Analysis of Random Oracle Instantiation Scenarios for OAEP and other Practical Schemes

Alexandra Boldyreva[1] and Marc Fischlin[2] *

[1] College of Computing, Georgia Institute of Technology,
801 Atlantic Drive, Atlanta, GA 30332, USA
aboldyre@cc.gatech.edu www.cc.gatech.edu/~aboldyre

[2] Institute for Theoretical Computer Science, ETH Zürich, Switzerland
marc.fischlin@inf.ethz.ch www.fischlin.de

Abstract. We investigate several previously suggested scenarios of instantiating random oracles (ROs) with "realizable" primitives in cryptographic schemes. As candidates for such "instantiating" primitives we pick perfectly one-way hash functions (POWHFs) and verifiable pseudorandom functions (VPRFs). Our analysis focuses on the most practical encryption schemes such as OAEP and its variant PSS-E and the Fujisaki-Okamoto hybrid encryption scheme. We also consider the RSA Full Domain Hash (FDH) signature scheme. We first show that some previous beliefs about instantiations for some of these schemes are not true. Namely we show that, contrary to Canetti's conjecture, in general one cannot instantiate either one of the two ROs in the OAEP encryption scheme by POWHFs without losing security. We also confirm through the FDH signature scheme that the straightforward instantiation of ROs with VPRFs may result in insecure schemes, in contrast to regular pseudorandom functions which can provably replace ROs (in a well-defined way). But unlike a growing number of papers on negative results about ROs, we bring some good news. We show that one can realize *one* of the two ROs in a variant of the PSS-E encryption scheme and *either one* of the two ROs in the Fujisaki-Okamoto hybrid encryption scheme through POWHFs, while preserving the IND-CCA security in both cases (still in the RO model). Although this partial instantiation in form of substituting only one RO does not help to break out of the random oracle model, it yet gives a better understanding of the necessary properties of the primitives and also constitutes a better security heuristic.

1 Introduction

The random oracle (RO) model, introduced by Fiat and Shamir [15] and refined by Bellare and Rogaway [4], has been suggested as a trade-off between provable security and practical requirements for efficiency. Schemes and proofs in this

* Part of the work done while both authors were at the University of California, San Diego. The second author was supported by the Emmy Noether Programme Fi 940/1-1 of the German Research Foundation (DFG).

nowadays well-established model make the idealized assumption that all parties have oracle access to a truly random function. Availability of such a random oracle often allows to find more efficient solutions than in the standard model. In practice, it is then assumed that the idealized random function is instantiated through a "good" cryptographic hash function, like SHA-1 or a variation thereof.

The random oracle methodology has gained considerable attention as a design method. Numerous cryptographic schemes proven secure in the RO model have been proposed and some of them are implemented and standardized. The best known example is presumably the OAEP encryption scheme [5, 18]. However, even though a RO-based scheme instantiated with a "good" hash function is usually believed to remain secure in the standard model, proofs in the RO model do not technically guarantee this, but merely provide some evidence of security.

Moreover, several recent works [10, 21, 23, 2, 19] raised concerns by proving that the random oracle model is not sound. Here lack of soundness refers to the situation when a scheme allows a security proof in the random oracle model but any instantiation of the scheme with any real function family is insecure in the standard model. Such schemes are called "uninstantiable" in [2]. While these results are certainly good reminders about the gap between the RO model and the standard model, the defenders of the RO model and practitioners are assured by the fact that most uninstantiable schemes involve somewhat esoteric examples, in terms of either a construction or sometimes with respect to a security goal.

Towards Instantiating Random Oracles for Practical Schemes. In this work we continue to study security of instantiated schemes designed in the RO model. But unlike the aforementioned works we turn our attention to the most practical cryptographic schemes such as OAEP encryption, the full domain hash (FDH) signature scheme, hybrid encryption schemes obtained via Fujisaki-Okamoto transform [17] and the PSS-E encryption scheme, an OAEP variant due to Coron et al. [12]. Our goal is different, too. We do not show that these schemes are uninstantiable (this would be really bad news). It also seems unrealistic to instantiate these schemes such that they are still efficient and provably secure in the standard model (though this would be great news). Rather, we investigate several possible instantiation scenarios for to these practical schemes somewhere in between.

As candidates for substituting random oracles we consider two primitives with known constructions whose security definitions capture various strong properties of the ideal random oracles, and which have actually been suggested as possible instantiations of random oracles [9, 13]. These are the perfectly one-way hash functions (POWHFs) [9, 24] and verifiable pseudorandom functions (VPRFs) [22].

The notion of perfectly one-way hash functions has been suggested by Canetti [9] (and was originally named "oracle hashing") to identify and realize useful properties of random oracles. POWHFs are special randomized collision-resistant one-way functions which hide all information about preimages. Canetti [9], and subsequently [24, 16], gave several constructions of such POWHFs, based on specific number-theoretic and on more general assumptions. Usually, these

POWHFs satisfy another property that requires the output look random, even to an adversary who knows "a little" about the inputs. We will refer to such POWHFs as pseudorandom. In [9] it is proved that a hybrid encryption scheme of Bellare and Rogaway [4] secure against chosen-plaintext attacks (IND-CPA secure) can be securely instantiated with a pseudorandom POWHF, and Canetti conjectured that one could also replace one of the two random oracles in OAEP by a POWHF without sacrificing security against chosen-ciphertext attacks (IND-CCA security) in the RO model.

Verifiable pseudorandom functions have been proposed by Micali et al. in [22]. They resemble pseudorandom functions in that their outputs look random. But their outputs also include proofs that allow verifying the correctness of the outputs with respect to a previously announced public key. In contrast to POWHFs, which are publicly computable given the inputs, VPRFs involve a secret key and therefore their global usage requires the participation of a third party or a device with a tamper-proof key. It is folklore that a secure RO scheme instantiated with a PRF implemented by a third party, will remain secure in the standard model. As suggested in [13] an application scenario for VPRFs, that lowers the amount of trust put on the third party, is a trusted third party implementing a VPRF, say, through a web interface. Now the correctness of the given image can be verified with the consistency proof, and this can be done locally, without further interactions with the third party. We note that this scenario is suitable mostly for digital signatures and not encryption schemes, as the third party has to know the inputs.

Negative Results. In this work we show that the above intuition about securely replacing random oracles by the aforementioned primitives may be incorrect. We first disprove Canetti's [9] conjecture for the OAEP encryption scheme [5] saying that one can instantiate one of the two RO in the OAEP scheme without losing security (still in the RO model). Recall that, in the OAEP scheme with a (partial one-way) trapdoor permutation f, a ciphertext is of the form $C = f(s||t)$ for $s = G(r) \oplus M||0^k$ and $t = r \oplus H(s)$ for random r. For the security proof of OAEP it is assumed that both G and H are modeled as random oracles.

We prove that, with respect to general (partial one-way) trapdoor permutations f, one cannot replace either of the two random oracles G, H in OAEP by arbitrary pseudorandom POWHFs without sacrificing chosen-ciphertext security. Our negative result follows Shoup's idea to identify weaknesses in the original OAEP security proof [26], and holds relative to a malleable trapdoor function oracle from which a specific function f is derived. Yet, unlike [26], we consider *partial* one-way functions f which suffice to prove OAEP to be IND-CCA in the random oracle model [18]. Our construction also requires to come up with a malleable yet pseudorandom POWHF. We note that our impossibility result is not known to hold for the special case of the RSA function $f : x \mapsto x^e \bmod N$, yet indicates that further assumptions about the RSA function may be necessary to replace one of the random oracles by a POWHF.

