Waters Signatures with Optimal Security Reduction

Dennis Hofheinz and Tibor Jager

Institut für Kryptographie und Sicherheit, Karlsruhe Institute of Technology, Germany
{dennis.hofheinz,tibor.jager}@kit.edu

Abstract

Waters signatures (Eurocrypt 2005) can be shown existentially unforgeable under chosen-message attacks under the assumption that the computational Diffie-Hellman problem in the underlying (pairing-friendly) group is hard. The corresponding security proof has a reduction loss of $O(\ell \cdot q)$, where ℓ is the bitlength of messages, and q is the number of adversarial signature queries. The original reduction could meanwhile be improved to $O(\sqrt{\ell} \cdot q)$ (Hofheinz and Kiltz, Crypto 2008); however, it is currently unknown whether a better reduction exists. We answer this question as follows:

(a) We give a simple modification of Waters signatures, where messages are encoded such that each two encoded messages have a suitably large Hamming distance. Somewhat surprisingly, this simple modification suffices to prove security under the CDH assumption with a reduction loss of $O(q)$.

(b) We also show that any black-box security proof for a signature scheme with *re-randomizable* signatures must have a reduction loss of at least $\Omega(q)$, or the underlying hardness assumption is false. Since both Waters signatures and our variant from (a) are re-randomizable, this proves our reduction from (a) optimal up to a constant factor.

Understanding and optimizing the security loss of a cryptosystem is important to derive concrete parameters, such as the size of the underlying group. We provide a complete picture for Waters-like signatures: there is an inherent lower bound for the security loss, and we show how to achieve it.

Keywords: Digital signatures, Waters signatures, provable security, black-box reductions.

1 Introduction

Waters signatures. Waters signatures [21] form a simple and efficient digital signature scheme in pairing-friendly groups. The existential unforgeability of the scheme can be proved under the computational Diffie-Hellman (CDH) assumption. Unfortunately, the corresponding security reduction from [21] suffers from a multiplicative loss of $O(\ell \cdot q)$, where ℓ is the bitlength of signed messages, and q is the number of adversarial signing queries. In other words, every signature forger with success probability ε can only be mapped to a CDH-solver with success probability $\Omega(\varepsilon/(\ell \cdot q))$.

From the proof of [21], it is not immediately clear whether this comparatively large security gap is inherent or an artifact of the used proof technique. In fact, [12, 13] used a rather different simulation setup to show the security of Waters signatures with a reduction loss of $O(\sqrt{\ell} \cdot q)$. However, it is not at all clear whether their reduction is optimal. There is no known lower bound on the reduction loss of Waters signatures.

Our contributions. Our contributions revolve around the possibility of achieving a better security reduction for Waters (and similar) signatures. Concretely:

(a) We first give a simple modification of Waters signatures. Essentially, we simply encode each message before signing. This guarantees that any two (encoded) messages have a suitably large Hamming distance. Perhaps somewhat surprisingly, this trivial modification can be shown secure under the CDH assumption with a reduction loss of $O(q)$. The price to pay for this improved reduction is a constant-factor blowup (caused by the encoding) of the public key size and signature/verification times.

(b) Building on work of Coron [6], we proceed to show that any security proof for a signature scheme with re-randomizable signatures must have a reduction loss of at least $\Omega(q)$, or the underlying complexity assumption is false. Coron showed that statement for deterministic signature schemes. We extend the statement to schemes in which any signature can be publicly re-randomized. Since both Waters signatures and our variant from (a) are re-randomizable, this proves our reduction from (a) optimal up to a constant factor.

Of course, the *practical* impact of our results is somewhat limited. In fact, it is a bit disappointing that one can only save a reduction factor of $\sqrt{\ell}$ (compared to the proof of [12, 13]), where ℓ itself is typically significantly smaller than the remaining reduction loss of $O(q)$. However, we stress that from a *conceptual* point of view, our results essentially give a complete picture: there is an inherent lower bound for the security loss of Waters-like signature schemes, *and* we show how to achieve this bound.

Other related work. There exist a number of tightly secure signature schemes, both with (e.g., [17, 1]) and without random oracles (e.g., [8, 4, 3, 15, 19]). However, to the best of our knowledge, there is no standard-model signature scheme whose security could be *tightly* reduced to the CDH problem. In particular, the only known results about the reduction tightness of Waters (or similar) signatures are the discussed works [21, 12, 13]. We do mention that Guo et al. [10] give a variant of Waters signatures and claim that this variant suffers from a reduction loss of only $O(\ell)$. However, their security proof is subtly flawed [11], as we sketch briefly in Section 1.1. It is not clear if and how their argument can be fixed.

Remark. Subsequent to publication of this work on the Cryptology ePrint Archive, it was pointed out to us that (b) was proven independently in [18, Chapter 7].

1.1 Technical overview

Partitioning. In order to present our techniques, we briefly recall the "partitioning" proof strategy used in the context of signature schemes, e.g., by Coron [5] and Waters [21]. A "partitioning" proof simulation partitions the message space into two sets: those messages that can be signed during the simulation, and those that cannot. Let us call those messages "signable," resp. "unsignable." Any forged signature for an unsignable message can then be used to solve a computational problem (e.g., a CDH challenge). The simulation thus succeeds if (a) all adversarial signature queries correspond to signable messages, and (b) the forger finally forges a signature for an unsignable message. For simplicity, assume that each message is set up as signable with a certain probability p. Assume further that these probabilities are independent for different messages. Then, it is not hard to see that the probability that the simulation succeeds is $P := p^q \cdot (1-p)$, where q is the number of signature queries. This probability is maximized if we set p suitably in the order of $1 - 1/q$, in which case $P = O(1/q)$.

Coron's results. Specifically, using a partitioning technique, the best we can hope for is a reduction with a loss of $O(q)$. In fact, Coron [5] shows how to achieve such a reduction for the RSA-FDH scheme in the (programmable) random oracle model. Furthermore, he shows that for any signature scheme with *unique* signature any reduction must essentially be partitioning, and thus the loss of $O(q)$ is inherent. See also [16].

Waters' results. Waters [21] conducts a similar partitioning simulation in the standard model, for a particular CDH-based signature scheme. For this outline, we will only give a very abstract and idealized breakdown of his strategy. (For more details, see Section 3.) Namely, whether a message $m = (m_1, \ldots, m_\ell) \in \{0,1\}^\ell$ is signable or not depends on the value

$$a(m) = a_0 + \sum_i m_i \cdot a_i \tag{1}$$

for $a_i \in \mathbb{Z}$ chosen suitably by the simulator. If $a(m) = 0$, then m is unsignable; else, m is signable. Waters gives a setup of the a_i as uniform values over a suitable domain. As a result, any given message is signable with probability $p = 1 - O(1/(\ell \cdot q))$. This leads to a simulation success of $P = O(1/(\ell \cdot q))$. Later, Hofheinz and Kiltz [12, 13] showed how to set up the a_i differently, as random walks of a suitable length. They achieved a simulation success of $P = O(1/(\sqrt{\ell} \cdot q))$.

Guo et al. describe another way to set-up the a_i in the proof of [10, Theorem 2], and claim that this set-up can used to give a tighter security reduction for Waters signatures. However, it turns out that this is not true [11]. The reason is that in the proof of [10, Theorem 2] the simulation is set up in a way that depends on the messages to be signed. (Specifically, the variables that correspond to our a_i are not statistically hidden in [10].) Thus, the view of the adversary is not independent of the event that the simulation succeeds. Concretely, the setup in [10] potentially allows adversaries who forge only signatures for messages m^* with $a(m^*) \neq 0$, in which case no solution to the CDH problem can be extracted.

