How Risky is the Random-Oracle Model?

Gaëtan Leurent* and Phong Q. Nguyen†

October 13, 2008

Abstract

RSA-FDH and many other schemes provably secure in the Random-Oracle Model (ROM) require a cryptographic hash function whose output size does not match any of the standard hash functions. We show that the random-oracle instantiations proposed in the literature for such general cases are insecure, including the two historical instantiations proposed by Bellare and Rogaway themselves in their seminal papers from ACM CCS '93 and EUROCRYPT '96: for instance, for 1024-bit digests, we present a 2^{68} preimage attack on BR93 and a 2^{106} collision attack on BR96. This leads us to study the potential security impact of such defects. While one might think that a hash collision may at worst give rise to an existential forgery on a signature scheme, we show that for several (real-world) schemes secure in the ROM, collisions or slight hash function defects can have much more dramatic consequences, namely key-recovery attacks. For instance, we point out that a hash collision discloses the master key in the Boneh-Gentry-Hamburg identity-based cryptosystem from FOCS '07, and the secret key in the Rabin-Williams signature scheme for which Bernstein proved tight security at EUROCRYPT '08. This problem can be fixed, but still, such schemes, as well as the Rabin-Williams variant implemented in the IEEE P1363 standard, strongly require that the hash function is immune to malleability variants of collision attacks, which does not hold for the BR93 instantiation. Our results suggest an additional criterion to compare schemes secure in the ROM: assessing the risks by carefully studying the impact of potential flaws in the random-oracle instantiation. In this light, RSA-PSS seems more robust than other RSA signatures secure in the ROM.
Keywords: Cryptanalysis, Hash Function, Random Oracle, Provable Security, Signature, Full-Domain Hash, PSS, Rabin, Rabin-Williams, RSA.

1 Introduction

The Random-Oracle Model (ROM) goes back to at least Fiat and Shamir [27]. It was popularized and made into an explicit design paradigm by Bellare and Rogaway [7]. Roughly speaking, a security proof in the ROM is a security proof in which the underlying hash functions are modeled as random oracles. In some sense, the design of schemes based on the ROM is a trade-off between schemes that are provably secure but hopelessly impractical and very efficient schemes for which no security property is known. Today, the ROM is one of the most controversial issues in cryptographic research. On the one hand, the ROM is in widespread use in both research papers and standards, because such schemes are usually more efficient, can be based on well-studied computational problems, and sometimes, their security proof can even be tight. In fact, many public-key

*DGA and ENS, France. http://www.eleves.ens.fr/leurent/
†INRIA and ENS, France. http://www.di.ens.fr/~pnguyen/

cryptographic schemes used in practice are only proven secure in the ROM, *e.g.* RSA-OAEP [8] and RSA-PSS [9]. On the other hand, many researchers have expressed doubts about the wisdom of relying on the ROM. In particular, Canetti, Goldreich and Halevi [17] proved ten years ago that there are signature and encryption schemes which are secure in the ROM, but insecure for any instantiation of the random oracle. However, all the known constructions [17, 31, 4] showing the limitations of the ROM are arguably "unnatural" and significantly differ from real-world constructions. This has led Koblitz and Menezes [41] to claim that "our confidence in the random oracle assumption is unshaken".

Two trends have emerged in the past ten years. One trend is to keep on improving the security guarantees offered by proofs in the ROM, by providing tighter and tighter security proofs, which means that any potential attack could be used to efficiently solve the underlying hard problem with essentially the same success probability. In fact, the promise of tight security proofs has always been one of the key selling points of the ROM (see [7, 9]). The other trend is to get away from the ROM, by designing alternative solutions in the standard model (such as in [25]): in particular, with the development of pairing-based cryptography [14], new schemes provably secure in the standard model under ad-hoc assumptions based on pairings have appeared (such as [16, 12, 13, 26]).

Recent breakthroughs in the cryptanalysis of hash functions [53, 52, 49] have shown that standard hash functions like MD5 or SHA-1 are far from behaving like random oracles. Yet, these results have so far had surprisingly little impact on the public-key world, including widespread public-key schemes proved in the ROM. The only public-key application known so far seems to be [49], which constructs two colliding X.509 certificates for different identities and public keys. Four years after the discovery of the first MD5 collision [53], the lack of serious attack on RSA-OAEP or RSA-PSS may paradoxically reinforce the ROM.

This stresses the importance of studying the actual security (in the standard model) of schemes provably secure in the ROM. There can be significant differences between the real world and an idealized security model, as the MD5 case illustrates: like all Merkle-Damgård hash functions based on Davies-Meyer compression functions, MD5 was provably collision-resistant [55, 11] (up to the birthday bound) in the ideal cipher model (with respect to the block cipher underlying the compression function). Yet, the MD5 block cipher turned out to be so far from an ideal cipher that computing MD5 collisions only costs a few seconds now [53, 39]. However, not all applications of MD5 are under threat: for instance, though the MD5 compression function does not seem to satisfy any of the assumptions required by HMAC security proofs [5, 3], only very theoretical attacks on HMAC-MD5 are known [19, 28, 51].

But in order to study the actual security, it is essential to know how the random oracle will be instantiated in practice, should the scheme ever be used. Often, the output size of the required random oracle matches that of standard cryptographic hash functions (like 160 bits for SHA-1). In this case, standard hash functions are most likely to be used in practice, despite well-known extension properties of MD-iterated hash functions (such as the derivation of $h(m_1||m_2)$ from $h(m_1)$ and m_2) which make them easily differentiable from a random oracle. But RSA-FDH [9] and many other schemes secure in the ROM (such as [34, 21, 22, 37, 10, 29, 15]) actually require a random oracle with "non-standard" output. First, the output may not be a uniformally distributed bitstring: it could be integers mod N like in RSA-FDH [9], or elliptic curve points like in pairing-based crypto, *etc.*, fortunately there are well-known tricks to deal with such situations provided that one has access to an instantiation with arbitrary output $\{0,1\}^n$. However, if the required output bit-length does not match any of the standard cryptographic hash functions, as is the case

of RSA-FDH which needs at least 1024 bits, it is unclear how the oracle will be instantiated in practice. This is related to the *output domain extension* problem, which asks how to extend the output of a good hash function.

Unfortunately, this issue does not seem to be well adressed in the literature: Katz and Lindell devote a chapter to the ROM in their recent textbook [36], but warn that a detailed discussion of how to instantiate a random oracle in practice is beyond the scope of the book. To the best of our knowledge, the only proposals of random-oracle instantiations supporting arbitrary outbit bit-length are the following: two historical instantiations by Bellare and Rogaway proposed respectively in their seminal papers [7] (on the ROM) and [9] (on RSA-FDH and RSA-PSS), recent constructions by Coron *et al.* in the full version [24] of [23], and the instantiations implicit in the PKCS#1 v2.1 [48] and IEEE P1363 [33] standards. Surprisingly, despite the importance of [7, 9], it seems that none of these instantiations have been analyzed in the literature, except that [24] has been analyzed in the indifferentiability framework of Maurer *et al.* [43].

This raises the question of the impact of potential defects in random-oracle instantiations. When a research article provides a security proof in the ROM, it usually does not say how to instantiate the random oracle, neither what might happen if the hash function is not a random oracle. Assume that Alice implements a scheme secure in the ROM under a well-known computational assumption. Several years later, the assumption still stands, but Alice learns that her random-oracle implementation is not as perfect as she thought: Should Alice worry? Are the risks confined to chosen-message existential forgery and ciphertext distinguishability, or are there cases where Alice could be in serious trouble? If Alice had the choice between two (equally efficient) schemes secure in the ROM under the same assumption, maybe Alice would rather choose the least risky one, in terms of robustness to defects in the hash function.

Our Results. We analyze for the first time all the concrete proposals [7, 9, 48, 33] of random-oracle instantiations supporting arbitrary outbit bit-length, and show that they are insecure, notably those proposed by Bellare and Rogaway themselves in [7, 9], back in 1993 and 1996. For instance, for 1024-bit digests, we give a 2^{68} preimage attack on the BR93 instantiation [7] and a 2^{106} collision attack on the BR96 instantiation [9]. It should be stressed that none of these instantiations made it clear what was exactly the expected security guarantee, but one might argue that a random-oracle instantiation with output bit-length n should offer $2^{n/2}$ resistance to collisions, and 2^n resistance to preimages. What we show is that the proposals fall short of those bounds. We note that the instantiations implicit in PKCS [48] and IEEE [33] standards are not collision-resistant: collisions follow directly from collisions on SHA-1. And we show that when applied to the compression functions of MD5 or SHA-1, the theoretical constructions of Coron *et al.* [24] are no more collision-resistant than MD5 or SHA-1 themselves.