The idea for OAEP can be also applied to the Full Domain Hash (FDH) signature scheme, where signatures are of the form $S = f^{-1}(H(M))$. Transferring our OAEP result shows that for a specific class of trapdoor permutations f the instantiation of the RO H through a POWHF can result in an insecure implementation. But here we also show that FDH becomes insecure when H is instantiated the obvious way with a VPRF, even for any trapdoor permutation f such as RSA. By obvious we mean that the pseudorandom value $H(M)$ and its correctness proof π is concatenated with the signature S, such that one can verify the signature's validity by verifying π and checking that $f(S) = H(M)$. Note that VPRFs already provide secure signatures directly, so substituting the random oracle by a VPRF in a signature scheme seems to be moot. However, our goal is to see if VPRFs are a good instantiation in general. Second, one might want additional properties of the signature scheme which FDH gives but not the VPRF, e.g., if used as a sub-protocol in Chaum's blind signature scheme [11]. We note that, independently of our work, [14] obtained a related result about FDH signatures, showing that *any* instantiation of H fails relative to a specific trapdoor function oracle f (whereas our result holds for arbitrary trapdoor functions such as RSA but for a specific instantiation candidate).

Positive Results. Our results show that the RO model is very demanding and even functions with extremely strong properties often cannot securely replace random oracles. However this does not mean that no real function family can be securely used in place of any random oracle. As mentioned, Canetti [9] for example shows how to instantiate an IND-*CPA* secure encryption scheme through POWHFs. Accordingly, we look beyond our negative results and present some positive results, but this time for IND-*CCA* secure encryption schemes.

We first show the following positive results for a variation of the PSS-E encryption scheme introduced by Coron et al. [12]. In the original PSS-E encryption scheme ciphertexts are given by $C = f(\omega||s)$ for $\omega = H(M||r)$ and $s = G(\omega) \oplus M||r$. The PSS transform has been originally proposed by Bellare and Rogaway in the RSA-based signature scheme with message recovery [6]. Coron et al. showed that PSS is a universal transform in that it can also be used for RSA-based encryption for random oracles G, H, achieving chosen-ciphertext security as an alternative to OAEP.

Here we consider a variation PSS-I, where ciphertexts have the form $(f(\omega), s)$ for $\omega = H(M||r)$ and $s = G(\omega) \oplus M||r$, i.e., where the s-part is moved outside of the trapdoor permutation. We prove that for any trapdoor function f the random oracle G can be *i*nstantiated (hence the name PSS-I) with a pseudorandom POWHF such that the scheme remains IND-CCA secure (in the RO model). Interestingly, this also comes with a weaker assumption about the function f. While the original PSS-E scheme has been proven secure for *partial* one-way trapdoor permutations, our scheme PSS-I (with the G-instantiation through a POWHF) works for *any* trapdoor permutation f. A similar observation was made in [20] for OAEP. Concerning the substitution of the H-oracle (even if G is assumed to be a random oracle) we were neither able to prove or disprove that this oracle can be instantiated by some primitive with known construction. We remark that this result about PSS-I is in sharp contrast to OAEP where neither oracle can be replaced by such a POWHF.

As an example where we can replace two random oracles (individually) we discuss the Fujisaki-Okamoto transformation [17] for combining asymmetric and symmetric encryption schemes, where a ciphertext is given by $C = (\mathcal{E}_{\text{asym}}(pk, \sigma; H(\sigma, M)), \mathcal{E}_{\text{sym}}(G(\sigma), M))$ for random σ. It provides an IND-CCA secure hybrid encryption under weak security properties of the two encryption schemes (for random oracles G, H). We show that the scheme remains IND-CCA secure in the RO model if the oracle G is instantiated with a pseudorandom POWHF. We also show that one can instantiate oracle H through a POWHF (for random oracle G) but this requires a strong assumption about the joint security of the POWHF and the asymmetric encryption scheme. Hence, for the Fujisaki-Okamoto transformation both random oracles can be instantiated separately (albeit under a very strong assumption in case of the H oracle).

Our technical results do not mean that one scheme is "more" or "less" secure than the other one, just because one can substitute one random oracle by a primitive like POWHFs. In our positive examples there are usually two random oracles and, replacing one, the resulting scheme is still cast in the random oracle model. Yet, we believe that attenuating the assumption is beneficial, as substituting even one oracle by more "down-to-earth" cryptographic primitives gives a better understanding of the required properties, and it also provides a better heuristic than merely assuming that the hash function behaves as a random oracle.

Organization. We give the basic definitions of the two primitives, POWHFs and VPRFs, in Section 2. In Section 3 we show our negative result about instantiating one of the random oracles in OAEP through a POWHF. We then show that in Section 4 that PSS-I admits such an instantiation for one oracle. Section 5 presents the Fujisaki-Okamoto transformation as an example of a scheme where we can replace both random oracles by POWHFs. The FDH scheme and its instantiation through VPRFs are discussed in Section 6.

2 Preliminaries

If x is a binary string, then $|x|$ denotes its length, and if $n \geq 1$ is an integer, then $|n|$ denotes the length of its binary encoding, meaning the unique integer ℓ such that $2^{\ell-1} \leq n < 2^{\ell}$. The string-concatenation operator is denoted "$\|$". If S is a set then $x \xleftarrow{\$} S$ means that the value x is chosen uniformly at random from S. More generally, if D is a probability distribution on S then $x \xleftarrow{D} S$ means that the value x is chosen from set S according to D. If $\mathcal{A}$ is a randomized algorithm with a single output then $x \xleftarrow{\$} \mathcal{A}(y, z, \dots)$ means that the value x is assigned the output of $\mathcal{A}$ for input $(y, z, \dots)$. We let $[A(y, z, \ldots)]$ denote the set of all points having positive probability of being output by A on inputs y, z, etc. A (possibly probabilistic) algorithm is called efficient if it runs in polynomial time in the input length (which, in our case, usually refers to polynomial time in the security parameter).

In the full version of the paper [8] we recall the definitions of asymmetric encryption schemes, their security against chosen-plaintext attacks (IND-CPA security) and chosen-ciphertext attacks (IND-CCA security), of deterministic symmetric encryption schemes, also known as data encapsulation mechanisms or one-time symmetric encryption schemes, and their IND-CPA security (that is a weaker notion than the standard IND-CPA security), and of digital signature schemes and their security against existential unforgeability under chosen-message attacks. For simplicity we give all definitions in the standard model. To extend these definitions to the random oracle model, all algorithms including the adversary get oracle access to one or more random functions $G, H, \ldots$, drawn from the set of all mappings from domain A_k to some range B_k (possibly distinct for different oracles). Here, the parameter k and therefore the domain and the range are usually determined by the cryptographic scheme in question.

2.1 Perfectly One-Way Hash Functions

Perfectly one-way hash functions describe (probabilistic) collision-resistant hash functions with perfect one-wayness. The latter refers to the strong secrecy of a preimage x, even if some additional information about x besides the hash value are known. For this purpose [9] introduces the notion of a function hint which captures these side information. One assumes, though, that it is infeasible to recover the entire value x from $\mathsf{hint}(x)$, else the notion becomes trivial. More formally, a (possibly randomized) function $\mathsf{hint}\colon \{0,1\}^{m(k)} \to \{0,1\}^{n(k)}$, where m, n are polynomials, is *uninvertible* with respect to a probability distribution $\mathcal{X} = (\mathcal{X}_k)_{k\in\mathbb{N}}$ if for any probabilistic polynomial-time adversary $\mathcal{I}$ and x taken from $\mathcal{X}_k$, the probability $\Pr\left[\,\mathcal{I}(1^k, \mathsf{hint}(x)) = x\,\right]$ is negligible in k.

In the sequel we usually restrict ourselves to efficient and sufficiently smooth distributions. That is, a probability distribution $\mathcal{X} = (\mathcal{X}_k)_{k\in\mathbb{N}}$ is efficient if it can be computed in polynomial time in k; it is *well-spread* if the min-entropy of $\mathcal{X}$ is superlogarithmic in k.

