On the Lossiness of the Rabin Trapdoor Function*

Yannick Seurin

ANSSI, Paris, France
yannick.seurin@m4x.org

April 15, 2014

Abstract. Lossy trapdoor functions, introduced by Peikert and Waters (STOC '08), are functions that can be generated in two indistinguishable ways: either the function is injective, and there is a trapdoor to invert it, or the function is lossy, meaning that the size of its range is strictly smaller than the size of its domain. Kakvi and Kiltz (EUROCRYPT 2012) proved that the Full Domain Hash signature scheme based on a lossy trapdoor function has a *tight* security reduction from the lossiness of the trapdoor function. Since Kiltz, O'Neill, and Smith (CRYPTO 2010) showed that the RSA trapdoor function is lossy under the Φ-Hiding assumption of Cachin, Micali, and Stadler (EUROCRYPT '99), this implies that the RSA Full Domain Hash signature scheme has a *tight* security reduction from the Φ-Hiding assumption (for public exponents $e < N^{1/4}$). In this work, we consider the Rabin trapdoor function, *i.e.* modular squaring over $\mathbb{Z}_N^*$. We show that when adequately restricting its domain (either to the set $\mathbb{QR}_N$ of quadratic residues, or to $(\mathbb{J}_N)^+$, the set of positive integers $1 \leq x \leq (N-1)/2$ with Jacobi symbol $+1$) the Rabin trapdoor function is lossy, the injective mode corresponding to Blum integers $N = pq$ with $p, q \equiv 3 \bmod 4$, and the lossy mode corresponding to what we call pseudo-Blum integers $N = pq$ with $p, q \equiv 1 \bmod 4$. This lossiness result holds under a natural extension of the Φ-Hiding assumption to the case $e = 2$ that we call the 2-Φ/4-Hiding assumption. We then use this result to prove that deterministic variants of Rabin-Williams Full Domain Hash signatures have a tight reduction from the 2-Φ/4-Hiding assumption. We also show that these schemes are unlikely to have a tight reduction from the factorization problem by extending a previous "meta-reduction" result by Coron (EUROCRYPT 2002), later corrected by Kakvi and Kiltz (EUROCRYPT 2012). These two results therefore answer one of the main questions left open by Bernstein (EUROCRYPT 2008) in his work on Rabin-Williams signatures.

Keywords: Rabin trapdoor function, lossy trapdoor function, Phi-Hiding assumption, provable security, Rabin-Williams signatures, meta-reduction

1 Introduction

1.1 Background

Lossy Trapdoor Functions. Lossy Trapdoor Functions (LTF) were introduced by Peikert and Waters [PW08] and have since then found a wide range of applications in cryptography such as deterministic public-key encryption [BFO08], hedged public-key encryption [BBN+09], and security against selective opening attacks [BHY09, FHKW10] to name a few. Informally, an LTF consists of two families of functions: functions in the first family are injective (and efficiently invertible using some trapdoor), while functions in the second family are non-injective and hence lose information on their input. The key requirement for an LTF is that functions sampled from the first and the second family be computationally indistinguishable. Many constructions of LTF are known from various hardness assumptions such as DDH, LWE, etc. [PW08]. In particular, Kiltz, O'Neill, and Smith showed [KOS10] that the RSA trapdoor function $f : x \mapsto x^e \bmod N$, where $N = pq$ is an RSA modulus, is lossy under the Φ-Hiding assumption, introduced by Cachin, Micali, and Stadler [CMS99]. When e is coprime with $\phi(N)$ ($\phi(\cdot)$ is Euler's totient function), f is injective on the domain $\mathbb{Z}_N^*$, while when e divides $\phi(N)$ (but e^2 does not), f is e-to-1 on $\mathbb{Z}_N^*$. The Φ-Hiding assumption states that given (N, e) where $e < N^{1/4}$, it is hard to tell whether $\gcd(e, \phi(N)) = 1$ or $e|\phi(N)$, which corresponds to respectively the injective and lossy modes of the RSA function.

Full Domain Hash Signatures. Full Domain Hash (FDH) signatures [BR93] are a class of signature schemes which can be based on any trapdoor function f: the signature of a message m is computed as $\sigma = f^{-1}(H(m))$, where H is some hash function (the secret signature key is the trapdoor enabling to invert f). For a long time, the only known security result for FDH signatures, due to Coron [Cor00] (improving on a previous result [BR93]), had been a *non-tight* reduction from the problem of inverting the trapdoor function, losing a factor q_s (the maximal number of signature queries made by the forger). Recently, Kakvi and Kiltz [KK12] showed that the FDH signature scheme, when based on a trapdoor function which is lossy, has a *tight* reduction from the problem of distinguishing the injective from the lossy mode of the LTF. In particular, this applies to RSA-FDH signatures with public exponents $e < N^{1/4}$, which hence have a tight security reduction from the Φ-Hiding problem.[1] Moreover, in the same paper, Kakvi and Kiltz corrected a previous "meta-reduction" result due to Coron [Cor02a] stating that the security reduction of [Cor00] losing a factor q_s is essentially optimal. More precisely, they showed that when the trapdoor function is *certified* (meaning that there is an efficient algorithm distinguishing injective from non-injective members of the function family), any security reduction from inverting the trapdoor function to breaking FDH signatures must lose a factor q_s (unless inverting the trapdoor function is easy). This applies in particular to RSA-FDH signatures with public exponents $e > N^{1/4}$ since RSA is certified for these parameters [KKM12].

1.2 Contributions of this Work

Lossiness of the Rabin Trapdoor Function. We show that the Rabin trapdoor function, *i.e.* modular squaring, is lossy (with exactly one or two bits of lossiness) when adequately

[1] Tight security reductions are important for adequately setting security parameters, see the discussion of this point in [KK12].

restricting its domain. Since any quadratic residue modulo an RSA modulus $N = pq$ has exactly four square roots, it is not immediately obvious how to render this function injective. It is well known that when N is a so-called Blum integer, *i.e.* $p, q \equiv 3 \bmod 4$, any quadratic residue has a unique square root which is also a quadratic residue, named its *principal* square root. Hence, in this case, modular squaring defines a permutation over the set of quadratic residues $\mathbb{QR}_N$. One potential problem with this definition of the injective mode is that the domain of the permutation is (presumably) not efficiently recognizable (this is exactly the Quadratic Residuosity assumption). A different way to restrict the domain of modular squaring is to consider the set $(\mathbb{J}_N)^+$ of integers $1 \leq x \leq (N-1)/2$ with Jacobi symbol $+1$ (which is efficiently recognizable). We show that when restricting its domain to either $\mathbb{QR}_N$ or $(\mathbb{J}_N)^+$ to make it injective, modular squaring becomes an LTF. The lossy mode corresponds to integers $N = pq$ such that $p, q \equiv 1 \bmod 4$, that we call pseudo-Blum integers. It can be shown that in that case, modular squaring becomes 4-to-1 over $\mathbb{QR}_N$ and 2-to-1 over $(\mathbb{J}_N)^+$. Indistinguishability of the injective and lossy modes is then exactly the problem of distinguishing Blum from pseudo-Blum integers, which is equivalent to tell whether 2 divides $\phi(N)/4$ or not. This can be seen as the extension of the traditional Φ-Hiding assumption to exponent $e = 2$, so that we call this problem the 2-Φ/4-Hiding problem. Details can be found in Sections 2 and 3.

Application to Rabin-Williams Signatures. We apply our finding to the security of deterministic Rabin-Williams Full Domain Hash signatures. The Rabin signature scheme [Rab79] is one of the oldest provably secure digital signature scheme. Its security relies on the difficulty of computing modular square roots, which is equivalent to factoring integers. Given an RSA modulus $N = pq$, the general principle of Rabin signatures is to first map the message $m \in \{0,1\}^*$ to a quadratic residue h modulo N using some hash function H, and then return a square root s of h. Since only 1/4 of integers in $\mathbb{Z}_N^*$ are quadratic residues, directly using $h = H(m) \bmod N$ will fail for roughly 3 out of 4 messages. This can be coped with using a randomized padding. The simplest one, Probabilistic Full Domain Hash with ℓ-bit salts (ℓ-PFDH) [Cor02a], computes $h = H(r, m)$ for random ℓ-bit salts r, until h is a quadratic residue (r is then included in the signature for verification). A way to avoid this probabilistic method is to use a tweak, as proposed by Williams [Wil80].[2] For any RSA modulus N, one can find four values $\alpha_1, \alpha_2, \alpha_3, \alpha_4 \in \mathbb{Z}_N^*$ such that for any $h \in \mathbb{Z}_N^*$, there is a unique $i \in [1;4]$ such that $\alpha_i^{-1} h \bmod N$ is a quadratic residue.[3] When $p \equiv 3 \bmod 8$ and $q \equiv 7 \bmod 8$, one can use the set of values $\{1, -1, 2, -2\}$. This way, the signature becomes a so-called tweaked square root (α, s), where s is a square root of $\alpha^{-1} H(m) \bmod N$ for the correct value $\alpha \in \{1, -1, 2, -2\}$, and the verification algorithm now checks whether $\alpha s^2 = H(m) \bmod N$. This enables to define FDH Rabin-Williams signatures.

Since any quadratic residue modulo an RSA modulus N has four square roots, one must also specify which (tweaked) square root of the hash to use as the signature. There are basically two ways to proceed. The first one is simply to pick a square root at random. However, when no randomization (or randomization with only a small number of bits) is used in the input to the hash function, one must be careful not to output two non-trivially distinct square

[2] Williams' paper [Wil80] was primarily concerned with public key encryption. The idea of using a tweak for deterministic signing is implicit in the ISO/IEC 9796 standard published in 1991, and was later made more explicit in a paper by Kurosawa and Ogata [KO99].

[3] The sufficient condition for this is that the pairs of Legendre symbols $\left(\left(\frac{\alpha_i}{p}\right), \left(\frac{\alpha_i}{q}\right)\right)$ take each of the four values $(1,1)$, $(-1,1)$, $(1,-1)$ and $(-1,-1)$ for exactly one α_i.

roots if the same message is signed twice, since this would reveal the factorization of the modulus N. In consequence, the signature algorithm must either be stateful and store all signatures previously output (which is cumbersome), or generate the bits for deciding which root to use pseudo-randomly.[4] However, in constrained environments, implementors might be reluctant to pay the additional cost of a pseudorandom function (moreover, how exactly this derandomization is done is not always precisely discussed, and may have security implications as explained in [LN09]).

The second option is to define some deterministic rule telling which square root to use as the signature. The most popular way to do so is to use for N a Blum integer and to use the principal square root. A variant is to use what we call the absolute principal square root, *i.e.* $|s \bmod N|$, where s is the principal square root represented by an integer in $[-(N-1)/2; (N-1)/2]$. This turns out to also be the unique square root in $(\mathbb{J}_N)^+$. We will call these ways to choose a square root Principal Rabin-Williams (PRW) and Absolute Principal Rabin-Williams (APRW) respectively.[5] When no randomization in the input to the hash function is used, the signature algorithm then becomes entirely deterministic (without having to appeal to an auxiliary pseudorandom function), which is attractive from an implementation point of view.