Next, we study the impact on schemes secure in the ROM. While it is often believed that a hash collision may at worst give rise to an existential forgery on a signature scheme, we show that for several provably secure schemes proposed in the literature [9, 34, 15, 10, 29], collisions or slight hash function defects can have much more dramatic consequences, namely key-recovery attacks. Our most interesting examples are related to Rabin and Rabin-Williams signatures. For instance, we remark that a hash collision discloses the master key in the Boneh-Gentry-Hamburg identity-based cryptosystem [15], and the secret key in the Rabin-Williams signature scheme for which Bernstein [10] recently proved tight security, which was not mentioned in [15, 10]. Since this might not be desirable, we show how to fix this problem (oblivious to the security proof), but even then, it should be noted that these schemes, as well as another Rabin-Williams variant

included in the IEEE P1363 standard [33], require that the hash function be immune to malleability variants of collision attacks, which is for instance not the case of BR93 [7]. This means that the Rabin-Williams signature included in IEEE P1363 would have been insecure if the standard had implemented the BR93 instantiation, or if it had replaced the signature by its variant [10] (since the random-oracle instantiation of IEEE P1363 is not collision-resistant). These issues are not restricted to factoring-based schemes: we show similar problems for a recent provably-secure lattice-based signature scheme [29].

Our results suggest an additional criterion to compare schemes secure in the ROM: assessing the risks by carefully studying the impact of potential flaws in the random-oracle instantiation. Along these lines, we also study the main RSA signature schemes provably secure in the ROM: RSA-FDH [9], RSA-PFDH [21], RSA-PSS [9], and RSA-KW [37]. While it is well-known that RSA-PFDH, RSA-PSS and RSA-KW all offer a tight security proof, it seems that RSA-PSS is more robust with respect to potential defects in the random-oracle instantiation.

Road map. We assume the reader is familiar with hash functions, the random-oracle model [7] and provable security for signatures [32]. The paper is organized as follows. In Section 2, we recall and attack the random-oracle instantiations that have been proposed in the literature for arbitrary output size. Next, we study the implications of such defects for several secure signature schemes. In Section 3, we study Rabin signatures, more precisely the ID-based cryptosystem of [15] and PRab [9]. In Section 4, we study Rabin-Williams signatures, more precisely the scheme [34] implemented in the IEEE P1363 standard [33] and its variant analyzed in [10]. In Section 5, we study RSA signatures. In Appendix A, we study a different kind of scheme: the GPV lattice-based signature scheme [29].

2 Random-Oracle Instantiations for Arbitrary Output

In this section, we present and analyze the random-oracle instantiations supporting arbitrary output bit-length that have been proposed in the literature, namely [7, 9, 24] and the instantiations implicit in the PKCS#1 v2.1 [48] and IEEE P1363 [33] standards. For completeness, we also briefly discuss provably collision-resistant hash functions such as [18, 42].

We note that some of the instantiations make use of MD5, but that alone is insufficient to discard them. Indeed, though the collision-resistance of MD5 is seriously broken [53, 39], many usages of MD5 are not threatened yet: for instance, there is still no practical attack on HMAC-MD5.

2.1 The 1993 Instantiation by Bellare and Rogaway

Description. In their seminal paper on the random-oracle methodology [7], Bellare and Rogaway actually proposed several guidelines to instantiate a random oracle (see [7, Section 6]), without committing too much: it seemed like any reasonable choice should work in practice. In fact, the only explicit construction given in [7, Section 6] is the following one, which we call BR93:

- Let $h_4 : \{0,1\}^{512} \to \{0,1\}^{128}$ be the first compression function of MD5, that is the compression function evaluated with the initial IV of MD5.
- Let $h' : \{0,1\}^{256} \to \{0,1\}^{64}$ defined by $h'(x)$ being the first 64 bits of $h_4((xx) \oplus C)$, for a randomly chosen 512-bit constant C. The function h' defines a pseudo-random number gener-

ator $h''(x) : \{0,1\}^{224} \to \{0,1\}^*$ by counter as follows: $h''(x) = h'(x\langle 0\rangle)||h'(x\langle 1\rangle)||h'(x\langle 2\rangle)\ldots$ where $\langle i\rangle$ is the encoding of i into 32 bits[1].

- Finally, the BR93 instantiation of the random oracle is the truncation (prefix) of $h(x) : \{0,1\}^* \to \{0,1\}^*$ defined as follows. First, one applies a padding to x by adding a bit 1 and enough bits 0 to obtain a bitstring x' whose bit-length is a multiple of 128. Then, if we divide x' into 128-bit blocks as $x' = x'_0 \ldots x'_{n-1}$, then $h(x) = h''(x'_0\langle 0\rangle) \oplus h''(x'_1\langle 1\rangle) \oplus \cdots \oplus h''(x'_{n-1}\langle n-1\rangle)$, that is, $h(x)$ is the XOR of the n streams produced by each of the x'_i.

Weaknesses. We show that BR93 is insecure with respect to collision resistance and preimage resistance, independently of the choice of the underlying hash function (MD5 here). Our attacks are based on Wagner's generalized birthday algorithm [50], which we now recall. The basic operation of the algorithm is the general join $\bowtie_j$: $L \bowtie_j L'$, which consists of all elements of $L \times L'$ such that their j least significant bits match:

$$L \bowtie_j L' = \left\{ l \oplus l' : (l, l') \in L \times L' \;\middle|\; (l \oplus l')^{[0..j-1]} = 0^j \right\}.$$

Assume that we are given several lists $L_0, L_1, ...$, each of size 2^r. Our goal is to find $l_0 \in L_0, l_1 \in L_1, ...$ such that $\bigoplus l_i = 0$. The idea is to join the lists using a binary tree. We build the first level with $L_{01} = L_0 \bowtie_r L_1$, $L_{23} = L_2 \bowtie_r L_3$, and so on. By the birthday paradox, these new lists should still contain about 2^r elements. On the next level, we build $L_{0123} = L_{01} \bowtie_{2r} L_{23}$. Since the elements of L_{01} and L_{23} already agree on their r lower bits, we are only doing a birthday paradox on the bits r to $2r$, so we still expect to find 2^r elements. If we start with 2^k lists, on the last level we end up with one list of 2^r elements which all begin with kr zeros. In this list, we expect to find two elements that agree on $2r$ extra bits, so that we have a collision on $(k+2)r$ bits. The algorithm for $k = 2$ is described by Figure 1.

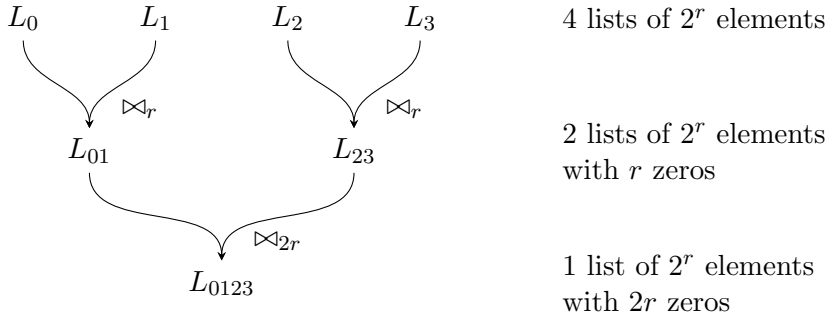

Figure 1: Wagner's algorithm for $k = 2$

We now use Wagner's algorithm to find collisions on BR93. For each message block x'_i, we will consider 2^r possible values, and build a list L_i with the resulting $h''(x'_i\langle i\rangle)$. Then we can use Wagner's attack on these lists. A collision attack on $(k+2)r$ bits will have a complexity of $2^k \cdot r2^r$ using messages of 2^k blocks. For instance, a collision attack on 1024 bits with messages of 2^{30} blocks costs $2^{30} \cdot 32 \cdot 2^{32} = 2^{67}$ elementary operations. If we limit the message size to 2^{14} blocks,

[1]The paper [7] actually says 64 bits, but that would be incompatible with the definition of h' as $224 + 64 = 288 > 256$.

the complexity is $2^{14} \cdot 64 \cdot 2^{64} = 2^{84}$. Note that this complexity does not depend on the size of the underlying hash function.

We can also use Wagner's algorithm to find preimages on BR93. If we want a preimage of H, we first replace L_0 by $L_0 \oplus H$. On the last level of the tree, we will still have one list of 2^r elements which all begins with kr zeros, but instead of looking for a collision on $2r$ extra bits, we look for an element with r extra zeroes. This element corresponds to a message x such that $H \oplus h(x) = H \oplus h''(x'_0\langle 0\rangle) \oplus h''(x'_1\langle 1\rangle) \oplus ...h''(x'_{2^k-1}\langle 2^k - 1\rangle) =_{(k+1)r} 0$, *i.e.* the $(k+1)r$ first bits of $h(x)$ agree with H. A preimage attack on $(k+1)r$ bits will have a complexity of $2^k \cdot r2^r$ using messages of 2^k blocks. A preimage attack on 1024 bits with messages of 2^{31} blocks costs $2^{31} \cdot 32 \cdot 2^{32} = 2^{68}$ elementary operations.

2.2 The 1996 Instantiation by Bellare and Rogaway

Description. In their paper [9] on PSS, Bellare and Rogaway proposed another instantiation [9, Appendix A], which we call BR96:

- Let H = MD5 or SHA-1.
- Define $h_{BR96}(x)$ as the appropriate truncation (prefix) of:

$$H(\texttt{const}\langle 0\rangle x)||H(\texttt{const}\langle 1\rangle x)||H(\texttt{const}\langle 2\rangle x)||\ldots$$

  where the constant `const` should be unique to h. If another instantiation is needed, one should change `const`.