Definition 1. ***[Perfectly One-Way Hash Function]*** *Let $\mathcal{K}$ be an efficient key generation algorithm that takes input 1^k for $k \in \mathbb{N}$ and outputs a function key K of length $l(k)$; let $\mathcal{H}$ be an efficient evaluation algorithm that takes a function key K, input $x \in \{0,1\}^{m(k)}$ and randomness $r \in Coins(K)$ for some fixed polynomial $m(k)$ and returns a hash value $y \in \{0,1\}^{n(k)}$; let $\mathcal{V}$ be an efficient verification algorithm that takes a function key K, an input $x \in \{0,1\}^{m(k)}$ and a hash value $y \in \{0,1\}^{n(k)}$ and outputs a decision bit. The tuple $\mathsf{POWHF} = (\mathcal{K}, \mathcal{H}, \mathcal{V})$ is called a perfectly one-way hash function (with respect to the well-spread, efficient distribution $\mathcal{X} = (\mathcal{X}_k)_{k\in\mathbb{N}}$ and the uninvertible function hint) if the following holds:*

1. *Completeness: For any $k \in \mathbb{N}$, any key $K \in [\mathcal{K}(1^k)]$, any $r \in Coins(K)$, any $x \in \{0,1\}^{m(k)}$ we have $\mathcal{V}(K, x, \mathcal{H}(K, x, r)) = 1$.*
2. *Collision-resistance: For every efficient adversary $\mathcal{C}$ the following holds. For $k \in \mathbb{N}$ pick $K \xleftarrow{\$} \mathcal{K}(1^k)$ and let $(x, x', y) \xleftarrow{\$} \mathcal{C}(K)$. Then $\Pr\ [\ \mathcal{V}(K, x, y) = 1 \wedge \mathcal{V}(K, x', y) = 1 \wedge x \neq x'\ ]$ is negligible in k.*

3. *Perfect one-wayness (with respect to* $\mathcal{X}, \mathsf{hint}$*): For any efficient adversary* $\mathcal{A}$ *with binary output the following random variables are computationally indistinguishable:*
   - *Let* $K \xleftarrow{\$} \mathcal{K}(1^k)$, $r \xleftarrow{\$} Coins(K)$, $x \xleftarrow{\mathcal{X}_k} \{0,1\}^{m(k)}$.
     Output $(K, x, \mathcal{A}(K,\ \mathsf{hint}(x),\ \mathcal{H}(K,x,r)))$.
   - *Let* $K \xleftarrow{\$} \mathcal{K}(1^k)$, $r \xleftarrow{\$} Coins(K)$, $x, x' \xleftarrow{\mathcal{X}_k} \{0,1\}^{m(k)}$.
     Output $(K, x, \mathcal{A}(K,\ \mathsf{hint}(x),\ \mathcal{H}(K,x',r)))$.

The perfectly one-way hash function may have the following additional properties:

4. *Public randomness:* $\mathcal{H}$ *can be written as* $\mathcal{H}(K,x,r) = (r, \mathcal{H}^{pr}(K,x,r))$ *for another function* $\mathcal{H}^{pr}\colon \{0,1\}^{l(k)} \times \{0,1\}^{m(k)} \times Coins(K) \to \{0,1\}^{n(k)-|r|}$ *for any* $k \in \mathbb{N}$, *any* $K \in [\mathcal{K}(1^k)]$, *any* $x \in \{0,1\}^{m(k)}$ *and any* $r \in Coins(K)$.
5. *Pseudorandomess (with respect to* $\mathcal{X}, \mathsf{hint}$*): The function acts a pseudorandom generator such that the following random variables are computationally indistinguishable:*
   - *Let* $K \xleftarrow{\$} \mathcal{K}(1^k)$, $r \xleftarrow{\$} Coins(K)$, $x \xleftarrow{\mathcal{X}_k} \{0,1\}^{m(k)}$.
     Output $(K, \mathsf{hint}(x), \mathcal{H}(K,\ x,\ r))$.
   - *Let* $K \xleftarrow{\$} \mathcal{K}(1^k)$, $x \xleftarrow{\mathcal{X}_k} \{0,1\}^{m(k)}$, *and* $U \xleftarrow{\$} \{0,1\}^{n(k)}$.
     Output $(K, \mathsf{hint}(x),\ U)$.

As pointed out in [9] the notion of an uninvertible function is weaker than the one of a one-way function. For example, $\mathsf{hint}(\cdot) = 0$, which reveals no information about x, is uninvertible but not one-way. We call this function the *trivial* uninvertible function. In fact, several constructions of POWHF based on the Decisional Diffie-Hellman assumption [9] and on more general assumptions like one-way permutations and regular hash functions [9, 24, 16] have been suggested in the literature. They are provably *pseudorandom* POWHFs with respect to trivial uninvertible function hint. For other uninvertible functions hint they are conjectured to remain secure, yet a formal proof is missing.

In this paper we will mostly consider perfectly one way function families with public randomness as this is a way to ensure correct function re-computation on the same input by different parties, needed for some encryption schemes functionality. All previous constructions [9, 24, 16] have been designed to meet this notion. For simplicity we will often use the notation $y \leftarrow \mathcal{H}_K(x,r)$ for $y \leftarrow \mathcal{H}(K,x,r)$ and $y \xleftarrow{\$} \mathcal{H}_K(x)$ for $r \xleftarrow{\$} \mathrm{Coins}(K), y \leftarrow \mathcal{H}(K,x,r)$, and we often define a hash function with public randomness by just specifying $\mathcal{H}^{\mathrm{pr}}$.

2.2 Verifiable Pseudorandom Functions

A verifiable pseudorandom function, defined in [22], is a pseudorandom function with an additional public key allowing to verify consistency of values. Any value for which one has not seen the proof should still look random:

Definition 2. ***[Verifiable Pseudorandom Function]*** *Let $\mathcal{K}$ be an efficient key generation algorithm that takes input 1^k for $k \in \mathbb{N}$ and outputs a function key and a verification key (fk, vk); let $\mathcal{H}$ be an efficient evaluation algorithm that takes the key fk, input $x \in \{0,1\}^*$ and returns the output $y \in \{0,1\}^{n(k)}$ and a proof $\pi \in \{0,1\}^{l(k)}$ for some fixed polynomials l, n; let $\mathcal{V}$ be an efficient verification algorithm that takes vk, x, y and π and returns a bit. The triple* $\mathsf{VPRF} = (\mathcal{K}, \mathcal{H}, \mathcal{V})$ *is called a verifiable pseudorandom function if the following holds:*

1. *Completeness: For any $(vk, fk) \in [\mathcal{K}(1^k)], x \in \{0,1\}^*$ and $(y, \pi) \in [\mathcal{H}(fk, x)]$, $\mathcal{V}(vk, x, y, \pi) = 1$.*
2. *Uniqueness: There exists a negligible function $\nu(\cdot)$ such that for any $(vk, fk) \in [\mathcal{K}(1^k)]$, any $x \in \{0,1\}^*, y_0 \neq y_1 \in \{0,1\}^{n(k)}, \pi_0, \pi_1 \in \{0,1\}^{l(k)}$ we have $\Pr[\, \mathcal{V}(vk, x, y_b, \pi_b) = 1\,] \leq \nu(k)$ for either $b = 0$ or $b = 1$.*
3. *Pseudorandomness: For any efficient algorithm $\mathcal{A}$ that has access to an oracle and the following experiment*

   Experiment $\mathbf{Exp}^{\text{vprf-}ind}_{\mathsf{VPRF},\mathcal{A}}(1^k)$

   $b \xleftarrow{\$} \{0,1\}$

   $(fk, vk) \xleftarrow{\$} \mathcal{K}(1^k)$

   $(x, state) \xleftarrow{\$} \mathcal{A}^{\mathcal{H}(fk,\cdot)}$ *where x has never been submitted to oracle $\mathcal{H}(fk, \cdot)$*

   If $b = 0$ then $(y, \pi) \xleftarrow{\$} \mathcal{H}(fk, x)$ else $y \xleftarrow{\$} \{0,1\}^{n(k)}$ EndIf

   $d \xleftarrow{\$} \mathcal{A}^{\mathcal{H}(fk,\cdot)}(y, state)$ *where x has never been submitted to oracle $\mathcal{H}(fk, \cdot)$*

   the difference $\Pr\left[\mathbf{Exp}^{\text{vprf-}ind}_{\mathsf{VPRF},\mathcal{A}}(1^k) = b\right] - 1/2$ *is negligible in k.*

3 (In)Security of OAEP Instantiations

Here we show that, for general trapdoor permutations, instantiating any of the two random oracles in OAEP with a pseudorandom POWHF does not yield a secure scheme.

