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

October 27, 2005

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.

*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).

1 Introduction

The random oracle (RO) model, introduced by Fiat and Shamir [16] and refined by Bellare and Rogaway [5], has been suggested as a trade-off between provable security and practical requirements for efficiency. Schemes and proofs in this 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 [6, 19]. 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, 22, 24, 3, 20] 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 [3]. 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 [18] and the PSS-E encryption scheme, an OAEP variant due to Coron et al. [13]. 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, 14]. These are the perfectly one-way hash functions (POWHFs) [9, 11] and verifiable pseudorandom functions (VPRFs) [23].

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 [11, 17], 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 [5] 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 [23]. 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 [14] an application scenario for VPRFs 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 [6] 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 [19]. 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 [12]. We note that, independently of our work, [15] 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 pseudorandom 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. [13]. 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 [7]. 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 [21] 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 [18] 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. In the body of the paper we usually present proof ideas only; the full proofs are given in the Appendix.

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 "$\|$". By $\langle i \rangle_k$ we denote the standard fixed-length binary encoding of numbers $i = 0, 1, \ldots, 2^k - 1$ with k bits. If x is a k-bit binary string then $x[i]$ for $0 \leq i \leq k-1$ denotes its i-th most significant bit.

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 $x \xleftarrow{\$} \mathcal{A}(y, z, \ldots)$ means that the value x is assigned the output of $\mathcal{A}$ for input $(y, z, \ldots)$. 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 Appendix A we recall the definitions of asymmetric encryption schemes, their security against chosen-plaintext attacks (IND-CPA security) and chosen-ciphertext attacks (IND-CCA security). We also specify 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 polynomi-

als, 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 2.1 ***[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: The function $\mathcal{H}$ 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's output is pseudorandom, i.e., 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, 11, 17] 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, as required for the functionality of some encryption schemes. All previous constructions [9, 11, 17] 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 [23], 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 2.2 ***[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 (possibly not generated according to $\mathcal{K}(1^k)$), 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 $\mathbf{Exp}^{\mathrm{vprf}\text{-}ind}_{\mathsf{VPRF},\mathcal{A}}(1^k)$
   $b \xleftarrow{\$} \{0,1\}$
   $(fk, vk) \xleftarrow{\$} \mathcal{K}(1^k)$
   $(x, state) \xleftarrow{\$} \mathcal{A}^{\mathcal{H}(fk,\cdot)}$ *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 $\Pr\left[\,\mathbf{Exp}^{\mathrm{vprf}\text{-}ind}_{\mathsf{VPRF},\mathcal{A}}(1^k) = b\,\right] - 1/2$ *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 [6]. 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:

- key generation $\mathcal{EK}(1^k)$: Pick a permutation f from F at random. Let pk specify f and let sk specify f^{-1}.
- encryption $\mathcal{E}(pk, M)$: Compute $r \xleftarrow{\$} \{0,1\}^{k_0}$, $s \leftarrow (m\|0^{k_1}) \oplus G(r)$ and $t \leftarrow r \oplus H(s)$. Output $C \leftarrow f(s||t)$.
- decryption $\mathcal{D}(sk, C)$: 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 [19]. Partial one-wayness is a stronger notion than one-wayness; for the definitions see [19].

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 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, 11, 17] 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 Appendix B.1. Our construction assumes that one-way permutations exist and employs the pseudorandom function tribe ensembles of [17] (which is 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, $\mathcal{A}$ needs to be able to 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 [19]. Our attacks work even if F is partial one way.

Theorem 3.1 *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*) and assume one-way permutations exist. 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 $\mathrm{OAEP}^{\mathsf{POWHF},H}[F]$, an instantiation of the G-oracle in the $\mathrm{OAEP}^{G,H}[F]$ encryption scheme with* POWHF*, is* not *IND-CCA in the RO model.*

We give a sketch of the proof. For the formal proof see Appendix B.2. 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 from the given POWHF' and one-way permutations in Appendix B.1). We now define a compliant XOR-malleable permutation family. We slightly strengthen the original definition of Shoup [26].

Definition 3.2 *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 3.2 with fixed δ:

Fact 3.3 ([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. 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

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

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

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

- $\mathcal{EK}(1^k)$: 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 the encryption algorithm and compute $s \leftarrow (m\|0^{k_1}) \oplus G(r)$. Pick randomness $r_{\mathcal{H}} \xleftarrow{\$} \text{Coins}(K)$ for the POWHF, compute $y \leftarrow \mathcal{H}^{\text{pr}}_K(s, r_{\mathcal{H}})$ and $t \leftarrow r \oplus y$. Compute $C \leftarrow f(s||t)$. Output $(r_{\mathcal{H}}, C)$.
- $\mathcal{D}(sk, (r_{\mathcal{H}}, C))$: Compute $s\|t \leftarrow f^{-1}(C), r \leftarrow t \oplus \mathcal{H}^{\text{pr}}_K(s, r_{\mathcal{H}})$ 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 $\perp$.

For substituting the H-oracle we obtain a similar insecurity result as for the case of G. However, the proof (presented in Appendix B.3) is slightly different as we have to transform both ciphertext parts.

Theorem 3.4 *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*) and assume one-way permutations exists. 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* $OAEP^{G,\mathsf{POWHF}}[F]$*, 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. [13] suggested that the transformation used by the PSS signature scheme [7] 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 [13] 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 $r_{\mathcal{G}} \xleftarrow{\$} \mathrm{Coins}(K)$. 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 $s \leftarrow \mathcal{G}_K^{\mathrm{pr}}(\omega, r_{\mathcal{G}}) \oplus (M\|r)$ and $C \leftarrow f(\omega)$. Output (C, s).

- $\mathcal{D}(sk, (C, s))$: Compute $\omega \leftarrow f^{-1}(C)$, $M\|r \leftarrow s \oplus \mathcal{G}_K^{\mathrm{pr}}(\omega, r_{\mathcal{G}})$. 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, 11], 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 pseudorandomness 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 4.1 *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 full proof is delegated to Appendix C. For the proof idea consider an adversary $\mathcal{A}$ attacking the encryption scheme and getting exactly one challenge ciphertext. Gradually modify the encryption algorithm to compute this challenge ciphertext

$$(C, s) = (f(\omega), \mathcal{G}_K^{\mathrm{pr}}(\omega, r_{\mathcal{G}}) \oplus (M\|r)), \quad \text{for } r \leftarrow \{0,1\}^k,\ \omega \leftarrow H(M\|r).$$

in a sequence of games as follows:

- In the first modification change the encryption process by substituting the ciphertext part $s \leftarrow \mathcal{G}_K^{\mathrm{pr}}(\omega, r_{\mathcal{G}}) \oplus (M\|r)$ through the computation of $s \leftarrow u \oplus (M\|r)$ for random u:

  $$(C, s) = (f(\omega), u \oplus (M\|r)), \quad \text{for } r \leftarrow \{0,1\}^k,\ \omega \leftarrow H(M\|r),\ u \leftarrow \{0,1\}^{k-k_1}.$$

  Since the POWHF value $\mathcal{G}_K^{\mathrm{pr}}(\omega, r_{\mathcal{G}})$ is pseudorandom with respect to the hint function $(f, f(\omega))$, the adversary's behavior in an attack in this game cannot change significantly.

- Next, instead of computing $\omega \leftarrow H(M\|r)$ we simply pick ω at random:

  $$(C, s) = (f(\omega), u \oplus (M\|r)), \quad \text{for } r \leftarrow \{0,1\}^k,\ \omega \leftarrow \{0,1\}^{k_1},\ u \leftarrow \{0,1\}^{k-k_1}.$$

  As the random oracle H is evaluated for a secret random string r this modification will not alter the adversary's output behavior noticeably.

We furthermore show that any decryption query of the adversary can be essentially answered in the random oracle model by inspecting $\mathcal{A}$'s communication with H (even if we now only have a single random oracle instead of two). This holds for the original scheme as well as for both modified games, but requires some care with decryption queries based on previously obtained ciphertexts of unknown messages. We finally note that ciphertexts in the last game are distributed independently of the message M and any adversary therefore cannot successfully attack this modified scheme. Because of the almost identical adverserial behavior in all games it follows that the original scheme is secure.

We note that our proof does not make use of 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, 8]. 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 and $f(x)$.

5 Security of Instantiating the Fujisaki-Okamoto Transformation

Fujisaki and Okamoto [18] 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 strong assumption about the combined security of the POWHF and the encryption scheme, 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 $\mathsf{AS} = (\mathcal{EK}_{\mathrm{asym}}, \mathcal{E}_{\mathrm{asym}}, \mathcal{D}_{\mathrm{asym}})$ and a deterministic symmetric encryption scheme $\mathsf{SS} = (\mathcal{EK}_{\mathrm{sym}}, \mathcal{E}_{\mathrm{sym}}, \mathcal{D}_{\mathrm{sym}})$, as well as two random oracles G, H. For parameter $k \in \mathbb{N}$ let $\mathrm{Coins}_{\mathrm{asym}}(k)$ and $\mathrm{MsgSp}_{\mathrm{asym}}(k)$ denote the set of random strings and the message space of the asymmetric encryption scheme, and $\mathrm{Keys}_{\mathrm{sym}}(k)$ and $\mathrm{MsgSp}_{\mathrm{sym}}(k)$ denote the key and message space of the symmetric encryption scheme. Let

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

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

- $\mathcal{EK}(1^k)$: Run $\mathcal{EK}_{\mathrm{asym}}(1^k)$ to generate a key pair (sk, pk).
- $\mathcal{E}(pk, M)$: Pick $\sigma \xleftarrow{\$} \mathrm{MsgSp}_{\mathrm{asym}}(k)$, compute $C_{\mathrm{asym}} \leftarrow \mathcal{E}_{\mathrm{asym}}(pk, \sigma; H(\sigma, M))$ and $C_{\mathrm{sym}} \leftarrow \mathcal{E}_{\mathrm{sym}}(G(\sigma), M)$. Output $C = (C_{\mathrm{asym}}, C_{\mathrm{sym}})$.

- $\mathcal{D}(sk, C)$: For $C = (C_{\text{asym}}, C_{\text{sym}})$ compute $\sigma \leftarrow \mathcal{D}(sk, C_{\text{asym}})$, $M \leftarrow \mathcal{D}_{\text{sym}}(G(\sigma), C_{\text{sym}})$. Recompute $c \leftarrow \mathcal{E}_{\text{asym}}(pk, \sigma; H(\sigma, M))$ and output M if $c = C_{\text{asym}}$, 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 $\text{FO}^{\mathsf{POWHF},H}$, works as follows:

- $\mathcal{EK}(1^k)$: Run $\mathcal{EK}_{\text{asym}}(1^k)$ to generate a key pair (sk, pk). Pick $K \xleftarrow{\$} \mathcal{K}(1^k)$ and $r \xleftarrow{\$} \text{Coins}_{\mathcal{G}}(K)$. Output $((sk, K, r), (pk, K, r))$.
- $\mathcal{E}((pk, K, r), M)$: Pick $\sigma \xleftarrow{\$} \text{MsgSp}_{\text{asym}}(k)$, compute $C_{\text{asym}} \leftarrow \mathcal{E}_{\text{asym}}(pk, \sigma, H(\sigma, M))$ and $C_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(\mathcal{G}^{\text{pr}}(K, \sigma, r), M)$. Output $C = (C_{\text{asym}}, C_{\text{sym}})$.
- $\mathcal{D}((sk, K, r), C)$: For $C = (C_{\text{asym}}, C_{\text{sym}})$ compute $\sigma \leftarrow \mathcal{D}_{\text{asym}}(sk, C_{\text{asym}})$ and $M \leftarrow \mathcal{D}_{\text{sym}}(\mathcal{G}^{\text{pr}}(K, \sigma, r), C_{\text{sym}})$. Recompute $c \leftarrow \mathcal{E}_{\text{asym}}(pk, \sigma; H(\sigma, M))$ and output M if $c = C_{\text{asym}}$, 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 5.1 *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* $(\text{MsgSp}_{asym}(k))_{k \in \mathbb{N}}$ *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 Appendix D. Similar to PSS-I the underlying idea is to consider an adversary and its behavior for slightly changing games when getting a single challenge ciphertext. Starting with the original encryption algorithm

$$C_{\text{asym}} \leftarrow \mathcal{E}_{\text{asym}}(pk, \sigma; H(\sigma, M)), C_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(\mathcal{G}^{\text{pr}}(K, \sigma, r), M) \quad \text{for random } \sigma.$$

we define the games with the modified encryption algorithm to compute the challenge ciphertext by:

- In the first game pick a random ω instead of using $H(\sigma, M)$ as the random coins for the computation of C_{asym}:

  $$C_{\text{asym}} \leftarrow \mathcal{E}_{\text{asym}}(pk, \sigma; \omega), C_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(\mathcal{G}^{\text{pr}}(K, \sigma, r), M) \quad \text{for random } \sigma, \omega.$$

  By the one-wayness of POWHF (for $\mathsf{hint}(\sigma) = (pk, C_{\text{asym}})$) it follows that any adversary attacking the original game cannot query H about (σ, M) during the attack, except with negligible probability. Given this, the modification with a random ω is unrecognizable to the adversary.