Weaknesses. First of all, we note that BR96 can easily be distinguished from a random oracle. More precisely, BR96 suffers from the same extension problems as any MD function: if the output size of h_{BR96} is an exact multiple of that of H, then $h_{BR96}(m_1||m_2)$ can be computed from $h_{BR96}(m_1)$ and m_2.

More importantly, we show that BR96 is insecure with respect to collision resistance and preimage resistance, independently of the choice of the underlying hash function (except the output size), thanks to Joux's multicollision technique [35], and we show how to (slightly) improve the results of [35]. This might come as a surprise, since [35, Sect. 5] stressed: "we were not able to find a single research paper that can be cryptanalyzed using the attacks presented here". Recall that [35] showed that the concatenation of two or more MD-iterated hash functions is not as secure as one might have expected: the security is essentially the same as that of a single hash function. More precisely, it was proved that if one concatenates k iterated hash functions with an internal size of n bits each, one can find a collision in the concatenation for a workload of $n^{k-1} \times 2^{n/2}$, and preimages for a workload of $\mathsf{poly}(n^k)2^n$.

In fact, these figures given in [35] are a bit conservative: a closer analysis reveals that the collision cost $n^{k-1} \times 2^{n/2}$ can be reduced to $[(n/2)^{k-1} + (n/2)^{k-2} + \cdots + 1] \times 2^{n/2} \leq (n/2)^{k-1} \times (n/2)/(n/2-1) \times 2^{n/2} \approx (n/2)^{k-1} \times 2^{n/2}$.

As for preimages, Joux described in [35, Appendix A] an attack on the concatenation of k iterated hash functions based on fixed points, whose exact cost was not precisely given: the cost was estimated to $\mathsf{poly}(n^k)2^n$. However, there seems to be a more efficient attack by generalizing the basic preimage attack against two hash functions as follows. First, build a $2^{n^{k-1}/2^{k-2}}$-multicollision on the first hash function F_1, and look for an extra block that maps this multicollision to the

target value of F_1. Then build a multicollision in F_2 using the messages of the first multicollision: each collision in F_2 requires a set of $2^{n/2}$ messages, which will be built from $n/2$ colliding pairs in F_1. Thus we should get a $2^{n^{k-2}}/2^{k-3}$-multicollision in F_1. We will also use the last n colliding pairs for a preimage search on F_1. This gives us a $2^{n^{k-2}}/2^{k-3}$-multicollision in $F_1||F_2$ which is also a preimage. We apply the technique iteratively to build a 2^n-multicollision for $F_1||F_2||...F_{k-1}$ which is also a preimage. If we compute F_k on the set of 2^n colliding messages, we expect to find one preimage against the full concatenation. The most expensive steps of this attack are the preimage search, because the collision finding steps all have a complexity which is $O(n^k \times 2^{n/2})$. The preimage step on F_i will require to compute F_i on 2^n message, which are made from n block pairs of length $n^{i-2}/2^{i-2}$ and one block of length $n^{i-2}/2i-3$. If we do an amortized analysis, each computation require to hash 2 blocks from message pairs, and the final block, which gives a cost of $n^{i-2}/2^{i-4} \times 2^n$. The cost of the full preimage search is roughly equivalent to the cost of the last preimage search, which is $n^{k-2}/2^{k-4} \times 2^n$.

We apply this result to BR96, by considering each of the $H_i : x \mapsto H(\texttt{const}\langle i\rangle x)$ as a distinct iterative hash function. For instance, if H is MD5, we can find collisions in 1024 bits of the output with a workload of essentially $64^7 \cdot 2^{64} = 2^{106}$, where the colliding messages will be of length $64^7 = 2^{42}$ blocks; and we can find preimages of 1024 bits of the output with a workload of $128^6/2^4 \cdot 2^{128} = 2^{166}$. Though impractical, these complexities are far lower than the theoretical security of a 1024-bit random oracle.

The BR96 construction is also malleable. Again with the multicollision technique, we can create pairs of messages x_0, x_1 such that $H(\texttt{const}\langle i\rangle x_0) = H(\texttt{const}\langle i\rangle x_1)$ for all i's except the last one. We will build a multicollision set of $2^{n/4}$ such messages, and we expect to find one quadruplet such that $H(x_0) \oplus H(x_1) \oplus H(x_2) \oplus H(x_3) = 0$. In Appendix A, we will see that this kind of malleability can be exploited to attack the GPV signature scheme.

2.3 Recent Instantiations by Coron *et al.* (CDMP)

Description. Coron *et al.* [23, 24] (CDMP) proposed several variations of Merkle-Damgård to build a random oracle from an (ideal) compression function or an (ideal) block-cipher using the Davies-Meyer mode. They proposed four variants of MD for *input* domain extensions (namely, *Prefix-Free Encoding*, *Dropping Some Output Bits*, *Using NMAC*, and *Using HMAC*) and one scheme (only in the full version [24]) for *output* domain extension. The output extension scheme is similar to the BR96 construction, but the counter is included after the message (which is reminiscent of the so-called MGF1 pseudo-random number generator used in several standards [33, 48]):

$$h_{CDMP}(x) = H(x\langle 0\rangle)||H(x\langle 1\rangle)||H(x\langle 2\rangle)||\dots$$

(here, H is one of the four input extension schemes). This choice is due to efficiency considerations, but we will see that it has a strong security implication. The main advantage of [23, 24] is its security proof: all the constructions are proved indifferentiable from a random oracle (in the sense of Maurer *et al.* [43]), if the underlying compression function is a random oracle, or if it uses the Davies-Meyer mode with an ideal block cipher. However, no recommendation is given in [23, 24] for the choice of the underlying compression function (or the underlying block cipher for Davies-Meyer). So strictly speaking, unlike [7, 9], there was no fully concrete proposal of a random-oracle instantiation: still, one may want to apply the constructions to usual compression functions.

Weaknesses. One should be careful not to overestimate the significance of indifferentiability security proofs: we currently do not know how to build an ideal compression function, and the security

guarantees on the hash function disappear if the compression function is not ideal. For instance, it was shown by Bellare and Ristenpart in [6] that none of the CDMP constructions necessarily preserve collision-resistance: they give (theoretical) examples of collision-resistant compression functions for which the resulting hash function is not collision-resistant.

We show further problems in the CDMP constructions. While [23, 24] was presented as a fix to the MD construction, we point out that if one applies these fixes to MD5 or SHA-1, one can still find collisions in the new hash function (independently of the chosen output length) with the same cost as the original MD5 or SHA-1. This means that [23, 24] do not address the main vulnerabilities in MD5 or SHA-1.

To see this, we first show that the four input extensions are not collision resistant if applied to the compression functions of MD5 or SHA-1. This is trivial for *Dropping Some Output Bits*, *Using NMAC*, and *Using HMAC*, because these constructions are nested: an inner collision becomes an outer collision. So the only potentially tricky case is *Prefix-Free Encoding*, for which [23, 24] proposed only two instantiations:

- prepend the message size as the first block. It turns out that MD5/SHA-1 collision attacks [53, 52] can be extended to this case, because the number of blocks of colliding messages produced is equal and already known in advance, and it is well-known that existing MD5/SHA-1 collision attacks can be extended to any given IV.

- use the first bit of each message block as a flag to distinguish the last message block. Since the number of blocks in MD5/SHA-1 colliding messages is very small, and the first bit of each block is random looking, we can simply produce random collisions until one has the required form.

Now, because of the iterated structure of the four input extensions, these collisions give rise to collisions in the output extension h_{CDMP}. More generally, while h_{CDMP} is indifferentiable from a random oracle if H is, any collision in H becomes a collision in h_{CDMP} if H has an iterative structure like MD or the four input extensions: namely, $H(x_0) = H(x_1)$ implies $H(x_0\langle i\rangle) = H(x_1\langle i\rangle)$ and therefore $h_{CDMP}(x_0) = h_{CDMP}(x_1)$.

Hence, we have shown that if the CDMP constructions are applied to the compression functions of MD5 or SHA-1 for an arbitrary output size, the cost of producing collisions remains essentially the same as in MD5 or SHA-1, even though the constructions are indifferentiable from a random oracle under the assumption that the underlying block cipher of MD5 or SHA-1 is an ideal cipher. Of course, one could try to apply the CDMP constructions with different components, but it should perhaps be noted that [23, 24] do not make any concrete recommendation for the compression function.

2.4 Instantiations in PKCS and IEEE Standards

Description. No cryptographic standard currently specifies a random oracle instantiation for arbitrary size. However, several instantiations are implicit in PKCS #1 v2.1 [48] and IEEE P1363 [33], because RSA-OAEP [8] and RSA-PSS [9] are standardized:

- RSA-OAEP requires two random oracles G and H with small input size (less than the RSA modulus), which are both instantiated in PKCS [48] by the MGF1 pseudo-random number generator [48]. Recall that MGF1 is simply a hash function in counter mode like h_{CDMP},

except that the counter is over four bytes: $\mathrm{MGF1}(x) = h(x\langle 0\rangle)||h(x\langle 1\rangle)||h(x\langle 2\rangle)||\ldots$, where h is either SHA-1 or a SHA-2.