The problem. We now explain that problem that prevents a better security reduction of Waters signatures. Recall the random variable $a(m)$ from (1). Assume m is a message to be signed and m^* is the message for which the forger eventually generates a signature. Assume further that m and m^* differ in only one bit, such that $a(m) - a(m^*) = a_i$ for some i. Then the probability for $a(m) \neq 0$ and $a(m^*) = 0$ simultaneously (such that m is signable, and m^* is not) is upper bounded by $p' \cdot (1 - p')$, where p' is the probability that a_i takes a particular value. Specifically, if we aim at a simulation success of, say, $O(1/q)$, then each a_i must come from a distribution of min-entropy at least $\log_2(q)$. But then, the probability that $a(m^*) = 0$ for, say, the all-one message m^* is at most $O(1/(\ell \cdot q))$, since then $a(m^*) = \sum_{i=0}^{\ell} a_i$. We note that the improved bound of $O(1/(\sqrt{\ell} \cdot q))$ of [12, 13] is obtained by aiming at a simulation success of $O(1/(\sqrt{\ell} \cdot q))$.

Our solution. Our solution to the problem above is simply to encode all messages prior to signing, using a code with suitably large minimum distance. This ensures that any two messages have large Hamming distance, and thus any difference $a(m) - a(m^*)$ consists of a constant fraction of all a_i. This allows to choose each individual a_i as a distribution with smaller min-entropy, and we end up with a simulation success of $O(1/q)$. The details are somewhat technical, and we postpone a more detailed exposition to Section 3.

Optimality of our solution. Naturally, one may ask whether it is possible to improve the reduction further. We answer this question in the negative. Concretely, we show that it is impossible to prove any re-randomizable signature scheme secure, using a black-box reduction to any of a large class of hardness assumptions, such that the security loss in the reduction is significantly better than $1/q$. Since both Waters signatures and our new variant are efficiently re-randomizable, this shows our reduction optimal.

The proof technique is based on the meta-reduction technique of Coron [6], which simulates a forger for $\mathcal{R}$ such that the simulation fails with probability at most $1/q$. For Coron's proof it is essential that the considered signature scheme is deterministic, and that for all public keys it is publicly verifiable that there exists only a single valid signature per message (as it is the case for instance for certified trapdoor permutations, cf. [16]). Since we want to consider probabilistic schemes, we lose this leverage and Coron's result does not apply.

Instead, we will show that it suffices that signatures are re-randomizable. In particular deterministic signature schemes with unique signature are re-randomizable, thus our result generalizes [6, 16].

Let us intuitively sketch the reason why re-randomizability suffices. Basically, if signatures are efficiently re-randomizable, then the only way left to prove security is to partition the message space into messages which can be signed by the reduction, and messages from which a solution to the given problem instance can be extracted. To see this, suppose that for a random message m^* it holds with high probability that the reduction can simulate one signature for m^*, but extract a solution to a hard problem from a different signature for m^*. Then the reduction could solve the hard problem even without interacting with the forger, by generating a simulated signature σ^* for m^*, re-randomizing it to obtain some random signature σ', and finally extracting the solution to the hard problem from σ'. Since the reduction would solve the problem without any additional assumption (i.e. the existence of a signature forger), this would contradict the assumption that the underlying problem is hard.

Further applications. We note that the analysis from Section 4 can also be applied to show that a security reduction from any hard problem to breaking Waters' identity-based encryption (IBE) scheme from [21] must lose a factor of $\Omega(q)$, if the adversary may issue q adaptive chosen-identity key queries are allowed.

However, this bound is only achievable using our techniques if one wants to prove that Waters' IBE scheme is *one-way* under adaptive chosen-identity attacks. The commonly accepted security notion for IBE is *indistinguishability* under adaptive chosen-identity attacks, and it seems that in this setting our techniques do not substantially improve on the results of [21, 2]. Therefore we do not elaborate this further.

1.2 Outline

We recall some notation, some standard definitions, and Waters' signature scheme in Section 2. In Section 3, we present our modified signature scheme and prove it secure with a reduction loss of $O(q)$. Finally, in Section 4, we show a lower bound of $\Omega(q)$ on the reduction loss of schemes with re-randomizable signatures.

2 Preliminaries

For $k \in \mathbb{N}$, we write 1^k for the string of k ones, and $[k]$ for $\{1, \ldots, k\}$. Moreover, $|x|$ denotes the length of a bitstring x, while $|S|$ denotes the size of a set S. Further, $s \xleftarrow{\$} S$ denotes the sampling a uniformly random element s of S. For an algorithm $\mathcal{A}$, we write $z \xleftarrow{\$} \mathcal{A}(x, y, \ldots)$ to indicate that $\mathcal{A}$ is a (probabilistic) algorithm that outputs z on input $(x, y, \ldots)$.

Digital signatures. A digital signature scheme $\mathsf{Sig} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vfy})$ consists of three algorithms. Key generation Gen generates a keypair $(pk, sk) \xleftarrow{\$} \mathsf{Gen}(1^k)$ for a secret signing key sk and a public verification key pk. The signing algorithm Sign inputs a message and the secret signing key, and returns a signature $\sigma \xleftarrow{\$} \mathsf{Sign}(sk, m)$ of the message. The verification algorithm Vfy takes a verification key and a message with corresponding signature as input, and returns $b \leftarrow \mathsf{Vfy}(pk, m, \sigma)$, where $b \in \{0, 1\}$. We say that a signature is *valid*, if $\mathsf{Vfy}(pk, m, \sigma) = 1$. We require the usual correctness properties.

Let us recall the *existential unforgeability against chosen message attacks* (EUF-CMA) security experiment [9], played between a challenger and a forger $\mathcal{F}$.

1. The challenger runs Gen to generate a keypair (pk, sk). The forger receives pk as input.
2. The forger may ask the challenger to sign a number of messages. To query the i-th signature, $\mathcal{F}$ submits a message $m^{(i)}$ to the challenger. The challenger returns a signature σ_i under sk for this

message.

3. The forger outputs a message m^* and signature σ^*.

$\mathcal{F}$ wins the game, if $1 \leftarrow \mathsf{Vfy}(pk, m^*, \sigma^*)$, that is, σ^* is a valid signature for m^*, and $m^* \neq m^{(i)}$ for all i.

Definition 2.1. We say that $\mathcal{F}$ (t, q, ϵ)-breaks the EUF-CMA security of Sig, if $\mathcal{F}$ runs in time t, makes at most q signing queries, and has success probability ϵ. Furthermore, we say that Sig is EUF-CMA secure if there is no PPT forger $\mathcal{F}$ that t, q, ϵ-breaks the EUF-CMA security of Sig for polynomials t, q and a non-negligible ϵ.

The Computational Diffie-Hellman Problem. Let $\mathbb{G}$ be a group of order p. The computational Diffie-Hellman problem is to compute the group element $g^{\alpha\beta}$, given random group elements $(g, g^\alpha, g^\beta) \in \mathbb{G}^3$.