Bernstein [Ber08] proposed an extensive study of possible variants of Rabin-Williams signature schemes depending on the length of the salt and the square root selection method. In particular, for FDH signatures, he showed a tight security reduction from the factoring assumption for the probabilistic square root selection method (Fixed Unstructured). On the other hand, for PRW and APRW, only a loose reduction from factoring is known using methods of Coron [Cor00, Ber08]. Our main result is a tight security reduction from the 2-Φ/4-Hiding problem for the PRW and APRW schemes, building on the results of [KK12]. Details can be found in Section 4.

Extending the Coron-Kakvi-Kiltz Meta-reduction Result. Recall that Coron's meta-reduction result [Cor02a] as corrected by Kakvi and Kiltz [KK12] states that when the trapdoor function is *certified*, any security reduction from inverting the trapdoor function to breaking FDH signatures must lose a factor q_s. Since this only applies for certified trapdoor functions, this leaves open the question of whether there might exist a tight reduction from inverting the trapdoor function to breaking FDH signatures when the trapdoor function is not certified. In particular, the question whether there exists a tight security reduction from factoring (or equivalently, computing modular square roots) for the PRW and APRW schemes was left as an open problem in [Ber08]. However, we observe that the meta-reduction result still holds (namely, any security reduction from inverting the trapdoor function to breaking FDH signatures must lose a factor q_s) when the underlying trapdoor function is *gap one-way*, meaning that inverting the injective mode of the function is hard even with the help of an oracle distinguishing injective from non-injective modes of the trapdoor function. This implies in particular that if factoring with the help of an oracle solving the 2-Φ/4-Hiding problem is hard, the PRW and APRW signature schemes cannot have a tight security reduction from

[4] This method was called Fixed Unstructured Rabin-Williams in [Ber08], and Probabilistic Rabin-Williams (PRW) in [LN09]. See Table 1 in Appendix A.

[5] PRW was called Fixed Principal in [Ber08] and Deterministic Rabin-Williams (DRW) in [LN09], while APRW was called Fixed |Principal| in [Ber08]. See Table 1 in Appendix A.

the factorization problem. This essentially answers the open question of [Ber08] regarding the security reductions for these schemes. Details can be found in Section 5.

1.3 Related and Future Work

Two constructions of lossy trapdoor functions based on modular squaring were previously proposed, however they are slightly more complicated than the basic Rabin trapdoor function. Mol and Yilek [MY10] gave a construction whose security relies on an assumption close in spirit (though more involved) to the 2-Φ/4-Hiding assumption. Freeman *et al.* [FGK+10] gave a construction relying on the Quadratic Residuosity problem.

The cryptographic applications of the set $(\mathbb{J}_N)^+$ when N is a Blum integer were previously considered by Goldwasser *et al.* [GMR88], Fischlin and Schnorr [FS00], and Hofheinz and Kiltz [HK09] (in this last paper, it was denoted $\mathbb{QR}_N^+$ and named *group of signed quadratic residues*). In particular, it was showed in [HK09] that under the factoring assumption, the Strong Diffie-Hellman problem [ABR01] is hard in this group.

The Coron-Kakvi-Kiltz meta-reduction result [Cor02a, KK12] was extended by Hofheinz *et al.* [HJK12] to the case where the signature scheme is re-randomizable (rather than with unique signatures).

Kiltz *et al.* [KOS10] showed that lossiness of RSA implies that the RSA-OAEP encryption scheme [BR94] meets indistinguishability under chosen-plaintext attacks in the standard model (under appropriate assumptions on the hash functions used to instantiate OAEP). An interesting question is whether lossiness of the Rabin trapdoor function can be used to argue about the security of Rabin-OAEP encryption as was done in [KOS10] for RSA. Though from a theoretical point of view the results of [KOS10] apply to OAEP used with any LTF, they provide some meaningful security insurance only when the amount of lossiness is sufficiently high. This requires more careful investigation in the case of Rabin-OAEP. As a first step in this direction, we note that if "multi-primes" pseudo-Blum integers $N = p_1 \cdots p_m$, with $p_1, \ldots, p_m \equiv 1 \bmod 4$ are indistinguishable from 2-primes pseudo-Blum integers, lossiness of the Rabin trapdoor function with domain $(\mathbb{J}_N)^+$ can be amplified from 1 bit to $m-1$ bits. Similar arguments were used for RSA in [KOS10].

2 Preliminaries

2.1 General Notation

The set of integers i such that $a \leq i \leq b$ will be denoted $[a; b]$. The security parameter will be denoted k. A function f of the security parameter is said *negligible* if for any $c > 0$, $f(k) \leq 1/k^c$ for sufficiently large k. When S is a non-empty finite set, we write $s \leftarrow_\$ S$ to mean that a value is sampled uniformly at random from S and assigned to s. By $z \leftarrow \mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2, \cdots}(x, y, \ldots)$ we denote the operation of running the (possibly probabilistic) algorithm $\mathcal{A}$ on inputs $x, y, \ldots$ with access to oracles $\mathcal{O}_1, \mathcal{O}_2, \ldots$ (possibly none), and letting z be the output.

2.2 Basic Definitions

Given an (odd for most of what follows) integer N, the multiplicative group of integers modulo N is denoted $\mathbb{Z}_N^*$. This group has order $\phi(N)$ where $\phi(\cdot)$ is the Euler function. We denote

$\mathbb{J}_N$ the subgroup of $\mathbb{Z}_N^*$ of all elements $x \in \mathbb{Z}_N^*$ with Jacobi symbol $\left(\frac{x}{N}\right) = 1$. This subgroup has index 2 and order $\phi(N)/2$ in $\mathbb{Z}_N^*$. Moreover it is efficiently recognizable even without the factorization of N since the Jacobi symbol is efficiently computable given only N. We also denote $\overline{\mathbb{J}}_N$ the coset of elements $x \in \mathbb{Z}_N^*$ such that $\left(\frac{x}{N}\right) = -1$. Finally, we denote $\mathbb{QR}_N$ the subgroup of quadratic residues of $\mathbb{Z}_N^*$. This subgroup is widely believed not to be efficiently recognizable when N is composite and its factorization is unknown: this is the Quadratic Residuosity assumption.

We will represent elements of $\mathbb{Z}_N$ as signed integers in $[-(N-1)/2, (N-1)/2]$. Given an integer x, we denote $|x \bmod N|$ the absolute value of $x \bmod N$. For any subset $S \subset \mathbb{Z}_N$, we denote $S^+ = S \cap [1; (N-1)/2]$ and $S^- = S \cap [-(N-1)/2; -1]$. Note that $(\mathbb{J}_N)^+$, $(\mathbb{J}_N)^-$, $(\overline{\mathbb{J}}_N)^+$ and $(\overline{\mathbb{J}}_N)^-$ form a partition of $\mathbb{Z}_N^*$.

We call an integer $N = pq$ which is the product of two distinct odd primes a *Blum integer* when $p, q \equiv 3 \bmod 4$, and a *pseudo-Blum integer* when $p, q \equiv 1 \bmod 4$, and we denote

$$\mathsf{Bl}(k) = \{(N, p, q) : N = pq,\ p, q \text{ are two distinct } \lfloor k/2 \rfloor\text{-bit primes with } p, q \equiv 3 \bmod 4\}$$
$$\widetilde{\mathsf{Bl}}(k) = \{(N, p, q) : N = pq,\ p, q \text{ are two distinct } \lfloor k/2 \rfloor\text{-bit primes with } p, q \equiv 1 \bmod 4\} \ .$$

We call a Blum integer $N = pq$ such that moreover $p \equiv 3 \bmod 8$ and $q \equiv 7 \bmod 8$ a *Williams integer*, and a pseudo-Blum integer such that $p \equiv 5 \bmod 8$ and $q \equiv 1 \bmod 8$ a *pseudo-Williams* integer. We denote

$$\mathsf{Wi}(k) = \{(N, p, q) \in \mathsf{Bl}(k) : p \equiv 3 \bmod 8, q \equiv 7 \bmod 8\}$$
$$\widetilde{\mathsf{Wi}}(k) = \{(N, p, q) \in \widetilde{\mathsf{Bl}}(k) : p \equiv 5 \bmod 8, q \equiv 1 \bmod 8\} \ .$$

Note that:

- when N is a Blum integer, $-1 \in \mathbb{J}_N \setminus \mathbb{QR}_N$;
- when N is a pseudo-Blum integer, $-1 \in \mathbb{QR}_N$;
- when N is a Williams or a pseudo-Williams integer, $2 \in \overline{\mathbb{J}}_N$.

A quadratic residue modulo an RSA modulus $N = pq$ has four square roots, two of which are in $(\mathbb{Z}_N^*)^+$ and two of which are in $(\mathbb{Z}_N^*)^-$. The two square roots in $(\mathbb{Z}_N^*)^+$ will be called the *absolute* square roots in what follows. The following lemma will be important when proving lossiness of the Rabin trapdoor function.

Lemma 1. *Let $N = pq$ be a RSA modulus with $N \equiv 1 \bmod 4$. Let $x \in \mathbb{QR}_N$, and let s_1 and s_2 be the two absolute square roots of x (the two other square roots being $-s_1$ and $-s_2$). Then:*

- *if N is a Blum integer, exactly one s_i is in $(\mathbb{J}_N)^+$ and the other is in $(\overline{\mathbb{J}}_N)^+$; moreover if $s_i \in (\mathbb{J}_N)^+$ then either $s_i \in \mathbb{QR}_N$ or $-s_i \in \mathbb{QR}_N$;*
- *if N is a pseudo-Blum integer, then either $s_1, s_2, -s_1, -s_2 \in \mathbb{QR}_N$, or $s_1, s_2, -s_1, -s_2 \in \mathbb{J}_N \setminus \mathbb{QR}_N$, or $s_1, s_2, -s_1, -s_2 \in \overline{\mathbb{J}}_N$.*

Proof. Consider $x \in \mathbb{QR}_N$. Denote $x_p = x \bmod p$ and $x_q = x \bmod q$. Let also $\pm r_p$ and $\pm r_q$ denote the two square roots of respectively $x_p \pmod p$ and $x_q \pmod q$. The four square roots of x modulo N are obtained by combining $\pm r_p$ and $\pm r_q$ by the Chinese Remainder Theorem, *i.e.* there are to integers c_p and c_q such that the four square roots of x are $\pm(pc_p r_q \pm qc_q r_p) \bmod N$. Assume that one of the two absolute square roots is $s_1 = (pc_p r_q + qc_q r_p) \bmod N$ (the reasoning

is similar if it is $-(pc_pr_q + qc_qr_p) \bmod N$). Then the other absolute square root satisfies $s_2 = \alpha(pc_pr_q - qc_qr_p) \bmod N$, with $\alpha = \pm 1$ so that:

$$\left(\frac{s_2}{p}\right) = \left(\frac{\alpha}{p}\right)\left(\frac{-1}{p}\right)\left(\frac{s_1}{p}\right) \quad \text{and} \quad \left(\frac{s_2}{q}\right) = \left(\frac{\alpha}{q}\right)\left(\frac{s_1}{q}\right) \ .$$

Consequently:

- when N is a Blum integer, s_1 and s_2 have opposite Jacobi symbols; moreover, assuming $s_1 \in (\mathbb{J}_N)^+$ then since -1 is a non-quadratic residue, either $s_1 \in \mathbb{QR}_N$ or $-s_1 \in \mathbb{QR}_N$;
- when N is a pseudo-Blum integer, we see that

  $$\left(\frac{s_1}{p}\right) = \left(\frac{-s_1}{p}\right) = \left(\frac{s_2}{p}\right) = \left(\frac{-s_2}{p}\right) \quad \text{and} \quad \left(\frac{s_1}{q}\right) = \left(\frac{-s_1}{q}\right) = \left(\frac{s_2}{q}\right) = \left(\frac{-s_2}{q}\right) \ ,$$

  from which the claim on the localization of the four square roots follows.