- In the second game we now pick a random key u for the symmetric part, instead of computing it via the POWHF $\mathcal{G}^{\text{pr}}(K, \sigma, r)$:

$$C_{\text{asym}} \leftarrow \mathcal{E}_{\text{asym}}(pk, \sigma; \omega), C_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(u, M) \quad \text{for random } \sigma, \omega, u.$$

  Indistinguishability of the two experiments follows by the pseudorandomness of the POWHF (for $\mathsf{hint}(\sigma) = (pk, C_{\text{asym}})$).

- Finally, replace the symmetric encryption of M by a trivial encryption of $0^{|M|}$:

$$C_{\text{asym}} \leftarrow \mathcal{E}_{\text{asym}}(pk, \sigma; \omega), C_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(u, 0^{|M|}) \quad \text{for random } \sigma, \omega, u.$$

  The security of the symmetric schemes guarantees that the adversary's output distribution is close to the in the previous game.

It follows analogously to the PSS-I case that decryption queries of the adversary in any of the games can already be simulated with the help of the remaining random oracle H. Yet, this time we also need to rely on the security of the asymmetric encryption scheme to prove this. Since ciphertexts in the final game are again independent of the actual message content —the length cannot be hidden— we conclude that the original scheme must be secure.

Recall that we demand perfect one-wayness and pseudorandomness of the POWHF to hold with respect to $\mathsf{hint}(\sigma) = (pk, C_{\text{asym}})$, i.e., for an encryption of random σ. By the security of the asymmetric scheme we can replace C_{asym} by an encryption of, say, $0^{|\sigma|}$ without affecting the one-wayness or pseudorandomness significantly. But this hint function is equivalent to trivial and we can therefore build such POWHF as in the claim of Theorem 5.1 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 $K \xleftarrow{\$} \mathcal{K}(1^k)$, $r \xleftarrow{\$} \text{Coins}(K)$, $x \xleftarrow{\chi_k} \{0,1\}^{m(k)}$, 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, $\text{FO}^{G,\mathsf{POWHF}}$, through a pseudorandom and strongly collision-resistant $\mathsf{POWHF} = (\mathcal{K}, \mathcal{H}, \mathcal{V})$:

- $\mathcal{EK}(1^k)$: Run $\mathcal{EK}_{\text{asym}}(1^k)$ to generate a key pair (sk, pk). Generate $K \xleftarrow{\$} \mathcal{K}(1^k)$ and $r \xleftarrow{\$} \text{Coins}_{\mathcal{H}}(k)$ for POWHF. Output (sk, K, r) and (pk, K, r).

- $\mathcal{E}((pk, K, r), M)$: Pick $\sigma \xleftarrow{\$} \mathcal{EK}_{\text{sym}}(1^k)$, compute $\omega \leftarrow \mathcal{H}^{\text{pr}}(K, \sigma||M, r)$ and $C_{\text{asym}} \leftarrow \mathcal{E}_{\text{asym}}(pk, \sigma, \omega)$ and $C_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(G(\sigma), M)$. Output $C = (C_{\text{asym}}, C_{\text{sym}})$.

- $\mathcal{D}((sk, K, r), C)$: For $C = (C_{\text{asym}}, C_{\text{sym}})$ compute $\sigma \leftarrow \mathcal{D}(sk, C_{\text{asym}})$, $M \leftarrow \mathcal{D}_{\text{sym}}(G(\sigma), C_{\text{sym}})$. Recompute $c \leftarrow \mathcal{E}_{\text{asym}}(pk, \sigma; \mathcal{H}^{\text{pr}}(K, \sigma||M, r))$ and output M if $c = C_{\text{asym}}$, 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) [4], i.e., for any efficient adversary $\mathcal{B}$ let $\kappa \xleftarrow{\$} \mathcal{EK}_{\text{sym}}(1^k)$, $C \xleftarrow{\$} \mathcal{B}^{\mathcal{E}_{\text{sym}}(\kappa,\cdot)}(1^k)$ and let $M \leftarrow \mathcal{D}_{\text{sym}}(\kappa, C)$. Then the probability that $M \neq \perp$ and that C has never been submitted by $\mathcal{B}$ to its oracle $\mathcal{E}_{\text{sym}}(\kappa, \cdot)$ is negligible. This INT-CTXT property can be accomplished for example by the encrypt-then-MAC paradigm [4]. We remark that this additional property, together with the IND-CPA security of the asymmetric and symmetric encryption schemes, 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 $(\mathcal{EK}_{\text{asym}}, \mathcal{E}_{\text{asym}}, \mathcal{D}_{\text{asym}})$. 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 $(sk, pk) \xleftarrow{\$} \mathcal{EK}_{\text{asym}}(1^k)$, $K \xleftarrow{\$} \mathcal{K}(1^k)$, $r \xleftarrow{\$} \text{Coins}_{\mathcal{G}}(k)$ and $(M, state) \xleftarrow{\$} \mathcal{M}(pk, K, r)$. Pick $\sigma \xleftarrow{\$} \text{MsgSp}_{\text{asym}}(k)$ and compute $\omega \leftarrow \mathcal{H}^{\text{pr}}(K, \sigma || M, r)$ and $C_{\text{asym}} \leftarrow \mathcal{E}_{\text{asym}}(pk, \sigma, \omega)$. Output $(pk, K, r, state, C_{\text{asym}})$.
- Let $(sk, pk) \xleftarrow{\$} \mathcal{EK}_{\text{asym}}(1^k)$, $K \xleftarrow{\$} \mathcal{K}(1^k)$, $r \xleftarrow{\$} \text{Coins}_{\mathcal{G}}(k)$ and $(M, state) \xleftarrow{\$} \mathcal{M}(pk, K, r)$. Pick $\sigma \xleftarrow{\$} \text{MsgSp}_{\text{asym}}(k)$ and $\omega \xleftarrow{\$} \text{Coins}_{\text{asym}}$ and compute $C_{\text{asym}} \leftarrow \mathcal{E}_{\text{asym}}(pk, \sigma, \omega)$. Output $(pk, K, r, state, C_{\text{asym}})$.

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 Appendix 5.2 and also relies on a sequence of games, modifying the original scheme to one where ciphertexts are independent of the message (yet, the analysis of these games is slightly more involved). Similar to the G-case the proof shows that regular one-wayness is enough for the pseudorandom POWHF.

Theorem 5.2 *Let* AS *and* SS *be IND-CPA public-key and private-key encryption schemes where* $\mathcal{E}_{sym}$ *is deterministic. Let* 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 3.1 and 3.4) 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.

6.1 Full Domain Hash Signature Scheme and Instantiation with VPRFs

We recall the well-known Full-domain hash (FDH) signature scheme [5]. The scheme $\mathrm{FDH}^H[F]$ makes use of a random permutation family F for domain $D = (D_k)_{k\in\mathbb{N}}$ and a random oracle H.

- $\mathcal{SK}(1^k)$: Pick a random permutation f on D_k from F. Let pk specify f and let sk specify f^{-1}.
- $\mathcal{S}^H(sk, M)$: Output $S \leftarrow f^{-1}(H(M))$.
- $\mathcal{V}^H(pk, M, S)$: If $f(S) = H(M)$ then return 1, else return 0

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 similarly to the case of schemes using PRFs, in the attack the adversary is not only given access to the signature oracle, but also to the VPRF oracle (just as it is allowed to query the random oracle). This viewpoint is also supported by the application as a third-party web interface providing such values or as a tamper-resistant device to which the adversary has local black-box access.[1]

[1] In the proceedings version we initially claimed that our result would also hold if the adversary is denied access to the VPRF; this, however, remains open.

6.2 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 6.1 *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. We prove this formally in Appendix F.

Acknowledgments

We thank Victor Shoup for clarifications on [26] and Mihir Bellare 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, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *ASIACRYPT 2001*, volume 2248, pages 566–582.

[3] M. Bellare, A. Boldyreva, and A. Palacio. An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In C. Cachin and J. Camenisch, editors, *Eurocrypt 2004*, volume 3027 of *LNCS*. Springer, 2004.

[4] M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In T. Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*. Springer, 2000.

[5] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *CCS '93*. ACM, 1993.

[6] M. Bellare and P. Rogaway. Optimal asymmetric encryption – how to encrypt with RSA. In A. De Santis, editor, *Eurocrypt '94*, volume 950, 1995.

[7] M. Bellare and P. Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In U. Maurer, editor, *Eurocrypt '96*, volume 1070 of *LNCS*. Springer, 1996.

[8] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM Journal of Computing*, 13:850–864, 1984.

[9] R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In B. Kaliski, editor, *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] R. Canetti, D. Micciancio, and O. Reingold. Perfectly one-way probabilistic hash functions. In *STOC '98*. ACM, 1998.

[12] D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. Rivest, and T. Sherman, editors, *CRYPTO '82*, 1983.

[13] J.-S. Coron, M. Joye, D. Naccache, and P. Paillier. Universal padding schemes for RSA. In M. Yung, editor, *CRYPTO 2002*, volume 2442. Springer, 2002.

[14] Y. Dodis. Efficient construction of (distributed) verifiable random functions. In Y. Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*. Springer, 2003.

[15] Y. Dodis, R. Oliveira, and K. Pietrzak. On the generic insecurity of full-domain hash. In V. Shoup, editor, *CRYPTO 2005*, LNCS, 2005.

[16] 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.

[17] M. Fischlin. Pseudorandom function tribe ensembles based on one-way permutations: Improvements and applications. In J. Stern, editor, *Eurocrypt '99*, volume 1592 of *LNCS*. Springer, 1999.

[18] E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In M. Weiner, editor, *CRYPTO '99*, volume 1666 of *LNCS*, 1999.

[19] E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA-OAEP is secure under the RSA assumption. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*. Springer, 2001.

[20] S. Goldwasser and Y. T. Kalai. On the (in)security of the Fiat-Shamir paradigm. In *FOCS 2003*. IEEE, 2003.

[21] 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.

[22] U. Maurer, R. Renner, and C. Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In M. Naor, editor, *TCC 2004*, volume 2951 of *LNCS*. Springer, 2004.

[23] S. Micali, M. Rabin, and S. Vadhan. Verifiable random functions. In *FOCS 1999*. IEEE, 1999.

[24] J. Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In M. Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*. Springer, 2002.

[25] V. Shoup. Lower bounds for discrete logarithms and related problems. In W. Fumy, editor, *Eurocrypt '97*, volume 1233 of *LNCS*. Springer, 1997.

[26] V. Shoup. OAEP reconsidered. In J. Kilian, editor, *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.

A Standard Definitions

A.1 Encryption Schemes and their Security

An asymmetric encryption scheme $\mathsf{AE} = (\mathcal{EK}, \mathcal{E}, \mathcal{D})$ is specified by three polynomial-time algorithms with the following functionalities. The randomized *key-generation* algorithm $\mathcal{EK}$ takes input 1^k, where k is the security parameter, and outputs a pair (pk, sk) consisting of a public

key and a matching secret key, respectively. The randomized *encryption* algorithm $\mathcal{E}$ takes input a public key pk and a message M, and outputs a ciphertext C. The deterministic *decryption* algorithm $\mathcal{D}$ takes input a secret key sk and a ciphertext C, and outputs a message M or a special symbol $\perp$ to indicate that the ciphertext is invalid. Associated to k is a *message space* $\mathrm{MsgSp}(k)$ from which M is allowed to be drawn. For any $(pk, sk) \in [\mathcal{EK}(1^k)]$, any $M \in \mathrm{MsgSp}(k)$, it is required that $\mathcal{D}(sk, \mathcal{E}(pk, M)) = M$.

The syntax of deterministic symmetric encryption schemes is very similar, except the same symmetric key K is used in place of public and secret keys ($pk = sk = K$) and the encryption algorithm is deterministic.

Definition A.1 [Security of Asymmetric Encryption] *Let* $\mathsf{AE} = (\mathcal{EK}, \mathcal{E}, \mathcal{D})$ *be an asymmetric encryption scheme. Consider experiments* $\mathbf{Exp}^{\text{enc-ind-cpa}}_{\mathsf{AE},\mathcal{A},b}(k), \mathbf{Exp}^{\text{enc-ind-cca}}_{\mathsf{AE},\mathcal{A},b}(k)$ *associated to* AE*, a bit* $b \in \{0,1\}$ *and an adversary* $\mathcal{A}$*. In both experiments* $\mathcal{A}$ *is given input a public key* pk*. In experiment* $\mathbf{Exp}^{\text{enc-ind-cca}}_{\mathsf{AE},\mathcal{A},b}(k)$ *the adversary is also given input a decryption oracle* $\mathcal{D}(sk, \cdot)$*. Here* pk *and* sk *are matching keys generated via* $(\mathrm{pk}, \mathrm{sk}) \xleftarrow{\$} \mathcal{EK}(1^k)$*. In the first phase* $\mathcal{A}$ *outputs a pair of messages* $M_0, M_1 \in \mathrm{MsgSp}(k)$ *of equal length, and some state information* st*. In the second phase* $\mathcal{A}$ *gets back the challenge ciphertext computed via* $C \xleftarrow{\$} \mathcal{E}(\mathrm{pk}, M_b)$ *and* st*, and outputs a guess* d *which is also the output of the experiment. In the seconf phase of the experiment* $\mathbf{Exp}^{\text{enc-ind-cca}}_{\mathsf{AE},\mathcal{A},b}(k)$ *the adversary is not allowed to query its decryption oracle on* C*.* AE *is said to be IND-CPA (resp. IND-CCA) secure if the the following*

$$\Pr[\mathbf{Exp}^{\text{enc-ind-atk}}_{\mathsf{AE},\mathcal{A},1}(k) = 1] - \Pr[\mathbf{Exp}^{\text{enc-ind-atk}}_{\mathsf{AE},\mathcal{A},0}(k) = 1] \ .$$

is negligible in k*.*

IND-CPA and IND-CCA security of symmetric encryption schemes is defined similarly, except that adversary is not given any key.

