K}$ a key space and let $\mathcal{T}$ be a set (tag space). A Message Authentication Code (MAC) MAC consists of the following three PPT algorithms $\mathsf{MAC} = (\mathsf{MACGen}, \mathsf{Tag}, \mathsf{Vrfy})$, whereby*

- *MACGen generates a key $k \in \mathcal{K}$ on input 1^λ: $k \leftarrow \mathsf{MACGen}(1^\lambda)$.*
- *Tag_k computes a tag $t \in \mathcal{T}$ for a given message $m \in \mathcal{M}$: $t \leftarrow \mathsf{Tag}_k(m)$.*
- *Given a message $m \in \mathcal{M}$ and a tag $t \in \mathcal{T}$, Vrfy_k, outputs a bit: $\{0, 1\} \leftarrow \mathsf{Vrfy}_k(m, t)$.*

We require MAC to be correct: For all $\lambda \in \mathbb{N}$, all keys k generated by $\mathsf{MACGen}(1^\lambda)$, all $m \in \mathcal{M}$ and all tags computed by $\mathsf{Tag}_k(m)$ we have $\Pr[\mathsf{Vrfy}_k(m, \mathsf{Tag}_k(m)) = 1] = 1$. For a fixed MAC and k, given message m we call a tag t / the tuple (m, t) valid, if $\mathsf{Vrfy}_k(m, t) = 1$.

Definition 3.4. *For an adversary $\mathcal{A}$ and a MAC $\mathsf{MAC} := (\mathsf{MACGen}, \mathsf{Tag}, \mathsf{Vrfy})$ we consider the $\mathsf{sUF\text{-}OT\text{-}CMA}_{\mathsf{MAC}}$ (strong unforgeability under one-time chosen message attacks) game given in Figure 4, where $\mathcal{A}$ is allowed to call* TAG *at most once. We define the advantage of $\mathcal{A}$ run in the $\mathsf{sUF\text{-}OT\text{-}CMA}_{\mathsf{MAC}}$ game as*

$$\mathbf{Adv}^{\mathsf{sUF\text{-}OT\text{-}CMA}}_{\mathsf{MAC}}(\mathcal{A}, \lambda) := \Pr[\mathsf{sUF\text{-}OT\text{-}CMA}^{\mathcal{A}}_{\mathsf{MAC}} \Rightarrow 1] \ .$$

MAC is sUF-OT-CMA secure, if $\mathbf{Adv}^{\mathsf{sUF\text{-}OT\text{-}CMA}}_{\mathsf{MAC}}(\mathcal{A}, \lambda) \leq \mathsf{negl}(\lambda)$ holds for all PPT adversaries $\mathcal{A}$.

Note that we only require one-time security, so a sUF-OT-CMA secure MAC can be constructed information-theoretically.

3.2 The Transformation

Before we prove our results on the selective opening security of schemes built from KEMs, we recall a well known transformation ([SBZ02]) to turn a given KEM into a PKE scheme. Notice, that we instantiate the symmetric encryption with a one-time-pad.

Let $\mathsf{KEM} = (\mathsf{KEMGen}, \mathsf{Encap}, \mathsf{Decap})$ be a KEM with key space $\mathcal{K}_{\mathsf{KEM}}$, randomness space $\mathcal{R}_{\mathsf{KEM}}$ and ciphertext space $\mathcal{C}_{\mathsf{KEM}}$. Let $\mathsf{MAC} = (\mathsf{MACGen}, \mathsf{Tag}, \mathsf{Vrfy})$ be a MAC with key space $\mathcal{K}_{\mathsf{KEM}}$, message

Procedure $\text{PKEGEN}(1^\lambda)$	**Procedure** $\text{ENC}(m)$	**Procedure** $\text{DEC}(c^{(1)}, c^{(2)}, c^{(3)})$
$(pk, sk) \xleftarrow{\$} \mathsf{KEMGen}(1^\lambda)$	$(k, c^{(1)}) \xleftarrow{\$} \mathsf{Encap}_{pk}$	$k \leftarrow \mathsf{Decap}_{sk}(c^{(1)})$
Return (pk, sk)	$(k^{sym}, k^{mac}) := \mathrm{H}(k)$	$(k^{sym}, k^{mac}) := \mathrm{H}(k)$
	$c^{(2)} := k^{sym} \oplus m$	if $\mathsf{Vrfy}_{k^{mac}}(c^{(2)}, c^{(3)}) = 0$
	$c^{(3)} := \mathsf{Tag}_{k^{mac}}(c^{(2)})$	Return $\perp$
	Return $(c^{(1)}, c^{(2)}, c^{(3)})$	else
		Return $c^{(2)} \oplus k^{sym}$

Figure 5: Transformation $\mathsf{PKE}_{\mathsf{KEM},\mathsf{MAC}}$ from KEM and MAC to PKE.

space $\{0,1\}^\ell$ and tag space $\mathcal{T}_{\mathsf{MAC}}$. Let $\mathrm{H} : \mathcal{K}_{\mathsf{KEM}} \to \{0,1\}^\ell \times \mathcal{K}_{\mathsf{MAC}}$ be a hash function. By applying the transformation given in Figure 5 we obtain a PKE $\mathsf{PKE}_{\mathsf{KEM},\mathsf{MAC}}$ for messages $\mathcal{M} = \{0,1\}^\ell$, with randomness space $\mathcal{R}_{\mathsf{KEM}}$ and ciphertexts in $\mathcal{C}_{\mathsf{KEM}} \times \{0,1\}^\ell \times \mathcal{T}_{\mathsf{MAC}}$.

It is well known, that the given construction turns a OW-PCA KEM into a IND-CCA secure PKE scheme in the random oracle model [SBZ02]. Our next theorem strengthens this results by showing that $\mathsf{PKE}_{\mathsf{KEM},\mathsf{MAC}}$ is even SIM-SO-CCA secure.

Theorem 3.5. *Let KEM be a OW-PCA secure KEM with unique encapsulations and let MAC be a sUF-OT-CMA secure MAC. Then $\mathsf{PKE}_{\mathsf{KEM},\mathsf{MAC}}$ is SIM-SO-CCA secure in the random oracle model. In particular, for any adversary $\mathcal{A}$ run in the REAL-SIM-SO-CCA$_{\mathsf{PKE}_{\mathsf{KEM},\mathsf{MAC}}}$ game, that issues at most $q_h \leq 2^{\lambda-1}$ hash and $q_d \leq 2^{\lambda-1}$ decryption queries and obtains n ciphertexts, and every PPT relation Rel, there exists a simulator $\mathcal{S}$, a forger $\mathcal{F}$ run in the sUF-OT-CMA$_{\mathsf{MAC}}$ game, and an adversary $\mathcal{B}$ run in the OW-PCA$_{\mathsf{KEM}}$ game with roughly the same running time as $\mathcal{A}$ such that*

$$\mathbf{Adv}^{\text{SIM-SO-CCA}}_{\mathsf{PKE}_{\mathsf{KEM},\mathsf{MAC}}}(\mathcal{A}, \mathcal{S}, \mathsf{Rel}, n, \lambda) \leq n \cdot \left(\frac{q_h + q_d}{2^{\lambda-1}} + \mathbf{Adv}^{\text{sUF-OT-CMA}}_{\mathsf{MAC}}(\mathcal{F}, \lambda) + \mathbf{Adv}^{\text{OW-PCA}}_{\mathsf{KEM}}(\mathcal{B}, \lambda) \right) .$$

Intuition for the proof of Theorem 3.5 Recall the definition of SIM-SO-CCA security. For any (fixed) adversary $\mathcal{A}$ run in the REAL-SIM-SO-CCA game we have to construct a simulator $\mathcal{S}$ that can compute the same output as $\mathcal{A}$, even though $\mathcal{S}$ is run in the IDEAL-SIM-SO-CCA game. To this end we gradually modify the REAL-SIM-SO-CCA game in a sequence of games such that, eventually, a simulator $\mathcal{S}$ can simulate the (modified) REAL-SIM-SO-CCA interface for adversary $\mathcal{A}$ while $\mathcal{S}$ itself is run in the IDEAL-SIM-SO-CCA game.

Game G_0 constitutes of the REAL-SIM-SO-CCA$^{\mathcal{A}}_{\mathsf{PKE}_{\mathsf{KEM},\mathsf{MAC}}}$ game up to some small syntactical changes, e.g. the random oracle is implemented by lazy sampling in an additional Hash procedure.

Observe that $\mathcal{S}$ can only learn m_i by querying $\text{Open}(i)$ to its challenger (*ideal* game). Further, note that the set of opened ciphertexts $\mathcal{I}$ may be part of $\mathcal{A}$'s output that $\mathcal{S}$ wants to simulate. Hence $\mathcal{S}$ must not query $\text{Open}(i)$ to its *ideal* game challenger unless $\mathcal{A}$ does not pose the same query.

Since all Open queries by $\mathcal{A}$ will happen *after* it received its ciphertexts $(c_1, \ldots, c_n)$ from $\mathcal{S}$, the simulator has to provide $\mathcal{A}$ with correctly distributed ciphertexts c_i independent of m_i. In game G_1 it is ensured that $c_i^{(2)} = k_i^{sym} \oplus m_i$ is uniformly random at the time $\mathcal{A}$ obtains its ciphertexts, in game G_2 we will rewrite encryption to sample $c_i^{(2)}$ uniformly at random.

Once $\mathcal{A}$ obtained its ciphertexts, it may call Open, Hash and Dec as it wishes. Observe, that fixing $c_i^{(2)}$ and m_i immediately determines $k_i^{sym} = c_i^{(2)} \oplus m_i$. Note that on a call $\text{Open}(i)$ by $\mathcal{A}$, simulator $\mathcal{S}$ has to reveal m_i and r_i such that $c_i = \mathsf{Enc}_{pk}(m_i; r_i)$. Therefore, in our construction, simulator $\mathcal{S}$ will program the first component of $H(k_i)$ to be $c_i^{(2)} \oplus m_i$. Thus, we have to ensure that k_i^{sym} remains undefined (i.e. the ciphertexts c_i remain non-commiting) as long as $\mathcal{A}$ does not call $\text{Open}(i)$. Only when $\mathcal{A}$ makes such a call, $\mathcal{S}$ can issue $\text{Open}(i)$ to its *ideal* game to obtain m_i and define k_i^{sym} accordingly.)

To make sure that k_i^{sym} remains undefined long enough, we have to consider that $\mathcal{A}$ might call $\text{Hash}(k_i)$ or issue a valid ciphertext $(c_i^{(1)}, \cdot, \cdot)$ to Dec since any such call would assign a value to k_i^{sym}. From game G_3 on, we will abort when $\mathcal{A}$ sends a valid ciphertext to Dec and $\mathcal{A}$ did not call $\text{Hash}(k_i)$ before. Since $\mathcal{A}$ did not call $\text{Hash}(k_i)$ before, k_i^{mac} is still uniformly random and the probability of abort can be bounded due to sUF-OT-CMA security of MAC. If $\mathcal{A}$ calls $\text{Hash}(k_i)$, it managed to decapsulate k_i hidden in $c_i^{(1)}$. From G_4 on, we will abort once $\mathcal{A}$ queries $\text{Hash}(k_i)$; the probability of abort is bounded via KEMs OW security.