Definition 2.2. We say that algorithm $\mathcal{A}$ (ϵ, t)-*solves* the computational Diffie-Hellman problem in $\mathbb{G}$, if

$$\Pr\left[\mathcal{A}(g, g^\alpha, g^\beta) = g^{\alpha\beta}\right] \geq \epsilon,$$

and $\mathcal{A}$ runs in time t.

Waters Signatures. Let us recall Waters' signature scheme $\mathsf{Sig}_{\mathsf{Wat}} = (\mathsf{Gen}_{\mathsf{Wat}}, \mathsf{Sign}_{\mathsf{Wat}}, \mathsf{Vfy}_{\mathsf{Wat}})$ from [21].

$\mathsf{Gen}_{\mathsf{Wat}}(1^k)$**:** The key generation algorithm selects a group $\mathbb{G}$ of prime order $p \approx 2^{2k}$ with generator g and bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Then it samples $h_0, h_1, \ldots, h_\ell \xleftarrow{\$} \mathbb{G}$ and $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$. The public key is defined as

$$pk := (\mathbb{G}, g, g^\alpha, g^\beta, h_0, h_1, \ldots, h_\ell),$$

and the secret key is $sk := (pk, g^{\alpha\beta})$.

In the sequel we will denote with $H : \{0,1\}^\ell \to \mathbb{G}$ the function mapping $m \mapsto h_0 \prod_{i=1}^{\ell} h_i^{m_i}$, where for $i \in [\ell]$, we denote by $m_i \in \{0,1\}$ the ith bit of m.

$\mathsf{Sign}_{\mathsf{Wat}}(sk, m)$**:** The signing algorithm takes as input a message $m \in \{0,1\}^\ell$. The algorithm samples $r \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\sigma_1 = g^r \qquad \text{and} \qquad \sigma_2 = g^{\alpha\beta} H(m)^r.$$

Then it returns the signature $\sigma = (\sigma_1, \sigma_2)$.

$\mathsf{Vfy}_{\mathsf{Wat}}(pk, m, \sigma)$**:** The verification algorithm returns 1 if the equation

$$e(g^\alpha, g^\beta) \cdot e(\sigma_1, H(m)) = e(g, \sigma_2)$$

holds. Otherwise 0 is returned.

Waters [21] proved that the above signature scheme is EUF-CMA secure under the computational Diffie-Hellman assumption in $\mathbb{G}$. The original reduction from [21] is not very tight. Concretely, it loses a factor of $(16(\ell+1)q)$, where ℓ is the bit-length of the message and q is (an upper bound on) the number of signature queries issued by the forger. The original analysis was slightly improved in [12], which gives the following theorem.

Theorem 2.3 ([21, 12])**.** *Suppose there exists a forger $\mathcal{F}$ that (t, q, ϵ)-breaks the EUF-CMA security of $\mathsf{Sig}_{\mathsf{Wat}}$. Then there exists an algorithm $\mathcal{A}$ (ϵ', t')-solving the computational Diffie-Hellman problem in $\mathbb{G}$ in time $t' \approx t$ with success probability $\epsilon' \geq \epsilon \cdot O(\frac{1}{\sqrt{\ell q}})$.*

3 A Variant of Waters' Signature Scheme

Waters signatures. Recall Waters' signature scheme from Theorem 2. A message $m = (m_1, \ldots, m_\ell)$ to be signed selects group elements h_i (for i with $m_i = 1$) that determine an intermediate hash value

$$H(m) = h_0 \prod_{m_i=1} h_i.$$

Depending on $H(m)$, the simulation in the security proof will be able to either generate a signature for m, or use any forged signature for m to solve a given CDH-challenge. Concretely, each h_i is associated with an (information-theoretically hidden) integer a_i. A message m in turn leads to an integer $a(m) := a_0 + \sum_{m_i=1} a_i$. If $a(m) \neq 0$, then the simulation can sign m; if $a(m) = 0$, then the simulation can use any forged signature for m to solve a CDH-challenge.

The programming of the hash function. Unfortunately, neither the messages that need to be signed, nor the message on which the adversary forges are known in advance. Hence, the crux in the security analysis is to set up the values a_i such that with significant probability (say, P) over the a_i,
(a) all q adversarial signature queries $m^{(1)}, \ldots, m^{(q)}$ can be answered (i.e., $a(m^{(i)}) \neq 0$ for all i), and
(b) the message m^* on which the adversary forges can be used to embed a challenge (i.e., $a(m^*) = 0$).
The probabilistic argument from [21] chooses the a_i uniformly over a suitable domain that depends on q. This results in a simulation success probability of $P = \Theta(1/(\ell \cdot q))$. Hofheinz and Kiltz [12, 13] show that by setting up the a_i as suitably long random walks, the success probability can be improved to $P = \Theta(1/(\sqrt{\ell} \cdot q))$.

The problem. The reason for the somewhat annoying ℓ, resp. $\sqrt{\ell}$ terms in these analyses is a bit subtle, and we will only try to give a brief idea here. Concretely, consider what happens when the forgery message m^* is "close" to a signed message $m^{(i)}$ in the sense that m^* and $m^{(i)}$ differ in only one bit. Then, $a(m^*)$ and $a(m^{(i)})$ differ by only one a_i. Now the analysis requires that the conditioned probability $\Pr\left[a(m^{(i)}) = 0 \mid a(m^*) = 0\right]$ is $O(1/q)$. (Otherwise, it becomes difficult to prove that the probability is significant that, say, q random messages $m^{(i)}$ can all be signed, given that m^* cannot.) But since $a(m^*)$ and $a(m^{(i)})$ differ by only one (a-priori unknown) a_i, *each* a_i must have a distribution with min-entropy at least $\log_2 q$. (That is, the probability that a_i takes a particular value must always be $O(1/q)$.) Hence, e.g., for the all-one message $m = (1, \ldots, 1)$, we get that $a(m) = a_0 + \sum_{i=1}^{\ell} a_i$, and we would expect that $a(m)$ has a much larger min-entropy than $\log_2 q$. (In particular, if m^* is the all-one message, then $\Pr[a(m^*) = 0]$ will be much smaller than $\Theta(1/q)$.)

Our solution. Intuitively, our solution is simply to encode all messages using a code with large minimum distance prior to signing. This avoids that two messages $m^*, m^{(i)}$ that are "close" even exist. Concretely, we will ensure that any two different $(a(m^*), a(m^{(i)}))$ will always differ by at least *a constant fraction* of all a_i. This allows to set up the a_i with lower min-entropy than in previous analyses, and allows us to set up a simulation with success probability $P = \Theta(1/q)$.

3.1 Our variant of Waters signatures

As mentioned before, our only modification will be to encode messages prior to signing. For each security parameter k, we will therefore assume a code $\mathcal{C} = \mathcal{C}_k$ of dimension k, length ℓ, and minimum distance $d \geq \alpha \cdot \ell$ for a fixed $\alpha > 0$. (For instance, one can use a family of expander codes with suitable parameters [20, 22].) We will apply $\mathcal{C}$ to k-bit messages, and we assume that each encoded message has Hamming weight at least d. (For instance, one could simply forbid any message that leads to an all-zero output.)