We adopt the convention that the *time complexity* of adversary $\mathcal{A}$ is the execution time of the entire experiment, including the time taken for key generation, and computation of answers to oracle queries. The same convention will be used implicitly in other definitions of the paper.

A.2 Digital Signature Schemes and Their Security

A digital signature scheme $\mathsf{DS} = (\mathcal{SK}, \mathcal{S}, \mathcal{V})$ is specified by four polynomial-time algorithms with the following functionalities. The randomized *key-generation* algorithm $\mathcal{SK}$ takes input 1^k, where k is the security parameter, and outputs a pair (pk, sk) consisting of a public key and a matching secret key, respectively. The (possibly) randomized *signing* algorithm $\mathcal{S}$ takes input a secret key sk and a message $M \in \{0,1\}^*$, and outputs a signature σ. The deterministic *verification* algorithm $\mathcal{V}$ takes input a public key pk, a message M and a candidate signature σ for M, and outputs a bit. We say that σ is a *valid* signature for M relative to pk if $\mathcal{V}(pk, M, \sigma) = 1$. For any pair of keys $(pk, sk) \in [\mathcal{SK}(1^k)]$ and any $M \in \{0,1\}^*$, it is required that $\mathcal{V}(pk, M, \mathcal{S}(sk, M)) = 1$.

Definition A.2 [Security of Digital Signatures] *Let* $\mathsf{DS} = (\mathcal{SK}, \mathcal{S}, \mathcal{V})$ *be a digital signature scheme. Consider an adversary* $\mathcal{A}$ *that is given input a public key* pk *and access to a signing oracle* $\mathcal{US}_{\mathcal{S}}(sk, \cdot)$*, where* pk *and* sk *are matching keys generated via* $I \xleftarrow{\$} \mathcal{SG}(1^k)$; $(pk, sk) \xleftarrow{\$} \mathcal{SK}(I)$*. The oracle takes input a message* M *and returns a signature* $\sigma \xleftarrow{\$} \mathcal{S}(sk, M)$*.* $\mathcal{A}$ *queries*

this oracle on messages of its choice, and eventually outputs a forgery (M, σ). DS *is said to be secure against existential forgery under adaptive chosen-message attacks (or, simply, secure) if the probability that the adversary outputs a pair* (M, σ) *such that* σ *is a valid signature for message* M *and this message was not queried to the signing oracle, is negligible in* k.

B Auxiliary Results for Section 3

B.1 Constructing Malleable POWHFs

Towards proving our negative result we need malleable POWHFs, i.e., for which $\mathcal{G}_K(x, r) \oplus \Delta = \mathcal{G}_K(x \oplus \delta, r)$ for some δ, Δ which we will specify later. Ironically, to construct such malleable POWHFs we start with POWHFs which are strongly xor-collision-resistant, which basically means that such bit modifications are impossible:

Definition B.1 *Let* $\mathsf{POWHF} = (\mathcal{K}, \mathcal{G}, \mathcal{V})$ *be a POWHF (with respect to* $\mathcal{X}, \mathsf{hint}$*). Then* POWHF *is strongly xor-collision-resistant if for every efficient adversary* $\mathcal{C}$ *and any sequence* $(z_k)_{k\in\mathbb{N}}$ *of values* $z_k \in \{0,1\}^{n(k)}$ *the following holds. For* $k \in \mathbb{N}$ *pick* $K \xleftarrow{\$} \mathcal{K}(1^k)$ *and let* $(x, x', y, y') \xleftarrow{\$} \mathcal{C}(K, z_k)$*. Then*

$$\Pr\left[\, \mathcal{V}(K, x, y) = 1 \wedge \mathcal{V}(K, x', y') = 1 \wedge x \neq x' \wedge y \oplus y' = z_k \,\right]$$

is negligible in k.

Strong xor-collision-resistance (which implies for example regular collision-resistance for the special case $z_k = 0^{n(k)}$) has not been considered in previous works about POWHFs. However, as we show below, this property is already satisfied by the pseudorandom function tribe ensembles in [17] (which is one possibility to build POWHFs), again under the assumption that the distribution $\mathcal{X}$ is uniform and that the uninvertible function hint is trivial. We actually note that the construction of a pseudorandom POWHF with respect to *some* $\mathcal{X}, \mathsf{hint}$ would suffice to get a strongly xor-collision-resistant POWHF with respect to the same pair $\mathcal{X}, \mathsf{hint}$ (if one-way permutations exist).

Proposition B.2 *If one-way permutations exist then there is a pseudorandom and strongly xor-collision-resistant POWHF (with respect to the uniform distribution and the trivial uninvertible function* hint*). If there is a pseudorandom POWHF (with respect to* some *distribution* $\mathcal{X}$ *and* some *uninvertible function* hint*) and one-way permutations exist, then there is also a pseudorandom and strongly xor-collision-resistant POWHFs (with respect to the same* $\mathcal{X}$ *and* hint*). All derived POWHFs have public randomness.*

Proof: To prove the proposition we recall some very basic facts about PRF tribe ensembles [11, 17]. Let $\mathcal{F}$ be a PRF family such that $\mathcal{F}(x, r)$ denotes the value of the pseudorandom function for secret key x at point r. A tribe ensemble now is a PRF family $\mathcal{F}$ with an additional *public* tribe key K such that for any secret keys $x \neq x'$ and r, we have $\mathcal{F}(K, x, r) \oplus \mathcal{F}(K, x', r) \neq z_k$ with overwhelming probability over the choice of K for any sequence $(z_k)_{k\in\mathbb{N}}$ of fixed values. In [11, 17] constructions of PRF tribe ensembles have been given based on any one-way permutation.[2]

[2] Originally defined only for the special case of $z_k = 0^{n(k)}$ but the solution in [17] for example achieves this property for any sequence of fixed values.

They immediately give POWHFs with strong xor-collision-resistance and public randomness via $\mathcal{H}(K,x,r) = (r, \mathcal{F}(K,x,r))$.

The first claim follows easily from the construction of the pseudorandom function tribe ensemble from any one-way permutation in [17] and the construction sketched above. Pseudorandomness of this POWHF is immediate from the pseudorandomness of $\mathcal{F}$. The strong XOR collision-resistance property follows from the random choice of the tribe key, as explained above. The idea for the second part of the claim is to convert x through the pseudorandom POWHF into a uniform-looking string and to use this string as input for the pseudorandom function tribe ensemble. The formalization and the security proof are straightforward. ▌

Given such strongly xor-collision-resistant POWHFs we can now construct our malleable POWHF:

Definition B.3 *Let* $\mathsf{POWHF}' = (\mathcal{K}', \mathcal{G}', \mathcal{V}')$ *be a pseudorandom and strongly XOR-collision-resistant POWHF with public randomness (for the uniform distribution and some function* hint*), where* $\mathcal{G}'\colon \{0,1\}^k \times \{0,1\}^{m-1} \times \mathit{Coins}(K) \to \{0,1\}^n$ *and* $\mathcal{V}\colon \{0,1\}^k \times \{0,1\}^{m-1} \times \{0,1\}^n \to \{0,1\}$*. For every* $K \in [\mathcal{K}'(1^k)]$ *and* $x \in \{0,1\}^m$ *and* $r \in \mathit{Coins}(K)$ *define:*

$$\mathcal{G}_K^{pr}(x,r) = \begin{cases} \mathcal{G}_K'^{pr}(x[2]\|\ldots\|x[m],r), & \text{if } x[1]=0 \\ \mathcal{G}_K'^{pr}(x[2]\|\ldots\|x[m],r) \oplus 1\|0^{n-1}, & \text{if } x[1]=1 \end{cases}$$

$$\mathcal{V}_K(x,(r,y)) = \begin{cases} \mathcal{V}_K'(x[2]\|\ldots\|x[m],(r,y)), & \text{if } x[1]=0 \\ \mathcal{V}_K'(x[2]\|\ldots\|x[m],(r,y\oplus 1\|0^{n-1})), & \text{if } x[1]=1 \end{cases}$$

Claim B.4 $\mathsf{POWHF} = (\mathcal{K}', \mathcal{G}, \mathcal{V})$ *as defined in Construction B.3 is a pseudorandom POWHF with public randomness, with respect to the uniform distribution and the uninvertible function* hint*.*

Proof: It is clear that POWHF satisfies the correctness property. Collision resistance follows from collision-resistance and XOR-collision-resistance of POWHF'. This is because if x, x', r, y are such that $x \neq x'$ and $\mathcal{V}_K(x,(r,y)) = \mathcal{V}_K(x',(r,y)) = 1$, then it is either

- $x[1] = x'[1]$ and $\mathcal{G}_K'^{\mathrm{pr}}(x,r) = \mathcal{G}_K'^{\mathrm{pr}}(x',r)$, or
- $x[1] \neq x'[1]$ and $\mathcal{G}_K'^{\mathrm{pr}}(x,r) \oplus \mathcal{G}_K'^{\mathrm{pr}}(x',r) = 1\|0^n$

But the former condition contradicts the collision-resistance of POWHF', and the latter condition contradicts XOR-collision-resistance of POWHF'. The pseudorandomness property of POWHF easily follows from the pseudorandomness property of POWHF', because the restriction of x to the last $m-1$ bits is also uniformly distributed and the value $1||0^{n-1}$ is added with probability $1/2$. ▌

We finally discuss that the derived POWHF is malleable. Given $\mathcal{G}_K^{\mathrm{pr}}(x,r)$ adding $1||0^{n-1}$ to this value flips the first bit in x:

$$\mathcal{G}_K^{\mathrm{pr}}(x,r) \oplus 1||0^{n-1} = \mathcal{G}_K^{\mathrm{pr}}(x \oplus 1||0^{m-1}, r) \qquad \text{for all } K,x,r.$$

Recall that we have shown in Proposition B.2 that the starting family POWHF' in Construction B.3 (with respect to the uniform distribution and the trivial uninvertible function) exists if

there are one-way permutations. Hence, under the same assumption we can construct our derived family POWHF. Under the more general assumption that such a family POWHF' with respect to the uniform distribution and *arbitrary* hint exists, the constructed POWHF is pseudorandom with respect to the uniform distribution and hint.

B.2 Proof of Theorem 3.1

Let $\mathsf{POWHF} = (\mathcal{K}, \mathcal{G}, \mathcal{V})$ be a pseudorandom POWHF with public randomness (with respect to the uniform distribution and hint), derive from POWHF' according to Construction B.3, where $m = k_0$ and $n = k - k_0$. Let F' be an XOR-malleable one-way permutation family that exists relative to an oracle according to Claim 3.3. For every $s\{0,1\}^{k-k_0}$, $t \in \{0,1\}^{k_0}$ define $f(s\|t) = f'_{\text{left}}(s)\|f'_{\text{right}}(t)$, where $f'_{\text{left}}, f'_{\text{right}}$ are random instances of F'. Clearly, the derived family F is a partial one-way permutation family.

We now show that $\text{OAEP}^{\mathsf{POWHF},H}[F]$ in *not* IND-CCA secure (in the RO model). An adversary $\mathcal{A}$ is given a public key (f, K). Without querying the decryption oracle in the first phase $\mathcal{A}$ outputs two arbitrary but distinct messages M_0, M_1. It gets back a challenge ciphertext $(r^*_{\mathcal{G}}, C^*_{\text{left}}\|C^*_{\text{right}})$ of message M_b, where $|C^*_{\text{left}}| = k - k_0$ and $|C^*_{\text{right}}| = k_0$. In the second phase algorithm $\mathcal{A}$ transforms the right ciphertext part through the XOR-malleable algorithm U to get $C_{\text{right}} \xleftarrow{\$} U(f'_{\text{right}}, C^*_{\text{right}}, 1\|0^{k_0-1})$. If U returns an answer then $\mathcal{A}$ queries $(r^*_{\mathcal{G}}, C^*_{\text{left}}\|C_{\text{right}})$ to its decryption oracle. If the returned message M equals $M_0 \oplus 1||0^{k-k_0-k_1-1}$ then $\mathcal{A}$ outputs 0, otherwise it outputs 1. If U fails then $\mathcal{A}$ outputs 1.

For the analysis assume that U succeeds to transform the ciphertext. This happens with non-negligible probability. Under this condition we have for the challenge ciphertext $f'_{\text{left}}(s^*)||f'_{\text{right}}(t^*)$ where $s^* = M_b||0^{k_0} \oplus \mathcal{G}^{\text{pr}}_K(r^*, r^*_{\mathcal{G}})$ and $t^* = r^* \oplus H(s^*)$:

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

Thus, the submitted ciphertext is valid and the decryption oracle returns $M_b \oplus 1||0^{k-k_0-k_1-1}$. In this case $\mathcal{A}$'s output according to the decision about the answer M is correct. In case U's transformation fails $\mathcal{A}$ outputs the fixed bit 1. It follows that the difference between the probabilities that $\mathcal{A}$ answers 1 for the case $b = 1$ and for the case $b = 0$, is at least the probability that U successfully transforms the ciphertext. This probability is non-negligible by assumption.