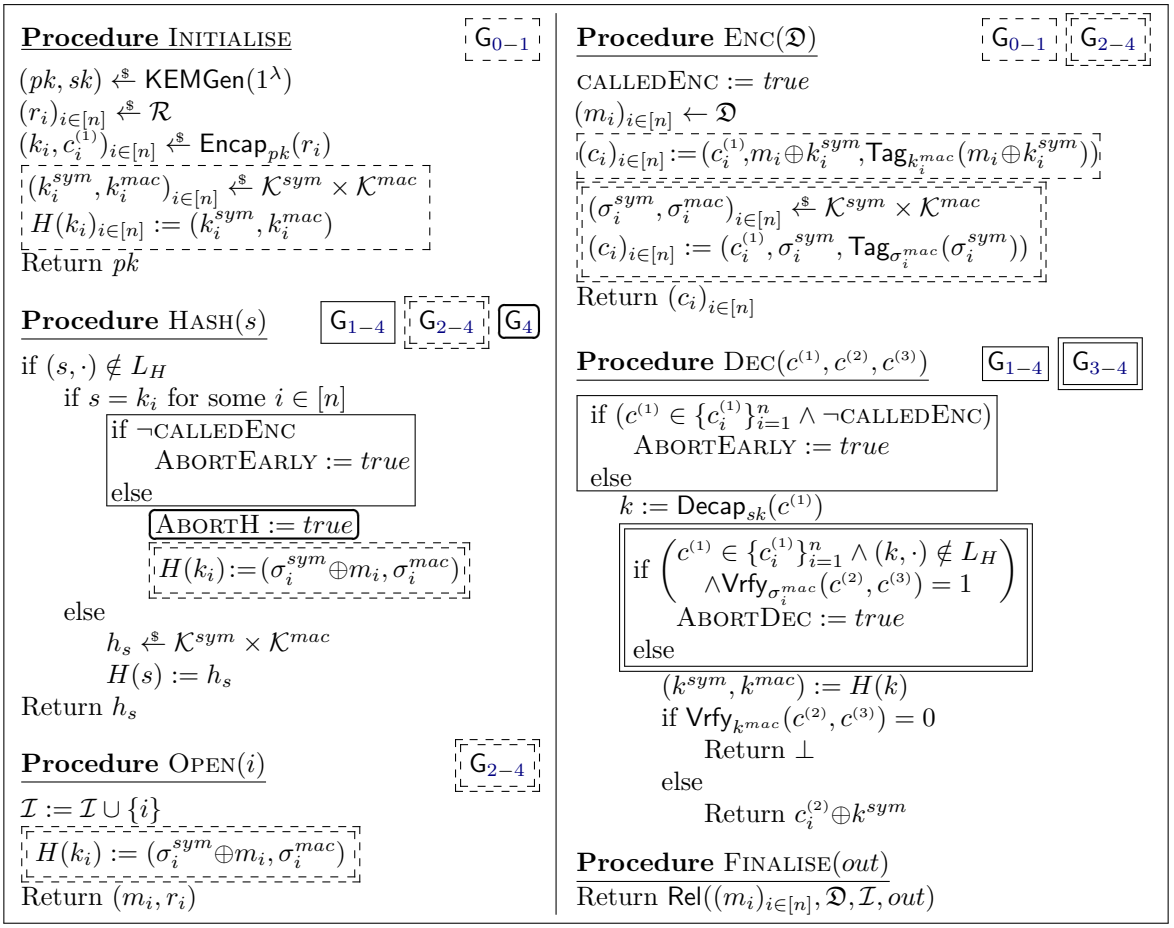

Figure 6: Sequence of games G_0 to G_4. Boxed code is only executed in the games indicated by the game names given in the same box style at the top right of every procedure, non-boxed code is always run.

Proof. Let q_h be the number of hash queries and let q_d be the number of decryption queries issued by $\mathcal{A}$, let $n = n(\lambda)$ be a polynomial in λ. For $i \in [n]$ let: m_i denote the i^{th} message sampled by the challenger, r_i the i^{th} randomness used by Encap: $(k_i, c_i^{(1)}) \leftarrow \mathsf{Encap}(r_i)$, $(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$ the i^{th} key-pair generated by hashing k_i and $c_i := (c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$ the i^{th} ciphertext. Without loss of generality, the games samples $(r_i)_{i\in[n]}$ as part of Initialise. We proceed with a sequence of games which is given in pseudocode in Figure 6.

Game 0 We model H as a random oracle. Challenger $\mathcal{C}_0$ keeps track of issued calls (either by the game or $\mathcal{A}$) of Hash(s) by maintaining a list L_H. For a query s, Hash(s) returns h_s if there is an entry $(s, h_s) \in L_H$, otherwise Hash samples h_s at random, adds (s, h_s) to L_H, and returns h_s; we write $H(s) := h_s$ only and implicitly assume an update operation $L_H := L_H \cup \{(s, h_s)\}$ to happen in the background.
We introduce small syntactical changes: Challenger $\mathcal{C}_0$ samples $(k_i^{sym}, k_i^{mac})_{i\in[n]}$ uniformly random and sets $(H(k_i))_{i\in[n]} := (k_i^{sym}, k_i^{mac})$ while Initialise is run. Additionally, G_0 runs Encap_{pk} to generate $(k_i, c_i^{(1)})_{i\in[n]}$ during Initialise.

Claim 0. $\mathbf{Adv}(\mathsf{REAL\text{-}SIM\text{-}SO\text{-}CCA}^{\mathcal{A}}_{\mathsf{PKE}_{\mathsf{KEM},\mathsf{MAC}}}, \mathsf{G}_0^{\mathcal{A}}) = 0$.

Proof. Apparently, it makes no difference if the challenger samples $(r_i)_{i\in[n]}$ and runs $\mathsf{Encap}(r_i)$ on demand as part of Enc or in advance while Initialise is run.
Since H is modeled as a random oracle, $H(s)$ is sampled uniformly random for every fresh query Hash(s).

Therefore $\mathcal{C}_0$ does not change the distribution by sampling (k_i^{sym}, k_i^{mac}) in the first place and setting $H(k_i) := (k_i^{sym}, k_i^{mac})$ afterwards. ■

Game 1 We add an abort condition. Challenger $\mathcal{C}_1$ raises the event AbortEarly and aborts[1], if $\mathcal{A}$ did not call Enc before calling either Hash(k_i) or Dec$(c_i^{(1)}, \cdot, \cdot)$ for some $i \in [n]$.

Claim 1. $\mathbf{Adv}(\mathsf{G}_0^{\mathcal{A}}, \mathsf{G}_1^{\mathcal{A}}) \leq n \cdot (q_h + q_d) \cdot 2^{-(\lambda-1)}$.

Proof. Since games G_0 and G_1 are identical until AbortEarly is raised, it follows that $\mathbf{Adv}(\mathsf{G}_0^{\mathcal{A}}, \mathsf{G}_1^{\mathcal{A}}) \leq \Pr[\text{AbortEarly}]$. Let viaHash and viaDec be the events that AbortEarly was caused by either a hash or a decryption query of $\mathcal{A}$. Let s_i denote the i^{th} hash and $d_i = (d_i^{(1)}, d_i^{(2)}, d_i^{(3)})$ the i^{th} decryption query of $\mathcal{A}$. It holds that

$$
\begin{aligned}
&\Pr[\text{AbortEarly}] = \Pr[\text{viaHash}] + \Pr[\text{viaDec}] \\
&\leq \Pr[s_1 \in \{k_i\}_{i=1}^n] + \sum_{i=2}^{q_h} \Pr[s_i \in \{k_i\}_{i=1}^n | \bigwedge_{j=1}^{i-1} s_j \notin \{k_i\}_{i=1}^n] \\
&+ \Pr[d_i^{(1)} \in \{c_i^{(1)}\}_{i=1}^n] + \sum_{i=2}^{q_d} \Pr[d_i^{(1)} \in \{c_i^{(1)}\}_{i=1}^n | \bigwedge_{j=1}^{i-1} d_j^{(1)} \notin \{c_i^{(1)}\}_{i=1}^n] \\
&= \sum_{i=1}^{q_h} \frac{n}{2^\lambda - (i-1)} + \sum_{i=1}^{q_d} \frac{n}{2^\lambda - (i-1)} \leq \sum_{i=1}^{q_h} \frac{n}{2^\lambda - q_h} + \sum_{i=1}^{q_d} \frac{n}{2^\lambda - q_d} \leq \frac{n(q_h + q_d)}{2^{\lambda-1}} \quad .
\end{aligned}
$$

Above holds since Encap samples $k \xleftarrow{\$} \mathcal{K}$ and KEM has unique encapsulations. ■

Game 2 We change the encryption procedure and answer hash queries in a different way. Challenger $\mathcal{C}_2$ does not program $H(k_i)$ for $i \in [n]$ anymore. Enc still samples m_i, and samples $\sigma_i^{sym} \xleftarrow{\$} \mathcal{K}^{sym}$, $\sigma_i^{mac} \xleftarrow{\$} \mathcal{K}^{mac}$, to compute $c_i = \mathsf{Enc}_{k_i}(m_i) := (c_i^{(1)}, \sigma_i^{sym}, \mathsf{Tag}_{\sigma_i^{mac}}(\sigma_i^{sym}))$. If $\mathcal{A}$ should call Hash(k_i) for $i \in [n]$ or Open(i), the challenger programs $H(k_i) := (\sigma_i^{sym} \oplus m_i, \sigma_i^{mac})$.

As from now on $(k_i, \cdot) \notin L_H$ implies that Open(i) was not called.

Claim 2. $\mathbf{Adv}(\mathsf{G}_1^{\mathcal{A}}, \mathsf{G}_2^{\mathcal{A}}) = 0$.

Proof. Assuming that AbortEarly does not happen in game G_2, the keys k_i^{sym} and k_i^{mac} are still uniformly random when $\mathcal{A}$ calls Enc. Therefore $(c_i^{(2)})_{i\in[n]} = m_i \oplus k_i^{sym}$ is uniform and $(c_i^{(3)})_{i\in[n]}$ is a valid tag of a uniformly random message under a key from the uniform distribution. Consequently, challenger $\mathcal{C}_2$ can sample $(c_i^{(2)})_{i\in[n]} := \sigma_i^{sym}$ uniformly and can compute the tags using a uniform key σ_i^{mac} without changing the distribution of the encryptions $(c_i)_{i\in[n]}$.
Challenger $\mathcal{C}_2$ does not program $H(k_i)$ for $i \in [n]$ anymore, but has to keep H consistent. If $\mathcal{A}$ calls Hash(k_i) or Open(i), $\mathcal{C}_2$ sets $H(k_i) := (\sigma_i^{sym} \oplus m_i, \sigma_i^{mac})$. ■

Game 3 We add another abort condition. If $\mathcal{A}$ already called Enc, issues a decryption query $(c_i^{(1)}, c^{(2)}, c^{(3)}) \notin \{c_i\}_{i=1}^n$, where $H(k_i)$ is not defined, and $\mathsf{Vrfy}_{\sigma_i^{mac}}(c_i^{(1)}, c^{(2)}, c^{(3)})$ verifies, challenger $\mathcal{C}_3$ raises AbortDec and aborts $\mathcal{A}$.

Claim 3. $\mathbf{Adv}(\mathsf{G}_2^{\mathcal{A}}, \mathsf{G}_3^{\mathcal{A}}) \leq n \cdot \mathbf{Adv}_{\mathsf{MAC}}^{\mathsf{sUF\text{-}OT\text{-}CMA}}(\mathcal{F}, \lambda)$.

Proof. Games G_2 and G_3 are identical until AbortDec happens, it suffices to bound $\Pr[\text{AbortDec}]$.

Let $\mathsf{MAC} := (\mathsf{MACGen}, \mathsf{Tag}, \mathsf{Vrfy})$ be the MAC used by the challenger in the sUF-OT-CMA game. We construct an adversary $\mathcal{F}$ against the sUF-OT-CMA security of MAC having success probability $\Pr[\text{AbortDec}]/n$. The reduction is straight forward: Forger $\mathcal{F}$ runs adversary $\mathcal{A}$ as in game G_3, but picks $i^* \xleftarrow{\$} [n]$ during Initialise. Computing the i^{*th} ciphertext, $\mathcal{F}$ queries its sUF-OT-CMA challenger for $t^* := \text{Tag}(\sigma_{i^*}^{sym})$ instead of using its own Tag procedure and sends $(c_i)_{i\in[n]}$ to $\mathcal{A}$. If $\mathcal{A}$ should call Open(i^*), challenger C_3 apparently was unlucky in hiding its own challenge and aborts the adversary. Querying its $\text{Vrfy}_k(\cdot, \cdot)$ oracle, $\mathcal{F}$ can detect when $\mathcal{A}$ issues a valid query Dec$(c_i^{(1)}, c^{(2)}, c^{(3)})$ for some $i \in [n]$, returns $(c^{(2)}, c^{(3)})$ to its sUF-OT-CMA challenger and aborts $\mathcal{A}$.

[1] Notice, that $\mathcal{C}_1$ aborts even if such a decryption query is invalid.