- RSA-PSS also requires two random oracles G and H, but while G still has small input size, H has a small output size but possibly large inputs. In PKCS [48], H is instantiated by SHA-1 or SHA-2, and G is instantiated by MGF1.

Thus, none of the oracles required by RSA-OAEP and RSA-PSS have both a large input and output as would be required by for instance RSA-FDH. Still, MGF1 is a potential random-oracle instantiation, because it supports arbitrarily large input and output.

There is another implicit instantiation in IEEE P1363 [33]. Indeed, the Rabin-Williams signature scheme implemented in IEEE P1363 uses a variant of the PSS encoding [9] (as described in [34]) called EMSA-PSS in [48] and EMSA4 in [33] (see Figure 2: the main difference between EMSA-PSS and PSS [9] is that the message is first hashed before going through the PSS encoding) but it is specified in [33] that the salt can optionally be set to zero, in which case "*the signature scheme is deterministic, similar to Full-Domain Hashing*". Thus, one can view EMSA-PSS with

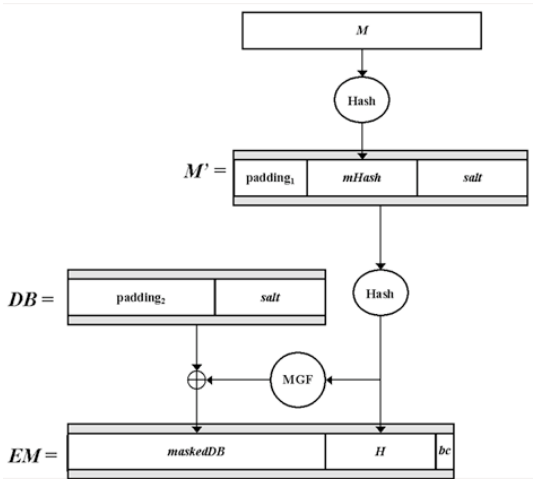

Figure 2: The EMSA-PSS encoding [33]

zero salt as an instantiation of a FDH: since the padding constants are zero, this amounts to essentially hash the message twice in a row, then apply MGF1; concatenate the output and the input of MGF1, and append the "BC" byte.

Weaknesses. The case of MGF1 has already been analyzed with the CDMP case in the previous subsection: using SHA-1 or any MD-iterated hash function, the cost of producing collisions in MGF1 remains as low as for the underlying hash function. And EMSA-PSS with zero salt is clearly no more collision-resistant than the underlying hash function. Of course, the "BC" byte makes it differentiable from a random oracle.

Hence, none of the instantiations in PKCS and IEEE standards can be considered collision-resistant with MD5 or SHA-1 as the underlying hash function.

2.5 Provably secure hash functions

To conclude this section, we briefly mention the case of hash functions which are provably collision-resistant under appropriate computational assumptions. Though they are not claimed to be random

oracles, they might be potential candidates since they usually support large output size, But it is folklore that none should be viewed nor used as a random oracle, because they all have special properties which are not satisfied by a random oracle, typically malleability. Consider for instance two recent collision-resistant hash functions:

- VSH [18], which is provably collision-resistant, provided that a certain problem related to integer factorization is hard. The output set is $\mathbb{Z}_N^\times$, where N is an integer hard to factor.
- SWIFFT [42], which is provably (asymptotically) collision-resistant and one-way, provided that certain lattice approximation problems are hard. The smallest output size is 528 bits, but larger sizes are possible.

These hash functions are easily malleable in the following sense. In [42], it is noted that for any two inputs x_1 and x_2 such that $x_1 + x_2$ is a valid input, we have $SWIFFT(x_1) + SWIFFT(x_2) = SWIFFT(x_1 + x_2)$. By definition of VSH [18], if N is the public modulus, it is trivial to generate two distinct messages M_0 and M_1 such that $4VSH(M_0) \equiv VSH(M_1) \pmod N$. More generally, for any product $s > 1$ of distinct very small primes (chosen among the primes used by the VSH compression function), it is easy to generate two distinct messages M_0 and M_1 such that $s^2 VSH(M_0) \equiv VSH(M_1) \pmod N$.

Curiously, we will see as a side result of subsequent sections that these malleability relationships are exactly the kind of properties which can be exploited to attack several signature schemes. For instance, the malleability of SWIFFT can be exploited to attack the GPV signature scheme [29], and the malleability of VSH can be exploited to attack Rabin and Rabin-Williams signatures. This is yet another warning that one should be very careful when plugging provably-secure primitives.

3 Security Robustness of Rabin Signatures

In 1979, Rabin [47] introduced the following signature scheme. Let $N = pq$ be an RSA-modulus, $h : \{0,1\}^* \to \mathbb{Z}_N$ be a hash function and $k \geq 1$ be an integer. To sign a message m, pick $r \in \{0,1\}^k$ uniformly at random until $h(m||r)$ is a quadratic residue mod N. Then select $s \in \mathbb{Z}_N$ uniformly at random among all the four square roots of $h(m||r)$. The signature of m is the pair (s, r). In modern terminology, this process can be viewed as a probabilistic-full-domain-hash (PFDH) [21], which was here necessary because the hash function h does not only map to quadratic residues.

There are now many variations of Rabin signatures which are provably secure in the ROM. In this section, we consider two of those, and study the potential impact of defects in the oracle instantiation. First, we consider the recent identity-based cryptosystem [15] by Boneh *et al.*, which uses secure Rabin signatures for key generation. While this cryptosystem is provably secure in the ROM, we notice that the scheme described in [15] is not tolerant to collisions in the hash function: namely, any hash collision gives rise to a very efficient chosen-ID attack which recovers the master key. This problem can be fixed, but the attack also shows that other malleability defects in the oracle would be equally deadly.

Next, we consider PRab [9], which is a PSS version of Rabin signatures by Bellare and Rogaway: we give a converse to the security proof of [9], namely that if the parameters are selected in such a way that the security reduction is not tight, then there is an efficient key-recovery attack on PRab. And we show how to improve this attack if there are defects in the oracle instantiation. Thus, in this case, it is extremely important to select parameters carefully so that the security proof is tight.

3.1 Rabin Signatures in the Boneh-Gentry-Hamburg ID-based scheme

Boneh *et al.* [15] recently presented an identity-based encryption without pairings, which uses Rabin signatures in the key generation process as follows.

Let $N = pq$ be an RSA modulus, and denote by $J(N)$ the subgroup of $\mathbb{Z}_N^\times$ of elements with Jacobi symbol equal to 1. Denote by $\mathrm{QR}(N)$ the subgroup of $J(N)$ of quadratic residues. The integer ℓ defines the bit-length of messages, and $\mathcal{ID}$ denotes the set of identities. The public parameters are (N, u, H) where u is chosen with uniform distribution in $J(N) \setminus \mathrm{QR}(N)$, and $H : \mathcal{ID} \times [1, \ell] \to J(N)$ is a hash function. The master key is the factorization of N and a random key K for a pseudorandom function $F_K : \mathcal{ID} \times [1, \ell] \to \{0, 1, 2, 3\}$.

The key generation is as follows. Given as input the master key and an identity ID, for each $j = 1, \ldots, \ell$ do: $R_j \leftarrow H(\mathrm{ID}, j) \in J(N)$ and $w \leftarrow F_K(\mathrm{ID}, j)$; let $a \in \{0, 1\}$ be such that $u^a R_j \in \mathrm{QR}(N)$; set $r_j \leftarrow z_w$ where $\{z_0, \ldots, z_3\}$ are the four square roots of $u^a R_j$ in $\mathbb{Z}_N$. The decryption key corresponding to ID is $(r_1, \ldots, r_\ell)$ together with the public parameters. The PRF F ensures that the key generator always outputs the same square roots for a given ID, but an adversary cannot tell ahead of time which of the four square roots will be output.

While this ID-based cryptosystem is provably secure in the ROM (see [15] for a proof and a full description of the scheme), we point out that it is not tolerant to hash collisions. Assume indeed that an adversary knows two pairs $(\mathrm{ID}_1, j_1) \neq (\mathrm{ID}_2, j_2)$ such that $H(\mathrm{ID}_1, j_1) = H(\mathrm{ID}_2, j_2)$. In a chosen-ID attack, we can retrieve the the secret keys corresponding to the identities ID_1 and ID_2. But because the pseudo-random function F is independent of H, we will thus obtain two (independent) random square roots of the same quadratic residue, which discloses the factorization of the master key with probability 1/2. To prevent this attack, one can modify the scheme by replacing $w \leftarrow F_K(\mathrm{ID}, j)$ with $w \leftarrow F_K(H(\mathrm{ID}, j))$ (and modifying the input domain of F_K accordingly): it seems that [15] chose $w \leftarrow F_K(\mathrm{ID}, j)$ to simplify the security proof. However, even with the fix, the scheme is still vulnerable to malleability variants of collisions. More precisely, one can easily extend the key-recovery attack to the more general case where one knows two pairs $(\mathrm{ID}_1, j_1) \neq (\mathrm{ID}_2, j_2)$ such that $H(\mathrm{ID}_1, j_1) \equiv k^2 H(\mathrm{ID}_2, j_2) \pmod N$ where k is known.