3.1 OAEP Encryption Scheme

We first recall the OAEP encryption scheme [5]. It is parameterized by integers k, k_0 and k_1 (where k_0, k_1 are linear in k) and makes use of a trapdoor permutation family F with domain and range $\{0,1\}^k$ and two random oracles

$$G\colon \{0,1\}^{k_0} \to \{0,1\}^{k-k_0} \quad \text{and} \quad H\colon \{0,1\}^{k-k_0} \to \{0,1\}^{k_0}$$

The message space is $\{0,1\}^{k-k_0-k_1}$. The scheme $\mathrm{OAEP}^{G,H}[F] = (\mathcal{EK}, \mathcal{E}, \mathcal{D})$ are defined as follows:

- $\mathcal{EK}(1^k)$: Pick a permutation f from F at random. Let pk specify f and let sk specify f^{-1}.
- $\mathcal{E}(pk, M)$: Compute $r \xleftarrow{\$} \{0,1\}^{k_0}$, $s \leftarrow (m\|0^{k_1}) \oplus G(r)$ and $t \leftarrow r \oplus H(s)$. Output $C \leftarrow f(s\|t)$.

– $\mathcal{D}(sk, C)$: Compute $s\|t \leftarrow f^{-1}(C), r \leftarrow t \oplus H(s)$ and $M \leftarrow s \oplus G(r)$. If the last k_1 bits of M are zeros, then return the first $k - k_0 - k_1$ bits of M. Otherwise, return $\perp$.

The encryption scheme $\mathrm{OAEP}^{G,H}[F]$ is proven to be IND-CCA secure in the RO model if the underlying permutation family F is partial one-way [18]. Partial one-wayness is a stronger notion than one-wayness; for the definitions see [18].

3.2 Insecurity of Instantiating the *G*-Oracle in OAEP with POWHFs

We first consider the OAEP scheme where the G-oracle is instantiated with a pseudorandom POWHF. Informally, a key specifying an instance of POWHF becomes a part of the public key and each invocation of the G-oracle is replaced with the function evaluation, such that in the encryption algorithm a new randomness for the function evaluation is picked and becomes part of the ciphertext, and in the decryption algorithm the function is re-computed using the given randomness. More formally:

Let $\mathsf{POWHF} = (\mathcal{K}, \mathcal{G}, \mathcal{V})$, where $\mathcal{K} : \{1^k | k \in \mathbb{N}\} \to \{0,1\}^k$, $\mathcal{G}\colon \{0,1\}^k \times \{0,1\}^{k_0} \times \mathrm{Coins}(K) \to \{0,1\}^{k-k_0}$ and $\mathcal{V}\colon \{0,1\}^k \times \{0,1\}^{k_0} \times \{0,1\}^{k-k_0} \to \{0,1\}$, be a perfectly one-way pseudorandom hash function with public randomness. An *instantiation of the G-oracle* in the $\mathrm{OAEP}^{G,H}[F]$ encryption scheme with $\mathsf{POWHF} = (\mathcal{K}, \mathcal{G}, \mathcal{V})$ results in the following encryption scheme $\mathrm{OAEP}^{\mathsf{POWHF},H}[F] = (\mathcal{EK}, \mathcal{E}, \mathcal{D})$

– $\mathcal{EK}(1^k)$: Pick a random permutation f on $\{0,1\}^k$ and sample a POWHF key $K \xleftarrow{\$} \mathcal{K}(1^k)$. Let pk specify f and also contain K, and let sk specify f^{-1} and also contain K.
– $\mathcal{E}(pk, M)$: Pick randomness $r \xleftarrow{\$} \{0,1\}^{k_0}$ for encryption and $r_{\mathcal{G}} \xleftarrow{\$} \mathrm{Coins}(K)$ for the POWHF. Compute $y \leftarrow \mathcal{G}_K^{\mathrm{pr}}(r, r_{\mathcal{G}})$, $s \leftarrow (M\|0^{k_1}) \oplus y$ and $t \leftarrow r \oplus H(s)$. Let $C \leftarrow f(s\|t)$ and output $(r_{\mathcal{G}}, C)$.
– $\mathcal{D}(sk, (r_{\mathcal{G}}, C))$: Compute $s\|t \leftarrow f^{-1}(C), r \leftarrow t \oplus H(s), M \leftarrow s \oplus\ \mathcal{G}^{\mathrm{pr}}(r, r_{\mathcal{G}})$. If the last k_1 bits of M are zeros, then return the first $k - k_0 - k_1$ bits of M. Otherwise, return $\perp$.

We note that for simplicity we assume that $r_{\mathcal{G}}$, the randomness output by $\mathcal{G}_K$, is a public part of the ciphertext. If it was possible to tamper this value $r_{\mathcal{G}}$ into $r'_{\mathcal{G}}$ for a given ciphertext, such that this yields the same hash value, $\mathcal{G}_K^{\mathrm{pr}}(r, r_{\mathcal{G}}) = \mathcal{G}_K^{\mathrm{pr}}(r, r'_{\mathcal{G}})$, then it would be obviously easy to mount a successful chosen-ciphertext attack. To prevent such attacks one can in principle demand that such collisions for the hash function are infeasible to find —most known constructions [9, 24, 16] have this additional property— or one can protect $r_{\mathcal{G}}$ by some other means. We do not complicate the instantiation here, as our attack already succeeds without changing $r_{\mathcal{G}}$, e.g., the attack would even work if $r_{\mathcal{G}}$ was encrypted (separately or inside f) or authenticated.

Intuition. Before we present our results in detail we provide some intuition. First we construct malleable POWHFs, i.e., for which $\mathcal{G}_K(x, r) \oplus \Delta =$

$\mathcal{G}_K(x \oplus \delta, r)$ for some δ, Δ. We show how to construct such primitives in [8]. Our construction assumes that one-way permutations exist and employs the pseudorandom function tribe ensembles of [16] (which are one possibility to build POWHFs). Assume that either RO in the $\mathrm{OAEP}^{G,H}[F]$ encryption scheme is instantiated with such a POWHF. Here F is a partial one-way trapdoor permutation family. Now given the challenge ciphertext $C^* = f(s^*\|t^*)$ of some message M_b where f is an instance of F, an adversary $\mathcal{A}$ can find δ, Δ such that $C = f((s^*\|t^*) \oplus \delta)$ is a valid encryption of $M_b \oplus \Delta$, and given the decryption of this ciphertext one can easily compute M_b.

The only problem is that, although flipping bits by penetrating the POWHF is easy by construction, how can $\mathcal{A}$ compute $f((s^*\|t^*) \oplus \delta)$ without knowing $s^*\|t^*$? Here we use the idea of Shoup [26] about the existence of XOR-malleable trapdoor permutations which allow such modifications. We note that the attack is not known to work for OAEP with the RSA trapdoor family, but it nevertheless shows that security may fail in general if a RO is instantiated with a POWHF.

Our approach is somewhat similar to the attacks Shoup used to show that for a XOR-malleable one-way trapdoor permutation family F the encryption scheme $\mathrm{OAEP}^{G,H}[F]$ is *not* IND-CCA secure in the RO model. However, Shoup's attack does not work if F is partial one way, and, moreover, for such F the scheme $\mathrm{OAEP}^{G,H}[F]$ has been proven IND-CCA secure in the RO model [18]. Our attacks work even if F is partial one way.

Theorem 1. *Let $\mathsf{POWHF}' = (\mathcal{K}', \mathcal{G}', \mathcal{V}')$ be a pseudorandom POWHF with public randomness (with respect to the uniform distribution and some uninvertible function hint). Then there exists a pseudorandom $\mathsf{POWHF} = (\mathcal{K}, \mathcal{G}, \mathcal{V})$ with public randomness (with respect to the uniform distribution and hint) and an oracle relative to which there is a partial one-way permutation family F, such that $OAEP^{\mathsf{POWHF},H}[F]$, an instantiation of the G-oracle in the $OAEP^{G,H}[F]$ encryption scheme with POWHF, is* not *IND-CCA in the RO model.*

Recall that we can assume that POWHF is malleable in the sense that $\mathcal{G}_K^{\mathrm{pr}}(x, r) \oplus 1||0^{n-1} = \mathcal{G}_K^{\mathrm{pr}}(x \oplus 1||0^{m-1}, r)$ for all k, x, r (we show how to construct such POWHFs form the given POWHF' in [8]). We now define a compliant XOR-malleable permutation family. We slightly strengthen the original definition of Shoup [26].