Therefore $\text{OAEP}^{\mathsf{POWHF},H}[F]$, an instantiation of the G-oracle in the $\text{OAEP}^{G,H}[F]$ encryption scheme with POWHF, is not IND-CCA in the RO model. ∎

B.3 Proof of Theorem 3.4

Let again $\mathsf{POWHF} = (\mathcal{K}, \mathcal{H}, \mathcal{V})$ is the pseudorandom POWHF with public randomness derive from POWHF' through Construction B.3 (but this time the evaluation function is called $\mathcal{H}$ since we replace the H-oracle). Let F' be once more an XOR-malleable one-way trapdoor family relative to an oracle according to Claim 3.3. For every $s \in \{0,1\}^{k-k_0}$, $t \in \{0,1\}^{k_0}$ let again $f(s\|t) = f'_{\text{left}}(s)\|f'_{\text{right}}(t)$, where $f'_{\text{left}}, f'_{\text{right}}$ are random instances of F'.

The adversary $\mathcal{A}$ is given a public key (f, K). It immediately outputs two arbitrary but distinct messages M_0, M_1 to receive the challenge ciphertext $(r^*_{\mathcal{H}}, C^*_{\text{left}} \| C^*_{\text{right}})$ of message M_b. Algorithm $\mathcal{A}$ now tries to transform both ciphertext parts through U to get $C_{\text{left}} \xleftarrow{\$} U(f'_{\text{left}}, C^*_{\text{left}}, 1\|0^{k-k_0-1})$ and $C_{\text{right}} \xleftarrow{\$} U(f'_{\text{right}}, C^*_{\text{right}}, 1\|0^{k_0-1})$. If successful then $\mathcal{A}$ queries the decryption oracle about $(r^*_{\mathcal{H}}, C_{\text{left}} \| C_{\text{right}})$. If the answer M equals $M_0 \oplus 1||0^{k-k_0-k_1-1}$ then $\mathcal{A}$ outputs 0, otherwise it outputs 1. If U fails to transform any of the two ciphertexts then $\mathcal{A}$ outputs 1.

For the analysis assume that U succeeds to transform both ciphertexts. This happens with non-negligible probability, because the preimage in the corresponding part is random or at least pseudorandom. Given that U succeeds in both cases let $s = s^* \oplus 1\|0^{k-k_0-1}$ and $t = t^* \oplus 1\|0^{k_0-1}$ denote the two values encrypted in C_{left} and C_{right}, respectively. Then,

$$\begin{aligned} t \oplus \mathcal{H}^{\text{pr}}_K(s, r^*_{\mathcal{H}}) &= t^* \oplus 1\|0^{k_0-1} \oplus \mathcal{H}^{\text{pr}}_K(s, r^*_{\mathcal{H}}) \\ &= t^* \oplus \mathcal{H}^{\text{pr}}_K(s \oplus 1||0^{k-k_0-1}, r^*_{\mathcal{H}}) = t^* \oplus \mathcal{H}^{\text{pr}}_K(s^*, r^*_{\mathcal{H}}) \\ &= r^* \end{aligned}$$

and the decryption oracle therefore returns M, where

$$M||0^{k_0} = s \oplus G(r^*) = s^* \oplus 1||0^{k-k_0-1} \oplus G(r^*) = M_b||0^{k_0} \oplus 1||0^{k-k_0-1}$$

Hence, $\mathcal{A}$'s output is correct for successful transformations. It follows as in the proof for the G-oracle that $\text{OAEP}^{G,\mathsf{POWHF}}[F]$, an instantiation of the H-oracle in the $\text{OAEP}^{G,H}[F]$ encryption scheme with POWHF, is not IND-CCA in the RO model. ▌

C Proof of Theorem 4.1

Let $\mathcal{A}$ be an arbitrary probabilistic polynomial-time algorithm. Let $\mathsf{Game}^0_{\mathcal{A},b}(k)$ denote the original attack of $\mathcal{A}$ on the encryption scheme $\text{PSS-I}^{\mathsf{POWHF},H}[F]$ where always message M_b is encrypted in the challenge ciphertext. Let $\mathsf{Game}^1_{\mathcal{A},b}(k)$ denote the game where we replace $\mathcal{G}^{\text{pr}}_K(\omega^*, r_{\mathcal{G}})$ in the challenge ciphertext by a uniformly and independently distributed string u^*. All games are described formally in Figure 1.

Note that in $\mathsf{Game}^2_{\mathcal{A},b}$ the distribution of the data is independent of bit b.Hence, the probabilities $\Pr\left[\,\mathsf{Game}^2_{\mathcal{A},1}(k) = 1\,\right]$ and $\Pr\left[\,\mathsf{Game}^2_{\mathcal{A},0}(k) = 1\,\right]$ for $b = 1$ and $b = 0$, respectively, are identical. Therefore,

$$\begin{aligned} &\Pr\left[\,\mathsf{Game}^0_{\mathcal{A},1}(k) = 1\,\right] - \Pr\left[\,\mathsf{Game}^0_{\mathcal{A},0}(k) = 1\,\right] \\ &\qquad = \sum_{i=0}^{1} \Pr\left[\,\mathsf{Game}^i_{\mathcal{A},1}(k) = 1\,\right] - \Pr\left[\,\mathsf{Game}^{i+1}_{\mathcal{A},1}(k) = 1\,\right] \\ &\qquad\quad + \Pr\left[\,\mathsf{Game}^2_{\mathcal{A},1}(k) = 1\,\right] - \Pr\left[\,\mathsf{Game}^2_{\mathcal{A},0}(k) = 1\,\right] \\ &\qquad\quad + \sum_{i=1}^{0} \Pr\left[\,\mathsf{Game}^{i+1}_{\mathcal{A},0}(k) = 1\,\right] - \Pr\left[\,\mathsf{Game}^i_{\mathcal{A},0}(k) = 1\,\right] \end{aligned}$$

and it suffices to show that $\Pr\left[\,\mathsf{Game}^0_{\mathcal{A},b}(k) = 1\,\right] - \Pr\left[\,\mathsf{Game}^1_{\mathcal{A},b}(k) = 1\,\right]$ and $\Pr\left[\,\mathsf{Game}^1_{\mathcal{A},b}(k) = 1\,\right] - \Pr\left[\,\mathsf{Game}^2_{\mathcal{A},b}(k) = 1\,\right]$ are negligible for any $b \in \{0,1\}$. by flip-

Experiment $\mathsf{Game}^0_{\mathcal{A},b}(k)$**:**
$((f^{-1}, K, r_{\mathcal{G}}), (f, K, r_{\mathcal{G}})) \xleftarrow{\$} \mathcal{EK}(1^k)$
$(M_0, M_1, \text{state}) \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)}(f, K, r_{\mathcal{G}})$
Compute ciphertext (C^*, s^*):
  Pick $r^* \xleftarrow{\$} \{0,1\}^k$
  Compute $\omega^* \leftarrow H(M_b||r^*)$
  Compute $C^* \leftarrow f(\omega^*)$
  Compute $s^* \leftarrow \mathcal{G}^{\text{pr}}_K(\omega^*, r_{\mathcal{G}}) \oplus M_b||r^*$
$d \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)-\{(C^*,s^*)\}}((C^*, s^*), \text{state})$

Experiment $\mathsf{Game}^1_{\mathcal{A},b}(k)$**:**
$((f^{-1}, K, r_{\mathcal{G}}), (f, K, r_{\mathcal{G}})) \xleftarrow{\$} \mathcal{EK}(1^k)$
$(M_0, M_1, \text{state}) \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)}(f, K, r_{\mathcal{G}})$
Compute ciphertext (C^*, s^*):
  Pick $r^* \xleftarrow{\$} \{0,1\}^k$
  Compute $\omega^* \leftarrow H(M_b||r^*)$
  Compute $C^* \leftarrow f(\omega^*)$
  Pick $u^* \xleftarrow{\$} \{0,1\}^{k-k_1}$
  Compute $s^* \leftarrow u^* \oplus M_b||r^*$
$d \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)-\{(C^*,s^*)\}}((C^*, s^*), \text{state})$

Experiment $\mathsf{Game}^2_{\mathcal{A},b}(k)$**:**
$((f^{-1}, K, r_{\mathcal{G}}), (f, K, r_{\mathcal{G}})) \xleftarrow{\$} \mathcal{EK}(1^k)$
$(M_0, M_1, \text{state}) \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)}(f, K, r_{\mathcal{G}})$
Compute ciphertext (C^*, s^*):
  Pick $r^* \xleftarrow{\$} \{0,1\}^k$
  Pick $\omega^* \xleftarrow{\$} \{0,1\}^{k_1}$
  Compute $C^* \leftarrow f(\omega^*)$
  Pick $u^* \xleftarrow{\$} \{0,1\}^{k-k_1}$
  Compute $s^* \leftarrow u^* \oplus M_b||r^*$
$d \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)-\{(C^*,s^*)\}}((C^*, s^*), \text{state})$

Figure 1: Games in the Proof of Theorem 5.1: Shaded areas indicate the differences between the games. It is always assumed that the output (M_0, M_1, state) of $\mathcal{A}$ in the first phase satisfies $|M_0| = |M_1|$.

ping $\mathcal{A}$'s output bit we can always assume that the differences are positive. In the sequel we fix the bit b.

Simulating the Decryption Oracle. We first describe how to simulate decryption queries in the games without knowing the secret key f^{-1}. This is accomplished through the random oracle mode and via one procedure D which works for all games. In addition to a ciphertext (C, s) this procedure gets the public data $f, K, r_{\mathcal{G}}$ and a list L_H, representing $\mathcal{A}$'s queries to H and the answers as input. The procedure checks if there is exactly one pair $(M||r, \omega)$ in L_H such that $C = f(\omega)$ and $s = \mathcal{G}^{\text{pr}}_K(\omega, r_{\mathcal{G}}) \oplus (M||r)$. If so, it outputs M, else it returns $\perp$.

We next prove that this decryption procedure may substitute the actual decryption oracle except with negligible simulation error probability in both games. More formally, this means that for every decryption request in the game we run D (on the list L_H of communication between $\mathcal{A}$ and H up to this point) instead of $\mathcal{D}$. Let $\mathsf{DecError}_i$ denote the event that D returns a different answer than $\mathcal{D}$ for the i-th decryption query in the corresponding game, given that the first $i-1$ replies were identical. It then suffices to show that the probability of $\mathsf{DecError}_i$ is negligible for

arbitrary i. Recall that we call a ciphertext valid iff $\mathcal{D}$ returns a message $M \neq \perp$.

First note that collisions $(M||r, \omega)$, $(M'||r', \omega)$ for different $M||r \neq M'||r'$ in the list L_H are unlikely and happen with negligible probability only at any point. This holds in all games, because the values ω are picked at random. So we can condition on the event that there are no such collision and analyze $\Pr[\,\mathsf{DecError}_i\,]$ under this condition, i.e., it suffices to discuss the case of missing entries in L_H, as this is the only case when D's behavior diverges; if there is a unique entry in L_H then D gives the same answer as the genuine decryption oracle.

Behavior in Game Zero. Assume that $\mathcal{A}$ submits some (C, s) to the decryption oracle with the i-th query in $\mathsf{Game}^0_{\mathcal{A},b}$ such that there is no matching entry in L_H. Let ω, M, r denote the unique values such that $f(\omega) = C$ and $s = \mathcal{G}^{\mathrm{pr}}_K(\omega, r_{\mathcal{G}}) \oplus (M||r)$. Let ω^*, M_b, r^* denote the corresponding values for the challenge ciphertext $(C^*, s^*) = (f(\omega^*), \mathcal{G}^{\mathrm{pr}}_K(\omega^*, r_{\mathcal{G}}) \oplus (M_b||r^*))$.

- If we are in the first phase of the game, before $\mathcal{A}$ receives the challenge ciphertext, and there is no value for $M||r$ in L_H, then $\omega = H(M||r)$ is an unknown random value and the probability that $f(\omega) = C$ is negligible.
- If we are in either phase and $M||r \neq M_b||r^*$ and there is no entry in L_H, then the value $\omega = H(M||r)$ is again independently distributed (in particular, independent of $H(M_b||r^*)$) and the probability that $f(\omega) = C$ is again negligible.
- If we are in the second phase and have $M||r = M_b||r^*$ then we must also have $\omega = \omega^* = H(M_b||r^*)$ and $s = s^* = \mathcal{G}^{\mathrm{pr}}_K(\omega^*, r_{\mathcal{G}}) \oplus (M_b||r^*)$. But then $(C, s) = (C^*, s^*)$ equals the challenge ciphertext, and we presume that $\mathcal{A}$ never submits this ciphertext to the decryption oracle.

Hence, the probability of event $\mathsf{DecError}_i$ is negligible and D simulates $\mathcal{D}$ correctly with overwhelming probability.

Behavior in Games One and Two. We address D's behavior in experiment $\mathsf{Game}^1_{\mathcal{A},b}$. This case is a bit more demanding, as the challenge ciphertext (C^*, s^*) now contains a fake value s^*, where the POWHF value is replaced by a random value, and the adversary may for example at some point submit (C^*, s) for the right value $s = \mathcal{G}^{\mathrm{pr}}_K(\omega^*, r_{\mathcal{G}}) \oplus M_b||r^*$. Denote again by ω, M, r and ω^*, M_b, r^* the values associated to the i-th decryption query and to the challenge ciphertext.