This concludes the proof. □

Hence when N is a Blum integer, the two absolute square roots can easily be distinguished through their Jacobi symbol. In the following, given a Blum integer N and $x \in \mathbb{QR}_N$, we will call the unique square root of x which is in $\mathbb{QR}_N$ the *principal* square root of x, and denote it $\texttt{psr}(x)$. We will also call the unique square root of x which is in $(\mathbb{J}_N)^+$ the *absolute principal* square root of x, and will denote it $|\texttt{psr}|(x)$. The notation is chosen so that $|\texttt{psr}|(x) = |\texttt{psr}(x) \bmod N|$.

Tweaked Square Roots. Let N be a Williams integer. Then for any $x \in \mathbb{Z}_N^*$ there is a unique $\alpha \in \{1, -1, 2, -2\}$ such that $\alpha^{-1}x \bmod N$ is a quadratic residue.[6] The four pairs $(\alpha, s_i)_{i=1,\ldots,4}$ where $(s_i)_{i=1,\ldots,4}$ are the four square roots of $\alpha^{-1}x \bmod N$ are named the *tweaked square roots* of x, and α is named the tweak. Hence, (α, s) with $\alpha \in \{1, -1, 2, -2\}$ is a tweaked square root of $x \in \mathbb{Z}_N^*$ *iff* $\alpha s^2 = x \bmod N$. By extension, the principal tweaked square root of x is the unique tweaked square root (α, s) such that $s \in \mathbb{QR}_N$, and the absolute principal tweaked square root is the unique tweaked square root (α, s) such that $s \in (\mathbb{J}_N)^+$. Overloading the notation, they will be denoted respectively $\texttt{psr}(x)$ and $|\texttt{psr}|(x)$.

2.3 Trapdoor Functions

We recall some formal definitions associated with trapdoor functions (we follow closely the ones of [KK12]). We also introduce the concept of *gap one-way trapdoor function*, which is informally a trapdoor function which is hard to invert even when given access to an oracle which tells whether a member of the family is injective or lossy.

Definition 1 (Trapdoor Function). *A trapdoor function (TDF) is a tuple of polynomial-time algorithms* $\texttt{TDF} = (\texttt{InjGen}, \texttt{Eval}, \texttt{Invert})$ *with the following properties:*

- $\texttt{InjGen}(1^k)$*: a probabilistic algorithm which on input the security parameter* 1^k*, outputs a public description* $\texttt{pub}$ *(with implicitly understood domain* $\mathcal{D}_{\texttt{pub}}$*) and a trapdoor* $\texttt{td}$*;*

[6] This follows easily from the fact that the pairs of Legendre symbols $\left(\left(\frac{\alpha}{p}\right), \left(\frac{\alpha}{q}\right)\right)$ for $\alpha = 1$, -1, 2, and -2 are respectively $(1,1)$, $(-1,-1)$, $(-1,1)$ and $(1,-1)$.

– $\texttt{Eval}(\texttt{pub}, x)$*: a deterministic algorithm which on input* $\texttt{pub}$ *and a point* $x \in \mathcal{D}_{\texttt{pub}}$*, outputs a point* $y \in \{0,1\}^*$*; we denote* $f_{\texttt{pub}} : x \mapsto \texttt{Eval}(\texttt{pub}, x)$*;*
– $\texttt{Invert}(\texttt{td}, y)$*: a deterministic algorithm which on input* $\texttt{td}$ *and a point* $y \in \{0,1\}^*$*, outputs a point* $x \in \mathcal{D}_{\texttt{pub}}$ *when* $y \in f_{\texttt{pub}}(\mathcal{D}_{\texttt{pub}})$ *(and* $\perp$ *otherwise).*

We require that for any k *and any* $(\texttt{pub}, \texttt{td})$ *possibly output by* $\texttt{InjGen}(1^k)$*, the function* $f_{\texttt{pub}} : x \mapsto \texttt{Eval}(\texttt{pub}, x)$ *be injective, and* $y \mapsto \texttt{Invert}(\texttt{td}, y)$ *be its inverse* $f_{\texttt{pub}}^{-1}$*. We also require that* $\mathcal{D}_{\texttt{pub}}$ *and* $f_{\texttt{pub}}(\mathcal{D}_{\texttt{pub}})$ *be efficiently samplable.*

Definition 2 (One-Way TDF). *A trapdoor function* $\texttt{TDF} = (\texttt{InjGen}, \texttt{Eval}, \texttt{Invert})$ *is said to be* (t, ε)*-one-way if for any adversary* $\mathcal{A}$ *running in time at most* t*, one has:*

$$\Pr\left[\texttt{pub} \leftarrow \texttt{InjGen}(1^k), x \leftarrow_{\$} \mathcal{D}_{\texttt{pub}}, x' \leftarrow \mathcal{A}(\texttt{pub}, \texttt{Eval}(\texttt{pub}, x)) : x' = x\right] \leq \varepsilon \ .$$

Definition 3 (Certified TDF). *A trapdoor function* $\texttt{TDF} = (\texttt{InjGen}, \texttt{Eval}, \texttt{Invert})$ *is said to be* certified *if there exists a deterministic polynomial-time algorithm* $\texttt{Certify}$ *which, on input an arbitrary string* $\texttt{pub}$ *(not necessarily generated by* $\texttt{InjGen}$*) returns 1* iff *the function* $x \mapsto \texttt{Eval}(\texttt{pub}, x)$ *is injective over* $\mathcal{D}_{\texttt{pub}}$*.*

Definition 4 (Lossy TDF). *A lossy trapdoor function (LTF) with absolute lossiness* ℓ *is a tuple of algorithms* $\texttt{LTF} = (\texttt{InjGen}, \texttt{LossyGen}, \texttt{Eval}, \texttt{Invert})$ *such that* $(\texttt{InjGen}, \texttt{Eval}, \texttt{Invert})$ *is a TDF as per Definition 1, and moreover* $\texttt{LossyGen}$ *is a probabilistic algorithm which on input* 1^k*, outputs a public description* $\texttt{pub}'$ *such that the range of the function* $f_{\texttt{pub}'} : x \mapsto \texttt{Eval}(\texttt{pub}', x)$ *over* $\mathcal{D}_{\texttt{pub}'}$ *satisfies:*

$$\frac{|\mathcal{D}_{\texttt{pub}'}|}{|f_{\texttt{pub}'}(\mathcal{D}_{\texttt{pub}'})|} \geq \ell \ .$$

We say that $\texttt{LTF}$ *is* (t, ε)*-secure if for any adversary* $\mathcal{A}$ *running in time at most* t*, the following advantage is less than* ε*:*

$$\left|\Pr[(\texttt{pub}, \texttt{td}) \leftarrow \texttt{InjGen}(1^k) : 1 \leftarrow \mathcal{A}(\texttt{pub})] - \Pr[\texttt{pub}' \leftarrow \texttt{LossyGen}(1^k) : 1 \leftarrow \mathcal{A}(\texttt{pub}')]\right| \ .$$

We say that $\texttt{LTF}$ *is a regular* (ℓ, t, ε)*-lossy trapdoor function if* $\texttt{LTF}$ *is* (t, ε)*-secure and all functions generated by* $\texttt{LossyGen}$ *are* ℓ*-to-1 on* $\mathcal{D}_{\texttt{pub}'}$*.*

Remark 1. One can easily show that if $\texttt{TDF}$ is a regular (ℓ, t, ε)-lossy TDF, then it is (t', ε')-one way with $t' \simeq t$ and $\varepsilon' \leq \varepsilon + 1/\ell$. Note in particular that asymptotically, if $\ell = \mathcal{O}(1)$ is constant (as is the case for the trapdoor functions considered in this paper), this only implies that $\texttt{TDF}$ is weakly one-way [Gol01].

Definition 5 (Gap One-Way TDF). *A trapdoor function* $\texttt{TDF} = (\texttt{InjGen}, \texttt{Eval}, \texttt{Invert})$ *is said* (t, ε, n)*-gap one-way if for any adversary* $\mathcal{A}$ *running in time at most* t *and making at most* n *queries to a* $\texttt{Certify}(\cdot)$ *oracle which on input a string* $\texttt{pub}$*, returns 1* iff *the function* $x \mapsto \texttt{Eval}(\texttt{pub}, x)$ *is injective over* $\mathcal{D}_{\texttt{pub}}$*, one has:*

$$\Pr\left[\texttt{pub} \leftarrow \texttt{InjGen}(1^k), x \leftarrow_{\$} \mathcal{D}_{\texttt{pub}}, x' \leftarrow \mathcal{A}^{\texttt{Certify}(\cdot)}(\texttt{pub}, \texttt{Eval}(\texttt{pub}, x)) : x' = x\right] \leq \varepsilon \ .$$

Informally, for a lossy TDF, being gap one-way means that inverting the injective mode of the function cannot be black-box reduced to the lossiness of the TDF. Note that for a certified TDF, being gap one-way is equivalent to being one-way since the $\texttt{Certify}$ oracle can be efficiently implemented.

2.4 Signature Schemes

We recall the formal definition and the security notion for a signature scheme.

Definition 6. *A signature scheme Σ is a tuple of algorithms ($\Sigma.\mathtt{KeyGen}$, $\Sigma.\mathtt{Sig}$, $\Sigma.\mathtt{Ver}$) with the following properties:*

- $\Sigma.\mathtt{KeyGen}(1^k)$*: a probabilistic algorithm which on input the security parameter 1^k, outputs a pair of public/secret key $(\mathtt{pk}, \mathtt{sk})$;*
- $\Sigma.\mathtt{Sig}(\mathtt{sk}, m)$*: a (possibly probabilistic) algorithm which on input a secret key $\mathtt{sk}$ and a message $m \in \{0,1\}^*$, outputs a signature σ;*
- $\Sigma.\mathtt{Ver}(\mathtt{pk}, m, \sigma)$*: a deterministic algorithm which on input a public key $\mathtt{pk}$, a message m, and a purported signature σ, either outputs 1 (accepts) or 0 (rejects).*

We require that the scheme be correct, i.e. *for all k and all messages m,*

$$\Pr[(\mathtt{pk}, \mathtt{sk}) \leftarrow \mathtt{KeyGen}(1^k), \sigma \leftarrow \mathtt{Sig}(\mathtt{sk}, m) : \mathtt{Ver}(\mathtt{pk}, m, \sigma) = 1] = 1 \ .$$

A signature scheme is said to have unique signatures *if for all k, for any public key $\mathtt{pk}$ possibly output by $\mathtt{KeyGen}(1^k)$, and any message $m \in \{0,1\}^*$, there is exactly one string σ such that $\mathtt{Ver}(\mathtt{pk}, m, \sigma)$ accepts.*

The usual security definition for a signature scheme is existential unforgeability under chosen-message attacks (EUF-CMA security). We recall this definition in Appendix B.