Definition 3. *A permutation family F is XOR-malleable if there exists an efficient algorithm U, such that on inputs a random instance permutation f from F with domain $\{0,1\}^k$ and $f(t)$ for random $t \in \{0,1\}^k$ and any $\delta \in \{0,1\}^k$, algorithm $U(f, f(t), \delta)$ outputs $f(t \oplus \delta)$ with non-negligible probability (in k).*

Even though Shoup uses a weaker definition of XOR-malleability, where U's success probability is also over the random choice of $\delta \in \{0,1\}^k$, his proof in [26] is also valid for the stronger Definition 3 with fixed δ:

Fact 1 ([26]). *There exists an oracle relative to which XOR-malleable one-way trapdoor permutations exist.*

Now we are ready to prove the theorem of the insecure instantiation of the G-oracle in OAEP. We present the formal proof in [8]. The idea is to construct the trapdoor permutation family F as $f(s\|t) = f'_{\text{left}}(s)\|f'_{\text{right}}(t)$ for random instances $f'_{\text{left}}, f'_{\text{right}}$ of the malleable family F'. Then an adversary $\mathcal{A}$ gets a challenge ciphertext $(r^*_{\mathcal{G}}, C^*_{\text{left}}\|C^*_{\text{right}})$ of one of two messages M_0, M_1, and invokes U to modify the right part to $C_{\text{right}} \leftarrow U(f'_{\text{right}}, C^*_{\text{right}}, 1\|0^{k_0-1})$. Submitting the ciphertext $(r^*_{\mathcal{G}}, C^*_{\text{left}}\|C_{\text{right}})$ to the decryption oracle is a valid ciphertext for the message $M_b \oplus 1\|0^{k-k_0-k_1-1}$ because for

$$(C^*_{\text{left}}||C^*_{\text{right}}) = (f'_{\text{left}}(s^*)||f'_{\text{right}}(t^*)),\ s^* = M_b||0^{k_0} \oplus \mathcal{G}^{\text{pr}}_K(r^*, r^*_{\mathcal{G}}),\ t^* = r^* \oplus H(s^*)$$

we have:

$$\begin{aligned}
C_{\text{right}} &= f'_{\text{right}}\big(t^* \oplus 1\|0^{k_0-1}\big) = f'_{\text{right}}\big((r^* \oplus 1\|0^{k_0-1}) \oplus H(s^*)\big) \\
C^*_{\text{left}} &= f'_{\text{left}}(s^*) = f'_{\text{left}}\big(M_b||0^{k_0} \oplus \mathcal{G}^{\text{pr}}_K(r^*, r^*_{\mathcal{G}})\big) \\
&= f'_{\text{left}}\big((M_b||0^{k_0} \oplus 1||0^{k-k_0-1}) \oplus (\mathcal{G}^{\text{pr}}_K(r^*, r^*_{\mathcal{G}}) \oplus 1||0^{k-k_0-1})\big) \\
&= f'_{\text{left}}\big((M_b||0^{k_0} \oplus 1||0^{k-k_0-1}) \oplus \mathcal{G}^{\text{pr}}_K(r^* \oplus 1||0^{k_0-1}, r^*_{\mathcal{G}})\big)
\end{aligned}$$

The answer of the decryption oracle now allows to determine the bit b easily.

3.3 Insecurity of Instantiating the *H*-Oracle in OAEP with POWHFs

For substituting the H-oracle we obtain a similar insecurity result as for the case of G. However, the proof (presented in [8]) is slightly different as we have to transform both ciphertext parts.

Theorem 2. *Let* $\mathsf{POWHF}' = (\mathcal{K}', \mathcal{H}', \mathcal{V}')$ *be a pseudorandom POWHFs with public randomness (with respect to the uniform distribution and some uninvertible function* hint*). Then there exists a pseudorandom* $\mathsf{POWHF} = (\mathcal{K}, \mathcal{H}, \mathcal{V})$ *with public randomness (with respect to the uniform distribution and* hint*), and there exists an oracle relative to which there is a partial one-way permutation family* F*, such that* $OAEP^{G,\mathsf{POWHF}}[F] = (\mathcal{EK}, \mathcal{E}, \mathcal{D})$*, an instantiation of the* H*-oracle in the* $OAEP^{G,H}[F]$ *encryption scheme with* POWHF*, is not IND-CCA in the RO model.*

4 Security of PSS-I Encryption Instantiations

In this section we show a positive result, allowing to replace one of the random oracles in our PSS-E variation, called PSS-I, by a pseudorandom POWHF. We were unable to prove or disprove that one can replace the other oracle in PSS-I.

4.1 The PSS-I Encryption Scheme

Coron et al. [12] suggested that the transformation used by the PSS signature scheme [6] can also be used for encrypting with RSA. Here we consider the following variation PSS-I. This scheme is parameterized by integers k, k_0 and k_1 (where k_0, k_1 are linear in k) and makes use of an instance of a trapdoor permutation family with domain and range $\{0,1\}^k$ (and it can be easily adapted for other domains like $\mathbf{Z}_N^*$ for the RSA permutation). The scheme also uses two random oracles

$$G\colon \{0,1\}^{k_1} \to \{0,1\}^{k-k_1} \quad \text{and} \quad H\colon \{0,1\}^{k-k_1} \to \{0,1\}^{k_1} .$$

The message space is $\{0,1\}^{k-k_0-k_1}$. The scheme PSS-I$^{G,H}[F]$ is given by the following algorithms:

- $\mathcal{EK}(1^k)$: Pick a random permutation f on $\{0,1\}^{k_1}$. Let pk specify f and let sk specify f^{-1}.
- $\mathcal{E}(pk, M)$: Compute $r \xleftarrow{\$} \{0,1\}^{k_0}$, $\omega \leftarrow H(M\|r)$ and $s \leftarrow G(\omega) \oplus (M\|r)$. Compute $C \leftarrow f(\omega)$ and output (C, s).
- $\mathcal{D}(sk, (C, s))$: Compute $\omega \leftarrow f^{-1}(C)$, $M\|r \leftarrow s \oplus G(\omega)$. If $\omega = H(M\|r)$ then return M. Otherwise, return $\perp$.

In the original PSS-E scheme [12] one computes f over both $\omega\|s$. We remark that our version here seems to be less secure than the original scheme at first, as the value s is now given in the clear. However, it nonetheless allows us to securely replace oracle G by a POWHF which we were unable to do in the original scheme. Moreover, we can prove security of our instantiation with respect to arbitrary trapdoor permutations, whereas the original scheme required partial one-way trapdoor permutations.

4.2 Instantiating the G-Oracle in PSS-I with POWHFs

An *instantiation of the G-oracle* in the PSS-I$^{G,H}[F]$ encryption scheme with a pseudorandom perfectly one-way hash function $\mathsf{POWHF} = (\mathcal{K}, \mathcal{G}, \mathcal{V})$ with public randomness results in the following encryption scheme PSS-I$^{\mathsf{POWHF},H}[F]=(\mathcal{EK}, \mathcal{E}, \mathcal{D})$

- $\mathcal{EK}(1^k)$: Pick a random permutation f on $\{0,1\}^{k_1}$ and sample a POWHF key $K \xleftarrow{\$} \mathcal{K}(1^k)$ and randomness $r_{\mathcal{G}} \xleftarrow{\$} \text{Coins}(K)$. Let pk specify f and also contain $K, r_{\mathcal{G}}$, and let sk specify f^{-1} and also contain $K, r_{\mathcal{G}}$.
- $\mathcal{E}(pk, M)$: Pick randomness $r \xleftarrow{\$} \{0,1\}^{k_0}$ for the encryption algorithm and compute $\omega \leftarrow H(M\|r)$. Compute $s \leftarrow \mathcal{G}_K^{\text{pr}}(\omega, r_{\mathcal{G}}) \oplus (M\|r)$ and $C \leftarrow f(\omega)$. Output (C, s).
- $\mathcal{D}(sk, (C, s))$: Compute $\omega \leftarrow f^{-1}(C)$, $M\|r \leftarrow s \oplus \mathcal{G}_K^{\text{pr}}(\omega, r_{\mathcal{G}})$. If $\omega = H(M\|r)$ then return M. Otherwise, return $\perp$.