Our scheme $\mathsf{Sig}_{\mathsf{tight}} = (\mathsf{Gen}_{\mathsf{tight}}, \mathsf{Sign}_{\mathsf{tight}}, \mathsf{Vfy}_{\mathsf{tight}})$ is almost identical to the one by Waters (see Theorem 2):

Keys. $\mathsf{Gen}_{\mathsf{tight}}(1^k)$ outputs $pk := (\mathbb{G}, g, g^\alpha, g^\beta, h_1, \ldots, h_\ell)$ and $sk := (pk, g^{ab})$ just like $\mathsf{Gen}_{\mathsf{Wat}}$, but without h_0. Now pk defines a hash function $H(M) := h_i^{M_i}$ for $M = (M_1, \ldots, M_\ell) \in \{0,1\}^\ell$.

Signing. $\mathsf{Sign}_{\mathsf{tight}}(sk, m)$ (for $m \in \{0,1\}^k$) first computes $M := \mathcal{C}_k(m) \in \{0,1\}^\ell$ and then outputs $\sigma := (\sigma_1, \sigma_2) := (g^r, g^{ab} H(M)^r)$.

Verification. $\mathsf{Vfy}_{\mathsf{tight}}(pk, m, \sigma)$ sets $M := \mathcal{C}_k(m)$ and then checks $e(g^\alpha, g^\beta) \cdot e(\sigma_1, H(M)) \stackrel{?}{=} e(g, \sigma_2)$.

Obviously, this defines a signature scheme. We also claim:

Theorem 3.1. *Suppose there exists a forger $\mathcal{F}$ that (t, q, ϵ)-breaks the EUF-CMA security of $\mathsf{Sig}_{\mathsf{tight}}$. Then there exists an algorithm $\mathcal{A}$ (ϵ', t')-solving the computational Diffie-Hellman problem in $\mathbb{G}$ in time $t' \approx t$ with success probability $\epsilon' \geq \epsilon \cdot \Theta(\frac{1}{q})$.*

The rest of this section will be devoted to proving Theorem 3.1.

3.2 A better bound on the success probability of the simulation

We start with our abstract setup and the analysis of the crucial variables a_i for our simulation. In the next subsection, we then proceed to outline how this setup is embedded in a simulation of the signature scheme.

In the following let $\ell, w \in \mathbb{N}$. In the simulation, ℓ will be the bitlength of (encoded) messages, and w will be an integer that determines how long each random walk a_i will be. For $i \in [\ell], j \in [w]$, let $a_{i,j}$ be independently and uniformly distributed random variables over $\{-1, 0, 1\}$. Let $a_i := \sum_{j=1}^{w} a_{i,j}$. Furthermore, for $S \subseteq [\ell]$, let $a(S) := \sum_{i \in S} a_i$. Note that $a(S)$ is a random walk (with $\{-1, 0, 1\}$-steps) of length $|S| \cdot w$. Hence, the following standard result about random walks applies:

Theorem 3.2. *There exist $\lambda, \Lambda \in \mathbb{R}$ that do not depend on ℓ, w, such that for any $S \subseteq [\ell]$ of size $s := |S|$, we have*

$$\frac{\lambda}{\sqrt{s \cdot w}} \quad \leq \quad \Pr[a(S) = 0] \quad \leq \quad \frac{\Lambda}{\sqrt{s \cdot w}}.$$

Furthermore, for any ℓ, w, S, the probability $\Pr[a(S) = i]$ is maximized for $i = 0$.

Proof. Although this is a standard fact about random walks (see [14, 7] for a thorough introduction), [13, Theorems 17 and 18] provide a direct proof of the theorem adjusted to our setting. □

We can now use Theorem 3.2 to derive the main technical lemma for the analysis of our variant of Waters' signature scheme. This result uses and extends techniques of [12, 13] to a setting in which there is a guaranteed "minimum distance" between two random walks. (Later, this "minimum distance" will correspond to the Hamming distance between two encoded messages to be signed.)

Lemma 3.3. *Let $X, Y \subseteq [\ell]$ such that $|X|, |Y| \geq d$, and $|(X \setminus Y) \cup (Y \setminus X)| \geq d$ for $d \geq 1$. Then, we have*

$$\Pr[a(Y) = 0 \mid a(X) = 0] \quad \leq \quad C \cdot \frac{\sqrt{\ell}}{d \cdot \sqrt{w}} \tag{2}$$

for a fixed constant C that does not depend on ℓ, w, d, X, Y.

See Appendix A for the proof.

Next, we can plug Lemma 3.3 into the existing analysis of Waters' signature scheme [21]. First, this means proving the following technical claim, which essentially bounds the probability that all signing queries can be answered, while the adversary's forgery solves a computational challenge. This claim roughly corresponds to [21, Claim 2] and is proven in Appendix B

Lemma 3.4. *Let $X, Y_1, \ldots, Y_q \subseteq [\ell]$ such that $|X|, |Y_i| \geq d$ and $|(X \setminus Y_i) \cup (Y_i \setminus X)| \geq d$ for some $d \geq 1$ and all i. Then, we have*

$$\Pr\left[a(X) = 0 \wedge \forall i \in [q] : a(Y_i) \neq 0\right] \quad \geq \quad \left(1 - C \cdot q \cdot \frac{\sqrt{\ell}}{d \cdot \sqrt{w}}\right) \cdot \frac{D}{\sqrt{d \cdot w}} \tag{3}$$

for fixed constants C, D that do not depend on ℓ, w, d, q, X, and the Y_i.

Note that if we set $d = \alpha \cdot \ell$ and $w = (2Cq/\alpha)^2$ (for some $\alpha > 0$) in (3), a quick calculation gives

$$\Pr\left[a(X) = 0 \wedge \forall i \in [q] : a(Y_i) \neq 0\right] \quad \geq \quad \frac{D\sqrt{\alpha}}{4C} \cdot \frac{1}{q}. \tag{4}$$

Hence, if α is a constant, then this probability lies in the order of $1/q$.

3.3 The full simulation

We now briefly sketch how to use Lemma 3.4 to prove Theorem 3.1. We are very brief because except for Lemma 3.4 and a few syntactic differences, the proof is identical to the one from [21].

Our goal is to build a CDH adversary $\mathcal{A}$ from an EUF-CMA forger $\mathcal{F}$ on $\mathsf{Sig}_{\mathsf{tight}}$ that makes at most $q = q(k)$ signature queries. Our CDH adversary $\mathcal{A}$ gets as input a CDH challenge (g, g^α, g^β) for a group $\mathbb{G}$ of order p with pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, and is supposed to output g^{ab}.

Public key. The first task of $\mathcal{A}$ is to prepare a $\mathsf{Sig}_{\mathsf{tight}}$ public key for $\mathcal{F}$. In order to do so, $\mathcal{A}$ sets $w := (2Cq/\alpha)^2$ for the parameter α of the code $\mathcal{C}$, and the constant C from Section 3.2. Then, $\mathcal{A}$ prepares random variables $a_1, \ldots, a_\ell$ as random walks (over $\{-1, 0, 1\}$) of length w, just as in Section 3.2. Finally, $\mathcal{A}$ chooses uniformly blinding exponents $b_1, \ldots, b_\ell \leftarrow [p]$ and sets

$$h_i := (g^\alpha)^{a_i} \, g^{b_i} \quad \text{(for } i = 1, \ldots, \ell\text{)}$$
$$pk := (\mathbb{G}, g, g^\alpha, g^\beta, h_1, \ldots, h_\ell).$$

This results in a public key that is distributed exactly as in $\mathsf{Sig}_{\mathsf{tight}}$.