Procedure INITIALISE:
$pk \xleftarrow{\$} \text{INITIALISE}_{OW\text{-}PCA}$
$i^* \xleftarrow{\$} [n]$
$(r_i)_{i\in[n]\setminus\{i^*\}} \xleftarrow{\$} \mathcal{R}$
$(k_i, c_i^{(1)})_{i\in[n]\setminus\{i^*\}} \leftarrow \mathsf{Encap}_{pk}(r_i)$
$c_{i^*}^{(1)} \xleftarrow{\$} \text{CHALLENGE}$
Return pk

Procedure HASH(s)
if $(s,\cdot) \notin L_H$
  if $s = k_i$ for some $i \in [n] \setminus \{i^*\}$
    if $\neg$CALLEDENC
      ABORTEARLY := *true*
    else
      ABORTH := *true*
  else
    if CHECK$(s, c_{i^*}^{(1)}) = 1$
      FINALISE$_{OW\text{-}PCA}(s)$
      ABORTH := *true*
    else
      if $\begin{pmatrix}\exists (c^{(1)},k^{sym},k^{mac}) \in H_{patch} \\ \text{s.t. CHECK}(s, c^{(1)}) = 1)\end{pmatrix}$
        $H(s) := (k^{sym}, k^{mac})$
      else
        $h_s \xleftarrow{\$} \mathcal{K}^{sym} \times \mathcal{K}^{mac}$
        $H(s) := h_s$
Return $H(s)$

Procedure OPEN(i)
if $i = i^*$
  ABORT := *true*
else
  $\mathcal{I} := \mathcal{I} \cup \{i\}$
  $H(k_i) := (\sigma_i^{sym} \oplus m_i, \sigma_i^{mac})$
  Return (m_i, r_i)

Procedure ENC($\mathfrak{D}$):
CALLEDENC := *true*
$(m_i)_{i\in[n]} \leftarrow \mathfrak{D}$
$(\sigma_i^{sym}, \sigma_i^{mac})_{i\in[n]} \xleftarrow{\$} \mathcal{K}^{sym} \times \mathcal{K}^{mac}$
$(c_i)_{i\in[n]} := (c_i^{(1)}, \sigma_i^{sym}, \mathsf{Tag}_{\sigma_i^{mac}}(\sigma_i^{sym}))$
Return $(c_i)_{i\in[n]}$

Procedure DEC($c^{(1)}, c^{(2)}, c^{(3)}$)
if $(c^{(1)} \in \{c_i^{(1)}\}_{i=1}^n \wedge \neg$CALLEDENC)
  ABORTEARLY := *true*
else
  if $c^{(1)} \in \{c_i^{(1)}\}_{i=1}^n$
    if $\mathsf{Vrfy}_{\sigma_i^{mac}}(c^{(2)}, c^{(3)}) = 1$
      ABORTDEC := *true*
    else
      Return $\perp$
  else
    if $\begin{pmatrix}\exists\ (s,\cdot,\cdot) \in L_H \text{ s.t.} \\ \text{CHECK}(s, c^{(1)}) = 1\end{pmatrix}$
      $(k^{sym}, k^{mac}) := H(s)$
    else
      $(k^{sym},k^{mac}) \xleftarrow{\$} \mathcal{K}^{sym} \times \mathcal{K}^{mac}$
      Add $(c^{(1)},k^{sym},k^{mac})$ to H_{patch}
    if $\mathsf{Vrfy}_{k^{mac}}(c^{(2)}, c^{(3)}) = 0$
      Return $\perp$
    else
      Return $c_i^{(2)} \oplus k^{sym}$

Procedure FINALISE(out)
Return $\mathsf{Rel}((m_i)_{i\in[n]}, \mathfrak{D}, \mathcal{I}, out)$

Figure 7: Reduction to KEM's OW-PCA security given by the game interface for $\mathcal{A}$.

Assume that ABORTDEC happens, i.e. $\mathcal{A}$ makes a valid decryption query $(c_i^{(1)}, c^{(2)}, c^{(3)}) \notin \{c_i\}_{i\in[n]}$, while $H(k_i)$ is still undetermined. Notice, that we must not allow $H(k_{i^*})$ to be fixed since $k_{i^*}^{mac}$ is only known to the sUF-OT-CMA challenger. Let $(c_i^{(1)}, \tilde{c}^{(2)}, \tilde{c}^{(3)}) \in \{c_i\}_{i\in[n]}$ be the ciphertext c_i, whose first component matches the first entry of $\mathcal{A}$'s valid decryption query. Hence, $c^{(3)}$ is either a new valid tag for $\tilde{c}^{(2)}$ or $c^{(3)}$ is a valid tag for a 'new' message $c^{(2)}$, since $(c^{(2)}, c^{(3)}) \neq (\tilde{c}^{(2)}, \tilde{c}^{(3)})$. In both cases $\mathcal{F}$ wins its sUF-OT-CMA challenge by returning $(c^{(2)}, c^{(3)})$, if $\mathcal{F}$ picks the right challenge ciphertext to embed t^*. The claim follows by rearranging

$$\mathbf{Adv}^{\mathsf{sUF\text{-}OT\text{-}CMA}}_{\mathsf{MAC}}(\mathcal{F}, \lambda) \geq \Pr[\text{ABORTDEC}]/n \ .$$

■

Game 4 We add one more abort condition. Challenger $\mathcal{C}_4$ raises the event ABORTH if $\mathcal{A}$ already called ENC, issues a hash query HASH(k_i) for $i \in [n]$ and did not call OPEN(i) before.

Claim 4. $\mathbf{Adv}(\mathsf{G}_3^{\mathcal{A}}, \mathsf{G}_4^{\mathcal{A}}) \leq n \cdot \mathbf{Adv}^{OW\text{-}PCA}_{\mathsf{KEM}}(\mathcal{B}, \lambda)$.

Proof. Games G_3 and G_4 are identical until AbortH happens. Given some adversary $\mathcal{A}$ run in the REAL-SIM-SO-CCA game, we construct an adversary $\mathcal{B}$ against the OW-PCA security of KEM having success probability $\Pr[\text{AbortH}]/n$ as depicted in Figure 7. Adversary $\mathcal{B}$ receives a pk and a challenge encapsulation $c^* \leftarrow$ Challenge of some key k^* and aims to output k, given access to an $\text{Check}(\cdot,\cdot)$ returning $\text{Check}_{sk}(k, c) := (\mathsf{Decap}_{sk}(c) \stackrel{?}{=} k)$.

$\mathcal{B}$ runs $\mathcal{A}$ as $\mathcal{A}$ is run in game G_3 except for the following differences: After calling Initialise, $\mathcal{B}$ guesses an index $i^* \xleftarrow{\$} [n]$. Then $\mathcal{B}$ creates c_i as before, but hides its own challenge in the first component of the i^{*th} ciphertext. Let's assume that AbortH happens. Since $\mathcal{B}$ knows $\{c_i^{(1)}\}$ for $i \in [n] \setminus \{i^*\}$, it can detect if $\mathcal{A}$ queries $\text{Hash}(s)$ for $s \in \{k_i\}$ where $i \in [n] \setminus \{i^*\}$, while $\mathcal{B}$ can invoke its Check oracle to detect the query $\text{Hash}(k_{i^*})$ since $\text{Check}(k_{i^*}, c_{i^*}^{(1)}) = 1$. Therefore $\mathcal{B}$ does not have to guess when AbortH happens. If $\mathcal{A}$ should call $\text{Open}(i^*)$, $\mathcal{B}$ apparently guessed i^* wrong[2] and aborts $\mathcal{A}$. Running the reduction, $\mathcal{B}$ has to maintain the conditions for AbortDec. Therefore it suffices to check if $c^{(1)} \in \{c_i^{(1)}\}_{i=1}^n$ and $\mathsf{Vrfy}_{\sigma_i^{mac}}(c^{(2)}, c^{(3)})$ hold, because $H(k)$ cannot be defined, since neither AbortH, nor Abort via Open happened.

It remains to explain how $\mathcal{B}$ (unable to compute $k = \mathsf{Decap}_{sk}(c^{(1)})$) answers decryption queries without knowing sk. To answer these queries we make use of the nifty "oracle patching technique" from [CS03]. If $\mathcal{A}$ calls $\text{Dec}(c^{(1)}, c^{(2)}, c^{(3)})$, $\mathcal{B}$ checks if $H(k)$ is already defined by querying $\text{Check}(s, c^{(1)})$ for every $(s, \cdot) \in L_H$. If there is such a s, $\mathcal{B}$ uses $(k^{sym}, k^{mac}) := H(s)$. If not, $\mathcal{B}$ picks (k^{sym}, k^{mac}) at random and has to keep an eye on upcoming hash queries, since $\mathcal{B}$ just committed to $H(k)$.

Therefore $\mathcal{B}$ maintains a dedicated list H_{patch} where $\mathcal{B}$ adds $(c^{(1)}, (^{sym}, k^{mac}))$. On every hash query $\text{Hash}(s)$, $\mathcal{B}$ checks if there is an entry $(c^{(1)}, k^{sym}, k^{mac}) \in H_{patch}$ s.t. $\text{Check}(s, c^{(1)}) = 1$ in order to fix the oracle by setting $H(s) := (k^{sym}, k^{mac})$. If $\mathcal{A}$ should call $\text{Dec}(c_{i^*}^{(1)}, \cdot, \cdot)$, challenger $\mathcal{C}_3$ treats it like every other decryption query. Considering that AbortH happens, $\mathcal{B}$ only has to pick the right ciphertext to hide its own OW-PCA challenge to win its game. Therefore $\mathcal{B}$ succeeds if AbortH happens and $\mathcal{B}$ guessed $i^* \in [n]$ correctly:

$$\mathbf{Adv}^{\text{OW-PCA}}_{\mathsf{KEM}}(\mathcal{B}, \lambda) \geq \Pr[\text{AbortH}]/n \ .$$

■

Claim 5. There exists a simulator $\mathcal{S}$ run in the IDEAL-SIM-SO-CCA game such that

$$\mathbf{Adv}(\mathsf{G}_4^{\mathcal{A}}, \text{IDEAL-SIM-SO-CCA}^{\mathcal{S}}_{\mathsf{PKE}_{\mathsf{KEM},\mathsf{MAC}}}) = 0 \ .$$

Proof. We construct a simulator $\mathcal{S}$ run in the ideal game. The simulator runs the adversary as it is run in game G_4. Especially, all abort event remain in the code. Thus, any abort event is as likely to occur as in game G_4. For simplicity we explain the functionality of $\mathcal{S}$ assuming that no abort event happens.

First, $\mathcal{S}$ runs Gen on its own and feeds pk to $\mathcal{A}$. On $\mathcal{A}$'s call of $\text{Enc}(\mathfrak{D})$ the simulator calls $\text{Enc}(\mathfrak{D})$ as well and creates dummy encryptions without knowing the sampled messages $(m_i)_{i\in[n]}$. If $\mathcal{A}$ calls $\text{Open}(i)$, the simulator forwards the query to its own game, learns m_i, and returns (m_i, r_i) to $\mathcal{A}$.
Because AbortEarly does not happen, $\mathcal{S}$ does not have to commit to $\mathsf{Dec}(c_i)$ before Enc is called. Since neither AbortH nor AbortDec happen, $\mathcal{A}$ calls $\text{Open}(i)$ before issuing 'critical' hash or decryption queries and $\mathcal{S}$ is able to learn m_i and can program H accordingly. Due to these changes and the dummy encryption introduced in game G_2, $\mathcal{A}$ cannot get information on some m_i without calling $\text{Open}(i)$, that is, 'avowing' $\mathcal{S}$ to call $\text{Open}(i)$ as well, allowing $\mathcal{S}$ to answer possibly upcoming hash or decryption queries consistently. ■

Collecting the advantages of $\mathcal{A}$ we get the claim as stated in (3.5). ■

3.3 Implications for practical encryption schemes

We now give specific instantiations of SIM-SO-CCA secure scheme via our generic transformation. We focus on two well known KEMs, namely the DH and RSA key encapsulation mechanism.

[2] $\mathcal{A}$ cannot ask to open every single challenge ciphertext, since AbortH occurs.

Procedure $\mathsf{Gen}(1^\lambda)$	**Procedure** $\mathsf{Enc}(m)$	**Procedure** $\mathsf{Dec}(c_1, c_2, c_3)$
$x \xleftarrow{\$} \mathbb{Z}_p$	$r \xleftarrow{\$} \mathbb{Z}_p$	$(k^{sym}, k^{mac}) \leftarrow H({c_1}^x)$
$X := g^x$	$(k^{sym}, k^{mac}) \leftarrow H(X^r)$	if $\mathsf{Vrfy}_{k^{mac}}(c_2, c_3) = 0$
$pk := (\mathbb{G}, g, p, X)$	$c_1 := g^r$	Return $\perp$
$sk := x$	$c_2 := k^{sym} \oplus m$	else
Return (pk, sk)	$c_3 := \mathsf{Tag}_{k^{mac}}(c_2)$	Return $c_2 \oplus k^{sym}$
	Return (c_1, c_2, c_3)	

Figure 8: The Diffie-Hellman Integrated Encryption Scheme DHIES instantiated with a one-time pad.

DHIES Let $\mathbb{G}$ be a group of prime-order p, and let g be a generator. The Diffie-Hellman KEM $\mathsf{DH\text{-}KEM} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is defined as follows. The key-generation algorithm Gen picks $x \xleftarrow{\$} \mathbb{Z}_p$ and defines $pk = X := g^x$ and $sk = x$; the encapsulation algorithm Encap_{pk} picks $r \xleftarrow{\$} \mathbb{Z}_p$ and returns $(c = g^r, k = X^r)$; the decapsulation algorithm $\mathsf{Decap}_{sk}(c)$ returns $k = c^x$.

OW-PCA security of the $\mathsf{DH\text{-}KEM}$ is equivalent to the strong Diffie-Hellman (sDH) assumption [ABR01]. The sDH assumption states that there is no PPT adversary $\mathcal{A}$ that, given two random group elements $U := g^u, V := g^v$ and a *restricted* DDH oracle $\mathcal{O}_v(\cdot,\cdot)$ where $\mathcal{O}_v(a,b) := (a^v \stackrel{?}{=} b)$ computes g^{uv} with non-negligible probability.