FDH Signatures Based on an Arbitrary TDF. Let $\mathtt{TDF} = (\mathtt{InjGen}, \mathtt{Eval}, \mathtt{Invert})$ be a trapdoor function. The Full Domain Hash signature scheme TDF-FDH is defined as follows: the key generation algorithm $\mathtt{KeyGen}(1^k)$ runs $\mathtt{InjGen}(1^k)$ to obtain $(\mathtt{pub}, \mathtt{td})$, selects a random hash function $\boldsymbol{H} : \{0,1\}^* \to f_{\mathtt{pub}}(\mathcal{D}_{\mathtt{pub}})$, and sets $\mathtt{pk} = (\mathtt{pub}, \boldsymbol{H})$ and $\mathtt{sk} = \mathtt{td}$. The signature algorithm, on input $\mathtt{td}$ and m, computes $h = \boldsymbol{H}(m)$ and returns $\sigma = \mathtt{Invert}(\mathtt{td}, h)$. The verification algorithm, on input $\mathtt{pub}$, m and σ, checks that $\mathtt{Eval}(\mathtt{pub}, \sigma) = \boldsymbol{H}(m)$. This scheme can be shown EUF-CMA secure in the Random Oracle Model under the assumption that $\mathtt{TDF}$ is (strongly) one-way [BR93, Cor00], but the reduction loses a factor q_s, where q_s is the maximal number of signature queries of the adversary, and this loss cannot be avoided assuming that $\mathtt{TDF}$ is certified [Cor02a, KK12].

3 The 2-Φ/4-Hiding Assumption and Lossiness of the Rabin Trapdoor Function

3.1 Definition

We introduce the 2-Φ/4-Hiding assumption, an extension of the traditional Φ-Hiding assumption to the case $e = 2$. The Φ-Hiding assumption, introduced by Cachin *et al.* in [CMS99], roughly states that given an RSA modulus $N = pq$ and a random prime $3 \leq e < N^{1/4}$, it is hard to distinguish whether e divides $\phi(N)$ or not (when $e \geq N^{1/4}$ and $e|\phi(N)$, N can be factored using Coppersmith's method for finding small roots of univariate modular equations [Cop96, CMS99]). Kiltz *et al.* [KOS10] were the first to observe that the Φ-Hiding assumption can be interpreted in terms of lossiness of the RSA trapdoor permutation.

The original definition of the Φ-Hiding assumption was formulated for primes e randomly drawn in $[3; N^{1/4}[$. Since in practice RSA is often used with a fixed, small prime e (*e.g.* $e = 3$ or $e = 2^{16} + 1$), Kakvi and Kiltz [KK12] introduced the Fixed-Prime Φ-Hiding assumption, which states, for a fixed prime e, that it is hard, given an RSA modulus $N = pq$, to distinguish whether e divides $\phi(N)$ or not (the exact statement of the assumption is slightly different for $e = 3$ and $e > 3$ in order to avoid trivial distinguishers). The 2-Φ/4-Hiding assumption is the extension of the Fixed-Prime Φ-Hiding assumption to the case $e = 2$. Since for an RSA modulus N (more generally for any number which has at least two distinct prime factors) one always has that 4 divides $\phi(N)$, the problem will be to distinguish whether 2 divides $\phi(N)/4$ or not. Moreover, when $N \equiv 3 \bmod 4$, one can check that 2 always divides $\phi(N)/4$, so that the instances will be restricted to RSA moduli such that $N \equiv 1 \bmod 4$. As a matter of fact, distinguishing whether 2 divides $\phi(N)/4$ or not when $N \equiv 1 \bmod 4$ turns out to be equivalent to distinguishing Blum integers from pseudo-Blum integers. Indeed, if N is a Blum integer, then $p = 4p' + 3$ and $q = 4q' + 3$, so that $\phi(N) = 4(2p' + 1)(2q' + 1)$ and $2 \nmid (\phi(N)/4)$. On the other hand, if N is a pseudo-Blum integer, then $p = 4p' + 1$ and $q = 4q' + 1$, so that $\phi(N) = 16p'q'$ and $2|(\phi(N)/4)$. We now precisely formalize the assumption.

Definition 7 (2-Φ/4-Hiding Assumption). *The 2-Φ/4-Hiding problem is said to be (t, ε)-hard if for any algorithm $\mathcal{A}$ running in time at most t, the following advantage is less than ε:*

$$\mathrm{Adv}^{2-\Phi/4}(\mathcal{A}) \stackrel{\text{def}}{=} \left| \Pr[(N, p, q) \leftarrow_{\$} \mathsf{BI}(k) : 1 \leftarrow \mathcal{A}(N)] - \Pr[(N, p, q) \leftarrow \widetilde{\mathsf{BI}}(k) : 1 \leftarrow \mathcal{A}(N)] \right| .$$

A variant of this problem is obtained by switching from Blum integers to Williams integers, *i.e.* replacing $\mathsf{BI}(k)$ and $\widetilde{\mathsf{BI}}(k)$ in the above definition by respectively $\mathsf{Wi}(k)$ and $\widetilde{\mathsf{Wi}}(k)$. Clearly, the hardness of this variant is polynomially related to the hardness of the original problem, under the plausible assumption that roughly half of Blum, resp. pseudo-Blum integers are Williams, resp. pseudo-Williams integers.

Remark 2. We note that, given an RSA modulus $N = pq$ such that $N \equiv 1 \bmod 4$, -1 is a quadratic residue *iff* N is a pseudo-Blum integer, while -1 is a pseudo-quadratic residue (i.e., $-1 \in \mathbb{J}_N \backslash \mathbb{QR}_N$) *iff* N is a Blum integer. Hence, the 2-Φ/4-Hiding problem can be equivalently phrased as the problem of telling, given $N \equiv 1 \bmod 4$, whether -1 is a quadratic residue or not in $\mathbb{Z}_N^*$. In particular, since the Quadratic Residuosity problem is random self-reducible, the 2-Φ/4-Hiding problem is at most as hard as the Quadratic Residuosity problem (restricted to RSA moduli such that $N \equiv 1 \bmod 4$): an instance N of the 2-Φ/4-Hiding problem can be turned into an instance (N, y) of the Quadratic Residuosity problem by simply drawing a random $x \leftarrow_{\$} \mathbb{Z}_N^*$ and letting $y = -x^2 \bmod N$.

3.2 Lossiness of the Rabin and Rabin-Williams Trapdoor Functions

We now show that the 2-Φ/4-Hiding assumption implies that squaring is a lossy trapdoor function over the domains $\mathbb{QR}_N$ or $(\mathbb{J}_N)^+$, for $N \equiv 1 \bmod 4$, with respectively two bits or one bit of lossiness. The injective mode corresponds to N being a Blum integer, and the lossy mode corresponds to N being a pseudo-Blum integer.

The Rabin LTFs. We first define two related LTFs, that we name respectively the Principal Rabin LTF `PR-LTF` and the Absolute Principal Rabin LTF `APR-LTF` as follows:

- on input 1^k, `PR-LTF.InjGen` and `APR-LTF.InjGen` both draw $(N, p, q) \leftarrow_\$ \mathsf{Bl}(k)$, and output $\mathtt{pub} = N$ and $\mathtt{td} = (p, q)$;
- on input 1^k, `PR-LTF.LossyGen` and `APR-LTF.LossyGen` both draw $(N, p, q) \leftarrow_\$ \widetilde{\mathsf{Bl}}(k)$, and output $\mathtt{pub}' = N$;
- the domain is $\mathcal{D}_N = \mathbb{QR}_N$ for `PR-LTF`, and $\mathcal{D}_N = (\mathbb{J}_N)^+$ for `APR-LTF`; the evaluation algorithms $\mathtt{PR\text{-}LTF.Eval}(N, x)$ and $\mathtt{APR\text{-}LTF.Eval}(N, x)$ both output $f_N(x) = x^2 \bmod N$; in both cases $f_N(\mathcal{D}_N) = \mathbb{QR}_N$ in injective mode;
- the inversion algorithm $\mathtt{PR\text{-}LTF.Invert}((p, q), y)$ outputs the principal square root $\mathtt{psr}(y)$, while $\mathtt{APR\text{-}LTF.Invert}((p, q), y)$ outputs the absolute principal square root $|\mathtt{psr}|(y)$ (for N a Blum integer and $y \in \mathbb{QR}_N$).

Theorem 1. *Assuming the 2-Φ/4-Hiding problem is (t, ε)-hard, the Principal Rabin trapdoor function* `PR-LTF` *is a regular $(4, t, \varepsilon)$-LTF, while the Absolute Principal Rabin trapdoor function* `APR-LTF` *is a regular $(2, t, \varepsilon)$-LTF.*

Proof. Indistinguishability of the injective and lossy modes is exactly the 2-Φ/4-Hiding problem. It follows from Lemma 1 that when N is a Blum integer, any $y \in \mathbb{QR}_N$ has exactly one pre-image in $\mathbb{QR}_N$ or $(\mathbb{J}_N)^+$, while when N is pseudo-Blum integer, any y in the range $f_N(\mathbb{QR}_N)$ has exactly 4 pre-images in $\mathbb{QR}_N$, and any y in the range $f_N((\mathbb{J}_N)^+)$ has exactly 2 pre-images in $(\mathbb{J}_N)^+$. □

Remark 3. Note that one can define a group structure on $(\mathbb{J}_N)^+$ as soon as N is a Blum or pseudo-Blum integer as follows. Since $-1 \in \mathbb{J}_N$, one can consider the quotient group $\mathbb{J}_N/\{-1, 1\}$. This quotient group can be identified with the set $(\mathbb{J}_N)^+$ equipped with the group operation $\circ$ defined as $a \circ b = |ab \bmod N|$. Note that the order of this group is $\phi(N)/4$. It is then easy to check that squaring, seen as a mapping from $(\mathbb{J}_N)^+$ to $\mathbb{QR}_N$, is a group homomorphism. When N is a Blum integer, its image is $\mathbb{QR}_N$, whereas when N is a pseudo-Blum integer, its image is a strict subgroup of $\mathbb{QR}_N$ of index 2. Similarly, when N is a pseudo-Blum integer, the image of $\mathbb{QR}_N$ is a strict subgroup of $\mathbb{QR}_N$ of index 4.