- Suppose $\mathcal{A}$ asks about $M||r$ in the first phase or about $M||r \neq M_b||r^*$ in either phase, and there is no entry in L_H, then it follows as before in $\mathsf{Game}^0_{\mathcal{A},b}$ that the submitted ciphertext is valid with negligible probability only.
- Assume $\mathcal{A}$ submits for the first time a ciphertext (C, s) in the second phase which correctly encodes $M||r = M_b||r^*$. First observe that the value s^* in the challenge ciphertext is distributed independently of $M_b||r^*$. Furthermore, since there is no corresponding entry in L_H the adversary has not asked H about $M_b||r^*$ before. But then the only information about $M_b||r^*$ available to the adversary (in an information-theoretical sense) is that this value must be different from all (at most polynomially many) entries in L_H. Hence, r^* still has superlogarithmic entropy, given the adversary's view, and the probability that $\mathcal{A}$ submits $M||r = M_b||r^*$ is negligible (over the distribution of r^*).

It follows as for $\mathsf{Game}^0_{\mathcal{A},b}$ that D gives the same answers as $\mathcal{D}$, except with negligible probability. The same holds for $\mathsf{Game}^2_{\mathcal{A},b}$, with exactly the same argument as for $\mathsf{Game}^1_{\mathcal{A},b}$. The only difference

between the games here occurs if $\mathcal{A}$ accidently queries H about $M_b||r^*$; but then the decryption simulation works perfectly anyway.

Comparing Games Zero and One. We next show that the difference between $\mathsf{Game}^0_{\mathcal{A},b}$ and $\mathsf{Game}^1_{\mathcal{A},b}$ is negligible. Assume towards contradiction that this probability was non-negligible. Then we construct an algorithm $\mathcal{B}_b$ (for fixed bit b), refuting the pseudorandomness of the POWHF.

Algorithm $\mathcal{B}_b$ is given $(K, r_{\mathcal{G}}, u^*)$ as input, where K is a random POWHF key and either $(r_{\mathcal{G}}, u^*) \xleftarrow{\$} \mathcal{G}_K(\omega^*, r_{\mathcal{G}})$ for random $r_{\mathcal{G}} \xleftarrow{\$} \mathrm{Coins}(K)$ and $\omega^* \xleftarrow{\$} \{0,1\}^{k_1}$, or $(r_{\mathcal{G}}, u^*)$ is completely random. In addition, $\mathcal{B}_b$ is given side information $\mathsf{hint}(\omega^*) = (f, f(\omega^*))$ where f is drawn from F, independently of whether the input $(r_{\mathcal{G}}, u^*)$ is pseudorandom or random. Note that this function hint is uninvertible by the one-wayness of f.

$\mathcal{B}_b$ starts a simulation of $\mathcal{A}$ for input $(f, K, r_{\mathcal{G}})$. Algorithm $\mathcal{B}_b$ perfectly simulates the random oracle H as usual by returning random answers for new queries and repeating replies for identical queries. All communication is stored in a list L_H. $\mathcal{B}_b$ also uses the decryption procedure D (on $L_H, f, K, r_{\mathcal{G}}$) to simulate $\mathcal{A}$'s decryption queries. If $\mathcal{A}$ outputs a message pair M_0, M_1 for the challenge ciphertext then $\mathcal{B}_b$ outputs (C^*, s^*) where $C^* \leftarrow f(\omega^*)$ is taken form $\mathsf{hint}(\omega^*)$ and $s^* \leftarrow u^* \oplus (M_b||r^*)$ for the given u^* and random $r^* \xleftarrow{\$} \{0,1\}^{k_0}$. Whenever $\mathcal{A}$ makes a query (M, r) to H then $\mathcal{B}_b$ checks whether $u^* = s^* \oplus (M||r)$ or not. If true, then $\mathcal{B}_b$ stops with output 1 immediately, else it continues the simulation. If this test never succeeds then $\mathcal{B}_b$ at the end outputs $\mathcal{A}$'s final answer.

We analyze $\mathcal{B}_b$'s success probability. Clearly, in case of a pseudorandom u^*, algorithm $\mathcal{B}_b$ returns 1 with probability at least $\Pr\left[\,\mathsf{Game}^0_{\mathcal{A},b}(k) = 1\,\right]$ (minus a negligible error probability for the simulation of $\mathcal{D}$ by procedure D). This is true since $\mathcal{B}_b$ sometimes even stops prematurely with output 1, namely, if $\mathcal{A}$ asks H about the right value $M_b||r^*$, and given that this does not happen the simulation perfectly mimics $\mathsf{Game}^0_{\mathcal{A},b}$.

On the other hand, if u^* is truly random, then (under the condition that no decryption errors occur) $\mathcal{A}$ asks H about $M_b||r^*$ with negligible probability only. This is true as the the adversary's view is essentially independent of $M_b||r^*$ —again, only the fact that $M_b||r^*$ is not among the previous H-queries is known— and therefore the probability that $\mathcal{A}$ at some point puts an H-query $M||r$ such that

$$s^* \oplus M||r = u^* \oplus M_b||r^* \oplus M||r = u^*$$

for this random and unknown r^*, is negligible. Given that $\mathcal{A}$ never queries H about $M_b||r^*$, algorithm $\mathcal{B}_b$ perfectly simulates experiment $\mathsf{Game}^1_{\mathcal{A},b}$. Hence, $\mathcal{B}_b$ outputs 1 for random u^* with probability at most $\Pr\left[\,\mathsf{Game}^1_{\mathcal{A},b}(k) = 1\,\right]$ (plus a negligible amount). Since both probabilities for the games are assumed to be non-negligibly apart, this contradicts the pseudorandomness of the POWHFs.

Comparing Games One and Two. The adversary can only notice a difference between the games if she queries H about the right value $M_b||r^*$. Since the distribution of the challenge ciphertext is independent of r^* the adversary only has a negligible success probability for such a query.

We conclude that the encryption scheme is IND-CCA in the RO model. ∎

D Proof of Theorem 5.1

Let $\mathcal{A}$ be an arbitrary probabilistic polynomial-time algorithm. We consider the following sequence of games, described formally in Figure 2:

- $\mathsf{Game}^0_{\mathcal{A},b}(k)$: Describes the chosen-ciphertext attack of $\mathcal{A}$ on the G-instantiation of the encryption scheme, where M_b is encrypted in the challenge ciphertext.
- $\mathsf{Game}^1_{\mathcal{A},b}(k)$: Same as in $\mathsf{Game}^0_{\mathcal{A},b}$, except that we substitute the hash value $H(\sigma^*, M_b)$ in the challenge ciphertext of the asymmetric part by an independent random value ω^*.
- $\mathsf{Game}^2_{\mathcal{A},b}(k)$: Similar to $\mathsf{Game}^1_{\mathcal{A},b}$, but in the challenge ciphertext we replace $\mathcal{G}^{\mathrm{pr}}(K, \sigma^*, r)$ by a uniformly distributed string u^*.
- $\mathsf{Game}^3_{\mathcal{A},b}(k)$: Same as $\mathsf{Game}^2_{\mathcal{A},b}$, but for the challenge ciphertext we replace the computation of $C^*_{\mathrm{sym}} \leftarrow \mathcal{E}_{\mathrm{sym}}(u^*, M_b)$ by an encryption of a fixed string $0^{|M_0|} = 0^{|M_1|}$.

Clearly,

$$\begin{aligned}
&\Pr\left[\,\mathsf{Game}^0_{\mathcal{A},1}(k) = 1\,\right] - \Pr\left[\,\mathsf{Game}^0_{\mathcal{A},0}(k) = 1\,\right] \\
&\qquad = \sum_{i=0}^{2} \Pr\left[\,\mathsf{Game}^i_{\mathcal{A},1}(k) = 1\,\right] - \Pr\left[\,\mathsf{Game}^{i+1}_{\mathcal{A},1}(k) = 1\,\right] \\
&\qquad\quad + \Pr\left[\,\mathsf{Game}^3_{\mathcal{A},1}(k) = 1\,\right] - \Pr\left[\,\mathsf{Game}^3_{\mathcal{A},0}(k) = 1\,\right] \\
&\qquad\quad + \sum_{i=2}^{0} \Pr\left[\,\mathsf{Game}^{i+1}_{\mathcal{A},0}(k) = 1\,\right] - \Pr\left[\,\mathsf{Game}^i_{\mathcal{A},0}(k) = 1\,\right]
\end{aligned}$$

In $\mathsf{Game}^3_{\mathcal{A},b}(k)$ the challenge ciphertext and therefore the execution is independent of the value b. Hence, the two probabilities $\Pr\left[\,\mathsf{Game}^3_{\mathcal{A},0}(k) = 1\,\right]$ and $\Pr\left[\,\mathsf{Game}^3_{\mathcal{A},1}(k) = 1\,\right]$ are equal. It therefore suffices to show that the transition from $\mathsf{Game}^i_{\mathcal{A},b}(k)$ to $\mathsf{Game}^{i+1}_{\mathcal{A},b}(k)$ for all $i = 0, 1, 2$ and $b \in \{0, 1\}$ cannot increase the advantage by more than a negligible amount. In the sequel we fix the bit b and prove similarity of the three adjacent games $\mathsf{Game}^i_{\mathcal{A},b}(k)$, $\mathsf{Game}^{i+1}_{\mathcal{A},b}(k)$ for $i = 0, 1, 2$.

Simulating the Decryption Oracle. As in the proof of Theorem 4.1 we first describe how to simulate decryption queries in the games without knowing the secret key sk in the random oracle model. Consider the following procedure D which has access to a list L_H of all oracle queries $\mathcal{A}$ makes in the corresponding experiment to H and all replies; D is also given pk, K, r as input. For a decryption request $(C_{\mathrm{asym}}, C_{\mathrm{sym}})$, procedure D checks if there is a pair $((\sigma, M), \omega)$ in L_H such that $C_{\mathrm{asym}} = \mathcal{E}_{\mathrm{asym}}(pk, \sigma, \omega)$. If not or if there is more than one such pair then D returns $\perp$. Else, it verifies that $M = \mathcal{D}_{\mathrm{sym}}(\mathcal{G}^{\mathrm{pr}}(K, \sigma, r), C_{\mathrm{sym}})$ and, if so, D returns M. Otherwise it outputs $\perp$.

We again consider the event $\mathsf{DecError}_i$ (relative to the game) that D gives a different answer for the i-th decryption query than $\mathcal{D}$, under the condition that both procedures coincide on the first $i-1$ decryption requests. Note that if D finds a unique matching entry $((\sigma, M), \omega)$ in L_H then it gives the same answer as the original decryption procedure. It thus suffices to analyze

Experiment $\mathsf{Game}^0_{\mathcal{A},b}(k)$**:**
$((sk,K,r),(pk,K,r)) \xleftarrow{\$} \mathcal{EK}(1^k)$
$(M_0,M_1,\text{state}) \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)}(pk,K,r)$
Compute ciphertext $C^* = (C^*_{\text{asym}}, C^*_{\text{sym}})$:
  Pick $\sigma^* \xleftarrow{\$} \text{MsgSp}_{\text{asym}}(k)$
  Compute $C^*_{\text{asym}} \leftarrow \mathcal{E}_{\text{asym}}(pk,\sigma^*;H(\sigma^*,M_b))$
  Compute $C^*_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(\mathcal{G}^{\text{pr}}(K,\sigma^*,r),M_b)$
$d \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)-\{C^*\}}(C^*,\text{state})$

Experiment $\mathsf{Game}^1_{\mathcal{A},b}(k)$**:**
$((sk,K,r),(pk,K,r)) \xleftarrow{\$} \mathcal{EK}(1^k)$
$(M_0,M_1,\text{state}) \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)}(pk,K,r)$
Compute ciphertext $C^* = (C^*_{\text{asym}}, C^*_{\text{sym}})$:
  Pick $\sigma^* \xleftarrow{\$} \text{MsgSp}_{\text{asym}}(k)$
  Pick $\omega^* \xleftarrow{\$} \text{Coins}_{\text{asym}}(k)$
  Compute $C^*_{\text{asym}} \leftarrow \mathcal{E}_{\text{asym}}(pk,\sigma^*;\omega^*)$
  Compute $C^*_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(\mathcal{G}^{\text{pr}}(K,\sigma^*,r),M_b)$
$d \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)-\{C^*\}}(C^*,\text{state})$

Experiment $\mathsf{Game}^2_{\mathcal{A},b}(k)$**:**
$((sk,K,r),(pk,K,r)) \xleftarrow{\$} \mathcal{EK}(1^k)$
$(M_0,M_1,\text{state}) \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)}(pk,K,r)$
Compute ciphertext $C^* = (C^*_{\text{asym}}, C^*_{\text{sym}})$:
  Pick $\sigma^* \xleftarrow{\$} \text{MsgSp}_{\text{asym}}(k)$
  Pick $\omega^* \xleftarrow{\$} \text{Coins}_{\text{asym}}(k)$
  Pick $u^* \xleftarrow{\$} \text{Keys}_{\text{asym}}(k)$
  Compute $C^*_{\text{asym}} \leftarrow \mathcal{E}_{\text{asym}}(pk,\sigma^*;\omega^*)$
  Compute $C^*_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(u^*,M_b)$
$d \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)-\{C^*\}}(C^*,\text{state})$

Experiment $\mathsf{Game}^3_{\mathcal{A},b}(k)$**:**
$((sk,K,r),(pk,K,r)) \xleftarrow{\$} \mathcal{EK}(1^k)$
$(M_0,M_1,\text{state}) \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)}(pk,K,r)$
Compute ciphertext $C^* = (C^*_{\text{asym}}, C^*_{\text{sym}})$:
  Pick $\sigma^* \xleftarrow{\$} \text{MsgSp}_{\text{asym}}(k)$
  Pick $\omega^* \xleftarrow{\$} \text{Coins}_{\text{asym}}(k)$
  Pick $u^* \xleftarrow{\$} \text{Keys}_{\text{asym}}(k)$
  Compute $C^*_{\text{asym}} \leftarrow \mathcal{E}_{\text{asym}}(pk,\sigma^*,\omega^*)$
  Compute $C^*_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(u^*, 0^{|M_b|})$
$d \xleftarrow{\$} \mathcal{A}^{H,\mathcal{D}(sk,\cdot)-\{C^*\}}(C^*,\text{state})$

Figure 2: Games in the Proof of Theorem 5.1: Shaded areas indicate the differences between the games. It is always assumed that the output (M_0,M_1,state) of $\mathcal{A}$ in the first phase satisfies $|M_0| = |M_1|$.

the probability of $\mathsf{DecError}_i$ under the condition that D does not find an entry $((\sigma,M),\omega)$ in L_H although the ciphertext is valid. Here we can again assume that collisions in the list L_H at any point, i.e., pairs $((\sigma,M),\omega)$, $((\sigma',M'),\omega)$ with $(\sigma,M) \neq (\sigma',M')$, do not occur. This is true with overwhelming probability as collisions among q values appear with probability at most $q^2/|\text{Coins}_{\text{asym}}(k)|$. Since q is assumed to be polynomial and $\text{Coins}_{\text{asym}}$ must be a superpolynomial set by the IND-CPA property, this term is negligible.