3.2 A Converse to the Security Proof of PRab

When introducing PSS, Bellare and Rogaway [9] also showed how to implement Rabin signatures [47] with a security proof in the ROM: the resulting scheme, called PRab in [9], is essentially Rabin-PSS, which we now describe. Let $N = pq$ be the Rabin public key of bit-length $k \geq k_0 + k_1$. Two hash functions $h : \{0,1\}^* \to \{0,1\}^{k_1}$ and $g : \{0,1\}^{k_1} \to \{0,1\}^{k-k_1}$ are used. We let g_1 be the function which on input $w \in \{0,1\}^{k_1}$ returns the first k_0 bits of $g(w)$, and let g_2 be the function which on input $w \in \{0,1\}^{k_1}$ returns the remaining $k - k_0 - k_1$ bits of $g(w)$. To sign a message $M \in \{0,1\}^*$, one first repeats the following process until y is a quadratic residue mod N:

1. Choose $r \in_R \{0,1\}^{k_0}$ and let $w = h(M\|r)$ and $r^* = g_1(w) \oplus r$;
2. Let $y = w\|r^*\|g_2(w)$.

Then the signature s is chosen uniformly at random among the four distinct square roots of $y \in \mathbb{Z}_N^\times$. Bellare and Rogaway [9] showed the security of PRab in the ROM, under the hardness of factoring N, and the tightness of their security proof was similar to that of their proof for RSA-PSS. More precisely, if ε and ε' denote the advantages for breaking respectively PRab and factoring, then:

$$\varepsilon = 2\varepsilon' + [4(q_{sig} + q_{hash})^2 + 2] \cdot (2^{-k_0} + 2^{-k_1}), \tag{1}$$

where q_{sig} and q_{hash} denote respectively the numbers of signature queries and hash queries. Under appropriate choices of k_0 and k_1, the right-hand term becomes negligible with respect to ε', in which case the security proof is said to be tight.

We now show that it is extremely important to select k_0 to make the security proof tight (see [40] for a discussion on the importance of tightness). Consider first the following elementary known-message key-recovery attack:

- Collect n signatures $s_1, \ldots, s_n$ of random messages $M_1, \ldots, M_n$. Denote by r_i the (unknown) random number used by the signer to generate s_i.
- Since the output of h is $\{0,1\}^{k_1}$ and $r \in_R \{0,1\}^{k_0}$, by the birthday paradox, as soon as $n \geq \Omega(2^{(k_0+k_1)/2})$, there should be $i \neq j$ such that $h(M_i||r_i) = h(M_j||r_j)$ and $r_i = r_j$. Such a collision can be detected by looking at all $s_i^2 \pmod N$.
- Then s_i and s_j are two random square roots of the same $y \in \mathbb{Z}_N^\times$. This is because the signing process of PRab does not check whether the same y has been "signed" before. Hence $\gcd(s_i - s_j, N)$ is a non-trivial factor of N with probability $1/2$.

This gives a generic attack whose cost is essentially $2^{(k_0+k_1)/2}$ signatures of random messages.

This attack can be improved in the chosen-message setting. If we submit $\Theta(2^{k_0/2})$ times the same message M to the signing oracle, we will obtain a collision on the r_i's, and the attack still applies. This chosen-message attack costs $O(2^{k_0/2})$ signature queries (on the same message), which proves that the security reduction of [9] summarized by Eq. (1) is essentially tight in a different sense: as soon as $q_{sig}^2 \geq \Omega(2^{k_0})$, an adversary can actually recover the factorization of N. In particular, if one wrongly applies Coron's security result [21] for RSA-PSS to PRab, one might believe that $k_0 = 30$ is a good choice, in which case there is an attack requiring only 2^{15} signature queries on the same message.

Note that if h is a Merkle-Damgård iterated hash function, the adversary does not even have to submit the same message to the signing oracle: he may even restrict to distinct messages. Indeed, by using Joux's multicollision technique [35], the adversary can generate n distinct messages $M_1, \ldots, M_n$ such that $h(M_i) = h(M_j)$ and all the M_i's have the same number of blocks. Then, by definition of Merkle-Damgård , if $r_i = r_j$, we have $h(M_i||r_i) = h(M_j||r_j)$, so we can still apply the attack. The cost of generating n multicollisions is $O(2^{k_1/2} \log n)$ (see [35]), which can become as low as $O(\log n)$ if it is easy to generate chosen-IV collisions (such as is the case for MD5 [53]). However, we still need to make $O(2^{k_0/2})$ signature queries. The attack becomes particularly efficient if h is MD5, and the size k_0 of the random salt is small.

4 Security Robustness of Rabin-Williams Signatures

We now turn our attention to Rabin-Williams signatures, which are variants of Rabin signatures based on tweaks [54]. We focus on what are arguably the most interesting provably-secure Rabin-Williams schemes: the scheme implemented in the IEEE P1363 standard [33], and the scheme for which Bernstein [10] proved tight security at EUROCRYPT '08, without requiring any randomization à la PSS. We call the first scheme IRW (for IEEE Rabin-Williams) and the second scheme TRW (for "Tight" Rabin-Williams).

4.1 Description of Rabin-Williams signatures

We describe both TRW and IRW: in the terminology of Bernstein [10], TRW is called "fixed unstructured $B = 0$", while IRW is called "| Principal |". Let $N = pq$ be a Rabin public key with the special requirement that $p \equiv 3 \pmod 8$ and $q \equiv 7 \pmod 8$. TRW and deterministic IRW uses a full-domain hash function $h : \{0,1\}^* \to \mathbb{Z}_N$. To sign a message $M \in \{0,1\}^*$, one applies the following process:

1. First, the message M is transformed into $m \in \mathbb{Z}_N$.
   - In TRW, m is obtained by hashing: $m = h(M) \in \mathbb{Z}_N$ where $h : \{0,1\}^* \to \mathbb{Z}_N$ is a full-domain hash function.
   - In IRW, m is instead obtained by the EMSA-PSS transform (see Figure 2 of Section 2.4). However, IEEE allows both deterministic (with zero salt) and randomized versions (with a parameterized random salt). In other words, IRW-PSS with zero salt can be viewed as an instantiation of IRW-FDH.
2. From the structure of N, there are exactly four triplets $(e, f, s) \in \{-1, 1\} \times \{1, 2\} \times \{0, \ldots, N-1\}$ such that $m \equiv efs^2 \pmod N$, and these so-called tweaked square roots can all be computed using p and q.
   - In TRW, one selects the (e, f, s) triplet uniformly at random, but if ever M has already been signed, one must select the same triplet as chosen previously (see [10, Section 2]). The signature of M is s.
   - In IRW, one computes the so-called *principal* tweaked square root of m, that is, the unique tweaked square root (e, f, s) such that e is 1 if m is a square modulo q, otherwise -1; f is 1 if em is a square modulo p, otherwise 2; and s is a square modulo $N = pq$. This is called the *principal* tweaked square root. The signature of M is the minimum of s and $N - s$.

Regarding Step 2 of TRW, the paper [10] does not say how one can make sure that if given the same message M again, the signer chooses the same signature again. If the implementor mistakenly forgets to do that check, there is a trivial key-recovery attack by simply submitting the same message twice. However, we note that this issue has been discussed by Katz and Wang in [37, Section 4.1] in the context of RSA signatures: they proposed to hash (with an independent random oracle) the concatenation of the secret key and the message M (or alternatively, $m = h(M)$), and uses the result to choose deterministically the triplet (e, f, s). Of course, another solution is to do like in the ID-based cryptosystem [15] discussed in Section 3.1: select (e, f, s) deterministically using a pseudo-random function indexed by an additional secret key. We will see that these subtleties in the implementation can have a major impact on the actual security of the scheme, though they are irrelevant to the security proof.

Both IRW and TRW are provably secure with respect to factoring, in the ROM. More precisely, it was shown in [34] that the security proof of IRW is loose if there is no randomization (zero salt), and tight if the randomization is large, and it was recently shown in [10] that the security proof of TRW is tight.

4.2 Robustness of Rabin-Williams with respect to collisions

We show that TRW is not tolerant to collisions on the hash function. Let $h : \{0,1\}^* \to \mathbb{Z}_N$ be the random-oracle instantiation. We have the following key-recovery attack:

- Assume that the attacker is able to generate a collision (M_0, M_1) on h.
- The attacker queries the signing oracle on M_0 and M_1.
- Let $(e_i, f_i, s_i) \in \{-1, 1\} \times \{1, 2\} \times \{0, \ldots, N-1\}$ be the (selected) tweaked square root of $h(M_i)$, for $0 \leq i \leq 1$.
- Depending on how TRW is exactly implemented, we claim that $(e_i, f_i, s_i)_{0 \leq i \leq 1}$ will be two random tweaked square roots (not necessarily equal) of the same element $h(M_0) = h(M_1) \in \mathbb{Z}_N$, in which case it is easy to obtain the factorization of N with probability $1/2$.