The Rabin-Williams LTFs. The `PR-LTF` and `APR-LTF` LTFs can be straightforwardly extended to what we call the Principal Rabin-Williams LTF `PRW-LTF` and Absolute Principal Rabin-Williams LTF `APRW-LTF` as follows:

- on input 1^k, `PRW-LTF.InjGen` and `APRW-LTF.InjGen` both draw a random Williams integer $(N, p, q) \leftarrow_\$ \mathsf{Wi}(k)$, and output $\mathtt{pub} = N$ and $\mathtt{td} = (p, q)$;
- on input 1^k, `PRW-LTF.LossyGen` and `APRW-LTF.LossyGen` both draw a random pseudo-Williams integer $(N, p, q) \leftarrow_\$ \widetilde{\mathsf{Wi}}(k)$ and output $\mathtt{pub}' = N$;
- the domain of `PRW-LTF` is $\mathcal{D}_N = \{1, -1, 2, -2\} \times \mathbb{QR}_N$, while the domain of `APRW-LTF` is $\mathcal{D}_N = \{1, -1, 2, -2\} \times (\mathbb{J}_N)^+$; the evaluation algorithms $\mathtt{PRW\text{-}LTF.Eval}(N, (\alpha, x))$ and $\mathtt{APRW\text{-}LTF.Eval}(N, (\alpha, x))$ both compute the function $f_N(\alpha, x) = \alpha x^2 \bmod N$; in both cases $f_N(\mathcal{D}_N) = \mathbb{Z}_N^*$ in injective mode;
- the inversion algorithm $\mathtt{PRW\text{-}LTF.Invert}((p, q), y)$ computes the principal tweaked square root $\mathtt{psr}(y)$, while $\mathtt{APRW\text{-}LTF.Invert}((p, q), y)$ computes the absolute principal tweaked square root $|\mathtt{psr}|(y)$ (for N a Williams integer and $y \in \mathbb{Z}_N^*$).

Theorem 2. *Under the assumption that Williams and pseudo-Williams integers are (t, ε)-indistinguishable, the Principal Rabin-Williams trapdoor function* `PRW-LTF` *is a regular $(4, t, \varepsilon)$-LTF, while the Absolute Principal Rabin-Williams trapdoor function* `APRW-LTF` *is a regular $(2, t, \varepsilon)$-LTF.*

Proof. Indistinguishability of the injective and lossy modes is exactly indistinguishability of Williams and pseudo-Williams integers, which follows from the 2-Φ/4-Hiding assumption and the additional (reasonable) assumption that roughly half of Blum, resp. pseudo-Blum integers, are Williams, resp. pseudo-Williams integers. Injectivity of f_N for both `PRW-LTF` and `APRW-LTF` follows directly from Lemma 1 and the discussion about tweaked square roots in Section 2. Assume now that N is a pseudo-Williams integer, and let $y \in f_N(\mathcal{D}_N)$ with $\mathcal{D}_N = \{1, -1, 2, -2\} \times \mathbb{QR}_N$. We show that y has exactly 4 pre-images in $\mathcal{D}_N$, which will establish that `PRW-LTF` is 4-to-1 on $\mathcal{D}_N$. Let $(\alpha, x) \in \mathcal{D}_N$ be such that $\alpha x^2 = y \bmod N$. Then by Lemma 1, y has at least 4 pre-images in $\mathcal{D}_N$, all with the same tweak α. Assume that y has an extra pre-image $(\alpha', x') \in \mathcal{D}_N$ with $\alpha' \neq \alpha$. Note that when $N = pq$ is a pseudo-Williams integer (*i.e.* $p \equiv 5 \bmod 8$ and $q \equiv 1 \bmod 8$), the pairs of Legendre symbols $(\left(\frac{\alpha}{p}\right), \left(\frac{\alpha}{q}\right))$ for $\alpha = 1$, -1, 2, and -2 are respectively $(1,1)$, $(1,1)$, $(-1,1)$ and $(-1,1)$. Hence it must be that $\alpha' = -\alpha$, so that $x^2 = -(x')^2 \bmod N$. Let a be any square root of -1 modulo N. Since $a^2 = -1 \bmod N$, we observe (denoting $p = 8p' + 5$ and $q = 8q' + 1$) that:

$$\left(\frac{a}{p}\right) \equiv a^{\frac{p-1}{2}} \equiv a^{\frac{8p'+4}{2}} \equiv (-1)^{2p'+1} \equiv -1 \bmod p$$
$$\left(\frac{a}{q}\right) \equiv a^{\frac{q-1}{2}} \equiv a^{\frac{8q'}{2}} \equiv (-1)^{2q'} \equiv 1 \bmod q \ ,$$

so that $a \in \overline{\mathbb{J}}_N$. Hence, we have that $x^2 = (ax')^2 \bmod N$, with $x, x' \in \mathbb{QR}_N$. Yet by Lemma 1, one should have $ax' \in \mathbb{QR}_N$ as well, which is impossible since $a \in \overline{\mathbb{J}}_N$. Hence y has exactly 4 pre-images in $\mathcal{D}_N$.

The proof that `APRW-LTF` is 2-to-1 on $\mathcal{D}_N = \{1, -1, 2, -2\} \times (\mathbb{J}_N)^+$ is very similar. Assume that N is a pseudo-Williams integer, and let $y \in f_N(\mathcal{D}_N)$ with $\mathcal{D}_N = \{1, -1, 2, -2\} \times (\mathbb{J}_N)^+$. Let $(\alpha, x) \in \mathcal{D}_N$ be such that $\alpha x^2 = y \bmod N$. Then by Lemma 1, y has at least 2 pre-images in $\mathcal{D}_N$, both with the same tweak α. Assume that y has an extra pre-image $(\alpha', x') \in \mathcal{D}_N$. Since by Lemma 1, the two additional square roots of $\alpha^{-1}y \bmod N$ are in $(\mathbb{J}_N)^-$, one must have $\alpha' \neq \alpha$. By the same considerations as above, there exists $a \in \overline{\mathbb{J}}_N$ such that $x^2 = (ax')^2 \bmod N$, with $x, x' \in (\mathbb{J}_N)^+$. Yet by Lemma 1, one should have $ax' \in \mathbb{J}_N$ as well, which is impossible since $a \in \overline{\mathbb{J}}_N$. Hence y has exactly 2 pre-images in $\mathcal{D}_N$. □

4 Application to Rabin-Williams Signatures

There are two very close ways to define deterministic Rabin-Williams FDH signatures, called principal and |principal| in the terminology of Bernstein [Ber08]. We will use the name *Absolute Principal* Rabin-Williams signatures for the latter in this paper. Before defining precisely these schemes, we stress that the exact definition of the verification algorithm is important, especially with respect to how a forgery is defined (since a forgery is exactly a string which is accepted by the verification algorithm). Hence, to be more precise, we will define in total four "real" signature schemes: Principal Rabin-Williams (PRW), Absolute Principal Rabin-Williams (APRW), as well as two slightly different variants that we call PRW* and APRW*,

which differ from respectively PRW and APRW only in their verification algorithm. We will also define a "theoretical" scheme PRW** where the verification algorithm is inefficient (this will be necessary to establish a clean security reduction). For the five schemes, the signing algorithm first hashes the message $h = \boldsymbol{H}(m)$; then, for the PRW, PRW*, and PRW** schemes, the signing algorithm returns the principal tweaked square root of h, whereas for the APRW and APRW* schemes, the signing algorithm returns the absolute principal tweaked square root of h. In all the following, we assume that if h is not coprime with N, the signing algorithm outputs some fixed signature, *e.g.* $(1, 1)$. Since this happens only with negligible probability when $\boldsymbol{H}$ is modeled as a random oracle, this does not affect the security analysis.

We now proceed to the formal definition. First, all the schemes share exactly the same key generation algorithm:

- (A)PRW(*,**).$\texttt{KeyGen}(1^k)$: on input the security parameter 1^k, draw uniformly at random $(N, p, q) \leftarrow_{\$} \mathsf{Wi}(k)$. Select a hash function $\boldsymbol{H} : \{0,1\}^* \to \mathbb{Z}_N$. The public key is $\texttt{pk} = (N, \boldsymbol{H})$ and the secret key is $\texttt{sk} = (p, q)$.

Note that the hash function will usually be selected once for each security parameter k and common to all public keys, but this affects the security proof only up to negligible terms, see Bernstein [Ber08].

The signing algorithm for PRW, PRW*, and PRW** on one hand, and for APRW and APRW* on the other hand, are the same, and are defined as follows:

- PRW(*,**).$\texttt{Sig}(\texttt{sk}, m)$: To sign a message m, compute $h = \boldsymbol{H}(m)$, and output the principal tweaked square root $\sigma = (\alpha, s) = \texttt{psr}(h)$.
- APRW(*).$\texttt{Sig}(\texttt{sk}, m)$: To sign a message m, compute $h = \boldsymbol{H}(m)$, and output the absolute principal tweaked square root $\sigma = (\alpha, s) = |\texttt{psr}|(h)$.

The verification algorithms for the five schemes are very close, and differ only with respect to an additional check on the Jacobi symbol of the signature made for PRW* and APRW*, and on the quadratic residuosity of the signature for PRW**. They are defined as follows:

- (A)PRW(*,**).$\texttt{Ver}(\texttt{pk}, m, \sigma)$: To check a purported signature $\sigma = (\alpha, s)$ on message m, first ensure that $s \in S$, and then check that $\alpha s^2 = \boldsymbol{H}(m) \bmod N$. Accept if this holds, and reject otherwise;

where the set S is defined as:

- $S = \mathbb{Z}_N^*$ for PRW, $S = \mathbb{J}_N$ for PRW*, and $S = \mathbb{QR}_N$ for PRW**;
- $S = (\mathbb{Z}_N^*)^+$ for APRW and $S = (\mathbb{J}_N)^+$ for APRW*.

Note that the verification algorithm is (presumably) inefficient for PRW** since it needs to decide whether the signature is indeed the principal square root, *i.e.* a quadratic residue.

The following claims are straightforward:

- in PRW, each message has exactly four valid signatures: $(\alpha, s_1) = |\texttt{psr}|(\boldsymbol{H}(m))$, $(\alpha, -s_1)$, and (α, s_2), $(\alpha, -s_2)$ with $s_2 \in (\overline{\mathbb{J}}_N)^+$;
- in PRW*, each message has exactly two valid signatures: $(\alpha, s) = |\texttt{psr}|(\boldsymbol{H}(m))$ and $(\alpha, -s)$;
- in PRW**, each message has a unique valid signature: $(\alpha, s) = \texttt{psr}(\boldsymbol{H}(m))$;
- in APRW, each message has exactly two valid signatures: $|\texttt{psr}|(\boldsymbol{H}(m))$ and (α, s_2) with $s_2 \in (\overline{\mathbb{J}}_N)^+$;

– in APRW*, each message has a unique valid signature: $|\mathtt{psr}|(\boldsymbol{H}(m))$.

We now relate the security of PRW, PRW*, and PRW** on one hand, and APRW and APRW* on the other hand.

Lemma 2. *The security of PRW, PRW* and PRW** on one hand, and APRW and APRW* on the other hand, is related as depicted in Figure 1, where an arrow labeled $(t, f(\varepsilon))$ from scheme A to scheme B means that if scheme A is $(t, \varepsilon, q_h, q_s)$-EUF-CMA secure in the ROM, then scheme B is $(t', f(\varepsilon), q_h, q_s)$-EUF-CMA secure for $t' \simeq t$.*

Proof. We prove each of the reductions in turn.