It is noteworthy that the randomness of the POWHF becomes part of the public key and is therefore fixed for each ciphertext. While this seems strange at first, it becomes clear in in light of the role of the randomness in POWHFs. Originally, POWHFs were designed to meet a stronger security requirement [9, 24], demanding pairs $(\mathcal{G}(x, r_1), \mathcal{G}(x, r_2))$ for a single random x to be indistinguishable from pairs $(\mathcal{G}(x, r_1), \mathcal{G}(x', r_2))$ for independent samples x, x'. This of course requires that the randomness r_1, r_2 is chosen independently for each function evaluation, else distinguishing would be easy. However, security of PSS-I relies on pseudo-randomness of the corresponding function family and does not require the above security property. Accordingly, putting the randomness for the function family in the public key does not compromise security of the encryption scheme.

Theorem 3. *Let F be a trapdoor permutation family and let $\mathsf{POWHF} = (\mathcal{K}, \mathcal{G}, \mathcal{V})$ be a pseudorandom POWHF with public randomness, where pseudorandomness holds with respect to the uniform distribution on and the uninvertible function $\mathsf{hint}(x) = (f, f(x))$ for random f drawn from F. Then PSS-I$^{\mathsf{POWHF},H}[F]$ is IND-CCA secure in the RO model.*

The proof is delegated to [8]. We note that our proof does not make use the collision-resistance of the POWHF. This is because the preimage ω of the POWHF is uniquely determined by the additional trapdoor function value $f(\omega)$ anyway. Hence, a pseudorandom generator for which distinguishing the output from random is infeasible, even if given $\mathsf{hint}(\omega)$, would actually suffice in this setting. In particular, such a generator G can be built in combination with the trapdoor permutation f via the Yao-Blum-Micali construction [27, 7]. Namely, let f be of the form $f(x) = g^n(x)$ for a trapdoor permutation g and define $G(x) = (\mathrm{hb}(x), \mathrm{hb}(g(x)), \ldots, \mathrm{hb}(g^{n-1}(x)))$ through the hardcore bits hb. Then the output of G is still pseudorandom, even given $f(x)$.

5 Security of Instantiating the Fujisaki-Okamoto Transformation

Fujisaki and Okamoto [17] suggested a general construction of hybrid encryption schemes in the random oracle model. It is based on two random oracles, G and H. Here we show that one can replace G by a pseudorandom POWHF and still obtain a secure scheme (for a random oracle H). We then prove, under a somewhat non-standard assumption, that one can also replace H by a POWHF to obtain a secure scheme for a random oracle G.

5.1 Fujisaki-Okamoto Scheme

The Fujisaki-Okamoto construction is based on an asymmetric encryption scheme $\mathsf{AS} = (\mathcal{EK}_{\text{asym}}, \mathcal{E}_{\text{asym}}, \mathcal{D}_{\text{asym}})$ and a deterministic symmetric encryption scheme $\mathsf{SS} = (\mathcal{EK}_{\text{sym}}, \mathcal{E}_{\text{sym}}, \mathcal{D}_{\text{sym}})$, as well as two random oracles G, H. For parameter $k \in \mathbb{N}$ let $\mathrm{Coins}_{\text{asym}}(k)$ and $\mathrm{MsgSp}_{\text{asym}}(k)$ denote the set of random strings and the message space of the asymmetric encryption scheme, and $\mathrm{Keys}_{\text{sym}}(k)$ and $\mathrm{MsgSp}_{\text{sym}}(k)$ denote the key and message space of the symmetric encryption scheme. Let

$$G\colon \mathrm{MsgSp}_{\text{asym}}(k) \to \mathrm{Keys}_{\text{sym}}(k) \quad \text{and} \quad H\colon \{0,1\}^k \times \{0,1\}^* \to \mathrm{Coins}_{\text{asym}}(k)$$

The message space is $\mathrm{MsgSp}_{\text{sym}}(k)$. The encryption scheme $\mathrm{FO}^{G,H}$ is given by the following algorithms:

- $\mathcal{EK}(1^k)$: Run $\mathcal{EK}_{\rm asym}(1^k)$ to generate a key pair (sk, pk).
- $\mathcal{E}(pk, M)$: Pick $\sigma \stackrel{\$}{\leftarrow} {\rm MsgSp}_{\rm asym}(k)$, compute $C_{\rm asym} \leftarrow \mathcal{E}_{\rm asym}(pk, \sigma; H(\sigma, M))$ and $C_{\rm sym} \leftarrow \mathcal{E}_{\rm sym}(G(\sigma), M)$. Output $C = (C_{\rm asym}, C_{\rm sym})$.
- $\mathcal{D}(sk, C)$: For $C = (C_{\rm asym}, C_{\rm sym})$ compute $\sigma \leftarrow \mathcal{D}(sk, C_{\rm asym})$, $M \leftarrow \mathcal{D}_{\rm sym}(G(\sigma), C_{\rm sym})$. Recompute $c \leftarrow \mathcal{E}_{\rm asym}(pk, \sigma; H(\sigma, M))$ and output M if $c = C_{\rm asym}$, else return $\perp$.

Security of this conversion has been shown under the assumption that the symmetric encryption scheme is IND-CPA (and that the symmetric encryption algorithm is deterministic), and that the public-key encryption scheme is one-way and γ-uniform, which roughly means that ciphertexts are almost uniform. Here we make different, yet "natural" assumptions about the encryption schemes, as specified below.

5.2 Instantiating the *G*-Oracle

An *instantiation of the G-oracle* in the Fujisaki-Okamoto scheme through a perfectly one-way hash function $\mathsf{POWHF} = (\mathcal{K}, \mathcal{G}, \mathcal{V})$ with public randomness, denoted by ${\rm FO}^{\mathsf{POWHF},H}$, works as follows:

- $\mathcal{EK}(1^k)$: Run $\mathcal{EK}_{\rm asym}(1^k)$ to generate a key pair (sk, pk). Pick $K \stackrel{\$}{\leftarrow} \mathcal{K}(1^k)$ and $r \stackrel{\$}{\leftarrow} {\rm Coins}_{\mathcal{G}}(k)$. Output $((sk, K, r), (pk, K, r))$.
- $\mathcal{E}((pk, K, r), M)$: Pick $\sigma \stackrel{\$}{\leftarrow} {\rm MsgSp}_{\rm asym}(k)$, compute $C_{\rm asym} \leftarrow \mathcal{E}_{\rm asym}(pk, \sigma, H(\sigma, M))$ and $C_{\rm sym} \leftarrow \mathcal{E}_{\rm sym}(\mathcal{G}^{\rm pr}(K, \sigma, r), M)$. Output $C = (C_{\rm asym}, C_{\rm sym})$.
- $\mathcal{D}((sk, K, r), C)$: For $C = (C_{\rm asym}, C_{\rm sym})$ compute $\sigma \leftarrow \mathcal{D}_{\rm asym}(sk, C_{\rm asym})$, $M \leftarrow \mathcal{D}_{\rm sym}(\mathcal{G}^{\rm pr}(K, \sigma, r), C_{\rm sym})$. Recompute $c \leftarrow \mathcal{E}_{\rm asym}(pk, \sigma; H(\sigma, M))$ and output M if $c = C_{\rm asym}$, else return $\perp$.

We note that we use the same trick as in the PSS-I case before and put the randomness r of the POWHF into the public key. See the remarks there for further discussion.