Signing queries. Next, $\mathcal{A}$ runs $\mathcal{F}$ on pk, and answers $\mathcal{F}$'s signing queries as follows. Suppose $\mathcal{F}$ asks for the signature of a message $m \in \{0,1\}^k$ that induces an encoded message $M = \mathcal{C}(m) \in \{0,1\}^\ell$. Let us view M as a subset of $[n]$, such that $i \in M$ iff the i-th bit of M is set. Write $a(M) := \sum_{i \in M} a_i$ and $b(M) := \sum_{i \in M} b_i$. Note that we can always write $H(M) = (g^\alpha)^{a(M)} g^{b(M)}$. Hence, valid signatures have the form

$$(g^r, g^{\alpha\beta} \cdot H(M)^r) \quad = \quad (g^r, g^{\alpha\beta + r \cdot (a \cdot a(M) + b(M))})$$

In particular, if we set $g^r = \left(g^\beta\right)^x g^y$, then valid signatures are of the form

$$(g^{x\beta+y}, g^{\alpha\beta + (x\beta+y)\cdot(\alpha \cdot a(M) + b(M))}) \; = \; \left(\left(g^\beta\right)^x g^y, \left(g^{\alpha\beta}\right)^{1 + x \cdot a(M)} (g^\alpha)^{x \cdot b(M)} \left(g^\beta\right)^{y \cdot a(M)} g^{y \cdot b(M)}\right). \tag{5}$$

Thus, depending on $a(M)$, we now distinguish two cases:

- if $a(M) \neq 0$, then the simulation can generate properly distributed valid signatures via (5) by setting $x = -a(M)^{-1} \bmod p$ and choosing y uniformly (notice that the g^{ab} term in (5) then vanishes);
- if $a(M) = 0$, then the simulation cannot generate a signature for m, and the simulation fails.

Extraction. Suppose that eventually, $\mathcal{F}$ generates a valid forged signature σ^* for a fresh message m^* with associated encoding $M^* := \mathcal{C}(m^*)$. Again, we can distinguish two cases:

- if $a(M^*) = 0$, then the simulation can extract g^{ab} by using

$$\sigma^* = (g^{r^*}, g^{\alpha\beta} \cdot H(M^*)^{r^*}) = \left(g^{r^*}, g^{\alpha\beta} \cdot \left(g^{b(M^*)}\right)^{r^*}\right) = \left(g^{r^*}, g^{\alpha\beta} \cdot \left(g^{r^*}\right)^{b(M^*)}\right)$$

  for some unknown r^* but known $b(M^*)$;
- if $a(M^*) \neq 0$, then the extraction fails.

Simulation success. Let $\texttt{fail}$ denote the event that the simulation fails (either because $a(M_i) = 0$ for a signature query, or because $a(M^*) \neq 0$). Then Lemma 3.4 immediately gives an upper bound of $1 - \Theta(1/q)$ on $\Pr[\texttt{fail}]$. Indeed, if we set $X := M^*$ and $Y_i := M_i$, then any two different encoded messages differ in at least $d = \alpha \cdot \ell$ bits. In particular, $|(X \setminus Y_i) \cup (Y_i \setminus X)| \geq d$. Substituting $d = \alpha \cdot \ell$ and $w = (2Cq/\alpha)^2$ in (3) yields (4), and thus a lower bound of $\Theta(1/q)$ on $\neg\texttt{fail}$. Furthermore, the a_i are information-theoretically hidden from $\mathcal{F}$, so conditioning on $\neg\texttt{fail}$ does not change $\mathcal{F}$'s success in the EUF-CMA experiment. Theorem 3.1 follows.

4 Lower Tightness Bounds for Re-Randomizable Signatures

In this section we show that it is impossible to prove security of a signature scheme with significantly smaller security loss than $\Omega(q)$, if the signature scheme is efficiently re-randomizable. To this end, we first define re-randomizable signatures. Then we give abstract definitions of computational problems, and reductions that reduce solving a given computational problem to breaking the security of a given signature scheme. All these results are generic, in the sense that they apply to any re-randomizable signature scheme. Finally, we show that both Waters' signature scheme from [21] and our modified scheme from Section 3 are efficiently re-randomizable, which implies that the reduction from Section 3 is optimal.

4.1 Re-Randomizable Signatures

The intuition behind re-randomizable signatures is the property that, given only the public key pk and a valid signature σ for some message m, one can efficiently generate a new signature σ' that is distributed uniformly over the set of all possible signatures for m.

Let $\mathsf{Sig} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vfy})$ be a signature scheme. For any string pk (which may either be a honestly generated public key, or a fake public key generated by a simulator in a security proof) let us denote with

$$\Sigma(pk, m) = \{\sigma : \mathsf{Vfy}(pk, m, \sigma) = 1\}$$

the set of signatures σ for message m that verify correctly under public key pk.

Definition 4.1. We say that Sig is *t-re-randomizable*, if there exists an algorithm ReRand running in time at most t, such that for all (pk, m, σ) with $\mathsf{Vfy}(pk, m, \sigma) = 1$ holds that the output distribution of

$$\mathsf{ReRand}(pk, m, \sigma)$$

is identical to the uniform distribution over $\Sigma(pk, m)$.

4.2 Computational Problems and Reductions

The definitions in this section follow [6].

Definition 4.2. A *computational problem* $\Pi = (C, \mathcal{S})$ consists of a set C and a family of sets $\mathcal{S} = (S_c)_{c \in C}$. We say that C is the set of *challenges* of Π, and for each $c \in C$ set S_c is the set of *solutions* for c. We say that an algorithm $\mathcal{A}$ $(\epsilon_{\mathcal{A}}, t_{\mathcal{A}})$-*solves* Π, if $\mathcal{A}$ runs in time $t_{\mathcal{A}}$ and

$$\Pr[\mathcal{A}(c) \in S_c : c \xleftarrow{\$} C] \quad \geq \quad \epsilon_{\mathcal{A}}.$$

As an example consider a group $\mathbb{G}$ of prime order p. Then the computational Diffie-Hellman problem in $\mathbb{G}$ is the problem $\Pi = (C, \mathcal{S})$ with $C = \mathbb{G} \times \mathbb{G} \times \mathbb{G}$ and where for each $c = (g, g^a, g^b) \in C$ we have $S_c = \{g^{ab}\}$.

Definition 4.3. We say that an algorithm $\mathcal{R}$ is a $(t_{\mathcal{F}}, \epsilon_{\mathcal{F}}, q, \epsilon_{\mathcal{R}}, t_{\mathcal{R}})$-*reduction* from problem Π to breaking the security of signature scheme Sig, if for *any* forger $\mathcal{F}$ that $(t_{\mathcal{F}}, \epsilon_{\mathcal{F}}, q)$-breaks the EUF-CMA security of Sig in the sense of Definition 2.1, algorithm $\mathcal{R}$ $(\epsilon_{\mathcal{R}}, t_{\mathcal{R}})$-solves Π.

Note that we require that the $\mathcal{R}$ works for any forger $\mathcal{F}$, in particular if $\mathcal{F}$ is given as a black-box.

For instance, in Section 3 we gave an example for an algorithm $\mathcal{R}$ that $(t_{\mathcal{F}}, \epsilon_{\mathcal{F}}, q, \epsilon_{\mathcal{F}} \cdot \Theta(1/q), t_{\mathcal{R}})$-reduces solving the computational Diffie-Hellman problem to breaking the security of Waters' signature scheme with $t_{\mathcal{R}} \approx t_{\mathcal{F}}$.