In the exact definition of TRW (see [10]), the signer should select the same tweaked square root if given the same message. However, note that $M_0 \neq M_1$ strictly speaking, which implies that $(e_i, f_i, s_i)_{0 \leq i \leq 1}$ will be two random tweaked square roots (not necessarily equal) of the same element $h(M_0) = h(M_1) \in \mathbb{Z}_N$. Furthermore, if one implements the first Katz-Wang suggestion [37] of hashing the concatenation of the secret key and the message to determine the tweaked square root, then $(e_i, f_i, s_i)_{0 \leq i \leq 1}$ will effectively be two random tweaked square roots of the same element: this is because $h(M_0) = h(M_1)$ does not necessarily imply $h'(S \| M_0) = h'(S \| M_1)$ where S denotes the secret key and h' is another random oracle. Of course, the problem is the same if one applies the solution of the ID-based cryptosystem [15]. However, if one applies instead the second Katz-Wang suggestion [37] of hashing the concatenation of the secret key and $h(M)$, the previous attack disappears.

Interestingly, IRW is immune to this attack, because the choice of (e, f, s) is deterministic for a given m. So, while IRW-FDH has a loose security proof, it is robust with respect to collisions, but TRW may not be robust depending on the way it is implemented. In this sense, IRW-FDH seems more secure than TRW.

It might help to see a concrete example. Assume that the we plug the compression function of MD5 into the CDMP random-oracle construction [24] (see Section 2.4), and that we use this instantiation as a full-domain hash for IRW-FDH and TRW. Then, because of the indifferentiability framework [43], the signature schemes become provably secure under the factoring assumption, in the ideal cipher model (with respect to the MD5 block cipher) or in the random oracle model (with respect to the MD5 compression function). But in practice, there is an instant chosen-message key-recovery attack on this TRW instantiation, which fails on IRW-FDH.

4.3 Robustness of Rabin-Williams with respect to malleability

4.3.1 The case of TRW

We show that TRW is not tolerant to malleability variants of collisions on the hash function. Assume that the attacker is able to generate a pair (M_0, M_1) of distinct messages such that:

$$4h(M_0) \equiv h(M_1) \pmod{N}. \tag{2}$$

From Section 2, we know that this is easy if h is VSH [18], even though it might be hard to find collisions on VSH; and that it is also possible (with more effort) if h is BR93 [7]. Again,

the attacker queries the signing oracle on M_0 and M_1, which gives rise to tweaked square roots $(e_i, f_i, s_i) \in \{-1, 1\} \times \{1, 2\} \times \{0, \ldots, N-1\}$ of $h(M_i)$. Note though that there is a one-to-one correspondance between the four tweaked square roots of $h(M_0)$ and the four tweaked square roots of $h(M_1)$, thanks to the congruence (2). More precisely, if (e, f, s) is a tweaked square root of $h(M_0)$, then $(e, f, 2s \bmod N)$ is a tweaked square root of $h(M_1)$. This implies that $(e_0, f_0, 2s_0 \bmod N)$ and (e_1, f_1, s_1) are two "independent" random tweaked square roots of $h(M_1)$, which means that one can factor N with probability $1/2$.

Note that this attack is independent on the way TRW is implemented regarding signatures of messages with identical digests, using Katz-Wang suggestions [37]. Obviously, the attack can be adapted to other malleability properties. For instance, similar attacks apply if one is able to find a pair (M_0, M_1) of distinct messages such that $h(M_0) \equiv -h(M_1) \pmod N$, or $k^2 h(M_0) \equiv h(M_1) \pmod N$ for some known $k \in \mathbb{Z}_N^\times$.

4.3.2 The case of IRW

We show that the previous attack also applies to IRW-FDH. Starting again from congruence (2), let $(e_i, f_i, s_i) \in \{-1, 1\} \times \{1, 2\} \times \{0, \ldots, N-1\}$ be the principal tweaked square root of $h(M_i)$. Because 4 is a square mod p and q, (2) implies that $e_0 = e_1$ and $f_0 = f_1$. Since $(e_0, f_0, 2s_0 \bmod N)$ is a tweaked square root of $h(M_1)$, we have $4s_0^2 \equiv s_1^2 \pmod N$, and therefore $s_1 \times (2s_0)^{-1} \bmod N$ is a square root of 1 mod N. But it must be a non-trivial square root because it has different Legendre symbols mod p and q: indeed, both s_0 and s_1 are squares mod N, while $\left(\frac{2}{p}\right) = 1$ and $\left(\frac{2}{q}\right) = -1$. Hence, this discloses the factorization of N. This attack can be generalized if congruence (2) is replaced by $k^2 h(M_0) \equiv h(M_1) \pmod N$ for any known $k \in \mathbb{Z}_N^\times$ such that $\left(\frac{k}{p}\right) \neq \left(\frac{k}{q}\right)$, which is slightly more restrictive than the TRW case.

4.4 Robustness of Rabin-Williams with respect to preimages

The previous attacks also show that both TRW and IRW-FDH become strongly insecure if the full-domain hash function h is not one-way, like BR93 [7]. Alternatively, one can simply select $(e, f, s) \in \{-1, 1\} \times \{1, 2\} \times \{0, \ldots, N-1\}$ uniformly at random, and compute $m = efs^2 \pmod N$. By inverting h, one obtains a message M such that $m = h(M)$. Finally, by signing the message M with either TRW or IRW-FDH, one will obtain another tweaked square root of m (principal or not), which will disclose the factorization of N with probability at least $1/2$ because (e, f, s) is a random tweaked square root.

5 Security Robustness of RSA Signatures

The previous sections suggest to look at what happens to the security of RSA signatures, when the hash function is flawed. Since there is a well-known gap between the RSA problem and integer factorization, it seems difficult to hope for a key-recovery attack: instead, we consider the resistance to chosen-message forgeries, either universal or existential. The main RSA signature schemes provably secure in the ROM under the RSA assumption are: RSA-FDH [9], RSA-PFDH [21], RSA-PSS [9], and RSA-KW [37]. It is well-known that RSA-FDH has a loose security proof [20], while RSA-KW has a tight security proof [37], and so do RSA-PFDH and RSA-PSS if the salt is sufficiently large (see [21]).

Let us briefly recall these schemes. The RSA-PSS transform has already been described in Section 3.2 (see also Figure 2 of Section 2.4 for the EMSA-PSS variant). All the other schemes use a full-domain hash $h : \{0,1\}^* \to \mathbb{Z}_N$ where N is the RSA modulus. Let $\sigma : \mathbb{Z}_N \to \mathbb{Z}_N$ denote the RSA raw signature primitive. Let $m \in \{0,1\}^*$ be a message and s its signature. For RSA-FDH, $s = \sigma(h(m))$. For RSA-PFDH, one selects a salt $r \leftarrow_{\mathcal{R}} \{0,1\}^k$, and let $s = (\sigma(h(m||r)), r)$. For RSA-KW, if m has never been signed, one selects a one-bit salt $r \leftarrow_{\mathcal{R}} \{0,1\}$, and let $s = \sigma(h(r||m))$.

The PKCS#1 v2.1 standard [48] uses RSA-PSS since Sept. 1999 (or more precisely, the variant RSA-EMSA-PSS [34] of RSA-PSS), and it has been reported that one of the main reasons why RSA-PSS was selected over RSA-FDH was the tightness of the security proof. If tightness was the main factor, one might now be tempted to select RSA-KW over RSA-PSS, because the salt in RSA-KW is reduced to one bit (which can be deterministically derived from the secret key and the message). However, by comparing the robustness of RSA signatures with respect to potential defects in the random-oracle instantiation, a different picture emerges.

5.1 Robustness with respect to collisions

Because RSA-FDH and RSA-EMSA-PSS are hash-and-sign schemes, they do not tolerate collisions: any collision obviously leads to a chosen-message existential forgery. Similarly, any collision leads to a chosen-message existential forgery on RSA-KW, with probability 1/2 because of the one-bit salt.

One may think that the probabilistic schemes RSA-PFDH and RSA-PSS are more robust. In this direction, Numayama *et al.* [46] showed that RSA-PFDH tolerates collisions in a weakened ROM, but their model does not take into account MD-iterated hash functions. We observe that if h is a MD-iterated hash function, then any collision in h with the same number of blocks gives rise to a chosen-message existential forgery on RSA-PFDH and RSA-PSS. This is because RSA-PFDH and RSA-PSS both use $h(m||r)$. And if $h(m_1) = h(m_2)$ where m_1 and m_2 have the same number of blocks, then $h(m_1||r) = h(m_2||r)$ for any r. This implies that for both RSA-PFDH and RSA-PSS, any signature of m_1 is also valid for m_2. It can be noted that if RSA-PFDH and RSA-PSS had used $h(r||m)$ instead of $h(m||r)$, then the ROM security proofs would remain valid, but the previous attack would fail.