Below we let $(C_{\text{asym}}, C_{\text{sym}})$ denote the i-th decryption query (which can be assumed to be valid) and denote by M,σ the unique values encrypted in $(C_{\text{asym}}, C_{\text{sym}})$. Recall that we consider the case that this pair (M,σ) does not appear in list L_H. Accordingly, $(C^*_{\text{asym}}, C^*_{\text{sym}})$ is the challenge ciphertext (computed according to the game). For each game we can assign a pair (M_b,σ^*) to this challenge ciphertext, as defined by Figure 2 (where M_b is determined by $\mathcal{A}$'s output in the first phase and σ^* through the encryption process).

Behavior in Game Zero. To analyze D's behavior in $\mathsf{Game}^0_{\mathcal{A},b}$ we use the fact that for a given pair C_{asym},σ the probability that $C_{\text{asym}} = \mathcal{E}_{\text{asym}}(pk,\sigma;\omega)$ for a random and independent value $\omega \leftarrow \text{Coins}_{\text{asym}}(k)$ is negligible. Otherwise it would contradict the IND-CPA security of the

encryption scheme in a straightforward way.[3] We distinguish between the following cases:

- If $(C_{\text{asym}}, C_{\text{sym}})$ is submitted in the first phase, before $\mathcal{A}$ sees the challenge, and there is no entry (M, σ) in L_H then $H(M, \sigma)$ is an independent, uniformly distributed and unknown string. The probability that $C_{\text{asym}} = \mathcal{E}_{\text{asym}}(pk, \sigma, H(\sigma, M))$ is therefore negligible.
- Assume $\mathcal{A}$'s query is made after having received the challenge ciphertext, but $C_{\text{asym}} \neq C^*_{\text{asym}}$ and there is no list entry for (M, σ). If $(\sigma, M) = (\sigma^*, M_b)$ then this would contradict $C_{\text{asym}} \neq C^*_{\text{asym}}$. Hence, the value $H(\sigma, M)$ is independent of $H(\sigma^*, M_b)$ and any other hash value. But since there is no value in L_H for (σ, M) it follows again that the probability $C_{\text{asym}} = \mathcal{E}_{\text{asym}}(pk, \sigma, H(\sigma, M))$ is negligible.
- Let $C^*_{\text{asym}} = C_{\text{asym}}$ for the query in the second phase. Then the symmetric parts must be distinct. In this case we have inequality $M_b \neq M$ for the encapsulated messages in $C^*_{\text{sym}}, C_{\text{sym}}$ (as both asymmetric parts encrypt the same value $\sigma = \sigma^*$). Once more, since (M, σ) does not appear in L_H and is different from (M_b, σ^*) the probability that the hash value $H(\sigma, M)$ yields C_{asym} is again negligible.

Overall, D gives the same answer for the i-th query as $\mathcal{D}$ does, except with negligible error.

Behaviors in Games One, Two and Three. D's behavior in these games is even simpler to analyze, as the challenge ciphertext is now independent of the hash function. If there is no entry in L_H for $M||r$, no matter whether $M||r = M_b||r^*$ or not, then the hash value $H(M||r)$ is an independent secret random value, and the probability for $C_{\text{asym}} = \mathcal{E}_{\text{asym}}(pk, \sigma; H(M||r))$ is negligible. Note that our analysis does not rely on the possibility that $\mathcal{A}$ realizes that the ciphertext in these games is fake or submits an invalid ciphertext derived as a modification of the challenge ciphertext; we only care about the difference in $\mathcal{D}$'s and D's answers for decryption queries here.

Comparing Games Zero and One. We show that the probability that $\mathcal{A}$ queries H about the challenge values (σ^*, M_b) during experiment $\mathsf{Game}^0_{\mathcal{A},b}$ is negligible. Assume that this was not the case. Then we construct an algorithm $\mathcal{B}_b$ (for fixed bit b), successfully attacking the one-wayness of the POWHF with respect to the following uninvertible function hint defined by

$$\mathsf{hint}(\sigma^*) = \left(pk, C^*_{\text{asym}}\right) \qquad \text{where } (sk, pk) \xleftarrow{\$} \mathcal{EK}_{\text{asym}}(1^k), C^*_{\text{asym}} \xleftarrow{\$} \mathcal{E}_{\text{asym}}(pk, \sigma^*).$$

Presuming the security of the encryption scheme this function is uninvertible for the well-spread distribution $\sigma^* \xleftarrow{\$} \text{MsgSp}_{\text{asym}}(k)$ since it is an independent encryption of σ^*. The IND-CPA property of the encryption scheme also implies that one can replace this function hint by the trivial uninvertible function $\mathsf{hint}(\sigma^*) = 0$.[4]

[3] If this probability was non-negligible for some σ, C_{asym} then we could construct a (non-uniform) CPA-attacker with "wired" σ, C_{asym}, and which gives σ and some $\sigma' \neq \sigma$ to the encryption oracle to receive a challenge ciphertext C^*_{asym}. If $C_{\text{asym}} = C^*_{\text{asym}}$ then this attacker outputs 1, else it returns 0. If σ' is encrypted in this challenge ciphertext then it can never equal C_{asym} which encrypts σ. In the other case the random encryption equals C_{asym} with non-negligible probability by assumption. Hence, this attacker would distinguish the two cases with non-negligible probability.

[4] Formally, we can replace the value $\mathsf{hint}(\sigma^*)$ by an independent encryption of the all-zero string, and this information can be simulated given the trivial value 0. This is done on the "POWHF level", of course, not here in the proof.

Attacker $\mathcal{B}_b$ (with binary output) is given as input $(K, \mathsf{hint}(\sigma^*), (r, u^*))$, where u^* is either $\mathcal{G}^{\mathrm{pr}}(K, \sigma^*, r)$ or $\mathcal{G}^{\mathrm{pr}}(K, \sigma', r)$ for an independent sample σ'. Algorithm $\mathcal{B}_b$ initially picks an index i between 1 and Q, the polynomial bound on the number of hash queries $\mathcal{A}$ makes. It starts an emulation of $\mathcal{A}$ mounting a chosen-ciphertext attack. In particular, $\mathcal{B}_b$ perfectly simulates $\mathcal{A}$'s access to random oracle H in the common way, i.e., by keeping track of previous queries in a list L_H, returning the same answer if queried twice, and generating random answers if queried for the first time. Any decryption query of $\mathcal{A}$ is answered by running procedure D described above for list L_H and K, r, pk. If $\mathcal{A}$ generates a pair of challenge messages (M_0, M_1) then $\mathcal{B}_b$ computes $C^*_{\mathrm{sym}} \leftarrow \mathcal{E}_{\mathrm{sym}}(u^*, M_b)$ for the given value u^* and returns $(C^*_{\mathrm{asym}}, C^*_{\mathrm{sym}})$ where C^*_{asym} is taken from $\mathsf{hint}(\sigma^*)$. If $\mathcal{A}$ (or D) makes the i-th query (σ, M) to H then $\mathcal{A}^*$ stops and outputs 1 if and only if $\mathcal{G}^{\mathrm{pr}}(K, \sigma, r) = u^*$.

Given input $u^* = \mathcal{G}^{\mathrm{pr}}(K, \sigma^*, r)$ and that $\mathcal{A}$ or D queries H about (σ^*, M_b) at some point, algorithm $\mathcal{B}_b$ guesses the right query with probability $1/Q$. This holds as the simulation up to the right query is identical to $\mathcal{A}$'s attack $\mathsf{Game}^0_{\mathcal{A},b}$, except for a negligible error probability due to the error in simulated decryption queries. Since the probability of asking H about the pair is non-negligible by assumption, attacker $\mathcal{B}_b$ thus outputs 1 with non-negligible probability for the case $u^* = \mathcal{G}^{\mathrm{pr}}(K, \sigma^*, r)$.

On the other hand, given $u^* = \mathcal{G}^{\mathrm{pr}}(K, \sigma', r)$ the probability that $\mathcal{G}^{\mathrm{pr}}(K, \sigma, r) = u^*$ for any query σ of $\mathcal{A}$ is negligible by the one-wayness and the collision-intractability of the POWHF. Specifically, the probability that some query includes the value σ' is negligible by the (regular) one-wayness of the POWHF. Any other H-query about (σ, M) for $\sigma \neq \sigma'$ resulting in the same value u^* would give a collision. Both arguments can be easily turned into formal algorithms refuting the one-wayness and collision-resistance, respectively. Thus, $\mathcal{B}_b$ outputs 1 for $u^* = \mathcal{G}^{\mathrm{pr}}(K, \sigma', r)$ with negligible probability only.

It follows that the random variables $(K, x, \mathcal{B}_b(K, \mathsf{hint}(\sigma^*), (r, u^*)))$ for the two cases are easy to distinguish, contradicting the one-wayness of the POWHF. Given that $\mathcal{A}$ never queries H about (σ^*, M_b) it is clear that the hash value is independently distributed, and that the outputs of $\mathcal{A}$ in $\mathsf{Game}^0_{\mathcal{A},b}$ and $\mathsf{Game}^1_{\mathcal{A},b}$ are indistinguishable.

Comparing Games One and Two. We show that any non-negligible difference in $\mathcal{A}$'s output for the two games would refute the pseudorandomness of the POWHF $(\mathcal{K}, \mathcal{G}, \mathcal{V})$. Specifically, suppose that $\Pr\left[\,\mathsf{Game}^1_{\mathcal{A},b}(k) = 1\,\right] - \Pr\left[\,\mathsf{Game}^2_{\mathcal{A},b}(k) = 1\,\right]$ is non-negligible as a function of parameter k (by flipping $\mathcal{A}$'s output bit we can always guarantee that the difference is non-negative for infinitely many k's). Then we construct the following probabilistic polynomial-time algorithm $\mathcal{B}_b$ (for fixed bit b) which gets as input a tuple $(K, \mathsf{hint}(\sigma^*), (r, u^*))$ and uses $\mathcal{A}$ to decide whether (r, u^*) is the output of $\mathcal{G}$ or random. Here the probabilistic function $\mathsf{hint}(\sigma^*) = (pk, C^*_{\mathrm{asym}})$ is defined as above.

Algorithm $\mathcal{B}_b$ runs a black-box simulation of $\mathcal{A}$ and emulates $\mathcal{A}$'s access to random oracle H with the usual technique. It also uses procedure D to simulate decryption queries. For the simulation $\mathcal{B}_b$ takes K, r from its input $(K, \mathsf{hint}(\sigma^*), (r, u^*))$ and pk from $\mathsf{hint}(\sigma^*)$. When $\mathcal{A}$ outputs the challenge message pair (M_0, M_1) algorithm $\mathcal{B}_b$ creates a ciphertext by letting $C^*_{\mathrm{sym}} \leftarrow \mathcal{E}_{\mathrm{sym}}(u^*, M_b)$ for the given value u^* and returns $C^* = (C^*_{\mathrm{asym}}, C^*_{\mathrm{sym}})$ where C^*_{asym} originates from $\mathsf{hint}(\sigma^*)$. Algorithm $\mathcal{B}_b$ continues $\mathcal{A}$'s attack as before and outputs $\mathcal{A}$'s prediction d.