4.3 Lower Tightness Bound for Re-Randomizable Signature Schemes

In this section we consider reductions that run Forger $\mathcal{F}$ only once, and show that any such reduction loses a factor of at least q. A generalization to reductions that run $\mathcal{F}$ repeatedly is straightforward, see Appendix C.

Theorem 4.4. *Let* Sig *be a* t_{ReRand}*-re-randomizable signature scheme and let* Π *be a computational problem in the sense of Definition 4.2. If there exists an* $(t_{\mathcal{F}}, \epsilon_{\mathcal{F}}, q, \epsilon_{\mathcal{R}}, t_{\mathcal{R}})$*-reduction* $\mathcal{R}$ *that runs* $\mathcal{F}$ *once and reduces* Π *to breaking* Sig*, then there exists an algorithm* $\mathcal{A}$ *that* $(\epsilon_{\mathcal{A}}, t_{\mathcal{A}})$*-solves* Π *with* $t_{\mathcal{A}} \approx 2t_{\mathcal{R}} + t_{\mathsf{ReRand}}$ *and*

$$\epsilon_{\mathcal{A}} \geq \epsilon_{\mathcal{R}} - \frac{\exp(-1)}{q}.$$

We will use the following lemma, which is due to Coron [6].

Lemma 4.5. *Let* $\mathcal{M}$ *be a set and let* Q *be a set of sequences of at most* q *elements of* $\mathcal{M}$*, such that for any sequence* $(m_1, \ldots, m_j) \in Q$ *we have* $(m_1, \ldots, m_{j-1}) \in Q$*. Let* $i \xleftarrow{\$} [q]$ *and* $(m_1, \ldots, m_q, m^*) \xleftarrow{\$} \mathcal{M}^{q+1}$ *be uniformly random. Then*

$$\Pr\left[(m_1, \ldots, m_q) \in Q \wedge (m_1, \ldots, m_{i-1}, m^*) \notin Q\right] \leq \frac{\exp(-1)}{q}.$$

See [6, Appendix D] for the proof.

Proof of Theorem 4.4. Consider an (imaginary) forger $\mathcal{F}$ that $(t_{\mathcal{F}}, \epsilon_{\mathcal{F}}, q)$-breaks the EUF-CMA security of a given signature scheme Sig with some success probability $\epsilon_{\mathcal{F}}$ in some time $t_{\mathcal{F}}$. Forger $\mathcal{F}$ works as follows.

1. $\mathcal{F}$ receives as input a public key pk from the challenger.

2. It selects $q+1$ random pairwise different messages $(m^{(1)}, \ldots, m^{(q)}, m^*)$ from the message space of Sig.

3. Then $\mathcal{F}$ queries the challenger for signatures of messages $(m^{(1)}, \ldots, m^{(q)})$.

4. $\mathcal{F}$ computes a valid signature σ^* for message m^*, such that σ^* is distributed uniformly over $\Sigma(pk, m^*)$. (Forger $\mathcal{F}$ may be inefficient, since it needs to forge a signature. However, we will later show how to simulate $\mathcal{F}$ efficiently.)

5. Finally $\mathcal{F}$ tosses a (biased) coin $b \xleftarrow{\$} \{0,1\}$ with $\Pr[b=1] = \epsilon_{\mathcal{F}}$.
   (a) If $b = 1$ then it outputs σ^*.
   (b) Otherwise it outputs error symbol $\perp$.

Note that any $(t_{\mathcal{F}}, \epsilon_{\mathcal{F}}, q, \epsilon_{\mathcal{R}}, t_{\mathcal{R}})$-reduction from some computational problem Π to breaking the security of Sig can use Forger $\mathcal{F}$ to $(\epsilon_{\mathcal{R}}, t_{\mathcal{R}})$-solve Π. In the sequel we will apply the rewinding technique of Coron [6] to show how to simulate $\mathcal{F}$, if Sig is re-randomizable.

Consider an algorithm $\mathcal{A}$ that uses $\mathcal{R}$ as follows.

1. $\mathcal{A}$ receives as input an instance c of Π, and starts $\mathcal{R}$ on input c.

2. $\mathcal{R}$ outputs a public key pk.

3. $\mathcal{A}$ selects a random integer $i \in [q]$ and $q+1$ random pairwise different messages $(m^{(1)}, \ldots, m^{(q)}, m^*)$.

4. It queries $\mathcal{R}$ for signatures for the sequence of messages $M_0 = (m^{(1)}, \ldots, m^{(i-1)}, m^*)$. If $\mathcal{R}$ aborts, then so does $\mathcal{A}$.

5. Then $\mathcal{A}$ rewinds $\mathcal{R}$ to the state after it has output the public key (i.e. the state after Step 2).

6. Now $\mathcal{A}$ queries for signatures for the sequence of messages $M_1 = (m^{(1)}, \ldots, m^{(q)})$. Again, if $\mathcal{R}$ aborts, then $\mathcal{A}$ aborts too.

7. Then $\mathcal{A}$ computes $\sigma' = \mathsf{ReRand}(pk, m^*, \sigma^*)$ and tosses a coin $b \xleftarrow{\$} \{0,1\}$ with $\Pr[b=1] = \epsilon_{\mathcal{F}}$.
   (a) If $b = 1$ then it submits σ' to $\mathcal{R}$.
   (b) Otherwise it submits error symbol $\perp$.

   Finally $\mathcal{A}$ returns outputs whatever $\mathcal{R}$ returns

Let $\mathcal{E}$ denote the event that $M_0 \notin Q$ and $M_1 \in Q$. Note that, due to the re-randomizability of Sig, $\mathcal{A}$ outputs a uniformly random signature σ' from the set $\Sigma(pk, m^*)$ of all valid signatures for m^* and public key pk. Therefore $\mathcal{A}$ simulates $\mathcal{F}$ perfectly (after the rewind), and thus can use $\mathcal{R}$ to solve Π, unless $\mathcal{E}$ occurs. By applying Lemma 4.5, we obtain that the success probability of $\mathcal{A}$ is at least

$$\begin{aligned}\epsilon_{\mathcal{A}} &\geq \epsilon_{\mathcal{R}} - \Pr[\mathcal{E}] \\ &= \epsilon_{\mathcal{R}} - \Pr[M_0 \notin Q \wedge M_1 \in Q] \\ &\geq \epsilon_{\mathcal{R}} - \frac{\exp(-1)}{q}.\end{aligned}$$

$\mathcal{A}$ essentially runs $\mathcal{R}$ twice and performs one re-randomization, therefore the running time of $\mathcal{A}$ is $t_{\mathcal{A}} \approx 2t_{\mathcal{R}} + t_{\mathsf{ReRand}}$. □

The above theorem directly gives rise to the following corollary.

Corollary 4.6. *Let Π be a $(\epsilon, 2t + t_{\mathsf{ReRand}})$-hard computational problem. Then the success probability $\epsilon_{\mathcal{R}}$ of any security reduction from Π to breaking a re-randomizable signature scheme that runs in time t is at most*

$$\epsilon_{\mathcal{R}} \leq \frac{\exp(-1)}{q} + \epsilon.$$

In particular, if ϵ is close to zero and signatures are efficiently re-randomizable, then this gives an upper bound on the success probability of the reduction of $\epsilon_{\mathcal{R}} \lessapprox \exp(-1)/q$ for all reductions running in time t.