– PRW** $\xrightarrow{(t,2\varepsilon)}$ PRW*: Assume there is an adversary $\mathcal{A}$ which $(t, \varepsilon, q_h, q_s)$-breaks the PRW* scheme. We build from it an adversary $\mathcal{A}'$ breaking the PRW** scheme. $\mathcal{A}'$ receives as input a public key N and runs $\mathcal{A}$ with the same public key. Denote $\boldsymbol{H}'$ the random oracle to which $\mathcal{A}'$ has access. $\mathcal{A}'$ simulates the PRW* security experiment to $\mathcal{A}$ by simply relaying its random oracle queries and signing queries to its own oracles. When $\mathcal{A}$ outputs a forgery $(\hat{\alpha}, \hat{s})$ for some message $\hat{m}$ where $\hat{s} \in \mathbb{J}_N$, $\mathcal{A}'$ simply draws a random bit b, and outputs $(\hat{\alpha}, (-1)^b \hat{s})$. The security experiment is perfectly simulated to $\mathcal{A}$ (since a PRW** signature oracle and a PRW* signature oracle are the same), and, assuming that the forgery output by $\mathcal{A}$ is valid (which happens with probability at least ε), the forgery output by $\mathcal{A}'$ is valid when $(-1)^b \hat{s}$ is a quadratic residue, which happens with probability 1/2. Hence $\mathcal{A}'$ $(t, \varepsilon/2, q_h, q_s)$-breaks PRW**.

– (A)PRW* $\xrightarrow{(t,2\varepsilon)}$ (A)PRW: We consider the PRW* → PRW reduction, the reasoning for the APRW* → APRW reduction is similar. Assume there is an adversary $\mathcal{A}$ which $(t, \varepsilon, q_h, q_s)$-breaks the PRW scheme. We build from it an adversary $\mathcal{A}'$ breaking the PRW* scheme. $\mathcal{A}'$ receives as input a public key N and runs $\mathcal{A}$ with the same public key. Denote $\boldsymbol{H}'$ the random oracle to which $\mathcal{A}'$ has access. We assume *wlog* that $\mathcal{A}$ always makes a random oracle query for m before asking the corresponding signature or returning a forgery for m (otherwise we let $\mathcal{A}'$ emulate this random oracle query). We now describe how $\mathcal{A}'$ simulates the random oracle $\boldsymbol{H}$ and the PRW signing oracle to $\mathcal{A}$. Each time $\mathcal{A}$ makes a query $\boldsymbol{H}(m)$, $\mathcal{A}'$ draws a random bit b_m. If $b_m = 0$, then $\mathcal{A}'$ makes the query $\boldsymbol{H}'(m)$ to its own random oracle and returns $\boldsymbol{H}(m) = \boldsymbol{H}'(m)$. If $b_m = 1$, then $\mathcal{A}'$ draws a random tweak $\alpha \leftarrow_\$ \{1, -1, 2, -2\}$ and a random $s \leftarrow_\$ \mathbb{QR}_N$ (by sampling $z \leftarrow_\$ \mathbb{Z}_N^*$ and letting $s = z^2 \bmod N$), and returns $\boldsymbol{H}(m) = \alpha s^2 \bmod N$ to $\mathcal{A}'$. If $\mathcal{A}$ makes a subsequent PRW signing query for m, then if b_m was 0, $\mathcal{A}'$ makes the same signing query to its own PRW* signing oracle and returns the corresponding signature. If b_m was 1, then $\mathcal{A}'$ simply outputs (α, s) as the signature, where α and s were randomly drawn to simulate $\boldsymbol{H}(m)$. Clearly, the simulation of the PRW security experiment is close to perfect (up to the fact that answers to random oracle queries $\boldsymbol{H}(m)$ are uniform in $\mathbb{Z}_N^*$ rather than $\mathbb{Z}_N$ when $b_m = 1$). Hence $\mathcal{A}$ outputs a forgery $(\hat{\alpha}, \hat{s})$ for some message $\hat{m}$ with probability at least ε. Note that this is a valid forgery for PRW, so that $\hat{s}$ may be in $\mathbb{J}_N$ or $\overline{\mathbb{J}}_N$. Since the view of $\mathcal{A}$ is independent of the bit $b_{\hat{m}}$, we can assume that this bit is randomly drawn after the forgery is returned. Two cases arise. In case where $\hat{s} \in \mathbb{J}_N$, then if $b_{\hat{m}} = 0$, $(\hat{\alpha}, \hat{s})$ is also a valid forgery for PRW* and $\mathcal{A}'$ can simply output the same forgery. Otherwise, in case where $\hat{s} \in \overline{\mathbb{J}}_N$, then if $b_{\hat{m}} = 1$, denoting s' the value randomly drawn in $\mathbb{QR}_N$ by $\mathcal{A}'$ to simulate the random oracle query $\boldsymbol{H}(\hat{m})$, we see that $\hat{s}$ and s' are two non-trivially distinct square

roots of the same quadratic residue, so that $\mathcal{A}'$ can factor N and forge a signature for a message of its choice. In both cases $\mathcal{A}'$ is successful with probability $1/2$, so that the overall success probability of $\mathcal{A}'$ is $\varepsilon/2$. Hence $\mathcal{A}'$ $(t, \varepsilon/2, q_h, q_s)$-breaks PRW*.

This proves the lemma. □

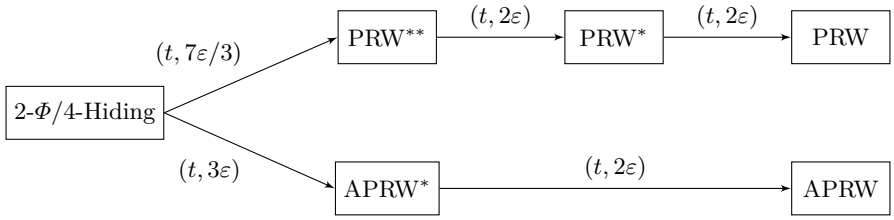

Fig. 1. Set of reductions proved in Lemma 2. An arrow labeled $(t, f(\varepsilon))$ from scheme A to scheme B means that if scheme A is $(t, \varepsilon, q_h, q_s)$-EUF-CMA secure in the ROM, then scheme B is $(t', f(\varepsilon), q_h, q_s)$-EUF-CMA secure for $t' \simeq t$. The reduction from 2-Φ/4-Hiding to breaking PRW** and APRW* is Theorem 4.

Hence, one can see that PRW and PRW* on one hand, and APRW and APRW* on the other hand, have the same security up to a factor 2. In other words, omitting the additional check on the Jacobi symbol has negligible impact on security. Since computing a Jacobi symbol might be costly (in particular, it is more expensive than modular squaring), we see that PRW and APRW are superior in terms of security/efficiency trade-off.

In the following, we give a tight reduction for PRW** and APRW* from the 2-Φ/4-Hiding assumption, which extends to PRW and APRW by Lemma 2. It is easy to see that the PRW**, resp. APRW* signature scheme is exactly the instantiation of the generic TDF-FDH scheme recalled in Section 2.4 with `PRW-LTF`, resp. `APRW-LTF`. In order to conclude about the security of these schemes, we appeal to the main result of [KK12]. This theorem was originally stated for trapdoor *permutations*, but it can be straightforwardly extended to trapdoor functions such that $\mathcal{D}_{\mathtt{pub}}$ and $f_{\mathtt{pub}}(\mathcal{D}_{\mathtt{pub}})$ are efficiently samplable.

Theorem 3 ([KK12]). *Assume* `LTF` *is a regular* (ℓ, t', ε')*-LTF for* $\ell \geq 2$*. Then for any* (q_h, q_s)*, the TDF-FDH signature scheme instantiated with* `LTF` *is* $(t, \varepsilon, q_h, q_s)$*-EUF-CMA secure in the ROM, where*

$$\varepsilon = \left(\frac{2\ell - 1}{\ell - 1}\right)\varepsilon' \quad \textit{and} \quad t = t' - q_h T_{\mathtt{Eval}} \ ,$$

where $T_{\mathtt{Eval}}$ *is the time to run algorithm* `Eval` *of* `LTF`*.*

Theorem 4. *Assuming the 2-Φ/4-Hiding problem is (t', ε')-hard, then for any (q_h, q_s), the PRW** signature scheme is $(t, \varepsilon, q_h, q_s)$-EUF-CMA secure, where $\varepsilon = 7\varepsilon'/3$ and $t = t' - \mathcal{O}(q_h k^3)$, and the APRW* signature scheme is $(t, \varepsilon, q_h, q_s)$-EUF-CMA secure, where $\varepsilon = 3\varepsilon'$ and $t = t' - \mathcal{O}(q_h k^3)$.*

Proof. This follows directly from Theorems 2 and 3 (noting that $\mathbb{QR}_N$ and $(\mathbb{J}_N)^+$ are efficiently samplable). Combined with Lemma 2, this yields tight security reductions for PRW and APRW (see Figure 1 for a clear picture). □

Remark 4. The global security reduction from the 2-Φ/4-Hiding assumption to breaking the signature scheme is slightly looser for PRW (factor 28/3) than for APRW (factor $6 = 18/3$). We also remark that a PRW signature oracle is (potentially) slightly more powerful than an APRW signature oracle because it reveals some non-trivial information regarding the quadratic residuosity of the square roots of the hash of the message (whereas this information, which is unnecessary for verifying signatures, is "canceled" in an APRW signature oracle). Since APRW signatures are not more costly than PRW signatures (and even slightly more communication efficient), these two observations make a case in favor of APRW signatures.

Remark 5. In Appendix C, we define a close variant of the APRW* scheme, where the tweak is re-computed by the verifier rather than transmitted with the signature. Since this involves a Jacobi Symbol computation on the verification side, this is not very attractive from an implementation point of view, but this allows a slightly different formalization based on a lossy trapdoor *permutation* over $(\mathbb{J}_N)^+$, the use of tweaks being seen as a way to deterministically hash into $(\mathbb{J}_N)^+$ (this formalization was already put forward in [Ber08]).

As explained in [KK12], these results can be extended to PSS-R [BR96], allowing a smaller overhead of the randomized signature under the 2-Φ/4-Hiding assumption. It seems also likely (though we have not checked the details) that the same techniques can be used to prove a tight security reduction from the 2-Φ/4-Hiding assumption for Rabin-Williams Partial Domain Hash signatures [Cor02b, Gen04].

5 Extending the Coron-Kakvi-Kiltz Meta-reduction Result

In this section, we complete the picture of the security of FDH signatures by extending Coron's meta-reduction result [Cor00] as corrected by Kakvi and Kiltz [KK12]. In a nutshell, this result says that if a trapdoor function `TDF` is certified, then any reduction from inverting the trapdoor function to breaking the EUF-CMA security of the TDF-FDH signature scheme must lose a factor q_s (the maximal number of signature queries made by the forger) in its time-to-success ratio. The theorem below extends this to trapdoor functions which are not necessarily certified, assuming that `TDF` is gap one-way. The proof is straightforwardly adapted from the one of [Cor00, KK12]: when simulating the forger, the meta-reduction checks as a preliminary step that the public key received from the reduction contains a parameter `pub` which defines an injective function. When `TDF` is certified, this can be done efficiently by the meta-reduction itself. In the variant below, the meta-reduction uses a `Certify` oracle for this step, hence breaking the gap one-wayness (rather than classical one-wayness) of the trapdoor function.