Let MAC be a MAC with message-space and key-space $\{0,1\}^\ell$ and let $H : \mathbb{G} \mapsto \{0,1\}^{2\ell}$ be a hash function. The security of $\mathrm{DHIES} = \mathsf{PKE}_{\mathsf{DH\text{-}KEM},\mathsf{MAC}}$ (depicted in Figure 8) instantiated with a one-time pad is stated in the following corollary, whose proof is a direct consequence of Theorem 3.5.

Corollary 3.6. *DHIES instantiated with a one-time pad is SIM-SO-CCA secure in the random oracle model, if* MAC *is sUF-OT-CMA and the sDH assumption holds in* $\mathbb{G}$.

RSA-KEM We obtain another selective-opening secure encryption scheme, if we plug the $\mathsf{RSA\text{-}KEM}$ into the generic transformation given in Figure 5. Thereby, OW-PCA security of the $\mathsf{RSA\text{-}KEM}$ holds under the RSA assumption [Sho04a]. Under the RSA assumption, $\mathsf{PKE}_{\mathsf{RSA\text{-}KEM},\mathsf{MAC}}$ (as described in ISO18033-2 [Sho04a]) is SIM-SO-CCA secure in the random oracle model.

Both reductions for the OW-PCA security of the $\mathsf{DH\text{-}KEM}$, $\mathsf{RSA\text{-}KEM}$, respectively, are tight.

4 The OAEP Transformation

In this section we show that OAEP is SIM-SO-CCA secure when instantiated with a *partial-domain one-way trapdoor permutation* (see Section 4.1). Since it is known [FOPS01] that the RSA permutation is partial-domain oneway under the RSA assumption, this implies that RSA-OAEP is SIM-SO-CCA secure under the RSA assumption. In fact, our result works not only for trapdoor permutations, but for *injective* trapdoor functions as well.

Since SIM-SO-CCA security implies IND-CCA security, our proof also provides an alternative to the IND-CCA security proof of [FOPS01].

4.1 Trapdoor Permutations and Partial-Domain Onewayness

Recall that a trapdoor permutation is a triple of algorithms $\mathcal{T} = (GK, F, F^{-1})$, where GK generates a key pair $(ek, td) \xleftarrow{\$} GK(1^\lambda)$, $F(ek,\cdot)$ implements a permutation

$$f_{ek} : \{0,1\}^k \to \{0,1\}^k \tag{1}$$

specified by ek, and $F^{-1}(td,\cdot)$ inverts f_{ek} using the trapdoor td. Let us write the function f_{ek} from (1) as a function

$$f_{ek} : \{0,1\}^{\ell+k_1} \times \{0,1\}^{k_0} \to \{0,1\}^k$$

with $k = \ell + k_1 + k_0$.

Procedure INITIALISE(1^λ)	**Procedure** CHALLENGE	**Procedure** FINALISE(s')
$(ek, td) \xleftarrow{\$} GK(1^\lambda)$ Return ek	$(s,t) \xleftarrow{\$} \{0,1\}^{\ell+k_1} \times \{0,1\}^{k_0}$ $y := F(ek,(s,t))$ Return y	Return $(s \stackrel{?}{=} s')$

Figure 9: $\mathsf{PD\text{-}OW}_{\mathcal{T}}$ game

Definition 4.1. *Let $\mathcal{T}$ be a trapdoor permutation as given above and $\mathcal{B}$ an adversary run in the $\mathsf{PD\text{-}OW}_{\mathcal{T}}$ game given in Figure 9. We restrict $\mathcal{B}$ to call* CHALLENGE *exactly one time and define $\mathcal{B}$'s advantage in winning the $\mathsf{PD\text{-}OW}_{\mathcal{T}}$ game as*

$$\mathbf{Adv}^{\mathsf{PD\text{-}OW}}_{\mathcal{T}}(\mathcal{B},\lambda) := \Pr[\mathsf{PD\text{-}OW}^{\mathcal{B}}_{\mathcal{T}} \Rightarrow 1] \ .$$

Moreover, if $\mathbf{Adv}^{\mathsf{PD\text{-}OW}}_{\mathcal{T}}(\mathcal{B},\lambda) \leq \mathsf{negl}(\lambda)$ for all probabilistic polynomial-time (in λ) adversaries $\mathcal{B}$, we say that $\mathcal{T}$ is a partial-domain secure *trapdoor permutation.*

4.2 Optimal Asymmetric Encryption Padding (OAEP)

Let $\mathcal{T} = (GK, F, F^{-1})$ be a trapdoor permutation. The OAEP encryption scheme is defined as follows.

- The key generation $\mathsf{Gen}(1^\lambda)$ computes a key pair $(ek, td) \leftarrow GK(1^\lambda)$ for the trapdoor permutation. It defines two hash functions

  $$G : \{0,1\}^{k_0} \to \{0,1\}^{\ell+k_1} \quad \text{and} \quad H : \{0,1\}^{\ell+k_1} \to \{0,1\}^{k_0}$$

  and outputs $sk = td$ and $pk = (ek, G, H)$.

- To encrypt a message $m \in \{0,1\}^\ell$, the sender draws a random value $r \xleftarrow{\$} \{0,1\}^{k_0}$. Then it computes

  $$s = m||0^{k_1} \oplus G(r) \qquad t = r \oplus H(s) \ .$$

  The ciphertext is $c = F(ek,(s,t)) = f_{ek}(s,t)$.

- To decrypt a ciphertext C, the decryption algorithm $\mathsf{Dec}_{sk}(c)$ uses $sk = td$ to apply the inverse permutation to c, and obtains $(s,t) = F^{-1}(td, c)$. Then it computes $r = t \oplus H(s)$ and $\mu = s \oplus G(r)$, and parses $\mu \in \{0,1\}^{\ell+k_1}$ as $\mu = m||\rho$ with $m \in \{0,1\}^\ell$ and $\rho \in \{0,1\}^{k_1}$. If $\rho = 0^{k_1}$, then the decryption algorithm outputs m. Otherwise $\perp$ is returned.

The OAEP padding process is illustrated in Figure 10.

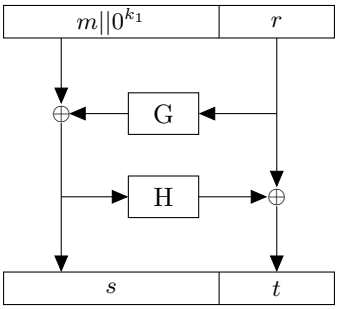

Figure 10: The OAEP padding process.

4.3 Security of OAEP against SO-CCA Attacks

In this section we will analyse the security of the OAEP scheme. We will prove that OAEP is SIM-SO-CCA secure in the random oracle model [BR93], assuming the partial-domain onewayness of the trapdoor permutation $\mathcal{T}$. Note that a proof in the random oracle model is the strongest result we can hope for, since SIM-SO-CCA security implies IND-CCA security, and it is known [KP09] that OAEP can not be proven IND-CCA secure without random oracles.

Theorem 4.2. *Let OAEP be the scheme described in Section 4.2 and $\mathcal{T} = (GK, F, F^{-1})$ be a trapdoor permutation. Then OAEP is SIM-SO-CCA secure in the random oracle model (where both hash functions G and H are modeled as random oracles). In particular, for every PPT relation Rel, every adversary $\mathcal{A}$ run in the $\mathsf{REAL\text{-}SIM\text{-}SO\text{-}CCA}_{OAEP}$ game that issues at most q_h queries to H, q_g queries to G, q_d decryption queries, and obtains n ciphertexts, there exists a simulator $\mathcal{S}$ and an adversary $\mathcal{B}$ in the $\mathsf{PD\text{-}OW}_{\mathcal{T}}$ experiment such that*

$$\mathbf{Adv}^{\mathsf{SIM\text{-}SO\text{-}CCA}}_{OAEP}(\mathcal{A}, \mathcal{S}, \mathsf{Rel}, n, \lambda) \leq \delta$$

where

$$\delta = q_d \cdot \left(2^{-k_1} + q_g \cdot 2^{-k_0}\right) + n(q_g + n) \cdot 2^{-k_0} + nq_h \cdot \mathsf{Adv}^{\mathcal{T}}_{\mathsf{pd}}(\mathcal{B}, \lambda) + nq_g \cdot 2^{-\ell - k_1} \quad .$$

Intuition for the proof of Theorem 4.2 We prove the theorem in a sequence of games, starting with the $\mathsf{REAL\text{-}SIM\text{-}SO\text{-}CCA}_{\mathsf{OAEP}}$ experiment. From game to game we gradually modify the challenger, until we end up in a game where the challenger can act as a simulator in the $\mathsf{IDEAL\text{-}SIM\text{-}SO\text{-}CCA}_{\mathsf{OAEP}}$ experiment. Our goal is to modify the challenger such that in the final game it does not need to know message m_i before the adversary asks $\textsc{Open}(i)$. To this end, we have to describe how the challenger is able to create "non-committing" ciphertexts $c_1, \ldots, c_n$ in the $\textsc{Enc}$ procedure, which can then be opened to any message m_i when $\mathcal{A}$ issues an $\textsc{Open}(i)$ query.

In a first step, we replace the original decryption procedure that uses the real trapdoor td with an equivalent (up to a negligible error probability) decryption procedure, which is able to decrypt ciphertexts by examining the sequence of random oracle queries made by adversary $\mathcal{A}$. Here we use that $\mathcal{A}$ is not able (except for some non-negligible probability) to create a new valid ciphertext $c = F(ek, (s, t))$, unless it asks the random oracle H on input s and G on input $H(s) \oplus t$. However, in this case the challenger is able to decrypt c by exhaustive search through all queries to H and G made by $\mathcal{A}$.

For $i \in [n]$ let $c_i = F(ek, (s_i, t_i)$ now denote the i^{th} challenge ciphertext that $\mathcal{A}$ receives in the security experiment. We show how to construct an attacker against the partial-domain one-wayness of $\mathcal{T}$, which is successful if the adversary $\mathcal{A}$ ever asks $H(s_i)$ before $\textsc{Open}(i)$ for any $i \in [n]$. Thus, assuming that $\mathcal{T}$ is secure in the sense of partial-domain one-wayness, it will never happen that $\mathcal{A}$ asks $H(s_i)$ before $\textsc{Open}(i)$, except for some negligible probability.

Finally, we conclude with the observation that from $\mathcal{A}$'s point of view all values of $H(s_i)$ remain equally likely until $\textsc{Open}(i)$ is asked, which implies also that it is very unlikely that $\mathcal{A}$ ever asks $G(t_i \oplus H(s_i))$ before $\textsc{Open}(i)$. This in turn means that the challenger does not have to commit to a particular value of $G(t_i \oplus H(s_i))$, and thus not to a particular message $m_i || 0^{k_1} = s_i \oplus G(t_i \oplus H(s_i))$, before $\textsc{Open}(i)$ is asked.

Proof of Theorem 4.2. The proof proceeds in a *sequence of games*, following [BR06, Sho04b], where Game G_0 corresponds to the $\mathsf{REAL\text{-}SIM\text{-}SO\text{-}CCA}^{\mathcal{A}}_{\mathsf{OAEP}}$ experiment with adversary $\mathcal{A}$ and a challenger, called $\mathcal{C}_0$. From game to game, we gradually modify the challenger, until we obtain a challenger which is able to act as a simulator in the $\mathsf{IDEAL\text{-}SIM\text{-}SO\text{-}CCA}^{\mathcal{S}}_{\mathsf{OAEP}}$ experiment.

Let us first fix some notation. We denote with q_g an upper bound on the number of queries issued by $\mathcal{A}$ to random oracle G, with q_h an upper bound on the number of queries to H, and with q_d an upper bound on the number of decryption queries. For $i \in [n]$ we will denote with c_i the i^{th} component of the challenge ciphertext vector $(c_i)_{i \in [n]}$, and we write c_i as $c_i = f_{ek}(s_i, t_i)$.