Note also that in principle any (probabilistic) signature scheme is re-randomizable, though not necessarily efficiently. However, the running time of the simulated forger depends on the running time of the re-randomization algorithm. Thus, in order to get a meaningful result, we need to require that signatures are *efficiently* re-randomizable.

4.4 Waters Signatures are Re-Randomizable

To show that any reduction from a computationally hard problem to the (t, q, ϵ)-EUF-CMA security of Waters signatures loses at least a factor $1/q$, it remains to show that Waters signatures are efficiently re-randomizable.

Note that the original Waters scheme from [21] and the variant from Section 3 differ only in the way the hash value $H(m) \in \mathbb{G}$ is computed. The following considerations do not depend on a specific function H. Therefore we consider a Waters signature scheme that uses some abstract hash function H in the sequel, which makes the analysis applicable to both schemes (and other similar constructions) simultaneously.

Lemma 4.7. *Waters signatures are t-re-randomizable, where t amounts to two exponentiations in $\mathbb{G}$ plus some minor additional operations.*

Proof. Let $pk = (\mathbb{G}, g, g^\alpha, g^\beta, H)$ be a given public key, and let m and $\sigma = (\sigma_1, \sigma_2)$ be a given message with valid Waters signature, i.e., σ satisfies

$$e(g^\alpha, g^\beta) \cdot e(\sigma_1, H(m)) = e(g, \sigma_2). \tag{6}$$

Since σ_1 is a group element, we can write $\sigma_1 = g^r$ for some integer $r \in \mathbb{Z}_p$, where $p = |\mathbb{G}|$ is the order of $\mathbb{G}$. Then Equation 6 implies that we can write σ_2 as $\sigma_2 = g^{\alpha\beta} H(m)^r$. The set of all (σ_1, σ_2) satisfying Equation 6 is therefore identical to the set

$$\Sigma(pk, m) = \{(g^r, g^{\alpha\beta} H(m)^r) : r \in \mathbb{Z}_p\}.$$

It remains to show that there exists an efficient algorithm ReRand that produces uniformly random elements of $\Sigma(pk, m)$ given only the public key pk, message m, and a valid signature $\sigma = (\sigma_1, \sigma_2)$. Consider algorithm ReRand taking as input pk, signature $(\sigma_1, \sigma_2) = (g^r, g^{\alpha\beta} H(m)^r)$ for some r, and message m. The algorithm samples $s \xleftarrow{\$} \mathbb{Z}_p$ and computes and returns (σ_1', σ_2') where

$$\sigma_1' := \sigma_1 \cdot g^s = g^{r+s} \qquad \text{and} \qquad \sigma_2' := \sigma_2 \cdot H(m)^s = g^{\alpha\beta} H(m)^{r+s}.$$

Since s is uniformly distributed over $\mathbb{Z}_p$, the resulting signature (σ_1', σ_2') is distributed uniformly over $\Sigma(sk, m)$, as required. □

Combining the above lemma with Theorem 4.4 yields the following result.

Theorem 4.8. *Let Π be a computational problem according to Definition 4.2. If there exists a $(t_{\mathcal{F}}, \epsilon_{\mathcal{F}}, q, \epsilon_{\mathcal{R}}, t_{\mathcal{R}})$-reduction $\mathcal{R}$ that reduces solving Π to breaking Waters signatures, then there exists an algorithm $\mathcal{A}$ that $(\epsilon_{\mathcal{A}}, t_{\mathcal{A}})$-solves Π with $t_{\mathcal{A}} \approx 2t_{\mathcal{R}}$ and*

$$\epsilon_{\mathcal{A}} \geq \epsilon_{\mathcal{R}} - \frac{\exp(-1)}{q}.$$

Thus, a reduction from any computational problem Π to breaking Waters signatures that runs in time t with success probability significantly better than $1/q$ implies that there exists an algorithm solving Π in time $\approx 2t$ with significant success probability.

References

[1] Nuttapong Attrapadung, Jun Furukawa, Takeshi Gomi, Goichiro Hanaoka, Hideki Imai, and Rui Zhang. Efficient identity-based encryption with tight security reduction. In David Pointcheval, Yi Mu, and Kefei Chen, editors, *CANS 06: 5th International Conference on Cryptology and Network Security*, volume 4301 of *Lecture Notes in Computer Science*, pages 19–36, Suzhou, China, December 8–10, 2006. Springer, Berlin, Germany.

[2] Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for Waters' IBE scheme. In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 407–424, Cologne, Germany, April 26–30, 2009. Springer, Berlin, Germany.

[3] 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, Istanbul, Turkey, April 13–17, 2008. Springer, Berlin, Germany.

[4] Benoît Chevallier-Mames and Marc Joye. A practical and tightly secure signature scheme without hash function. In Masayuki Abe, editor, *Topics in Cryptology – CT-RSA 2007*, volume 4377 of *Lecture Notes in Computer Science*, pages 339–356, San Francisco, CA, USA, February 5–9, 2007. Springer, Berlin, Germany.

[5] 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, Santa Barbara, CA, USA, August 20–24, 2000. Springer, Berlin, Germany.

[6] 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, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer, Berlin, Germany.

[7] William Feller. *An Introduction to Probability Theory and Its Applications, Vol. 1, 3rd Edition*. Wiley, 1968.

[8] Rosario Gennaro, Shai Halevi, and Tal Rabin. Secure hash-and-sign signatures without the random oracle. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 123–139, Prague, Czech Republic, May 2–6, 1999. Springer, Berlin, Germany.

[9] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988.

[10] Fuchun Guo, Yi Mu, and Willy Susilo. How to prove security of a signature with a tighter security reduction. In Josef Pieprzyk and Fangguo Zhang, editors, *ProvSec 2009: 3rd International Conference on Provable Security*, volume 5848 of *Lecture Notes in Computer Science*, pages 90–103, Guangzhou, China, November 11–13, 2009. Springer, Berlin, Germany.

[11] Fuchun Guo, Yi Mu, and Willy Susilo. Personal communication, November 2011.

[12] Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 21–38, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Berlin, Germany.

[13] Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. *Journal of Cryptology*, ??(??):???–???, 2012. Appears; online available at `http://eprint.iacr.org/2011/270`.

[14] Barry D. Hughes. *Random Walks and Random Environments: Volume 1: Random Walks*. Oxford University Press, 1995.

[15] Marc Joye. An efficient on-line/off-line signature scheme without random oracles. In Matthew K. Franklin, Lucas Chi Kwong Hui, and Duncan S. Wong, editors, *CANS 08: 7th International Conference on Cryptology and Network Security*, volume 5339 of *Lecture Notes in Computer Science*, pages 98–107, Hong-Kong, China, December 2–4, 2008. Springer, Berlin, Germany.

[16] Saqib A. Kakvi and Eike Kiltz. Optimal security proofs for full domain hash, revisited. In *Advances in Cryptology - EUROCRYPT 2012*, Lecture Notes in Computer Science. Springer, 2012.

[17] Jonathan Katz and Nan Wang. Efficiency improvements for signature schemes with tight security reductions. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *ACM CCS 03: 10th Conference on Computer and Communications Security*, pages 155–164, Washington D.C., USA, October 27–30, 2003. ACM Press.

[18] Edward Knapp. On pairing-based signature and aggregate signature schemes. Master's thesis, University of Waterloo, Ontario, Canada, 2008.