5.2 Robustness with respect to preimages

It is easy to prove that if the full-domain hash is not one-way, then there are chosen-message universal forgery attacks on RSA-FDH, RSA-PFDH and RSA-KW. On the other hand, preimages in h do not seem to provide stronger attacks than chosen-message existential forgeries on RSA-PSS. Hence, among all the ROM-secure RSA signature schemes, it seems that RSA-PSS is the most robust one, with respect to flaws in the random-oracle instantiation.

Acknowledgements. We thank Charles Bouillaguet, Alfred Menezes and Martijn Stam for very helpful discussions and comments.

References

[1] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, pages 601–610 (electronic), New York, 2001. ACM.

[2] L. Babai. On Lovász lattice reduction and the nearest lattice point problem. *Combinatorica*, 6:1–13, 1986.

[3] M. Bellare. New proofs for NMAC and HMAC: Security without collision-resistance. In *Advances in Cryptology - Proc. CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 602–619. Springer, 2006.

[4] M. Bellare, A. Boldyreva, and A. Palacio. An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In *Advances in Cryptology - Proc. EUROCRYPT '04*, volume 3027 of *Lecture Notes in Computer Science*, pages 171–188. Springer, 2004.

[5] M. Bellare, R. Canetti, and H. Krawczyk. Keying hash functions for message authentication. In *Advances in Cryptology - Proc. CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 1996.

[6] M. Bellare and T. Ristenpart. Multi-property-preserving hash domain extension and the EMD transform. In *Advances in Cryptology - Proc. ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 299–314. Springer, 2006.

[7] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *Proc. of the 1st CCS*, pages 62–73. ACM Press, 1993.

[8] M. Bellare and P. Rogaway. Optimal asymmetric encryption. In *Proc. of Eurocrypt '94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. IACR, Springer, 1995.

[9] M. Bellare and P. Rogaway. The exact security of digital signatures - how to sign with RSA and Rabin. In *Proc. of Eurocrypt '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. IACR, Springer, 1996.

[10] D. J. Bernstein. Proving tight security for Rabin-Williams signatures. In *Advances in Cryptology – Proc. EUROCRYPT '08*, Lecture Notes in Computer Science. Springer, 2008.

[11] J. Black, P. Rogaway, and T. Shrimpton. Black-box analysis of the block-cipher-based hash-function constructions from PGV. In *Advances in Cryptology - Proc. CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 320–335. Springer, 2002.

[12] D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In *Advances in Cryptology - Proc. CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 443–459. Springer, 2004.

[13] D. Boneh and X. Boyen. Short signatures without random oracles. In *Advances in Cryptology - Proc. EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73. Springer, 2004.

[14] D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In *Advances in cryptology—CRYPTO 2001 (Santa Barbara, CA)*, volume 2139 of *Lecture Notes in Comput. Sci.*, pages 213–229. Springer, Berlin, 2001.

[15] D. Boneh, C. Gentry, and M. Hamburg. Space-efficient identity based encryptionwithout pairings. In *Proc. 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007)*, pages 647–657. IEEE Computer Society, 2007.

[16] D. Boneh, I. Mironov, and V. Shoup. A secure signature scheme from bilinear maps. In *Topics in Cryptology - Proc. CT-RSA 2003*, volume 2612 of *Lecture Notes in Computer Science*, pages 98–110. Springer, 2003.

[17] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594 (electronic), 2004. Preliminary version at STOC '98.

[18] S. Contini, A. K. Lenstra, and R. Steinfeld. VSH, an efficient and provable collision-resistant hash function. In *Advances in Cryptology - Proc. EUROCRYPT '06*, volume 4004 of *Lecture Notes in Computer Science*, pages 165–182. Springer, 2006.

[19] S. Contini and Y. L. Yin. Forgery and partial key-recovery attacks on HMAC and NMAC using hash collisions. In *Advances in Cryptology - Proc. ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 37–53. Springer, 2006.

[20] J.-S. Coron. On the exact security of full domain hash. In *Advances in Cryptology – Proc. CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 229–235. Springer, 2000.

[21] J.-S. Coron. Optimal security proofs for PSS and other signature schemes. In *Advances in Cryptology - Proc. EUROCRYPT '02*, volume 2332 of *Lecture Notes in Computer Science*, pages 272–287. Springer, 2002.

[22] J.-S. Coron. Security proof for partial-domain hash signature schemes. In *Advances in Cryptology - Proc. CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 613–626. Springer, 2002.

[23] J.-S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-Damgård revisited: How to construct a hash function. In *Advances in Cryptology - Proc. CRYPTO '05*, volume 3621 of *Lecture Notes in Computer Science*, pages 430–448. Springer, 2005.

[24] J.-S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-Damgård revisited: How to construct a hash function. Full version of [23]. Draft of September 2007 available on Dodis' webpage, 2007.

[25] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Advances in Cryptology - Proc. CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer, 1998.

[26] Y. Dodis and A. Yampolskiy. A verifiable random function with short proofs and keys. In *Public Key Cryptography - Proc. PKC 2005*, volume 3386 of *Lecture Notes in Computer Science*, pages 416–431. Springer, 2005.

[27] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology - Proc. CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1987.

[28] P.-A. Fouque, G. Leurent, and P. Q. Nguyen. Full key-recovery attacks on HMAC/NMAC-MD4 and NMAC-MD5. In *Advances in Cryptology - Proc. CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 13–30. Springer, 2007.

[29] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. Cryptology ePrint Archive, Report 2007/432, 2007. Also in Proc. STOC '08.

[30] O. Goldreich, S. Goldwasser, and S. Halevi. Public-key cryptosystems from lattice reduction problems. In *Proc. of Crypto '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 112–131. IACR, Springer, 1997.

[31] S. Goldwasser and Y. T. Kalai. On the (in)security of the Fiat-Shamir paradigm. In *FOCS '03: 44th Symposium on Foundations of Computer Science*. IEEE Computer Society, 2003.

[32] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988. Special issue on cryptography.

[33] IEEE. P1363: Standard specifications for public-key cryptography. Available at `http://grouper.ieee.org/groups/1363/`.

[34] J. Jonsson. Security proofs for the RSA-PSS signature scheme and its variants. In *Proc. 2nd NESSIE Workshop*, 2001. Full version available as Report 2001/053 of the Cryptology ePrint Archive.

[35] A. Joux. Multicollisions in iterated hash functions. application to cascaded constructions. In *Advances in Cryptology - Proc. CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 306–316. Springer, 2004.

[36] J. Katz and Y. Lindell. *Introduction to modern cryptography*. Chapman & Hall/CRC Cryptography and Network Security. Chapman & Hall/CRC, Boca Raton, FL, 2008.

[37] J. Katz and N. Wang. Efficiency improvements for signature schemes with tight security reductions. In *Proc. 10th ACM Conference on Computer and Communications Security, CCS 2003*, pages 155–164. ACM, 2003.

[38] P. N. Klein. Finding the closest lattice vector when it's unusually close. In *Proc. SODA*, pages 937–941, 2000.

[39] V. Klima. Tunnels in Hash Functions: MD5 Collisions Within a Minute. Cryptology ePrint Archive, Report 2006/105, 2006. `http://eprint.iacr.org/`.

[40] N. Koblitz and A. Menezes. Another look at "provable security". ii. In *Progress in Cryptology - Proc. INDOCRYPT 2006*, volume 4329 of *Lecture Notes in Computer Science*, pages 148–175. Springer, 2006. Also available as Cryptology ePrint Archive Report 2006/229.

[41] N. Koblitz and A. J. Menezes. Another look at "provable security". *J. Cryptology*, 20(1):3–37, 2007.

[42] V. Lyubashevsky, D. Micciancio, C. Peikert, and A. Rosen. SWIFFT: A modest proposal for FFT hashing. In *Fast Software Encryption – Proc. FSE '08*, volume 5086 of *Lecture Notes in Computer Science*. Springer, 2008.

[43] U. M. Maurer, R. Renner, and C. Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In *Theory of Cryptography - Proc. TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 21–39. Springer, 2004.

[44] P. Q. Nguyen and O. Regev. Learning a Parallelepiped: Cryptanalysis of GGH and NTRU Signatures. In *Advances in Cryptology – Proceedings of EUROCRYPT '06*, volume 4004 of *Lecture Notes in Computer Science*, pages 215–233. Springer, 2006.

[45] P. Q. Nguyen and T. Vidick. Sieve algorithms for the shortest vector problem are practical. *J. of Mathematical Cryptology*, 2(2), 2008.

[46] A. Numayama, T. Isshiki, and K. Tanaka. Security of digital signature schemes in weakened random oracle models. In *Public Key Cryptography - Proc. PKC '08*, volume 4939 of *Lecture Notes in Computer Science*, pages 268–287. Springer, 2008.

[47] M. Rabin. Digital signatures and public key functions as intractable as factorization. Technical report, MIT Laboratory for Computer Science, 1979. Report TR-212.

[48] RSA Laboratories. PKCS #1 v2.1: RSA cryptography standard. June 14, 2002.

[49] M. Stevens, A. K. Lenstra, and B. de Weger. Chosen-prefix collisions for MD5 and colliding X.509 certificates for different identities. In *Proc. EUROCRYPT '07*, volume 4515 of *Lecture Notes in Computer Science*, pages 1–22. Springer, 2007.