Game 0 Challenger $\mathcal{C}_0$ executes the $\mathsf{REAL\text{-}SIM\text{-}SO\text{-}CCA}$ experiment with attacker $\mathcal{A}$ by implementing the procedures described in Figure 11. Note that $\mathcal{C}_0$ also implements procedures to simulate the random oracles G and H. To this end, it maintains four lists

$$L_G \subseteq \{0,1\}^{k_0} \times \{0,1\}^{\ell + k_1} \qquad L_H \subseteq \{0,1\}^{\ell + k_1} \times \{0,1\}^{k_0}$$
$$L^{\mathcal{A}}_G \subseteq \{0,1\}^{k_0} \qquad L^{\mathcal{A}}_H \subseteq \{0,1\}^{\ell + k_1}$$

Procedure INITIALISE
$(ek, td) \xleftarrow{\$} GK(1^\lambda)$
Return ek

Procedure ENC($\mathfrak{D}$)
$(m_i)_{i\in[n]} \leftarrow \mathfrak{D}$
for $i \in [n]$:
  $r_i \xleftarrow{\$} \{0,1\}^{k_0}$
  $s_i := m||0^{k_1} \oplus G_{\mathsf{int}}(r_i)$
  $t_i := r_i \oplus H_{\mathsf{int}}(s_i)$
  $c_i := F(ek, (s_i, t_i))$
Return $(c_i)_{i\in[n]}$

Procedure DEC(c)
$(s, t) := F^{-1}(td, c)$
$r := t \oplus H_{\mathsf{int}}(s)$
$m||\rho := s \oplus G_{\mathsf{int}}(r)$
if $\rho = 0^{k_1}$
  Return m
else
  Return $\perp$

Internal procedure $H_{\mathsf{int}}(s)$
If $(s, h_s) \notin L_H$
  $h_s \xleftarrow{\$} \{0,1\}^{k_0}$
  $L_H := L_H \cup (s, h_s)$
Return h_s

Procedure $H(s)$
$L_H^{\mathcal{A}} := L_H^{\mathcal{A}} \cup \{s\}$
Return $H_{\mathsf{int}}(s)$

Procedure OPEN(i)
$\mathcal{I} := \mathcal{I} \cup \{i\}$
Return (m_i, r_i)

Internal procedure $G_{\mathsf{int}}(r)$
if $(r, h_r) \notin L_G$
  $h_r \xleftarrow{\$} \{0,1\}^{\ell+k_1}$
  $L_G := L_G \cup (r, h_r)$
Return h_r

Procedure $G(r)$
$L_G^{\mathcal{A}} := L_G^{\mathcal{A}} \cup \{r\}$
Return $G_{\mathsf{int}}(r)$

Procedure FINALISE(out)
Return $\mathsf{Rel}((m_i)_{i\in[n]}, \mathfrak{D}, \mathcal{I}, out)$

Figure 11: Procedures of Game 0.

which are initialised to the empty set in the INITIALISE procedure.

To simulate the random oracle G, the challenger uses the *internal* procedure G_{int}, which uses list L_G to ensure consistency of random oracle responses.

The adversary does not have direct access to procedure G_{int}, but only via procedure G, which stores all values r queried by $\mathcal{A}$ in an additional list $L_G^{\mathcal{A}}$. This allows us to keep track of all values queried by $\mathcal{A}$. Random oracle H is implemented similarly, with procedures H_{int} and H, using list s L_H and $L_H^{\mathcal{A}}$.

By definition we have

$$\mathbf{Adv}(\mathsf{REAL\text{-}SIM\text{-}SO\text{-}CCA}^{\mathcal{A}}_{\mathsf{OAEP}}, \mathsf{G}_0^{\mathcal{A}}) = 0 \ .$$

In the following games we will replace $\mathcal{C}_0$ with challenger $\mathcal{C}_i$ in Game i. In the last game, we replace the challenger with a simulator.

Game 1 In this game, $\mathcal{C}_1$ proceeds exactly as $\mathcal{C}_0$, except that instead of implementing procedure DEC, it uses procedure DEC_1 from Figure 12 to respond to decryption-queries. Note that procedure DEC_1 does not require the trapdoor td to perform decryption.

Claim 1. It holds that $\mathbf{Adv}(\mathsf{G}_0^{\mathcal{A}}, \mathsf{G}_1^{\mathcal{A}}) \leq q_d \cdot \left(2^{-k_1} + q_g \cdot 2^{-k_0}\right)$.

Proof. Game G_1 is perfectly indistinguishable from Game G_0, unless $\mathcal{A}$ makes a decryption query with ciphertext c, such that $\text{DEC}(c) \neq \text{DEC}_1(c)$. Note that this can only hold if $\mathcal{A}$ queries a ciphertext c with $(s, t) = F^{-1}(td, c)$, such that

$$(s, \cdot) \notin L_H \quad \text{or} \quad (t \oplus H(s), \cdot) \notin L_G$$

where $\cdot$ is any value, but it holds that $G(t \oplus H(s)) \oplus s = m||\rho$ with $\rho = 0^{k_1}$.

Consider a single chosen-ciphertext $c = F(ek, (s, t))$. Suppose that $(s, \cdot) \notin L_H$. In this case $H(s)$ is uniform and independent from $\mathcal{A}$'s view. The probability that there exists $(r, \cdot) \in L_G$ such that $r = H(s) \oplus t$ is therefore at most $q_g \cdot 2^{-k_0}$, since we assumed that the adversary issues at most q_g queries to G.

If $(r, \cdot) \notin L_G$ then $G(r)$ is uniform and independent from $\mathcal{A}$'s view, thus the probability that $G(r) \oplus s = m||0^{k_1}$ has the correct syntax is at most 2^{-k_1}.

Since the adversary issues at most q_d chosen-ciphertext queries, we have $\mathbf{Adv}(\mathsf{G}_0^{\mathcal{A}}, \mathsf{G}_1^{\mathcal{A}}) \leq q_d \cdot \left(2^{-k_1} + q_g \cdot 2^{-k_0}\right)$. ∎

Procedure $\textsc{Dec}_1(c)$	**Procedure** $\textsc{Enc}_2(\mathfrak{D})$
For $(r, h_r, s, h_s) \in L_G \times L_H$:	$(m_i)_{i\in[n]} \leftarrow \mathfrak{D}$
if $\begin{pmatrix} c = F(ek,(s, r \oplus h_s)) \\ \wedge\ s \oplus h_r = m\|\|0^{k_1} \end{pmatrix}$	For $i \in [n]$:
Return m	$s_i \xleftarrow{\$} \{0,1\}^{\ell+k_1}$, $t_i \xleftarrow{\$} \{0,1\}^{k_0}$
Return $\perp$	$c_i := F(ek,(s_i, t_i))$
	$r_i := H(s_i) \oplus t_i$
	if $r_i \in L_G$
	$\textsc{AbortG} := \mathit{true}$
	$h_{r_i} := s_i \oplus m_i\|\|0^{k_1}$
	$L_G := L_G \cup \{(r_i, h_{r_i})\}$
	Return $(c_i)_{i\in[n]}$

Figure 12: Replacement procedures $\textsc{Dec}_1$ and $\textsc{Enc}_2$.

Game 2 Challenger $\mathcal{C}_2$ proceeds exactly like $\mathcal{C}_1$, except that it implements procedure $\textsc{Enc}_2$ from Figure 12 instead of $\textsc{Enc}$. Note that this procedure first samples (s_i, t_i) uniformly random, then computes $c_i = F(ek,(s_i, t_i))$, and finally programs the random oracle G such that c_i decrypts to m_i.

Claim 2. It holds that $\mathbf{Adv}(\mathsf{G}_1^{\mathcal{A}}, \mathsf{G}_2^{\mathcal{A}}) \leq n(q_g + n) \cdot 2^{-k_0}$.

Proof. Note that procedure $\textsc{Enc}_2$ first defines $r_i := H(s_i) \oplus t_i$ for uniformly random $t_i \xleftarrow{\$} \{0,1\}^{k_0}$. Thus, r_i is distributed uniformly over $\{0,1\}^{k_0}$, exactly as in Game G_1. Now suppose that $r_i \notin L_G$, thus $\textsc{Enc}_2$ does not terminate. In this case the hash function G is programmed such that $G(r_i) = h_{r_i} = s_i \oplus m_i\|\|0^{k_1}$. Since s_i is uniformly distributed, so is $G(r_i)$, exactly as in Game G_1. Thus, $\textsc{Enc}_2$ simulates procedure $\textsc{Enc}$ from Game G_1 perfectly, provided that it does not terminate.

Note that the procedure terminates only if $r_i \in L_G$. Since all values $r_1, \ldots, r_n$ are distributed uniformly, because the s_i-values are uniformly random, this happens with probability at most $n(q_g + n) \cdot 2^{-k_0}$. ■

Game 3 We add an abort condition to the $\textsc{Open}$ procedure (see the left-hand side of Figure 13). Challenger $\mathcal{C}_3$ proceeds exactly like $\mathcal{C}_2$, except that it raises event $\textsc{AbortS}$ and terminates, if $\mathcal{A}$ ever queried s_i to H for some $i \in [n]$ *before* querying $\textsc{Open}(i)$.

Note that in Game G_3, the attacker never evaluates H on input s_i for any $i \notin \mathcal{I}$, or the game is aborted.

Claim 3. It holds that $\mathbf{Adv}(\mathsf{G}_2^{\mathcal{A}}, \mathsf{G}_3^{\mathcal{A}}) \leq n \cdot q_h \cdot \mathsf{Adv}_{\mathsf{pd}}^{\mathcal{T}}(\mathcal{B}, \lambda)$.

Proof. Game G_3 proceeds identically to Game G_2, until event $\textsc{AbortS}$ is raised. Thus we have

$$\mathbf{Adv}(\mathsf{G}_2^{\mathcal{A}}, \mathsf{G}_3^{\mathcal{A}}) \leq \Pr[\textsc{AbortS}] \ .$$

We construct an adversary $\mathcal{B}$ against the partial-domain onewayness of $\mathcal{T}$. Adversary $\mathcal{B}$ receives as input ek and $y = f_{ek}(s,t)$ for uniformly random $(s,t) \xleftarrow{\$} \{0,1\}^{\ell+k_1} \times \{0,1\}^{k_0}$. It proceeds exactly like $\mathcal{C}_3$,

Procedure $\textsc{Open}(i)$	**Procedure** $\textsc{Open}(i)$
$\mathcal{I} := \mathcal{I} \cup \{i\}$	$\mathcal{I} := \mathcal{I} \cup \{i\}$
if $s_i \in L_H^{\mathcal{A}}$	if $s_i \in L_H^{\mathcal{A}}$
$\textsc{AbortS} := \mathit{true}$	$\textsc{AbortS} := \mathit{true}$
Return (m_i, r_i)	if $r_i \in L_G^{\mathcal{A}}$
	$\textsc{AbortR} := \mathit{true}$
	Return (m_i, r_i)

Figure 13: Modified $\textsc{Open}$ procedures of Games 3 (left) and 4 (right).

Procedure $\textsc{Enc}(\mathfrak{D})$	**Procedure** $\textsc{Open}(i)$
$(m_i)_{i\in[n]} \leftarrow \mathfrak{D}$	$\mathcal{I} := \mathcal{I} \cup \{i\}$
For $i \in [n]$:	if $s_i \in L_H^{\mathcal{A}}$
$s_i \xleftarrow{\$} \{0,1\}^{\ell+k_1}$, $t_i \xleftarrow{\$} \{0,1\}^{k_0}$	$\textsc{AbortS} := true$
$c_i := F(ek, (s_i, t_i))$	if $r_i \in L_G^{\mathcal{A}}$
$r_i := H(s_i) \oplus t_i$	$\textsc{AbortR} := true$
if $r_i \in L_G$	$h_{r_i} := s_i \oplus m_i \| 0^{k_1}$
$\textsc{AbortG} := true$	$L_G := L_G \cup \{(r_i, h_{r_i})\}$
Return $(c_i)_{i\in[n]}$	Return (m_i, r_i)

Figure 14: New procedures for Game G_5.

except for the following. At the beginning of the game it sets $pk := ek$ and guesses two indices $j \xleftarrow{\$} [n]$ and $q \xleftarrow{\$} [q_h]$ uniformly random, and sets $c_j := y$. Note that c_j is correctly distributed (cf. the changes introduced in Game G_2). When $\mathcal{A}$ makes its q^{th} query s^* to H, then $\mathcal{B}$ returns s^* and terminates.

Assume that $\textsc{AbortS}$ happens. Then, at some point in the game, $\mathcal{A}$ makes the *first* query s' to H such that $s' = s_i$ is a partial-domain preimage of some c_i. With probability $1/q_h$ it holds that $s^* = s_i$. Moreover, with probability $1/n$ we have $i = j$. In this case $\mathcal{B}$ obtains the partial preimage $s = s_j$ of $y = c_j$. Thus, $\mathcal{B}$ succeeds, if $\textsc{AbortS}$ happens and if it has guessed $j \in [n]$ and $q \in [q_h]$ correctly. This happens with probability $\Pr[\textsc{AbortS}]/(nq_h)$, which implies that

$$\Pr[\textsc{AbortS}] \leq nq_h \cdot \mathsf{Adv}_{\mathsf{pd}}^{\mathcal{T}}(\mathcal{B}, \lambda) \ .$$

■

Game 4 We add another abort condition to the $\textsc{Open}$ procedure (see the right-hand side of Figure 13). Challenger $\mathcal{C}_4$ raises event $\textsc{AbortR}$ and terminates, if $\mathcal{A}$ ever queries r_i to $G_{\mathcal{A}}$ for some $i \in [n]$, before querying $\textsc{Open}(i)$. Otherwise it proceeds like $\mathcal{C}_3$.

Claim 4. It holds that $\mathbf{Adv}(\mathsf{G}_3^{\mathcal{A}}, \mathsf{G}_4^{\mathcal{A}}) \leq nq_g \cdot 2^{-\ell-k_1}$.