It is clear that $\mathcal{B}_b$ outputs 1 for a hash function input $(K, \mathsf{hint}(\sigma^*), \mathcal{G}(K, \sigma^*, r))$ with the

same probability as $\mathcal{A}$ does in experiment $\mathsf{Game}^1_{\mathcal{A},b}$ (except for a negligible error for simulating the decryption requests). On the other hand, $\mathcal{B}_b$ outputs 1 with the same probability as $\mathcal{A}$ in experiment $\mathsf{Game}^2_{\mathcal{A},b}$ for random values (r, u^*) (again, up to a negligible error). By assumption the probabilities have non-negligible difference. This, however, contradicts the pseudorandomness of the POWHF and our initial assumption about $\mathcal{A}$'s advantage must have been wrong.

Comparing Games Two and Three. It follows straightforwardly from the IND-CPA security of the symmetric encryption scheme that $\mathcal{A}$'s output in games $\mathsf{Game}^2_{\mathcal{A},b}$ and $\mathsf{Game}^3_{\mathcal{A},b}$ is indistinguishable. If $\mathcal{A}$'s advantage would be non-negligible for the two experiments then it would be easy to distinguish encryptions of M_b and $0^{|M_b|}$ in a chosen-plaintext attack, by emulating the experiments with knowledge of the asymmetric decryption key.

This proves that the derived scheme is IND-CCA. ∎

E Proof of Theorem 5.2

Let $\mathcal{A}$ be a probabilistic polynomial-time algorithm. We consider again a sequence of games, formally described in Figure 3, such that the starting game corresponds to $\mathcal{A}$'s attack scenario and the final game is independent of b:

- $\mathsf{Game}^0_{\mathcal{A},b}$ describes the attack on the encryption scheme when message M_b is encrypted.
- $\mathsf{Game}^1_{\mathcal{A},b}$ describes the game where we replace the symmetric encryption key $G(\sigma^*)$ in the challenge ciphertext by an independent key κ^*.
- $\mathsf{Game}^2_{\mathcal{A},b}$ modifies $\mathsf{Game}^1_{\mathcal{A},b}$ insofar as the challenge ciphertext now contains an (asymmetric) encryption $\mathcal{E}_{\mathrm{asym}}(pk, \sigma^*, u^*)$ with an independent random value u^* instead of $\mathcal{H}(K, \sigma^*||M_b, r)$.
- $\mathsf{Game}^3_{\mathcal{A},b}$ is identical to $\mathsf{Game}^2_{\mathcal{A},b}$, except that the symmetric part of the challenge ciphertext now contains an encryption of $0^{|M_b|}$ instead of M_b.

With the same argument as in the proof of Theorem 5.1 it suffices to show that the differences $\Pr\left[\,\mathsf{Game}^i_{\mathcal{A},b} = 1\,\right](k) - \Pr\left[\,\mathsf{Game}^{i+1}_{\mathcal{A},b}(k) = 1\,\right]$ are negligible for all $i = 0, 1, 2$ and $b \in \{0, 1\}$. Fix again the bit b for this.

Simulating the Decryption Oracle. Analogously to the proofs of Theorems 4.1 and 5.1 we again first describe a procedure to simulate decryption queries. This procedure D takes as input a ciphertext $(C_{\mathrm{asym}}, C_{\mathrm{sym}})$, a list L_G of queries of $\mathcal{A}$ and answers random oracle G as well as pk, K, r. It checks if there are pairs (σ, κ) in L_G such that for $m \leftarrow \mathcal{D}_{\mathrm{sym}}(\kappa, C_{\mathrm{sym}})$ it holds $C_{\mathrm{asym}} = \mathcal{E}_{\mathrm{asym}}(pk, \sigma, \mathcal{H}^{\mathrm{pr}}(K, \sigma||m, r))$. If there is a unique pair then D returns m, else if returns $\perp$.

Let $\mathsf{DecError}_i$ once more denote the event that D returns a different answers than $\mathcal{D}$ for the i-th query, conditioning on equal replies for the first $i-1$ decryption requests (when D is run instead of $\mathcal{D}$ in the corresponding game). The i-th query $(C_{\mathrm{asym}}, C_{\mathrm{sym}})$, which we can assume to be valid, uniquely determines some σ and therefore some M. In particular, collisions in the list L_G cannot occur. Since D yields the same answer for this query if there is exactly one entry

Experiment $\mathsf{Game}^0_{\mathcal{A},b}$**:**
$((sk, K, r), (pk, K, r)) \xleftarrow{\$} \mathcal{EK}(1^k)$
$(M_0, M_1, \text{state}) \xleftarrow{\$} \mathcal{A}^{G,\mathcal{D}(sk,\cdot)}(pk)$
Compute ciphertext $C^* = (C^*_{\text{asym}}, C^*_{\text{sym}})$:
  Pick $\sigma^* \xleftarrow{\$} \text{Keys}(1^k)$
  Compute $C^*_{\text{asym}} \xleftarrow{\$} \mathcal{E}_{\text{asym}}(pk, \sigma^*; \mathcal{H}^{\text{pr}}_K(\sigma^*||M_b, r^*))$
  Compute $C^*_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(G(\sigma^*), M_b)$
$d \xleftarrow{\$} \mathcal{A}^{G,\mathcal{D}(sk,\cdot)-\{C^*\}}(C^*, \text{state})$

Experiment $\mathsf{Game}^1_{\mathcal{A},b}$**:**
$((sk, K, r), (pk, K, r)) \xleftarrow{\$} \mathcal{EK}(1^k)$
$(M_0, M_1, \text{state}) \xleftarrow{\$} \mathcal{A}^{G,\mathcal{D}(sk,\cdot)}(pk)$
Compute ciphertext $C^* = (C^*_{\text{asym}}, C^*_{\text{sym}})$:
  Pick $\sigma^* \xleftarrow{\$} \text{MsgSp}_{\text{asym}}(k)$
  Compute $C^*_{\text{asym}} \xleftarrow{\$} \mathcal{E}_{\text{asym}}(pk, \sigma^*; \mathcal{H}^{\text{pr}}_K(\sigma^*||M_b, r^*))$
  Pick $\kappa^* \xleftarrow{\$} \text{Keys}_{\text{sym}}(k)$
  Compute $C^*_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(\kappa^*, M_b)$
$d \xleftarrow{\$} \mathcal{A}^{G,\mathcal{D}(sk,\cdot)-\{C^*\}}(C^*, \text{state})$

Experiment $\mathsf{Game}^2_{\mathcal{A},b}$**:**
$((sk, K, r), (pk, K.r)) \xleftarrow{\$} \mathcal{EK}(1^k)$
$(M_0, M_1, \text{state}) \xleftarrow{\$} \mathcal{A}^{G,\mathcal{D}(sk,\cdot)}(pk)$
Compute ciphertext $C^* = (C^*_{\text{asym}}, C^*_{\text{sym}})$:
  Pick $\sigma^* \xleftarrow{\$} \text{MsgSp}_{\text{asym}}(k)$
  Pick $u^* \xleftarrow{\$} \text{Coins}_{\text{asym}}(1^K)$
  Compute $C^*_{\text{asym}} \xleftarrow{\$} \mathcal{E}_{\text{asym}}(pk, \sigma^*; u^*)$
  Pick $\kappa^* \xleftarrow{\$} \text{Keys}_{\text{sym}}(k)$
  Compute $C^*_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(\kappa^*, M_b)$
$d \xleftarrow{\$} \mathcal{A}^{G,\mathcal{D}(sk,\cdot)-\{C^*\}}(C^*, \text{state})$

Experiment $\mathsf{Game}^3_{\mathcal{A},b}$**:**
$((sk, K, r), (pk, K.r)) \xleftarrow{\$} \mathcal{EK}(1^k)$
$(M_0, M_1, \text{state}) \xleftarrow{\$} \mathcal{A}^{G,\mathcal{D}(sk,\cdot)}(pk)$
Compute ciphertext $C^* = (C^*_{\text{asym}}, C^*_{\text{sym}})$:
  Pick $\sigma^* \xleftarrow{\$} \text{MsgSp}_{\text{asym}}(k)$
  Pick $u^* \xleftarrow{\$} \text{Coins}_{\text{asym}}(1^K)$
  Compute $C^*_{\text{asym}} \xleftarrow{\$} \mathcal{E}_{\text{asym}}(pk, \sigma^*; u^*)$
  Pick $\kappa^* \xleftarrow{\$} \text{Keys}_{\text{sym}}(k)$
  Compute $C^*_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(\kappa^*, 0^{|M_b|})$
$d \xleftarrow{\$} \mathcal{A}^{G,\mathcal{D}(sk,\cdot)-\{C^*\}}(C^*, \text{state})$

Figure 3: Games in the Proof of Theorem 5.2: Shaded areas indicate the differences between the games. It is always assumed that the output (M_0, M_1, state) of $\mathcal{A}$ in the first phase satisfies $|M_0| = |M_1|$.

(σ, κ) in L_G we condition again on that there is no value (σ, κ) in L_G. Let σ^* be the value that is encrypted in C^*_{asym}.

Behavior in Game Zero. We distinguish between the cases that $\sigma \neq \sigma^*$ and $\sigma = \sigma^*$. Possibly, σ^* has not been determined at this point, if we are still in the first phase of the attack.

- Suppose $\sigma \neq \sigma^*$, which happens also with overwhelming probability if the query is submitted before seeing the ciphertext. Since there is no entry in L_G about σ the value $G(\sigma)$ has not been defined yet, not even implicitly through σ^* in the challenge ciphertext since $\sigma \neq \sigma^*$. Since we can assume that $G(\sigma)$ is given through a random and unknown key κ the probability that C_{sym} is a valid ciphertext under this key is negligible by the INT-CTXT property. This is straightforward to formalize. Hence, rejecting such a query through D is correct with overwhelming probability.
- Suppose $\sigma = \sigma^*$ and $C^*_{\text{sym}} = C_{\text{sym}}$. Then we must also have $M = M_b$ and $C^*_{\text{asym}} \neq C^*_{\text{asym}}$. But then C_{asym} cannot be a valid encryption, and rejecting this ciphertext is correct.

The final case, $\sigma = \sigma^*$ and $C^*_{\text{sym}} \neq C_{\text{sym}}$, needs a more thorough treatment. Assume that there is no entry in L_G and that the ciphertext is valid, i.e., should not be rejected by D. We first claim

that the probability that $\mathcal{A}$ has queried random oracle G about σ^* at this point is negligible.

If $\mathcal{A}$ would query G about σ^* with non-negligible probability then it would be straightforward to derive an algorithm $\mathcal{B}_b$ (with fixed bit b) refuting the POWHF-encryption assumption. Namely, algorithm $\mathcal{B}_b$ is given as input $(pk, K, r, C^*_{\text{asym}})$ and simulates $\mathcal{A}$'s attack by supplying a perfect simulation of the random oracle G as usual, and simulating the decryption queries through procedure D. If $\mathcal{B}_b$ is given a message pair M_0, M_1 and state from $\mathcal{A}$ —which describes an efficient distribution $\mathcal{M}(pk, K, r)$ on the messages— then it picks κ^* at random and computes $C^*_{\text{sym}} \leftarrow \mathcal{E}_{\text{sym}}(\kappa^*, M_b)$ and returns $(C^*_{\text{asym}}, C^*_{\text{sym}})$ to continue $\mathcal{A}$'s simulation. $\mathcal{B}_b$ initially also guesses a number j among the at most polynomially many G-queries and aborts the simulation immediately if $\mathcal{A}$ puts the j-th query σ to G. Then $\mathcal{B}_b$ checks whether $C^*_{\text{asym}} = \mathcal{E}(pk, \sigma, \mathcal{H}^{\text{pr}}(K, \sigma||M_b, r))$. If so, $\mathcal{B}_b$ outputs the guess 1, else it outputs the guess 0.

For the analysis note that, if the probability that $\mathcal{A}$ queries G about σ^* before submitting the i-th decryption query is non-negligible, then $\mathcal{B}_b$ guesses the right query with non-negligible probability, too. In this case $\mathcal{B}_b$ predicts the type of input correctly, and for the other cases $\mathcal{B}_b$ outputs the fixed bit 0. Overall the advantage of $\mathcal{B}_b$ is thus non-negligible, and we conclude that $\mathcal{A}$ cannot query G about σ^* with more than negligible probability.

Conditioning on that $\mathcal{A}$ has never queried G about σ^* replacing the value $G(\sigma^*)$ by an independent random key κ^* generates the same view for $\mathcal{A}$, and thus $\mathcal{A}$ triggers event $\mathsf{DecError}_i$ also with non-negligible probability if we substitute $G(\sigma^*)$ by κ^*. We next prove that this, together with $\sigma = \sigma^*$ and $C^*_{\text{sym}} \neq C_{\text{sym}}$, contradicts the integrity of ciphertexts of the symmetric scheme.

We construct again an algorithm $\mathcal{B}_b$, but this time $\mathcal{B}_b$ refutes the INT-CTXT property of the symmetric encryption scheme. Algorithm $\mathcal{B}_b$ is given access to an encryption oracle $\mathcal{E}_{\text{sym}}(\kappa^*, \cdot)$ for an unknown key κ^*. It tries to predict the index i by guessing a random j and then starts to simulate $\mathcal{A}$'s attack. $\mathcal{B}_b$ again provides a simulated random oracle G and answers decryption requests with D. All other complementary inputs to $\mathcal{A}$ like the public key (pk, K, r) are chosen by $\mathcal{B}_b$. For the challenge ciphertext $\mathcal{B}_b$ computes an encryption C^*_{asym} for random σ^* itself and queries the given oracle about M_b to get C^*_{sym}. It returns $(C^*_{\text{asym}}, C^*_{\text{sym}})$ to $\mathcal{A}$. If $\mathcal{A}$ eventually outputs the i-th decryption query $(C_{\text{asym}}, C_{\text{sym}})$ then $\mathcal{B}_b$ stops with output C_{sym}.