[50] D. Wagner. A generalized birthday problem. In *Advances in Cryptology - Proc. CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 288–303. Springer, 2002.

[51] L. Wang, K. Ohta, and N. Kunihiro. New key-recovery attacks on HMAC/NMAC-MD4 and NMAC-MD5. In *Advances in Cryptology - Proc. EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 237–253. Springer, 2008.

[52] X. Wang, Y. L. Yin, and H. Yu. Finding collisions in the full SHA-1. In *Advances in cryptology— Proc. CRYPTO '05*, volume 3621 of *Lecture Notes in Comput. Sci.*, pages 17–36. Springer, Berlin, 2005.

[53] X. Wang and H. Yu. How to break MD5 and other hash functions. In *Advances in Cryptology – Proc. EUROCRYPT '05*, volume 3494 of *Lecture Notes in Computer Science*, pages 19–35. Springer, 2005.

[54] H. C. Williams. A modification of the RSA public-key encryption procedure. *IEEE Trans. Inform. Theory*, 26(6):726–729, 1980.

[55] R. S. Winternitz. A secure one-way hash function built from DES. In *Proc. IEEE Symposium on Security and Privacy*, pages 88–90, 1984.

A Security Robustness of the GPV Lattice-based Signature Scheme

A.1 Description of GPV

Gentry, Peikert and Vaikuntanathan (GPV) [29] recently proposed a provably-secure variant of the GGH lattice-based signature scheme [30]. Nguyen and Regev [44] obtained a polynomial-time key-recovery known-message attack against GGH signatures, but their attack exploited the deterministic choices made by the signature algorithm. In constrast, GPV is probabilistic, based on Klein's randomized variant [38] of Babai's nearest plane algorithm [2]. The tight security proof of [29] is strikingly similar to Bernstein's tight security proof [10] for the TRW scheme. This is because both signature schemes are actually based on the same primitive: a trapdoor collision-resistant hash function with preimage sampling. In [10], collision resistance is based on factoring, while in [29], it is based on the hardness of finding very short vectors in certain lattices.

We now recall the GPV signature scheme [29], using as less lattice terminology as possible: much more details can be found in [29]. There is a main integer parameter n, and two additional integers m and q such that $q = \mathsf{poly}(n)$ and $m = \Theta(n \log q)$. The signer selects a matrix $A \in \mathbb{Z}_q^{n \times m}$ in such a way that: the distribution of A is statistically close to the uniform distribution, and the signer knows a secret set S of m very short linearly independent vectors in the m-dimensional lattice L formed by all vectors $\mathbf{x} \in \mathbb{Z}^m$ such that $A\mathbf{x} \equiv 0 \pmod{q}$.

There is a full-domain hash function $h : \{0,1\}^* \to \mathbb{Z}_q^n$. To sign a message $M \in \{0,1\}^*$, the signer applies the following process:

1. Compute the hash $\mathbf{h} = h(M) \in \mathbb{Z}_q^n$.
2. Thanks to S, the signer is able to compute in a probabilistic way (with overwhelming probability) a vector $\mathbf{e} \in \mathbb{Z}^m$ such that $A\mathbf{e} \equiv \mathbf{h} \pmod q$ and $\mathbf{e}$ is very short, namely $\|\mathbf{e}\| \leq s\sqrt{m}$ where the parameter $s \in \mathbb{R}^+$ is related to S and the lattice L. There are actually many possible $\mathbf{e}$'s, and the one selected by the signer has a Gaussian distribution, which is the main reason why one can obtain a security proof in the ROM. Again, like in TRW, it is assumed that if ever M has already been signed, then the same $\mathbf{e}$ is chosen.
3. Output $\mathbf{e} \in \mathbb{Z}^m$ as the signature of M.

To check that $\mathbf{e} \in \mathbb{Z}^m$ is a valid signature of M, one verifies that $A\mathbf{e} \equiv h(M) \pmod q$ and that $\mathbf{e}$ is sufficiently short, namely $\|\mathbf{e}\| \leq s\sqrt{m}$.

The security proof of [29] in the ROM is with respect to the following hard lattice problem: finding non-zero lattice vectors $\mathbf{x} \in L$ such that $\|\mathbf{x}\| \leq 2s\sqrt{m}$.

We now present chosen-message attacks on the GPV signature scheme, when the underlying hash function is weak. We first present attacks that break the underlying computational assumption, by finding very short vectors in the lattice L, then we discuss why such attacks might lead to forgery attacks and key-recovery attacks. In the Rabin-Williams case, breaking the underlying computational assumption (that is, factoring) directly leads to key recovery. This may not be the case with GPV [29].

A.2 Robustness with respect to collisions

Let $h : \{0,1\}^* \to \mathbb{Z}_q^n$ be the random-oracle instantiation. Assume that the attacker is able to generate a collision (M_0, M_1) on h. The attacker queries the signing oracle on M_0 and M_1, and obtains two signatures $\mathbf{e}_0, \mathbf{e}_1 \in \mathbb{Z}^m$. Then:

$$A\mathbf{e}_0 \equiv A\mathbf{e}_1 \equiv h(M_0) \equiv h(M_1) \pmod q,$$

which implies that $A(\mathbf{e}_0 - \mathbf{e}_1) \equiv 0 \pmod q$. Thus, $\mathbf{e}_0 - \mathbf{e}_1$ belongs to the m-dimensional lattice L, of norm $\leq 2s\sqrt{m}$, but could it be zero? In the exact description of GPV scheme [29, Section 5], it is written that if a message M has already been signed, the same signature should be output, but similarly to TRW, it does not say that this should also be the case if the digest $h(M)$ has already been "signed". Thus, we may assume that both $\mathbf{e}_0$ and $\mathbf{e}_1$ have been generated independently, as $M_0 \neq M_1$: since the distribution of both $\mathbf{e}_0$ and $\mathbf{e}_1$ is Gaussian, the difference $\mathbf{e}_0 - \mathbf{e}_1$ is nonzero with overwhelming probability. In other words, using only two signature queries, we are likely to have obtained a non-zero lattice vector of L of norm $\leq 2s\sqrt{m}$, which is exactly the underlying hard problem of the GPV security proof.

A.3 Robustness with respect to malleability

Similarly to the Rabin-Williams case, the previous attack can be adapted to malleability properties of the hash function. Assume indeed that the attacker is able to find two distinct messages M_0 and M_1 such that:

$$h(M_0) \equiv h(M_1) + A\mathbf{x} \pmod q,$$

where $\mathbf{x} \in \mathbb{Z}^m$ is non-zero and very short, say $\|\mathbf{x}\| \ll s\sqrt{m}$. Again, let $\mathbf{e}_0, \mathbf{e}_1 \in \mathbb{Z}^m$ be signatures corresponding to M_0 and M_1. Then:

$$A\mathbf{e}_0 \equiv A\mathbf{e}_1 + A\mathbf{x} \pmod{q},$$

which implies that $\mathbf{e}_0 - \mathbf{e}_1 - \mathbf{x}$ is a very short vector of L, whose norm might be nearly as short as $2s\sqrt{m}$.

In a similar vein, assume that the attacker is able to generate two distincts pairs of distinct messages (M_0, M_1) and (M_0', M_1') such that:

$$h(M_0) - h(M_1) \equiv h(M_0') - h(M_1') \pmod{q}.$$

In Section 2, we showed that this was easy with SWIFFT [42] and one can easily build instantiations of h based on MD5/SHA-1 with similar properties, thanks to near-collisions techniques. Then, if we denote by $\mathbf{e}_i$ and $\mathbf{e}_i'$ the respective signatures of M_i and M_i', we obtain that $\mathbf{e}_0 - \mathbf{e}_1 - \mathbf{e}_0' + \mathbf{e}_1'$ is a short vector of L, of norm $\leq 4s\sqrt{m}$.

A.4 Forgery and Key recovery

The previous attacks made it possible to find very short nonzero vectors in the lattice L, possibly sufficiently short to break the computational assumption of GPV. However, this alone may not be sufficient to recover the secret key S, nor to generate new signatures. We now explain why the attacker may still be hopeful.

First of all, the attacker may try to use his collection of newly obtained short lattice vectors to compute even shorter lattice vectors. This is exactly the principle of sieve algorithms [1, 45], which can be heuristically improved by performing strong lattice reduction of low-dimensional sublattices spanned by subsets of vectors.

The ability of the attacker to forge signatures depends on his ability to approximate the closest vector problem (CVP) in the lattice L. This approximation is usually obtained by Babai's algorithm [2] or its variants [38], which in turn depends on how short and how orthogonal are the lattice vectors known by the attacker. However, the performances of Babai's algorithm can be improved by exhaustive search: more precisely, Babai's algorithm is based on a repeated use of a one-dimensional CVP subroutine, which can heuristically be improved by using higher-dimensional CVP subroutines.

The success of these methods will depend on the exact choice of GPV parameters. Since there is currently no concrete proposal for such parameters, it is difficult to assess the difficulty of forgery or key recovery, based on the previous attacks.