Proof. Note that $\mathcal{A}$ never queries s_i before querying $\textsc{Open}(i)$ (or the game is aborted), due to the changes introduced in Game G_3. Thus, for all $i \notin \mathcal{I}$, $H(s_i)$ is uniformly random and independent of $\mathcal{A}$'s view. Therefore, all $r_i = t_i \oplus H(s_i)$ are uniformly random and independent of $\mathcal{A}$'s view. Since $\mathcal{A}$ issues at most q_g queries to G, and we have $1 \leq i \leq n$, this implies $\mathbf{Adv}(\mathsf{G}_3^{\mathcal{A}}, \mathsf{G}_4^{\mathcal{A}}) \leq nq_g \cdot 2^{-\ell-k_1}$. ■

Game 5 Note that the attacker in Game G_4 never issues a query $G(r_i)$ before asking $\textsc{Open}(i)$, as otherwise the game is aborted. Thus, the challenger does not have to define the hash value $G(r_i)$ before $\textsc{Open}(i)$ is asked. Therefore we can move the definition of $G(r_i)$ from the $\textsc{Enc}_2$ procedure to the $\textsc{Open}$ procedure.

Therefore we replace the procedures $\textsc{Enc}_2$ and $\textsc{Open}$ from Game G_4 with procedures $\textsc{Enc}$ and $\textsc{Open}$ described in Figure 14.

Note that the only difference is that for each $i \in [n]$ the hash value $G(r_i)$ is not defined in the $\textsc{Enc}$ procedure, but in the $\textsc{Open}$ procedure. Moreover, this modification is completely oblivious to $\mathcal{A}$, which implies

$$\mathbf{Adv}(\mathsf{G}_4^{\mathcal{A}}, \mathsf{G}_5^{\mathcal{A}}) = 0 \ .$$

Game 6 Note that in Game G_5 the encryption procedure samples a message vector $(m_i)_{i\in[n]}$, but the messages are only used in the $\textsc{Open}$ procedure. This allows us to construct a simulator, whose procedures are described in Figure 15. Note that the view of $\mathcal{A}$ when interacting with the simulator is *identical* to its view when interacting with challenger $\mathcal{C}_5$, which implies

$$\mathbf{Adv}(\mathsf{G}_5^{\mathcal{A}}, \mathsf{G}_6^{\mathcal{A}}) = 0 \ .$$

Procedure INITIALISE
INITIALISE()
$(ek, td) \xleftarrow{\$} GK(1^\lambda)$
$L_G := L_H := L_G^{\mathcal{A}} := L_H^{\mathcal{A}} := \emptyset$
Return ek

Procedure ENC($\mathfrak{D}$)
ENC($\mathfrak{D}$)
for $i \in [n]$:
  $s_i \xleftarrow{\$} \{0,1\}^{\ell+k_1}$
  $t_i \xleftarrow{\$} \{0,1\}^{k_0}$
  $c_i := F(ek, (s_i, t_i))$
  $r_i := H(s_i) \oplus t_i$
  If $r_i \in L_G$ then
    ABORTG := *true*
Return $(c_i)_{i\in[n]}$

Procedure $\text{DEC}_1(c)$
For $(r, h_r, s, h_s) \in L_G \times L_H$ do
  if $\begin{pmatrix} c = F(ek, (s, r \oplus h_s)) \\ \wedge\ s \oplus h_r = m||0^{k_1} \end{pmatrix}$
    Return m
Return $\perp$

Internal procedure $H_{\mathsf{int}}(s)$
if $(s, h_s) \notin L_H$
  $h_s \xleftarrow{\$} \{0,1\}^{k_0}$
  $L_H := L_H \cup (s, h_s)$
Return h_s

Procedure $H(s)$
$L_H^{\mathcal{A}} := L_H^{\mathcal{A}} \cup \{s\}$
Return $H_{\mathsf{int}}(s)$

Procedure OPEN(i)
$m_i :=$ OPEN(i)
$\mathcal{I} := \mathcal{I} \cup \{i\}$
if $s_i \in L_H^{\mathcal{A}}$
  ABORTS := *true*
if $r_i \in L_G^{\mathcal{A}}$
  ABORTR := *true*
$h_{r_i} := s_i \oplus m_i||0^{k_1}$
$L_G := L_G \cup \{(r_i, h_{r_i})\}$
Return (m_i, r_i)

Internal procedure $G_{\mathsf{int}}(r)$
if $(r, h_r) \notin L_G$
  $h_r \xleftarrow{\$} \{0,1\}^{\ell+k_1}$
  $L_G := L_G \cup (r, h_r)$
Return h_r

Procedure $G(r)$
$L_G^{\mathcal{A}} := L_G^{\mathcal{A}} \cup \{r\}$
Return $G_{\mathsf{int}}(r)$

Procedure FINALISE(out)
FINALISE(out)

Figure 15: Procedures used by the simulator to implement the $\mathsf{REAL\text{-}SIM\text{-}SO\text{-}CCA}^{\mathcal{A}}_{\mathsf{OAEP}}$ experiment. Instructions in boxes correspond to calls to the $\mathsf{IDEAL\text{-}SIM\text{-}SO\text{-}CCA}^{\mathcal{S}}_{\mathsf{OAEP}}$ experiment made by the simulator.

5 The Fujisaki-Okamoto Transformation

In this section we study the Fujisaki-Okamoto transformation [FO99, FO13] that combines a one-way secure PKE scheme with high min-entropy of ciphertexts and a one-time secure symmetric encryption scheme to obtain an IND-CCA secure PKE scheme. We instantiate the symmetric encryption with the one-time pad and consider the FO transformation as a transformation of PKE schemes. We show that the same transformation also yields SIM-SO-CCA secure PKEs under the same complexity assumptions as in [FO13].

5.1 One-wayness and ciphertext distribution

Definition 5.1. *Let* $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be a PKE scheme with message space* $\mathcal{M}$ *and randomness space* $\mathcal{R}$*. For an adversary* $\mathcal{B}$ *we consider the one-wayness game* OW_{PKE} *as given in Figure 16, where* $\mathcal{B}$ *calls* CHALLENGE *exactly once. To* PKE*,* $\mathcal{B}$*, and* λ *we associate the advantage function* $\mathbf{Adv}^{OW}_{\mathsf{PKE}}(\mathcal{B}, \lambda) := \Pr[OW^{\mathcal{B}}_{\mathsf{PKE}} \Rightarrow 1]$.

Procedure INITIALISE(1^λ)
$(pk, sk) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$
Return pk

Procedure CHALLENGE
$m \xleftarrow{\$} \mathcal{M}$
$r \xleftarrow{\$} \mathcal{R}$
$c := \mathsf{Enc}_{pk}(m; r)$
Return c

Procedure FINALISE(m')
Return $(m \stackrel{?}{=} m')$

Figure 16: One-way game $\mathsf{OW}_{\mathsf{PKE}}$ for PKE scheme PKE.

Procedure $\mathsf{FOGen}(1^\lambda)$	**Procedure** $\mathsf{FOEnc}_{pk}(m)$	**Procedure** $\mathsf{FODec}_{sk}(e, c)$
$(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$	$r \xleftarrow{\$} \mathcal{R}_{\mathsf{FO}}$	$\hat{r} := \mathsf{Dec}_{sk}(e)$
Return (pk, sk)	$c := m \oplus \mathrm{G}(r)$	if $\hat{r} \notin \mathcal{R}_{\mathsf{FO}}$
	$h := \mathrm{H}(r, c)$	Return $\perp$
	$e := \mathsf{Enc}_{pk}(r; h)$	$\hat{h} := \mathrm{H}(\hat{r}, c)$
	Return (e, c)	if $e \neq \mathsf{Enc}_{pk}(\hat{r}; \hat{h})$
		Return $\perp$
		$m := c \oplus \mathrm{G}(\hat{r})$
		Return m

Figure 17: Fujisaki-Okamoto Transformation for PKE scheme PKE.

Definition 5.2. *Let* $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *PKE scheme with message space* $\mathcal{M}$ *and randomness space* $\mathcal{R}$*. Let* $m \in \mathcal{M}$ *and* $pk \leftarrow \mathsf{Gen}$*. We define the* min-entropy of $\mathsf{Enc}_{pk}(m)$ *as*

$$\gamma(pk, m) := -\log \max_{c \in \{0,1\}^*} \left\{ \Pr_{r \xleftarrow{\$} \mathcal{R}} [c = \mathsf{Enc}_{pk}(m; r)] \right\} .$$

PKE *is* γ-spread *if for all* $pk \leftarrow \mathsf{Gen}$ *and all* $m \in \mathcal{M}$ *we have* $\gamma(pk, m) \geq \gamma$.

Note that for any γ-spread PKE scheme appending γ' random bits to encryptions of messages immediately ensures that the scheme is $(\gamma + \gamma')$-spread at the cost of longer ciphertexts as mentioned in [FO99]. Hence, the following transformation allows one to construct a SIM-SO-CCA secure PKE from any one-way secure PKE.

5.2 The Transformation

Let $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a PKE with message space $\mathcal{M}$, randomness space $\mathcal{R}$ and ciphertext space $\mathcal{C}$. Consider the FO Transformation $\mathsf{FO}_{\mathsf{PKE}} = (\mathsf{FOGen}, \mathsf{FOEnc}, \mathsf{FODec})$, given in Figure 17 with message space $\mathcal{M}_{\mathsf{FO}} := \{0,1\}^\ell$, randomness space $\mathcal{R}_{\mathsf{FO}} := \mathcal{M}$ and ciphertext space $\mathcal{C}_{\mathsf{FO}} := \mathcal{C} \times \{0,1\}^\ell$. Therefore, let $\mathrm{G} : \mathcal{R}_{\mathsf{FO}} \to \{0,1\}^\ell$ and $\mathrm{H} : \mathcal{R}_{\mathsf{FO}} \times \{0,1\}^\ell \to \mathcal{R}$ be hash functions.

The FO encryption process is illustrated in Figure 18.

Theorem 5.3. *Let* PKE *be a OW,* γ*-spread* PKE *where* $\gamma = \mathsf{poly}(n)$*, then* $\mathsf{FO}_{\mathsf{PKE}}$ *is SIM-SO-CCA secure in the random oracle model. In particular, for any adversary* $\mathcal{A}$ *run in the* $\mathsf{REAL\text{-}SIM\text{-}SO\text{-}CCA}_{\mathsf{FO}_{\mathsf{PKE}}}$ *game that issues at most* q_h *hash queries,* q_d *decryption queries and obtains* n *ciphertexts, and every efficient PPT relation* Rel*, there exists a simulator* $\mathcal{S}$ *and an adversary* $\mathcal{B}$ *run in the* $\mathsf{OW}_{\mathsf{PKE}}$ *game with roughly the same running time as* $\mathcal{A}$ *such that*

$$\mathbf{Adv}^{\mathsf{SIM\text{-}SO\text{-}CCA}}_{\mathsf{FO}_{\mathsf{PKE}}}(\mathcal{A}, \mathcal{S}, \mathsf{Rel}, n, \lambda) \leq n \cdot \left(q_d \cdot 2^{-\gamma} + q_h \cdot \left(\frac{1}{|\mathcal{R}| - q_h} + \mathbf{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathcal{B}, \lambda) \right) \right) .$$

Intuition for the proof of Theorem 5.3 The proof idea is similar to the proof of our result on DHIES from Theorem 3.5. In the first game hop we will replace the decryption procedure such that

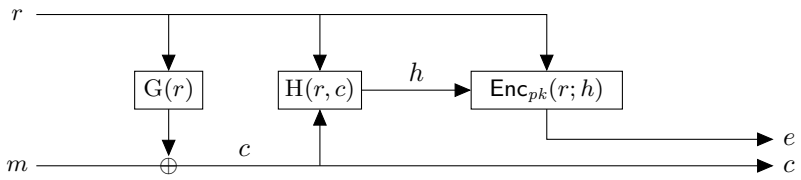

Figure 18: Structure of the Fujisaki-Okamoto encryption. We have $(e, c) = \mathsf{FOEnc}_{pk}(m; r)$.

(almost all) decryption queries can be answered correctly without the secret key. Then a statistical argument ensures that $\mathrm{H}(r_i, \cdot)$ and $\mathrm{G}(r_i)$ are still uniformly random for all $i \in [n]$ from $\mathcal{A}$'s point of view when it calls Enc. After rewriting the encryption of challenge messages and moving the programming of G and H from Enc to the G, H and Open procedure, we will use Π's one-wayness to ensure that $\mathcal{A}$ will not query $\mathrm{H}(r_i, c_i)$ or $\mathrm{G}(r_i)$ for $i \in [n]$ even after seeing the challenge ciphertexts. In a last game transition we will construct a simulator run in the ideal experiment.

Proof. We proceed in a sequence of games. Our proof of SIM-SO-CCA security is similar to the proof of IND-CCA security given in [FO13]. Note that q_h denotes the number of hash queries to G and H issued by $\mathcal{A}$. We continue with detailed descriptions of the games that are given as pseudocode in Figure 19.