[19] Sven Schäge. Tight proofs for signature schemes without random oracles. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 189–206, Tallinn, Estonia, May 15–19, 2011. Springer, Berlin, Germany.

[20] Michael Sipser and Daniel A. Spielman. Expander codes. *IEEE Transactions on Information Theory*, 42(6):1710–1722, 1996.

[21] Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany.

[22] Gilles Zémor. On expander codes. *IEEE Transactions on Information Theory*, 47(2):835–837, 2001.

A Proof of Lemma 3.3

We distinguish the two cases $|X \setminus Y| \geq d/2$ and $|Y \setminus X| \geq d/2$:
Case $|Y \setminus X| \geq d/2$**:**

$$\begin{aligned}
\Pr[a(Y) = 0 \mid a(X) = 0] &\overset{(a)}{\leq} \max_i \Pr[a(Y) = 0 \mid a(Y \cap X) = i] \\
&\overset{(b)}{\leq} \max_i \Pr[a(Y \setminus X) = -i \mid a(Y \cap X) = i] \\
&\overset{(c)}{=} \max_i \Pr[a(Y \setminus X) = -i] \\
&\overset{(d)}{=} \Pr[a(Y \setminus X) = 0] \\
&\overset{(e)}{\leq} \frac{\sqrt{2} \cdot \Lambda}{\sqrt{d \cdot w}} \overset{(f)}{\leq} \sqrt{2} \cdot \Lambda \cdot \frac{\sqrt{\ell}}{d \cdot \sqrt{w}}
\end{aligned}$$

Here, (a) holds because $a(X)$ only depends on $a(Y \cap X)$ but not on $a(X \setminus Y)$; (b) uses $a(Y) = a(Y \setminus X) + a(Y \cap X)$; (c) uses that $a(Y \setminus X)$ and $a(Y \cap X)$ are independent; (d) and (e) apply Theorem 3.2, using that $a(Y \setminus X)$ is a random walk of length at least $(d/2) \cdot w$; finally, (f) uses $d \leq \ell$.
Case $|X \setminus Y| \geq d/2$**:**

$$\begin{aligned}
\Pr[a(Y) = 0 \mid a(X) = 0] &\overset{(a)}{=} \Pr[a(X) = 0 \mid a(Y) = 0] \cdot \frac{\Pr[a(Y) = 0]}{\Pr[a(X) = 0]} \\
&\overset{(b)}{\leq} \frac{\sqrt{2} \cdot \Lambda}{\sqrt{d \cdot w}} \cdot \frac{\Pr[a(Y) = 0]}{\Pr[a(X) = 0]} \\
&\overset{(c)}{\leq} \frac{\sqrt{2} \cdot \Lambda}{\sqrt{d \cdot w}} \cdot \frac{\sqrt{2} \cdot \Lambda}{\sqrt{d \cdot w}} \cdot \frac{\sqrt{\ell \cdot w}}{\sqrt{2} \cdot \lambda} \\
&= \frac{\sqrt{2} \cdot \Lambda^2}{\lambda} \cdot \frac{\sqrt{\ell}}{d \cdot \sqrt{w}}
\end{aligned}$$

Here, (a) uses Bayes' theorem; (b) uses what we have proved for the case $|Y \setminus X| \geq d/2$ (with swapped X, Y); (c) apply Theorem 3.2, using that $a(X)$ and $a(Y)$ are random walks of length at least $d \cdot w$ and at most $\ell \cdot w$.
Since we have $\Lambda \geq \lambda$, setting $C := \sqrt{2} \cdot (\Lambda^2/\lambda)$ proves (2).

B Proof of Lemma 3.4

We have

$$\begin{aligned}
\Pr\left[a(X) = 0 \wedge \forall i : a(Y_i) \neq 0\right] &= \Pr\left[\forall i : a(Y_i) \neq 0 \mid a(X) = 0\right] \cdot \Pr\left[a(X) = 0\right] \\
&\overset{(a)}{\geq} \Pr\left[\forall i : a(Y_i) \neq 0 \mid a(X) = 0\right] \cdot \frac{\lambda}{\sqrt{d \cdot w}} \\
&= \left(1 - \Pr\left[\exists i : a(Y_i) = 0 \mid a(X) = 0\right]\right) \cdot \frac{\lambda}{\sqrt{d \cdot w}} \\
&\overset{(b)}{\geq} \left(1 - \sum_{i=1}^{q} \Pr\left[a(Y_i) = 0 \mid a(X) = 0\right]\right) \cdot \frac{\lambda}{\sqrt{d \cdot w}} \\
&\overset{(c)}{\geq} \left(1 - q \cdot C \cdot \frac{\sqrt{\ell}}{d \cdot \sqrt{w}}\right) \cdot \frac{\lambda}{\sqrt{d \cdot w}}
\end{aligned}$$

Here, (a) applies Theorem 3.2, using that $a(X)$ is a random walk of length at least $d \cdot w$; (b) uses a union bound; (c) denotes a q-wise application of Lemma 3.3. Setting $D := \lambda$ yields (3).

C Reductions that run $\mathcal{F}$ more than once

So far we have only considered reductions that run the forger once. While the reduction from [21] is of this type, it may be possible that there exist a tighter reduction that runs $\mathcal{F}$ several times with different public keys. Fortunately, following [6, 16] it is very simple to generalize the result of Section 4.3 to reductions that run $\mathcal{F}$ repeatedly.

Theorem C.1. *Let* Sig *be a* t_{ReRand}*-re-randomizable signature scheme and let* Π *be a computational problem in the sense of Definition 4.2. If there exists a* $(t_{\mathcal{F}}, \epsilon_{\mathcal{F}}, q, \epsilon_{\mathcal{R}}, t_{\mathcal{R}})$*-reduction* $\mathcal{R}$ *that runs* $\mathcal{F}$ *at most* r *times and reduces* Π *to breaking* Sig*, then there exists an algorithm* $\mathcal{A}$ *that* $(\epsilon_{\mathcal{A}}, t_{\mathcal{A}})$*-solves* Π *with* $t_{\mathcal{A}} \approx 2 \cdot t_{\mathcal{R}} + r \cdot t_{\mathsf{ReRand}}$ *and*

$$\epsilon_{\mathcal{A}} \geq \epsilon_{\mathcal{R}} - \frac{r \cdot \exp(-1)}{q}.$$

The proof is very similar to the proof of Theorem 4.4, the only difference is that now $\mathcal{A}$ needs to simulate r executions of $\mathcal{F}$. Consider an adversary $\mathcal{A}$ which proceeds exactly like in the proof of Theorem 4.4, and let $\mathcal{E}_i$ denote the event that the simulation of $\mathcal{F}$ fails in the i-th execution. Then we have

$$\begin{aligned}
\epsilon_{\mathcal{A}} &\geq \epsilon_{\mathcal{R}} - \sum_{i=1}^{r} \Pr[\mathcal{E}_i] \\
&= \epsilon_{\mathcal{R}} - \sum_{i=1}^{r} \Pr[M_{i,0} \notin Q \wedge M_{i,1} \in Q] \\
&\geq \epsilon_{\mathcal{R}} - \frac{r \cdot \exp(-1)}{q},
\end{aligned}$$

where $M_{i,0}$ and $M_{i,1}$ are the sequences of chosen-message queries issued by the simulated forger in the i-th execution of $\mathcal{F}$.