Note that if the probability for $\mathsf{DecError}_i$ and $\sigma = \sigma^*$, $C_{\text{sym}} \neq C^*_{\text{sym}}$ for a valid C_{sym}, is non-negligible for some i, then $\mathcal{B}_b$ guesses the right index with non-negligible probability. Under this condition $\mathcal{B}_b$ generates a valid but new ciphertext under the unknown key κ^*, contradicting the ciphertext integrity of the symmetric scheme.

In summary, events $\mathsf{DecError}_i$ only occur with negligible probability. It follows that D simulates decryption queries with overwhelming probability.

Behavior in Games One, Two and Three. The indistinguishable behavior of D and $\mathcal{D}$ in these games follows as in the previous case from the INT-CTXT property of the symmetric encryption scheme. The reason is that the challenge ciphertext is independent of oracle G, and if there is no entry for σ in L_G then the key of the symmetric scheme is an undetermined random value.

Comparing the Games. The indistinguishability of $\mathcal{A}$'s output primarily follows from the analysis of D's behavior in $\mathsf{Game}^0_{\mathcal{A},b}$. Specifically,

- For $\mathsf{Game}^0_{\mathcal{A},b}$ and $\mathsf{Game}^1_{\mathcal{A},b}$ we have already shown in the analysis of D for $\mathsf{Game}^0_{\mathcal{A},b}$ that $\mathcal{A}$ queries G about σ^* with negligible probability only. We can thus replace the value $G(\sigma^*)$

in the challenge ciphertext by an independent key κ^* without altering $\mathcal{A}$'s output behavior significantly.

- For $\mathsf{Game}^1_{\mathcal{A},b}$ and $\mathsf{Game}^2_{\mathcal{A},b}$ this follows analogously to the analysis of D in $\mathsf{Game}^0_{\mathcal{A},b}$ and the POWHF-encryption property.
- For $\mathsf{Game}^2_{\mathcal{A},b}$ and $\mathsf{Game}^3_{\mathcal{A},b}$ indistinguishability for these two games obviously follows from the IND-CPA property of the symmetric encryption scheme.

This proves that the H-instantiation of the Fujisaki-Okamoto transformation is IND-CCA for random oracle G. ▮

F Proof of Theorem 6.1

Recall that our idea is to construct a "bad" VPRF (which exists if any VPRF exists) which reveals signatures for free, but whose outputs are still pseudorandom. To ensure this pseudorandomness, even though the VPRF outputs signatures which usually have some structure, we use the key privacy trick of [2]. There, to make sure that RSA-encryptions for users with different k-bit moduli $N_0, N_1, N_2 \ldots$ do not reveal the actual receiver, the sender searches for ciphertexts belonging to $\bigcap_i \mathbf{Z}_{N_i}$. One possibility to ensure this privacy —which we also take advantage of to achieve pseudorandomness— is to look for RSA values of $k-1$ bits, because all moduli have at least k bits. We first define an intermediate VPRF which is then modified to obtain our "bad" VPRF:

Definition F.1 *Let $(\mathcal{K}^*, \mathcal{H}^*, \mathcal{V}^*)$ be any verifiable pseudorandom function with unbounded input length and output length $\ell(k) = 2k$. Then construct $(\mathcal{K}, \mathcal{H}, \mathcal{V})$ as follows:*

1. *$\mathcal{K}(1^k)$: The key generation algorithm remains unchanged, $\mathcal{K}(1^k) = \mathcal{K}^*(1^k)$.*
2. *$\mathcal{H}(fk, x)$: Parse x as $x = (N, e, m)$ for a k-bit modulus N and a $(k+1)$-bit prime e.[5] If x is not of the form (N, e, m) then compute $(z_0, \pi_0) \xleftarrow{\$} \mathcal{H}^*(fk, \langle 0 \rangle_k \,||x))$, let y be the first $k-1$ bits of z_0 and return $(y, (z_0, \pi_0))$.*

   Else, if $x = (N, e, m)$, then compute $(z_i, \pi_i) \xleftarrow{\$} \mathcal{H}^*(fk, \langle i \rangle_k \,||x)$ and $y_i \leftarrow z_i^e \bmod N$ for all $i = 1, 2, \ldots, k$. Take y to be the first among the values y_i such that $|y_i| = k-1$ or, if no such y_i exists, then set $y \leftarrow 0^{k-1}$. Output y and the vector of preimages and proofs $\pi = (z_i, \pi_i)_{i=1,2,\ldots,k}$ as the proof for y.
3. *$\mathcal{V}(vk, x, y, \pi)$: Parse x as $x = (N, e, m)$. If this fails then check that y equals the first $k-1$ bits in the proof $\pi = (z_0, \pi_0)$ and that $\mathcal{V}^*(vk, \langle 0 \rangle_k \,||x, z_0, \pi_0) = 1$. Accept if both tests succeed.*

   Else, if $x = (N, e, m)$, then $\mathcal{V}'$ parses π as $\pi = (z_i, \pi_i)_{i=1,2,\ldots,k}$. It recomputes all $y_i = z_i^e \bmod N$ and checks that y equals the first y_i with $k-1$ bits (or that $y = 0^{k-1}$ if no such y_i exists). V also verifies each proof π_i for z_i by running $\mathcal{V}^*(vk, \langle i \rangle_k \,||x, z_i, \pi_i)$. Accept if all tests succeed.

[5] We presume an appropriate encoding, e.g., encode strings $s_1 \ldots s_n \in \{0,1\}^n$ by doubling the individual bits to $s_1 s_1 \ldots s_n s_n$ and using 01 as a separation mark between blocks.

Lemma F.2 *The tuple $(\mathcal{K}, \mathcal{H}, \mathcal{V})$ in Construction F.1 is a verifiable pseudorandom function with unbounded input length and output length $\ell(k) = k - 1$.*

Proof: Completeness and unique provability are clear. For the pseudorandomness assume for the moment that $\mathcal{H}^*$ returns truly random values. Note that a random $2k$-bit value reduced modulo the k-bit integer N is statistically close to a uniform value on $\mathbf{Z}_N$. Since e is relatively prime to N the e-th power of this value is almost uniform on $\mathbf{Z}_N$ as well, and therefore we obtain a $(k-1)$-bit string y_i with probability close to $2^{k-2}/2^k \geq 1/4$. If the probability for a *pseudo*random $\mathcal{H}^*$ would diverge from this significantly, say drop by a factor of $1/2$, then it would be easy to distinguish random from pseudorandom values. Hence, with probability at least $1 - (7/8)^k$ our algorithm $\mathcal{H}$ finds a $(k-1)$-bit string in k trials and does not output 0^{k-1}. We can thus simply condition on the event that $\mathcal{H}$ never returns 0^{k-1} because of unsuccessful trials.

We next show that a successful attack on our protocol would contradict the pseudorandomness of the underlying ensemble $(\mathcal{K}^*, \mathcal{H}^*, \mathcal{V}^*)$. Let $\mathcal{A}$ be an adversary attacking our protocol as in Definition 2.2 and outputting $d = b$ with non-negligible probability $\epsilon(k)$ over $1/2$ (without having queried the challenge x). We construct adversary $\mathcal{A}^* = (\mathcal{A}_1^*, \mathcal{A}_2^*)$, running in two phases, for $(\mathcal{K}^*, \mathcal{H}^*, \mathcal{V}^*)$ as follows:

- $\mathcal{A}_1^*$ gets vk generated by $(vk, fk) \leftarrow \mathcal{K}^*(1^k)$ as input.
- $\mathcal{A}_1^*$ starts a simulation of $\mathcal{A}_1$ for input vk. For any oracle query x of $\mathcal{A}_1$ to $\mathcal{H}(fk, \cdot)$, algorithm $\mathcal{A}_1^*$ tries to parse $x = (N, e, m)$ and asks $\mathcal{H}^*(fk, \cdot)$ either about $\langle 0 \rangle_k \,||x$ to get (z_0, π_0) or about values $\langle i \rangle_k \,||x$ to receive answers (z_i, π_i) for $i = 1, 2, \ldots, k$. In the first case, $\mathcal{A}_1^*$ sets y to be the first $k-1$ bits of z_0 and returns $(y, (z_0, \pi_0))$. In the other case, if $x = (N, e, m)$, for each answer (z_i, π_i) of $\mathcal{H}^*(fk, \cdot)$ adversary $\mathcal{A}_1^*$ computes $y_i \leftarrow z_i^e \bmod N$ and lets y be the first among these values such that $|y_i| = k - 1$. Return $(y, (y_i, \pi_i)_{i=1,2,\ldots,k})$ to $\mathcal{A}_1$.
- If $\mathcal{A}_1$ outputs a challenge x and *state* then $\mathcal{A}_1^*$ first determines again whether x is of the form (N, e, m). If not, then $\mathcal{A}^*$ submits $\langle 0 \rangle_k \,||x$ as its own challenge with state *state*. The answer z, either pseudorandom ($b^* = 0$) or random ($b^* = 1$), is truncated by $\mathcal{A}_2^*$ to the first $k-1$ bits and returned to $\mathcal{A}_2$ with *state*.

  If $x = (N, e, m)$, on the other hand, then $\mathcal{A}_1^*$ first chooses $j \leftarrow \{1, 2, \ldots, k\}$ at random. For $i = 1, 2, \ldots, j-1, j+1, \ldots, k$ it submits queries $\langle i \rangle_k \,||x$ to oracle $\mathcal{H}^*(fk, \cdot)$ to receive answers z_i. $\mathcal{A}_1^*$ then outputs $\langle j \rangle_k \,||x$ as the challenge and state information *state*. Given the pseudorandom or random answer z_j, depending on bit b^*, and the remaining $k-1$ values z_i adversary $\mathcal{A}_2^*$ computes the reply y for $\mathcal{A}_2$ by returning the first y_i in the list with $k-1$ bits. If this index is different from j then $\mathcal{A}_2^*$ stops immediately with random output $d^* \xleftarrow{\$} \{0, 1\}$.
- Simulate all further oracle queries as before, and finally output the guess $d^* = d$ of $\mathcal{A}_2$.

The simulation is perfect if the challenge x of $\mathcal{A}$ is *not well-formed*, i.e., not of the form (N, e, m). In particular, since we use a fixed-length encoding for the numbers $\langle 0 \rangle_k, \langle 1 \rangle_k, \ldots, \langle k \rangle_k$ adversary $\mathcal{A}^*$ only submits its own challenge to the oracle $\mathcal{H}^*(fk, \cdot)$ if $\mathcal{A}$ does so with its challenge in its

simulated attack against $\mathcal{H}(fk, \cdot)$. Hence, in the case that the challenge is not of the form (N, e, m) adversary $\mathcal{A}^*$ outputs $d^* = b^*$ with the same probability as $\mathcal{A}$ manages to output $d = b$.

Suppose that the adversary outputs a *well-formed* challenge $x = (N, e, m)$. Given that $\mathcal{H}$ never outputs 0^{k-1} because of unsuccessful trials the adversary $\mathcal{A}^*$ guesses the challenge index j correctly with probability $1/k$. In this case, the simulation perfectly mimics an attack on $(\mathcal{K}, \mathcal{H}, \mathcal{V})$ and $\mathcal{A}^*$ outputs $d^* = b^*$ whenever $\mathcal{A}$ outputs $d = b$. If the choice of $\mathcal{A}'$ is wrong, which happens with probability $1 - 1/k$, then $\mathcal{A}^*$ outputs a correct guess with probability $1/2$. Overall, the probability that $a' = b'$ therefore equals $1/2 + \epsilon(k)/k$ in this case.

In both cases the success probability of $\mathcal{A}^*$ attacking $(\mathcal{K}^*, \mathcal{H}^*, \mathcal{V}^*)$ would be non-negligibly larger than $1/2$. This, however, would contradict the pseudorandomness of $(\mathcal{K}^*, \mathcal{H}^*, \mathcal{V}^*)$. ∎

The idea of the VPRF is that it gives away valid signatures with very high probability. If queried about some (N, e, x) then, except with very small probability with which the answer y equals 0^{k-1}, the proof π_i contains a value $y_i = y$ together with an e-th root z_i of y_i. This root is a valid signature under the key (N, e). We are now ready to prove the insecurity of the RSA-FDH instantiation with respect to our "bad" VPRF in Construction F.1 and complete the proof:

The adversary gets the public key $pk = (N, e)$ of FDH-RSA and the verification key vk of the VPRF as input. It also gets oracle access to $\mathcal{H}(fk, \cdot)$. The adversary submits $x \leftarrow (N, e, m)$ for some message m to the VPRF to receive (y, π) where $\pi = (z_i, \pi_i)_{i=1,2,\ldots,k}$. Except with negligible probability (with which the VPRF returns 0^{k-1}) the adversary gets a value $y = y_i$ and a value z_i in π such that $y_i = z_i^e \bmod N$. The adversary outputs the message x and the signature $\sigma = z_i$. Apparently, these values satisfy $\sigma^e = z_i^e = y_i = y = \mathcal{H}(fk, x) \bmod N$ and fulfill the verification equation of the signature scheme.