Theorem 5. *Let* `TDF` *be a trapdoor function. Let* $t_R, \varepsilon_R, n, \varepsilon_F, q_h, q_s$ *be functions of the security parameter with* $q_h > q_s$*. Assume there exists a reduction* $\mathcal{R}$ *which* $(t_R, \varepsilon_R, n, \varepsilon_F, q_h, q_s)$*-reduces breaking the one-wayness of* `TDF` *to breaking EUF-CMA security of the TDF-FDH signature scheme. Then there exists a meta-reduction* $\mathcal{M}$ *which* (t_M, ε_M, n)*-breaks the gap one-wayness of* `TDF`*, where:*

$$t_M \leq (n+1)t_R$$
$$\varepsilon_M \geq \varepsilon_R - \varepsilon_F \cdot \frac{n \cdot \exp(-1)}{q_s} \left(1 - \frac{q_s}{q_h}\right)^{-1} .$$

Proof. A precise definition of a (black-box) reduction and a sketch of the proof are provided in Appendix D. □

Remark 6. Theorem 7 of [KK12] directly follows from this result, since when `TDF` is certified, the oracle `Certify` can be efficiently implemented, and `TDF` is gap one-way *iff* it is one-way.

Remark 7. Theorem 5 above can be straightforwardly extended to any non-interactive computational problem which is hard relative to a `Certify` oracle (instead of the one-wayness of the underlying trapdoor function).

Consequences for RSA and Rabin-Williams FDH Signatures. We know by Theorem 8 of [KK12] that RSA-FDH with public exponents $e < N^{1/4}$ has a tight security reduction from the Φ-Hiding assumption. By Theorem 7 of [KK12] we also know that RSA-FDH with public exponents $e > N^{1/4}$ cannot have a tight security reduction from the problem of inverting RSA —nor any non-interactive hard problem— since RSA is certified for this class of exponents [KKM12]. Theorem 5 above implies that it is unlikely as well that RSA-FDH with $e < N^{1/4}$ can have a tight security reduction from inverting RSA: unless inverting RSA with the help of an oracle solving the Φ-Hiding problem is easy, any reduction from inverting RSA to breaking the EUF-CMA security of RSA-FDH with $e < N^{1/4}$ must lose a factor q_s.

This extends to Rabin-Williams FDH signatures as follows: unless computing modular square roots (or equivalently factoring) with the help of an oracle solving the 2-Φ/4-Hiding problem is easy, any reduction from factoring to breaking the EUF-CMA security of the PRW and APRW schemes must lose a factor q_s.

Acknowledgments

The author is thankful to the anonymous reviewers of EUROCRYPT 2013 and PKC 2014 for useful comments.

References

[ABR01] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In David Naccache, editor, *Topics in Cryptology - CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 143–158. Springer, 2001.

[BBN+09] Mihir Bellare, Zvika Brakerski, Moni Naor, Thomas Ristenpart, Gil Segev, Hovav Shacham, and Scott Yilek. Hedged Public-Key Encryption: How to Protect against Bad Randomness. In Mitsuru Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 232–249. Springer, 2009.

[Ber08] Daniel J. Bernstein. Proving Tight Security for Rabin-Williams Signatures. In Nigel P. Smart, editor, *Advances in Cryptology - EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 70–87. Springer, 2008.

[BFO08] Alexandra Boldyreva, Serge Fehr, and Adam O'Neill. On Notions of Security for Deterministic Encryption, and Efficient Constructions without Random Oracles. In David Wagner, editor, *Advances in Cryptology - CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 335–359. Springer, 2008.

[BHY09] Mihir Bellare, Dennis Hofheinz, and Scott Yilek. Possibility and Impossibility Results for Encryption and Commitment Secure under Selective Opening. In Antoine Joux, editor, *Advances in Cryptology - EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 1–35. Springer, 2009.

[BR93] Mihir Bellare and Phillip Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[BR94] Mihir Bellare and Phillip Rogaway. Optimal Asymmetric Encryption. In Alfredo De Santis, editor, *Advances in Cryptology - EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. Springer, 1994.

[BR96] Mihir Bellare and Phillip Rogaway. The Exact Security of Digital Signatures - How to Sign with RSA and Rabin. In Ueli M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. Springer, 1996.

[CMS99] Christian Cachin, Silvio Micali, and Markus Stadler. Computationally Private Information Retrieval with Polylogarithmic Communication. In Jacques Stern, editor, *Advances in Cryptology - EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 402–414. Springer, 1999.

[Cop96] Don Coppersmith. Finding a Small Root of a Univariate Modular Equation. In Ueli M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 155–165. Springer, 1996.

[Cor00] Jean-Sébastien Coron. On the Exact Security of Full Domain Hash. In Mihir Bellare, editor, *Advances in Cryptology - CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 229–235. Springer, 2000.

[Cor02a] Jean-Sébastien Coron. Optimal Security Proofs for PSS and Other Signature Schemes. In Lars R. Knudsen, editor, *Advances in Cryptology - EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 272–287. Springer, 2002.

[Cor02b] Jean-Sébastien Coron. Security Proof for Partial-Domain Hash Signature Schemes. In Moti Yung, editor, *Advances in Cryptology - CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 613–626. Springer, 2002.

[FGK+10] David Mandell Freeman, Oded Goldreich, Eike Kiltz, Alon Rosen, and Gil Segev. More Constructions of Lossy and Correlation-Secure Trapdoor Functions. In Phong Q. Nguyen and David Pointcheval, editors, *Public Key Cryptography - PKC 2010*, volume 6056 of *Lecture Notes in Computer Science*, pages 279–295. Springer, 2010.

[FHKW10] Serge Fehr, Dennis Hofheinz, Eike Kiltz, and Hoeteck Wee. Encryption Schemes Secure against Chosen-Ciphertext Selective Opening Attacks. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 381–402. Springer, 2010.

[FS00] Roger Fischlin and Claus-Peter Schnorr. Stronger Security Proofs for RSA and Rabin Bits. *Journal of Cryptology*, 13(2):221–244, 2000.

[Gen04] Craig Gentry. How to Compress Rabin Ciphertexts and Signatures (and More). In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 179–200. Springer, 2004.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[Gol01] Oded Goldreich. *Foundations of Cryptography - Volume 1, Basic Tools*. Cambridge University Press, 2001.

[HJK12] Dennis Hofheinz, Tibor Jager, and Edward Knapp. Waters Signatures with Optimal Security Reduction. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography - PKC 2012*, volume 7293 of *Lecture Notes in Computer Science*, pages 66–83. Springer, 2012.

[HK09] Dennis Hofheinz and Eike Kiltz. The Group of Signed Quadratic Residues and Applications. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 637–653. Springer, 2009.

[KK12] Saqib A. Kakvi and Eike Kiltz. Optimal Security Proofs for Full Domain Hash, Revisited. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 537–553. Springer, 2012.

[KKM12] Saqib A. Kakvi, Eike Kiltz, and Alexander May. Certifying RSA. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology - ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 404–414. Springer, 2012.

[KO99] Kaoru Kurosawa and Wakaha Ogata. Efficient Rabin-type Digital Signature Scheme. *Des. Codes Cryptography*, 16(1):53–64, 1999.

[KOS10] Eike Kiltz, Adam O'Neill, and Adam Smith. Instantiability of RSA-OAEP under Chosen-Plaintext Attack. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 295–313. Springer, 2010.

[LN09] Gaëtan Leurent and Phong Q. Nguyen. How Risky Is the Random-Oracle Model? In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 445–464. Springer, 2009.

[MY10] Petros Mol and Scott Yilek. Chosen-Ciphertext Security from Slightly Lossy Trapdoor Functions. In Phong Q. Nguyen and David Pointcheval, editors, *Public Key Cryptography - PKC 2010*, volume 6056 of *Lecture Notes in Computer Science*, pages 296–311. Springer, 2010.

[PW08] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Cynthia Dwork, editor, *Symposium on Theory of Computing - STOC 2008*, pages 187–196. ACM, 2008.

[Rab79] Michael O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report 212, MIT Laboratory for Computer Science, 1979.

[Wil80] Hugh C. Williams. A modification of the RSA public-key encryption procedure. *IEEE Transactions on Information Theory*, 26(6):726–729, 1980.

A Rabin-Williams Signature Schemes Terminology

Table 1. Terminology used in [Ber08], [LN09], and this paper depending on the square root selection method used to produce the signature.

square root s selection method	[Ber08]	[LN09]	this paper
random and stateful/pseudorandom	Fixed Unstructured	Probabilistic (PRW)	-
$s \in \mathbb{QR}_N$	Fixed Principal	Deterministic (DRW)	Principal (PRW)
$s \in (\mathbb{J}_N)^+$	Fixed |Principal|	-	Absolute Principal (APRW)

B Security Definition for a Signature Scheme

The usual security definition for a signature scheme is existential unforgeability under chosen-message attacks, which in the Random Oracle Model is defined as follows.

A signature scheme Σ is said to be $(t, \varepsilon, q_h, q_s)$-secure against existential forgery under chosen message attacks (EUF-CMA secure) if for any adversary $\mathcal{A}$ running in time at most t, making at most q_h random oracle queries and q_s signature queries, the advantage $\mathrm{Adv}^{\mathrm{euf-cma}}_{\Sigma}(\mathcal{A}) = \Pr[1 \leftarrow \mathbf{Exp}^{\mathrm{euf-cma}}_{\Sigma,\mathcal{A}}(k)]$ is less then ε, where the experiment is defined as:

Experiment $\mathbf{Exp}^{\mathrm{euf-cma}}_{\Sigma,\mathcal{A}}(k)$:
$(\mathtt{pk}, \mathtt{sk}) \leftarrow \mathtt{KeyGen}(1^k)$
$(\hat{m}, \hat{\sigma}) \leftarrow \mathcal{A}^{\boldsymbol{H}, \mathtt{Sig}(\mathtt{sk}, \cdot)}(\mathtt{pk})$
$b \leftarrow \mathtt{Ver}(\mathtt{pk}, \hat{m}, \hat{\sigma})$
If $b = 1$ and $\hat{m}$ was not queried to oracle $\mathtt{Sig}(\mathtt{sk}, \cdot)$
  return 1
Else return 0

C A Variant of the APRW* Scheme

We describe a close variant of the APRW* scheme in terms of lossy trapdoor *permutation* over $(\mathbb{J}_m)^+$. For this, we define the mapping:

$$\begin{aligned} f :(\mathbb{J}_N)^+ &\mapsto (\mathbb{J}_N)^+ \\ x &\mapsto |x^2 \bmod N| \end{aligned}$$

This is clearly a permutation over $(\mathbb{J}_N)^+$ when N is a Blum integer (this was already noted in [FS00, Section 6]). The inverse mapping maps $y \in (\mathbb{J}_N)^+$ to the absolute principal square root of $\pm y$ depending on whether $y \in \mathbb{QR}_N$ or not. When N is a pseudo-Blum integer, the mapping is 2-to-1.