Theorem 4. *Let* AS *and* SS *be IND-CPA asymmetric and symmetric encryption schemes, where* $\mathcal{E}_{sym}$ *is deterministic. Let* $\mathsf{POWHF} = (\mathcal{K}, \mathcal{G}, \mathcal{V})$ *be a pseudorandom POWHF with public randomness (with respect to the uniform distribution on* $({\rm MsgSp}_{asym}(k))_{k \in \mathbb{N}}$ *and the trivial uninvertible function* hint*). Then the instantiation of the G-oracle in the Fujisaki-Okamoto scheme,* $FO^{\mathsf{POWHF},H}$*, is IND-CCA in the random oracle model.*

The proof is in [8]. Recall that such POWHF as in the claim can be built from any one-way permutation. We can thus instantiate the G-oracle under this condition. In fact, the proof actually shows that regular one-wayness (instead of perfect one-wayness) is sufficient for the pseudorandom POWHF, where for any efficient algorithm $\mathcal{A}$ the probability that $\mathcal{A}$ returns x on input $(K, \mathsf{hint}(x), \mathcal{H}(K, x, r))$ for $K \stackrel{\$}{\leftarrow} \mathcal{K}(1^k)$, $r \stackrel{\$}{\leftarrow} {\rm Coins}(K)$, $x \stackrel{\mathcal{X}_k}{\leftarrow} \{0,1\}^{m(k)}$, is negligible. Clearly, perfect one-wayness implies regular one-wayness.

5.3 Instantiating the H-Oracle

Instantiating the H-oracle is technically more involved and requires a strong assumption about the combination of the POWHF and the public-key encryption scheme. Our construction also requires a stronger (yet mild) assumption about the symmetric encryption scheme.

Before presenting our assumptions we first define the H-instantiation of the Fujisaki -Okamoto transformation. We call the encryption scheme below an *instantiation of the H-oracle* in the Fujisaki-Okamoto scheme, $\mathrm{FO}^{G,\mathsf{POWHF}}$, through a pseudorandom and strongly collision-resistant $\mathsf{POWHF} = (\mathcal{K}, \mathcal{H}, \mathcal{V})$:

- $\mathcal{EK}(1^k)$: Run $\mathcal{EK}_{\mathrm{asym}}(1^k)$ to generate a key pair (sk, pk). Generate $K \xleftarrow{\$} \mathcal{K}(1^k)$ and $r \xleftarrow{\$} \mathrm{Coins}_{\mathcal{H}}(k)$ for POWHF. Output (sk, K, r) and (pk, K, r).
- $\mathcal{E}((pk, K, r), M)$: Pick $\sigma \xleftarrow{\$} \mathcal{EK}_{\mathrm{sym}}(1^k)$, compute $\omega \leftarrow \mathcal{H}^{\mathrm{pr}}(K, \sigma||M, r)$ and $C_{\mathrm{asym}} \leftarrow \mathcal{E}_{\mathrm{asym}}(pk, \sigma, \omega)$ and $C_{\mathrm{sym}} \leftarrow \mathcal{E}_{\mathrm{sym}}(G(\sigma), M)$. Output $C = (C_{\mathrm{asym}}, C_{\mathrm{sym}})$.
- $\mathcal{D}((sk, K, r), C)$: For $C = (C_{\mathrm{asym}}, C_{\mathrm{sym}})$ compute $\sigma \leftarrow \mathcal{D}(sk, C_{\mathrm{asym}})$, $M \leftarrow \mathcal{D}_{\mathrm{sym}}(G(\sigma), C_{\mathrm{sym}})$. Recompute $c \leftarrow \mathcal{E}_{\mathrm{asym}}(pk, \sigma; \mathcal{H}^{\mathrm{pr}}(K, \sigma||M, r))$ and output M if $c = C_{\mathrm{asym}}$, else return $\perp$.

To show that this instantiation is secure we need the following additional assumption about the symmetric encryption scheme. We assume that the symmetric encryption scheme provides *integrity of ciphertexts* (INT-CTXT) [3], i.e., for any efficient adversary $\mathcal{B}$ let $\kappa \xleftarrow{\$} \mathcal{EK}_{\mathrm{sym}}(1^k)$, $C \xleftarrow{\$} \mathcal{B}^{\mathcal{E}_{\mathrm{sym}}(\kappa,\cdot)}(1^k)$ and let $M \xleftarrow{\$} \mathcal{D}_{\mathrm{sym}}(\kappa, C)$. Then the probability that $M \neq \perp$ and that C has never been submitted by $\mathcal{B}$ to its oracle $\mathcal{E}_{\mathrm{sym}}(\kappa, \cdot)$ is negligible. This INT-CTXT property can be accomplished for example by the encrypt-then-MAC paradigm [3]. We remark that this additional property, together with the IND-CPA security of the asymmetric encryption scheme, does not necessarily imply IND-CCA security of hybrid schemes; it is easy to construct counterexamples.

For our instantiation we also need a very strong assumption about the combination of POWHF and the public-key encryption scheme $(\mathcal{EK}_{\mathrm{asym}}, \mathcal{E}_{\mathrm{asym}}, \mathcal{D}_{\mathrm{asym}})$. That is, we assume that the following random variables are indistinguishable for any efficient message distribution $\mathcal{M}$ (which also outputs some information *state* about the sampling process):

- Let $(sk, pk) \xleftarrow{\$} \mathcal{EK}_{\mathrm{asym}}(1^k)$, $K \xleftarrow{\$} \mathcal{K}(1^k)$, $r \xleftarrow{\$} \mathrm{Coins}_{\mathcal{G}}(k)$ and $(M, state) \xleftarrow{\$} \mathcal{M}(pk, K, r)$. Pick $\sigma \xleftarrow{\$} \mathrm{MsgSp}_{\mathrm{asym}}\ (k)$ and compute $\omega \leftarrow \mathcal{H}^{\mathrm{pr}}(K, \sigma||M, r)$ and $C_{\mathrm{asym}} \leftarrow \mathcal{E}_{\mathrm{asym}}(pk, \sigma, \omega)$. Output $(pk, K, r, state, C_{\mathrm{asym}})$.
- Let $(sk, pk) \xleftarrow{\$} \mathcal{EK}_{\mathrm{asym}}(1^k)$, $K \xleftarrow{\$} \mathcal{K}(1^k)$, $r \xleftarrow{\$} \mathrm{Coins}_{\mathcal{G}}(k)$ and $(M, state) \xleftarrow{\$} \mathcal{M}(pk, K, r)$. Pick $\sigma \xleftarrow{\$} \mathrm{MsgSp}_{\mathrm{asym}}\ (k)$ and $\omega \xleftarrow{\$} \mathrm{Coins}_{\mathrm{asym}}$ and compute $C_{\mathrm{asym}} \leftarrow \mathcal{E}_{\mathrm{asym}}(pk, \sigma, \omega)$. Output $(pk, K, r, state, C_{\mathrm{asym}})$.

We call this the *POWHF-encryption assumption for* POWHF *and* AS.

Informally, if one views the POWHFs as a pseudorandom generator, the assumption basically says that encrypting the seed σ of a pseudorandom generator with the pseudorandom output ω is indistinguishable from an encryption of the seed with independent randomness. Note that this assumption would be false in general if one is also given ω in clear (which is either pseudorandom or truly random). For example, for ElGamal encryption $(g^{\omega}, pk^{\omega} \cdot \sigma)$ one could easily recover σ if given ω (by dividing out pk^{ω} in the right part), and try to recompute ω through the pseudorandom generator applied to σ. However, if one is not given ω then such generic attacks (in the sense of [25]) fail.

Note also that our POWHF-encryption assumption is certainly not stronger than assuming that the pseudorandom generator is perfect and given by a random oracle. On the contrary, our result shows that seeing the adversary's queries to function H is not necessary to simulate attacks and to prove security. This holds, of course, as long as G is still a random oracle and the simulator learns the queries to this oracle. The proof of the following theorem is in [8]. Similar to the G-case the proof shows that regular one-wayness is enough for the pseudorandom POWHF.