Procedure Initialise
$(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$
$(r_i)_{i\in[n]} \xleftarrow{\$} \mathcal{R}_{\mathsf{FO}}$
Return pk

Procedure $\mathrm{H}(s_1, s_2)$ G_{2-4} G_{3-4} G_4
if $s_1 \in \{r_i\}_{i\in[n]}$
  if $\neg$calledEnc
    AbortEarly $:= true$
  else
    let i s.t. $s_1 = r_i$
    if $s_2 = c_i$
      if $i \notin \mathcal{I}$
        AbortHash $:= true$
      $L_{\mathrm{H}} := L_{\mathrm{H}} \cup \{s_1, s_2, \sigma_i^h\}$
if $(s_1, s_2, \cdot) \notin L_{\mathrm{H}}$
  $h_s \xleftarrow{\$} \mathcal{R}$
  $L_{\mathrm{H}} := L_{\mathrm{H}} \cup \{(s_1, s_2, h_s)\}$
Return h_s

Procedure $\mathrm{G}(t)$ G_{2-4} G_{3-4} G_4
if $t \in \{r_i\}_{i\in[n]}$
  if $\neg$calledEnc
    AbortEarly $:= true$
  else
    let i s.t. $t = r_i$
    if $i \notin \mathcal{I}$
      AbortHash $:= true$
    $L_{\mathrm{G}} := L_{\mathrm{G}} \cup \{t, \sigma_i^g \oplus m_i\}$
if $(t, \cdot) \notin L_{\mathrm{G}}$
  $g_t \xleftarrow{\$} \{0,1\}^\ell$
  $L_{\mathrm{G}} := L_{\mathrm{G}} \cup \{(t, g_t)\}$
Return g_t

Procedure Finalise(out)
Return $\mathsf{Rel}((m_i)_{i\in[n]}, \mathfrak{D}, \mathcal{I}, out)$

Procedure Enc$(\mathfrak{D})$ G_{0-2} G_{3-4}
calledEnc $:= true$
$(m_i)_{i\in[n]} \leftarrow \mathfrak{D}$
$c_i := \mathrm{G}(r_i) \oplus m_i$
$h_i := \mathrm{H}(r_i, c_i)$
$e_i := \mathsf{Enc}_{pk}(r_i; h_i)$
$\sigma_i^h \xleftarrow{\$} \{0,1\}^\ell$
$\sigma_i^h \leftarrow \mathcal{R}$
$c_i := \sigma_i^g$
$e_i := \mathsf{Enc}_{pk}(r_i; \sigma_i^h)$
Return $((e_i, c_i)_{i\in[n]})$

Procedure Open(i) G_{3-4}
$\mathcal{I} := \mathcal{I} \cup \{i\}$
$L_{\mathrm{G}} := L_{\mathrm{G}} \cup \{(r_i, \sigma_i^g \oplus m_i)\}$
$L_{\mathrm{H}} := L_{\mathrm{H}} \cup \{(r_i, c_i, \sigma_i^h)\}$
Return (m_i, r_i)

Procedure Dec(e, c) G_0
$\hat{r} := \mathsf{Dec}_{sk}(e)$
if $\hat{r} \notin \mathcal{R}_{\mathsf{FO}}$
  Return $\perp$
$\hat{h} := \mathrm{H}(\hat{r}, c)$
if $e \neq \mathsf{Enc}_{pk}(\hat{r}; \hat{h})$
  Return $\perp$
$m := c \oplus \mathrm{G}(\hat{r})$
Return m

Procedure Dec(e, c) G_{1-4}
if $\nexists(\hat{r}, c, \hat{h}) \in L_{\mathrm{H}}$ s.t. $e = \mathsf{Enc}_{pk}(\hat{r}; \hat{h})$
  Return $\perp$
let $\hat{r}$ s.t. $(\hat{r}, c, \hat{h}) \in L_{\mathrm{H}} \wedge e = \mathsf{Enc}_{pk}(\hat{r}; \hat{h})$
$m := c \oplus \mathrm{G}(\hat{r})$
Return m

Figure 19: Sequence of games used in the proof of Theorem 5.3. Boxed code is only run in those games that are given in the top right of each procedure within a box of the same style. Code not inside a box is always executed. Note that for Dec the whole procedure is replaced for better readability.

Game 0 REAL-SIM-SO-CCA$_{\mathsf{FO}_{\mathsf{PKE}}}$ game run with adversary $\mathcal{A}$ whereby hash functions G and H are modelled as random oracles. Additionally, we already moved the sampling of random coins r_i from Enc to Initialise and added a flag at the beginning of Enc. Clearly, $\mathbf{Adv}(\mathsf{REAL\text{-}SIM\text{-}SO\text{-}CCA}^{\mathcal{A}}, \mathsf{G}_0^{\mathcal{A}}) = 0$.

Game 1 We change the decryption procedure such that decryption queries can be answered without the secret key. Instead of decrypting e to obtain $\hat{r}$ and querying $(\hat{r}, c)$ to H, adversary $\mathcal{A}$ is aborted if it did not submit some tuple $(\hat{r}, c)$ to H s.t. $e = \mathsf{Enc}_{pk}(\hat{r}; \mathrm{H}(\hat{r}, c))$.

Claim 1. $\mathbf{Adv}(\mathsf{G}_0^{\mathcal{A}}, \mathsf{G}_1^{\mathcal{A}}) \leq n \cdot q_d \cdot 2^{-\gamma}$.

Proof. We call a ciphertext (e, c) *valid* if it passes decryption without aborting, i.e. for $r := \mathsf{Dec}_{sk}(e)$ we have $r \in \mathcal{R}_{\mathsf{FO}}$ and $e = \mathsf{Enc}_{pk}(r; \mathrm{H}(r, c))$. A ciphertext (e, c) is *invalid* if it is not valid.

Consider the new decryption procedure and observe that invalid ciphertexts are aborted in both procedures. Further, if a valid ciphertext is correctly decrypted in Game G_1 the same holds for Game G_0. On the other hand, a valid ciphertext in Game G_0 might result in $\perp$ in Game G_1 if $\mathcal{A}$ did not query $\mathrm{H}(\hat{r}, c)$ before, i.e. there is no entry $(\hat{r}, c, \cdot)$ in list L_{H}. Thus, $\mathbf{Adv}(\mathsf{G}_0^{\mathcal{A}}, \mathsf{G}_1^{\mathcal{A}})$ is upper-bounded by the probability that $\mathcal{A}$ submits a *valid* (e, c) to Dec while $\mathrm{H}(\hat{r}, c)$ is still undefined.

We show that $(r, c) \neq (r_i, c_i)$ for all $i \in [n]$, i.e. $\mathrm{H}(r, c)$ is independent of $\mathrm{H}(r_i, c_i)$ for all $i \in [n]$. $\mathcal{A}$ has to submit a ciphertext $(e, c) \notin \{(e_i, c_i)\}_{i \in [n]}$. If $c \neq c_i$ for all $i \in [n]$ the claim follows. Hence, assume that for some fixed $i \in [n]$ we have $c = c_i$, then $e \neq e_i$. Because $e = \mathsf{Enc}_{pk}(r; \mathrm{H}(r, c))$ it follows $(r, \mathrm{H}(r, c)) \neq (r_i, \mathrm{H}(r_i, c_i))$ by definition. Thus, either $r \neq r_i$ or $\mathrm{H}(r, c) \neq \mathrm{H}(r_i, c_i)$, implying $r \neq r_i$ as well since $c = c_i$ by assumption. Remember that Π is γ-spread, hence $\mathcal{A}$'s probability of submitting a valid (e, c) without querying H on $(\hat{r}, c)$ is at most $n \cdot 2^{-\gamma}$ for a single decryption query. It follows $\mathbf{Adv}(\mathsf{G}_0^{\mathcal{A}}, \mathsf{G}_1^{\mathcal{A}}) \leq n \cdot q_D \cdot 2^{-\gamma}$. ∎

Game 2 We add an abort condition: AbortEarly. If $\mathcal{A}$ queries $\mathrm{G}(r_i)$ or $\mathrm{H}(r_i, \cdot)$ for some $i \in [n]$ and did not call Enc, game G_2 aborts $\mathcal{A}$.

Claim 2. $\mathbf{Adv}(\mathsf{G}_1^{\mathcal{A}}, \mathsf{G}_2^{\mathcal{A}}) \leq (n \cdot q_h)/(|\mathcal{R}| - q_h)$.

Proof. Observe that r_i for all $i \in [n]$ is uniformly random from $\mathcal{A}$'s point of view when obtaining pk. Note that the game aborts if $\mathcal{A}$ chooses $r \in \{r_i\}_{i \in [n]}$ and queries $\mathrm{H}(r, c)$ for any $c \in \{0,1\}^{\ell}$ or $\mathrm{G}(r)$. Thus, $\Pr[\text{AbortEarly}]$ is upper-bounded by the sum over the probability of aborting on query i conditioned on 'AbortEarly did not happen'. Hence,

$$\Pr[\text{AbortEarly}] \leq n \cdot \sum_{i=1}^{q_h} \frac{1}{|\mathcal{R}_{\mathsf{FO}}| - (i-1)} \leq \frac{n \cdot q_h}{|\mathcal{R}| - q_h} \quad .$$

∎

Game 3 We rewrite the Enc procedure. Instead of querying $\mathrm{H}(r_i, c_i)$ (resp. $\mathrm{G}(r_i)$) we pick $\sigma_i^h \xleftarrow{\$} \mathcal{R}$ (resp. $\sigma_i^g \xleftarrow{\$} \{0,1\}^{\ell}$) uniformly random. The challenge ciphertexts will be computed as $(e_i, c_i) = (\mathsf{Enc}_{pk}(r_i; \sigma_i^h), \sigma_i^g)$ while the programming $\mathrm{H}(r_i, c_i) := \sigma_i^h$ and $\mathrm{G}(r_i) := \sigma_i^g \oplus m_i$ will only happen on $\mathcal{A}$'s call of Open(i) or query $\mathrm{H}(r_i, c_i)$, $\mathrm{G}(r_i)$, respectively.

Claim 3. $\mathbf{Adv}(\mathsf{G}_2^{\mathcal{A}}, \mathsf{G}_3^{\mathcal{A}}) = 0$.

Proof. Note that *on call* of Enc the values $\mathrm{H}(r_i, c_i)$ and $\mathrm{G}(r_i)$ are uniformly random for all $i \in [n]$. Hence, we can replace the evaluation of $\mathrm{H}(r_i, c_i)$ with some value σ_i^h chosen uniformly at random. The same argument applies for G. Thus, $G(r_i) \oplus m_i$ is uniform and we can replace it by same σ_i^g with identical distribution. The additional code within H and G ensures that for all $i \in [n]$ $\mathrm{H}(r_i, c_i)$ and $G(r_i)$ are programmed consistently, the same argument applies for the additional lines within Open. ∎

Observe that the encryptions (e_i, c_i) are independent of the sampled messages when $\mathcal{A}$ obtains $((e_i, c_i)_{i \in [n]})$. However, ciphertext (e_i, c_i) does not remain independent of m_i if $\mathcal{A}$ queries $\mathrm{H}(r_i, c_i)$, $\mathrm{G}(r_i)$ or Open(i). While the latter is not harmful, we will block aforementioned hash queries in the next game hop.

Game 4 We add another abort condition: AbortHash. Adversary $\mathcal{A}$ is aborted if it already called Enc and queries $\mathrm{H}(r_i, c_i)$ or $\mathrm{G}(r_i)$ for some $i \in [n]$ and did not call Open(i).

Claim 4. $\mathbf{Adv}(\mathsf{G}_3^{\mathcal{A}}, \mathsf{G}_4^{\mathcal{A}},) \leq n \cdot q_h \cdot \mathbf{Adv}^{\mathsf{OW}}(\mathcal{B}, \lambda)$.

Proof. Games G_3 and G_4 proceed identical until AbortHash happens. Thus we have $\mathbf{Adv}(\mathsf{G}_3^{\mathcal{A}}, \mathsf{G}_4^{\mathcal{A}}) \leq \Pr[\text{AbortHash}]$. We construct an adversary $\mathcal{B}$ against the one-wayness of Π. First, $\mathcal{B}$ obtains (pk, y) from its OW game. Adversary $\mathcal{B}$ picks $i^* \xleftarrow{\$} [n]$, $j^* \xleftarrow{\$} [q_h]$ and forwards pk to $\mathcal{A}$. On $\mathcal{A}$'s j^{*th} hash query $\mathrm{H}(s_1, \cdot)$ (resp. $\mathrm{G}(t)$), $\mathcal{B}$ returns s_1 (resp. t) to its OW_Π game.

Receiving $\mathfrak{D}$ from $\mathcal{A}$, $\mathcal{B}$ samples a message vector according to $\mathfrak{D}$ and encrypts every message as given in game G_3 for $i \in [n] \setminus \{i^*\}$. For the i^{*th} message, $\mathcal{B}$ returns (y, σ_i^g).