In order to define a FDH scheme based on this trapdoor permutation, we need to (deterministically) hash into $(\mathbb{J}_N)^+$. This can be done by "tweaking" a hash function with range $\mathbb{Z}_N$ as follows. Given a hash function $\boldsymbol{H} : \{0,1\}^* \to \mathbb{Z}_N$, one can construct a hash function $\boldsymbol{H}'$ defined as:

$$\boldsymbol{H}'(m) = \begin{cases} |\boldsymbol{H}(m) \bmod N| & \text{if } \boldsymbol{H}(m) \in \mathbb{J}_N \\ |\left(\frac{N+1}{2}\right) \boldsymbol{H}(m) \bmod N| & \text{if } \boldsymbol{H}(m) \in \overline{\mathbb{J}}_N \\ 1 \text{ if } \boldsymbol{H}(m) \notin \mathbb{Z}_N^* \end{cases}$$

Outputs of $\boldsymbol{H}'$ can easily be seen to be in $(\mathbb{J}_N)^+$.

Then, the FDH signature scheme based on the trapdoor permutation described above works as follows: to sign m, one first computes $h = \boldsymbol{H}'(m)$, and the signature is $s = f^{-1}(h) \in (\mathbb{J}_N)^+$. The verification algorithm checks whether $\boldsymbol{H}'(m) = |s^2 \bmod N|$. Note that the verifier needs to compute the Jacobi symbol of $\boldsymbol{H}(m)$ with respect to N, which significantly slows verification.

D Proof of Theorem 5

We first give a precise definition of a (black-box) reduction.

Definition 8. *Let* $\mathtt{TDF}$ *be a trapdoor function. An algorithm* $\mathcal{R}$ *is said to* $(t_R, \varepsilon_R, n, \varepsilon_F, q_h, q_s)$*-reduce breaking the one-wayness of* $\mathtt{TDF}$ *to breaking EUF-CMA security of the TDF-FDH signature scheme if on input* $(\mathtt{pub}, \mathtt{Eval}(\mathtt{pub}, x))$ *where* $\mathtt{pub} \leftarrow \mathtt{InjGen}(1^k)$ *and* $x \leftarrow_{\$} \mathcal{D}_{\mathtt{pub}}$*, and after running sequentially at most* n *times any forger which* $(t_F, \varepsilon_F, q_h, q_s)$*-breaks EUF-CMA security of the TDF-FDH signature scheme,* $\mathcal{R}$ *outputs* x *with probability at least* ε_R*, within an additional running time* t_R *(the forger is considered as a black-box oracle and its running time is not taken into account). Moreover, when the forger is probabilistic, the reduction has some limited control over the random tape of the forger, namely it might either run the forger with fresh random tapes, or replay the forger with a previous random tape.*[7] *The probability* ε_R *is taken over the random draw of* $\mathtt{pub}$ *and* x*, the random tape of* $\mathcal{R}$ *and the random tape(s) of the forger.*

Proof (of Theorem 5). As usual with meta-reduction results, we first describe an (inefficient) forger $\mathcal{F}^*$ which $(t_F, \varepsilon_F, q_h, q_s)$-breaks the EUF-CMA security of the TDF-FDH signature scheme, and then show how the meta-reduction can efficiently simulate this forger to any reduction. This forger $\mathcal{F}^*$ proceeds as follows:

[7] For technical reasons, the reduction is not allowed to, say, modify a few bits of a previous random tape.

1. On input $\mathtt{pk} = \mathtt{pub}$, $\mathcal{F}^*$ first checks whether $x \mapsto \mathtt{Eval}(\mathtt{pub}, x)$ is injective over $\mathcal{D}_{\mathtt{pub}}$ (note that this is always possible in finite time, although maybe not efficiently), and returns $\perp$ if this is not the case.
2. $\mathcal{F}^*$ makes random oracle queries for q_h arbitrary distinct messages $m_1, \ldots, m_{q_h}$ (say, m_i is the binary encoding of i), receiving respective answers $y_1, \ldots, y_{q_h}$.
3. $\mathcal{F}^*$ draws $\beta \leftarrow_{\$} [1; q_h]$, a tuple $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{q_s}) \leftarrow_{\$} ([1; q_h] \setminus \{\beta\})^{q_s}$, and a biased coin τ with probability ε_F of yielding 1. More precisely, β, $\boldsymbol{\alpha}$, and τ are computed as the outputs of random functions taking $(\mathtt{pub}, y_1, \ldots, y_{q_h})$ as input. Note that the random tape of this forger, which consists of all the random coins necessary to specify these random functions, has exponential size (yet the meta-reduction will only need a polynomially bounded amount of randomness to simulate $\mathcal{F}^*$).
4. $\mathcal{F}^*$ queries the signing oracle on the messages $m_{\alpha_1}, \ldots, m_{\alpha_{q_s}}$, and checks that all signatures are correct (aborting if this is not the case).
5. If $\tau = 0$ then $\mathcal{F}^*$ returns $\perp$, otherwise it returns a forgery $\hat{\sigma}$ for $\hat{m} = m_\beta$ (again, this is always possible in finite time by exhaustively searching for $\sigma \in \mathcal{D}_{\mathtt{pub}}$ such that $\mathtt{Eval}(\mathtt{pub}, \sigma) = \boldsymbol{H}(m_\beta)$).

Note that an execution of the forger is completely specified by a random tape, an input $\mathtt{pub}$, and a sequence of random oracle answers $(y_1, \ldots, y_{q_h})$. Indeed, since the FDH scheme has unique signatures, the interaction of the forger with the signing oracle is completely deterministic after the last random oracle answer has been received (for a fixed random tape).

Clearly, this forger $(t_F, \varepsilon_F, q_h, q_s)$-breaks the EUF-CMA security of the TDF-FDH scheme for some irrelevant time bound t_F. We now describe how the meta-reduction $\mathcal{M}$ can (efficiently) simulate the forger $\mathcal{F}^*$ to the reduction $\mathcal{R}$. $\mathcal{M}$, on input a pair $(\mathtt{pub}, y)$, runs $\mathcal{R}$ with the same input $(\mathtt{pub}, y)$ and a uniform random tape. We assume *wlog* that any two interactions of the simulated forger with the reduction are distinct, which means that they must differ in either the random tape (*i.e.* the random tapes are uniform and independent in the two executions), or the input $\mathtt{pub}'$ to the forger, or the sequence $(y_1, \ldots, y_{q_h})$ of answers provided by $\mathcal{R}$ to the random oracle queries of the forger. With this in mind, we describe the simulation. When $\mathcal{R}$ invokes the forger for the i-th time on input $\mathtt{pub}'_i$, $\mathcal{M}$ tries to simulate $\mathcal{F}^*$ as follows:

1. $\mathcal{M}$ first (perfectly) simulates step 1 thanks to its $\mathtt{Certify}$ oracle.
2. $\mathcal{M}$ then makes the random oracle queries as would $\mathcal{F}^*$, receiving answers $(y_{i,1}, \ldots, y_{i,q_h})$.
3. $\mathcal{M}$ draws $\beta_i \leftarrow_{\$} [1; q_h]$, a tuple $\boldsymbol{\alpha_i} = (\alpha_{i,1}, \ldots, \alpha_{i,q_s}) \leftarrow_{\$} ([1; q_h] \setminus \{\beta_i\})^{q_s}$, and a biased coin τ_i with probability ε_F of yielding 1.

At this point, $\mathcal{M}$ first tries to obtain a valid forgery for m_{β_i} as follows. It draws a random integer $\ell_i \leftarrow_{\$} [1; q_s]$, defines $\boldsymbol{\alpha'_i} = (\alpha_{i,1}, \ldots, \alpha_{i,\ell_i - 1}, \beta_i)$, and makes the signature queries for messages $m_{\alpha_{i,1}}, \ldots, m_{\alpha_{i,\ell_i - 1}}, m_{\beta_i}$. Let $\hat{\sigma}_i$ denote the signature received for m_{β_i}. $\mathcal{M}$ then rewinds the reduction to the point just after the random oracle queries and resumes the simulation of $\mathcal{F}^*$ as specified in steps 4 and 5. If the following three conditions are satisfied:

i) $\tau_i = 1$,
ii) all signature queries for messages $m_{\alpha_{i,1}}, \ldots, m_{\alpha_{i,q_s}}$ are correctly answered by the reduction,
iii) the signature $\hat{\sigma}_i$ is invalid for message m_{β_i},

then $\mathcal{M}$ aborts since it cannot simulate $\mathcal{F}^*$ correctly. Otherwise, $\mathcal{M}$ is always able to simulate $\mathcal{F}^*$. In particular, if conditions *i*) and *ii*) are satisfied and moreover $\hat{\sigma}_i$ is the correct signature

for m_{β_i}, then $\mathcal{M}$ returns this signature as forgery. This is valid since the signature oracle is never queried on m_{β_i} in the "main" execution of the forger. We stress that, by the above observation that all executions of the forger must be distinct, the values β_i, $\boldsymbol{\alpha_i}$, and τ_i drawn at step 3 are independent for each simulation of the forger. It remains to upper bound the probability that $\mathcal{M}$ fails to simulate the forger correctly in any of the executions, *i.e.* that conditions *i*), *ii*), *iii*) are satisfied simultaneously. It can be shown (cf. [Cor00, KK12]) that this probability is less than:

$$\varepsilon_F \cdot \frac{n \cdot \exp(-1)}{q_s} \left(1 - \frac{q_s}{q_h}\right)^{-1} \quad ,$$

from which the lower bound on ε_M follows.

The time bound follows from the fact that $\mathcal{M}$ rewinds $\mathcal{R}$ at most n times. □

We conclude with a word of interpretation about Theorem 5. Assuming that TDF is asymptotically gap one-way, any polynomial time algorithm breaks the gap one-wayness of TDF with only negligible probability. Assume there exists a polynomial time reduction which $(t_R, \varepsilon_R, n, \varepsilon_F, q_h, q_s)$-reduces breaking the one-wayness of TDF to breaking the EUF-CMA security of the TDF-FDH signature scheme, with $t_R, n, q_h, q_s \in \mathtt{poly}(k)$. Then by the assumption that TDF is gap one-way one must have that the success probability ε_M of the meta-reduction must be negligible, hence by Theorem 5:

$$\begin{aligned}
\varepsilon_R &\leq \mathtt{negl}(k) + \varepsilon_F \cdot \frac{n \cdot \exp(-1)}{q_s} \left(1 - \frac{q_s}{q_h}\right)^{-1} \\
&\leq \mathtt{negl}(k) + \varepsilon_F \cdot \frac{n}{q_s} \\
&\leq (1+\delta)\varepsilon_F \cdot \frac{n}{q_s} \quad ,
\end{aligned}$$

where for the second inequality we made the additional assumption that $q_h \geq 2q_s$, and the third inequality holds for any constant $\delta > 0$ and k sufficiently large (assuming ε_F non-negligible and $n, q_s \in \mathtt{poly}(k)$).

Since the total running time of the reduction is $t_R + nt_F \geq nt_F$, we see that the time-to-success ratio ρ_R of the reduction satisfies

$$\rho_R \geq \frac{nt_F}{(1+\delta)\varepsilon_F n/q_s} = \frac{q_s}{1+\delta} \cdot \rho_F \quad ,$$

where ρ_F is the time-to-success ratio of the forger. In other words, the reduction loses a factor q_s in its time-to-succes ratio compared with the forger.