Theorem 5. *Let* AS *and* SS *be IND-CPA public-key and private-key encryption schemes where* $\mathcal{E}_{sym}$ *is deterministic. Let* $\mathsf{POWHF} = (\mathcal{K}, \mathcal{H}, \mathcal{V})$ *be a pseudorandom POWHF with public randomness (with respect to the uniform distribution and the trivial uninvertible function). Assume further that the symmetric encryption scheme provides integrity of ciphertexts and that the POWHF-encryption assumption holds for* POWHF *and* AS*. Then the instantiation of the* H*-oracle in the Fujisaki-Okamoto transformation,* $FO^{G,\mathsf{POWHF}}$*, yields an IND-CCA encryption scheme in the random oracle model.*

6 (In)Security of FDH Signature Scheme Instantiations

In this section we consider the Full Domain Hash (FDH) signature scheme which is provably secure in the random oracle model if the associated permutation is one-way. We show that replacing the random oracle by a verifiable pseudorandom function does not necessarily yield a secure instantiation. For sake of concreteness we explain our negative result for the RSA case. The result can be transferred, mutatis mutandis, to other trapdoor permutations.

We note that one can easily transfer our negative result about OAEP (Theorems 1 and 2) to show that the FDH instantiated with a POWHF is insecure with respect to a specific trapdoor permutation oracle. But our result here for the VPRFs works for any trapdoor permutation, including RSA for example.

Full Domain Hash Signature Scheme and Instantiation with VPRFs. Due to lack of space we omit the formal description of the well-known Full-domain hash (FDH) signature scheme [4] Basically, a signature S for a message M is given as $S = f^{-1}(H(M))$ and verification requires checking $f(S) = H(M)$. An instantiation of the FDH scheme with $\mathsf{VPRF} = (\mathcal{K}, \mathcal{H}, \mathcal{V})$ is the following signature scheme $\mathrm{FDH}^{\mathsf{VPRF}}[F] = (\mathcal{SK}, \mathcal{S}, \mathcal{V})$:

– $\mathcal{SK}(1^k)$: pick a random permutation f on D_k from F, pick $(fk, vk) \xleftarrow{\$} \mathcal{K}(1^k)$. Let pk specify f and contain vk and let sk specify f^{-1} and contain vk.

– $\mathcal{S}^{\mathcal{H}(fk,\cdot)}(sk, M)$: $(y, \pi) \xleftarrow{\$} \mathcal{H}(fk, M)$, $S \leftarrow f^{-1}(y)$. Output (S, π, y).
– $\mathcal{V}^{\mathcal{V}(fk,\cdot)}(pk, M, (S, \pi))$: If $f(S) = y$ and $\mathcal{V}(vk, M, y, \pi) = 1$ then return 1, else return 0

It is important to note that in the attack the adversary is only given access to the signature oracle but not to the VPRF oracle. Although the application as a third-party web interface providing such values indicate that the adversary can get additional VPRF values, our result even holds in the setting where the adversary is denied such values.

On the Insecurity of RSA-FDH with VPRFs. A special case is the RSA-FDH signature scheme (and its instantiation through a VPRF) where f, f^{-1} are given by the RSA function $x \mapsto x^e \bmod N$ and its inverse $y \mapsto y^d \bmod N$. Here we consider the case *with large prime exponents* where the RSA exponent e has to be a prime of $(k+1)$ bits and therefore larger than the k-bit modulus N. We denote this function by $\mathrm{RSA}_{\text{large-exponent}}$. According to the recent result about deterministic primality testing [1], this prerequisite allows to verify deterministically that a pair (N, e) really constitutes a permutation. We also remark that this RSA version is not known to be weaker than RSA with other exponents.

For the RSA-FDH scheme we construct a "bad" VPRF such that, when instantiated with this VPRF, RSA-FDH becomes insecure:

Theorem 6. *Suppose VPRFs exist. Then there exists a verifiable pseudorandom function* $\mathsf{VPRF} = (\mathcal{K}, \mathcal{H}, \mathcal{V})$ *such that* $FDH^{\mathsf{VPRF}}[RSA_{large\text{-}exponent}]$ *is subject to existential forgeries in chosen-message attacks.*

The basic idea is that the "bad" VPRF (which exists if any VPRF exists) itself will reveal signatures for free as part of the correctness proof. Thus, giving the signature oracle the right message will force the signer to query the VPRF at the right input which, in turn, allows to forge signatures. We prove this formally in [8] .

Acknowledgements

We thank Victor Shoup for clarifications on [26] and the anonymous reviewers of Crypto 2005 for useful comments.

References

1. M. Agrawal, N. Kayal, and N. Saxena. Primes is in P. `http://www.cse.iitk.ac.in/news/primality.html`, 2002.
2. M. Bellare, A. Boldyreva, and A. Palacio. An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In *Eurocrypt 2004*, volume 3027 of *LNCS*. Springer, 2004.
3. M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In *ASIACRYPT 2000*, volume 1976 of *LNCS*. Springer, 2000.

4. M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *CCS '93*. ACM, 1993.
5. M. Bellare and P. Rogaway. Optimal asymmetric encryption – how to encrypt with RSA. In *Eurocrypt '94*, volume 950, 1995.
6. M. Bellare and P. Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In *Eurocrypt '96*, volume 1070 of *LNCS*. Springer, 1996.
7. M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM Journal of Computing*, 13:850–864, 1984.
8. A. Boldyreva and M. Fischlin. Analysis of random-oracle instantiation scenarios for OAEP and other practical schemes. Full version of this paper. Available at `http://www.cc.gatech.edu/~aboldyre/publications.html`.
9. R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In *CRYPTO '97*, volume 1294 of *LNCS*. Springer, 1997.
10. R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. In *STOC '98*. ACM, 1998.
11. D. Chaum. Blind signatures for untraceable payments. In *CRYPTO '82*, 1983.
12. J.-S. Coron, M. Joye, D. Naccache, and .P. Paillier. Universal padding schemes for RSA. In *CRYPTO 2002*, volume 2442. Springer, 2002.
13. Y. Dodis. Efficient construction of (distributed) verifiable random functions. In *PKC 2003*, volume 2567 of *LNCS*. Springer, 2003.
14. Y. Dodis, R. Oliveira, and K. Pietrzak. On the generic insecurity of full-domain hash. In *CRYPTO 2005*, LNCS, 2005.
15. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature schemes. In *Crypto '86*, volume 263 of *LNCS*. Springer, 1986.
16. M. Fischlin. Pseudorandom function tribe ensembles based on one-way permutations: Improvements and applications. In *Eurocrypt '99*, volume 1592 of *LNCS*. Springer, 1999.
17. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *CRYPTO '99*, volume 1666 of *LNCS*, 1999.
18. E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA-OAEP is secure under the RSA assumption. In *CRYPTO 2001*, volume 2139 of *LNCS*. Springer, 2001.
19. S. Goldwasser and Y. T. Kalai. On the (in)security of the Fiat-Shamir paradigm. In *FOCS 2003*. IEEE, 2003.
20. K. Kobara and H. Imai. OAEP++: A very simple way to apply OAEP to deterministic ow-cpa primitives. *Cryptology ePrint Archive, Report 2002/130.*, 2002.
21. U. Maurer, R. Renner, and C. Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In *TCC 2004*, volume 2951 of *LNCS*. Springer, 2004.
22. S. Micali, M. Rabin, and S. Vadhan. Verifiable random functions. In *FOCS 1999*. IEEE, 1999.
23. J. Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *CRYPTO 2002*, volume 2442 of *LNCS*. Springer, 2002.
24. D. Micciancio R. Canetti and O. Reingold. Perfectly one-way probabilistic hash functions. In *STOC '98*. ACM, 1998.
25. V. Shoup. Lower bounds for discrete logarithms and related problems. In *Eurocrypt '97*, volume 1233 of *LNCS*. Springer, 1997.
26. V. Shoup. OAEP reconsidered. In *CRYPTO 2001*, volume 2139 of *LNCS*. Springer, 2001.
27. A. Yao. Theory and applications of trapdoor functions. In *FOCS 1982*, pages 80–91. IEEE, 1982.