Assume that AbortHash happens. Then, with probability $1/n$ it will happen for $i = i^*$. In particular, $\mathcal{A}$ will not call Open(i^*) and will query $\mathrm{H}(r_{i^*}, \cdot)$ or $\mathrm{G}(r_{i^*})$ whereby r_{i^*} is the decryption of y. With probability $1/q_h$ $\mathcal{A}$ will make that query as its j^{*th} and $\mathcal{B}$ wins its OW_Π game by returning r_i. It follows $\mathbf{Adv}_\Pi^{\mathsf{OW}}(\mathcal{B}, \lambda) \geq \Pr[\text{AbortHash}] \cdot 1/(n \cdot q_{Hash})$. The claim follows by rearranging. ∎

Note that from now on the ciphertexts remain independent of the sampled messages until $\mathcal{A}$ makes an Open query.

Claim 4. There exists a simulator $\mathcal{S}$ that runs in the IDEAL-SIM-SO-CCA game such that we have $\mathbf{Adv}(\mathsf{G}_4^{\mathcal{A}}, \text{IDEAL-SIM-SO-CCA}^{\mathcal{S}}) = 0$.

Proof. We construct a simulator $\mathcal{S}$ run in the IDEAL-SIM-SO-CCA game. At first, $\mathcal{S}$ runs Gen on its own to obtain (pk, sk). Then $\mathcal{S}$ forwards pk to $\mathcal{A}$. The simulator will answer hash queries as in game G_4. Once it receives $\mathfrak{D}$ from $\mathcal{A}$, the simulator calls Enc$(\mathfrak{D})$. Simulator $\mathcal{S}$ creates 'dummy ciphertexts' as in game G_4 and sends them to $\mathcal{A}$. Assuming that $\mathcal{A}$ does not cause any Abort, $\mathcal{S}$ does not have to reveal a message m_i before $\mathcal{A}$ calls Open(i). If $\mathcal{A}$ calls Open(i), $\mathcal{S}$ issues the same query to its game, obtains m_i and programs the random oracles accordingly and forwards (r_i, m_i) to $\mathcal{A}$. $\mathcal{S}$ forwards whatever $\mathcal{A}$ outputs. ∎

Collecting the probabilities we obtain the bound as given in Theorem 5.3. ∎

Acknowledgments

We thank the reviewers of the Journal of the IET for their helpful feedback. Further, we thank Zhengan Huang and Shengli Liu for their valuable comments. Felix Heuer and Eike Kiltz were (partially) funded by a Sofja Kovalevskaja Award of the Alexander von Humboldt Foundation and the German Federal Ministry for Education and Research. Felix Heuer was also partially funded by German Research Foundation (DFG) SPP 1736, Algorithms for BIG DATA. Eike Kiltz was partially funded by ERC Project ERCC (FP7/615074). Sven Schäge is supported by Ubicrypt, the research training group 1817/1 funded by the DFG. Part of this work was done while he was employed at University College London and supported by EPSRC grant EP/J009520/1.

References

[ABR01] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In Naccache [Nac01], pages 143–158. (Cited on page 2, 4, 12.)

[BDU08] Michael Backes, Markus Dürmuth, and Dominique Unruh. OAEP is secure under key-dependent messages. In Josef Pieprzyk, editor, *ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 506–523, Melbourne, Australia, December 7–11, 2008. Springer, Berlin, Germany. (Cited on page 3.)

[BDWY12] Mihir Bellare, Rafael Dowsley, Brent Waters, and Scott Yilek. Standard security does not imply security against selective-opening. In Pointcheval and Johansson [PJ12], pages 645–662. (Cited on page 2.)

[Bea97] Donald Beaver. Plug and play encryption. In Kaliski Jr. [Kal97], pages 75–89. (Cited on page 1, 3.)

[BF06] Alexandra Boldyreva and Marc Fischlin. On the security of OAEP. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 210–225, Shanghai, China, December 3–7, 2006. Springer, Berlin, Germany. (Cited on page 3.)

[BH92] Donald Beaver and Stuart Haber. Cryptographic protocols provably secure against dynamic adversaries. In Rainer A. Rueppel, editor, *EUROCRYPT'92*, volume 658 of *LNCS*, pages 307–323, Balatonfüred, Hungary, May 24–28, 1992. Springer, Berlin, Germany. (Cited on page 1, 3.)

[BHK12] Florian Böhl, Dennis Hofheinz, and Daniel Kraschewski. On definitions of selective opening security. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 522–539, Darmstadt, Germany, May 21–23, 2012. Springer, Berlin, Germany. (Cited on page 2.)

[BHY09] Mihir Bellare, Dennis Hofheinz, and Scott Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In Joux [Jou09], pages 1–35. (Cited on page 2, 3.)

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press. (Cited on page 14.)

[BR94] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *EUROCRYPT'94*, volume 950 of *LNCS*, pages 92–111, Perugia, Italy, May 9–12, 1994. Springer, Berlin, Germany. (Cited on page 2, 3.)

[BR06] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Berlin, Germany. (Cited on page 3, 14.)

[Bro06] Daniel R. L. Brown. What hashes make RSA-OAEP secure? Cryptology ePrint Archive, Report 2006/223, 2006. `http://eprint.iacr.org/`. (Cited on page 3.)

[BWY11] Mihir Bellare, Brent Waters, and Scott Yilek. Identity-based encryption secure against selective opening attack. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 235–252, Providence, RI, USA, March 28–30, 2011. Springer, Berlin, Germany. (Cited on page 3.)

[CA06] T. Clancy and W. Arbaugh. Extensible Authentication Protocol (EAP) Password Authenticated Exchange. RFC 4746 (Informational), November 2006. (Cited on page 3.)

[CDNO97] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In Kaliski Jr. [Kal97], pages 90–104. (Cited on page 1, 3.)

[CFGN96] Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *28th ACM STOC*, pages 639–648, Philadephia, Pennsylvania, USA, May 22–24, 1996. ACM Press. (Cited on page 1, 3.)

[CHK05] Ran Canetti, Shai Halevi, and Jonathan Katz. Adaptively-secure, non-interactive public-key encryption. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 150–168, Cambridge, MA, USA, February 10–12, 2005. Springer, Berlin, Germany. (Cited on page 1, 3.)

[CS03] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM J. Comput.*, 33(1):167–226, 2003. (Cited on page 11.)

[DR08] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), August 2008. Updated by RFCs 5746, 5878, 6176. (Cited on page 3.)

[FHKW10] Serge Fehr, Dennis Hofheinz, Eike Kiltz, and Hoeteck Wee. Encryption schemes secure against chosen-ciphertext selective opening attacks. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 381–402, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany. (Cited on page 2, 3.)

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 537–554, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Berlin, Germany. (Cited on page 4, 18, 19.)

[FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, January 2013. (Cited on page 3, 4, 18, 20.)

[FOPS01] Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern. RSA-OAEP is secure under the RSA assumption. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 260–274, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Berlin, Germany. (Cited on page 3, 12.)

[Fuj12] Eiichiro Fujisaki. All-but-many encryptions: A new framework for fully-equipped UC commitments. Cryptology ePrint Archive, Report 2012/379, 2012. http://eprint.iacr.org/. (Cited on page 3.)

[Har06] B. Harris. RSA Key Exchange for the Secure Shell (SSH) Transport Layer Protocol. RFC 4432 (Proposed Standard), March 2006. (Cited on page 3.)

[HLOV11] Brett Hemenway, Benoît Libert, Rafail Ostrovsky, and Damien Vergnaud. Lossy encryption: Constructions from general assumptions and efficient selective opening chosen ciphertext security. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 70–88, Seoul, South Korea, December 4–8, 2011. Springer, Berlin, Germany. (Cited on page 2, 3.)

[Hof12] Dennis Hofheinz. All-but-many lossy trapdoor functions. In Pointcheval and Johansson [PJ12], pages 209–227. (Cited on page 2, 3.)

[Hou03] R. Housley. Use of the RSAES-OAEP Key Transport Algorithm in Cryptographic Message Syntax (CMS). RFC 3560 (Proposed Standard), July 2003. (Cited on page 3.)

[HR14] Dennis Hofheinz and Andy Rupp. Standard versus selective opening security: Separation and equivalence results. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 591–615, San Diego, CA, USA, February 24–26, 2014. Springer, Berlin, Germany. (Cited on page 1, 2.)

[Jou09] Antoine Joux, editor. *EUROCRYPT 2009*, volume 5479 of *LNCS*, Cologne, Germany, April 26–30, 2009. Springer, Berlin, Germany. (Cited on page 23, 25.)

[JQY01] Marc Joye, Jean-Jacques Quisquater, and Moti Yung. On the power of misbehaving adversaries and security analysis of the original EPOC. In Naccache [Nac01], pages 208–222. (Cited on page 4.)

[Kal97] Burton S. Kaliski Jr., editor. *CRYPTO'97*, volume 1294 of *LNCS*, Santa Barbara, CA, USA, August 17–21, 1997. Springer, Berlin, Germany. (Cited on page 23.)

[KOS10] Eike Kiltz, Adam O'Neill, and Adam Smith. Instantiability of RSA-OAEP under chosen-plaintext attack. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 295–313, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Berlin, Germany. (Cited on page 3.)

[KP09] Eike Kiltz and Krzysztof Pietrzak. On the security of padding-based encryption schemes - or - why we cannot prove OAEP secure in the standard model. In Joux [Jou09], pages 389–406. (Cited on page 3, 14.)

[LDL+14] Junzuo Lai, Robert H. Deng, Shengli Liu, Jian Weng, and Yunlei Zhao. Identity-based encryption secure against selective opening chosen-ciphertext attack. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 77–92, Copenhagen, Denmark, May 11–15, 2014. Springer, Berlin, Germany. (Cited on page 3.)

[Nac01] David Naccache, editor. *CT-RSA 2001*, volume 2020 of *LNCS*, San Francisco, CA, USA, April 8–12, 2001. Springer, Berlin, Germany. (Cited on page 22, 24, 25.)

[NSF05] T. Nadeau, C. Srinivasan, and A. Farrel. Multiprotocol Label Switching (MPLS) Management Overview. RFC 4221 (Informational), November 2005. (Cited on page 3.)

[OP01] Tatsuaki Okamoto and David Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. In Naccache [Nac01], pages 159–175. (Cited on page 5.)

[OU98] Tatsuaki Okamoto and Shigenori Uchiyama. A new public-key cryptosystem as secure as factoring. In Kaisa Nyberg, editor, *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, volume 1403 of *Lecture Notes in Computer Science*, pages 308–318. Springer, 1998. (Cited on page 4.)

[Pei14] Chris Peikert. Lattice cryptography for the internet. In Michele Mosca, editor, *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014, Waterloo, ON, Canada, October 1-3, 2014. Proceedings*, volume 8772 of *Lecture Notes in Computer Science*, pages 197–219. Springer, 2014. (Cited on page 3.)

[PJ12] David Pointcheval and Thomas Johansson, editors. *EUROCRYPT 2012*, volume 7237 of *LNCS*, Cambridge, UK, April 15–19, 2012. Springer, Berlin, Germany. (Cited on page 22, 24.)

[PW08] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 187–196, Victoria, British Columbia, Canada, May 17–20, 2008. ACM Press. (Cited on page 3.)

[Rae05] K. Raeburn. Encryption and Checksum Specifications for Kerberos 5. RFC 3961 (Proposed Standard), February 2005. (Cited on page 3.)

[Res02] E. Rescorla. Preventing the Million Message Attack on Cryptographic Message Syntax. RFC 3218 (Informational), January 2002. (Cited on page 3.)

[RT10] B. Ramsdell and S. Turner. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification. RFC 5751 (Proposed Standard), January 2010. (Cited on page 3.)

[SBZ02] Ron Steinfeld, Joonsang Baek, and Yuliang Zheng. On the necessity of strong assumptions for the security of a class of asymmetric encryption schemes. In Lynn Margaret Batten and Jennifer Seberry, editors, *ACISP 02*, volume 2384 of *LNCS*, pages 241–256, Melbourne, Victoria, Australia, July 3–5, 2002. Springer, Berlin, Germany. (Cited on page 2, 4, 6, 7.)

[Sho02] Victor Shoup. OAEP reconsidered. *Journal of Cryptology*, 15(4):223–249, 2002. (Cited on page 3.)

[Sho04a] Victor Shoup. ISO 18033-2: An emerging standard for public-key encryption. `http://shoup.net/iso/std6.pdf`, December 2004. Final Committee Draft. (Cited on page 12.)

[Sho04b] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs, 2004. shoup@cs.nyu.edu 13166 received 30 Nov 2004, last revised 18 Jan 2006. (Cited on page 14.)

[ST02] Kouichi Sakurai and Tsuyoshi Takagi. A reject timing attackon an IND-CCA2 public-key cryptosystem. In Pil Joong Lee and Chae Hoon Lim, editors, *Information Security and Cryptology - ICISC 2002, 5th International Conference Seoul, Korea, November 28-29, 2002, Revised Papers*, volume 2587 of *Lecture Notes in Computer Science*, pages 359–373. Springer, 2002. (Cited on page 4.)