




SoK: Comparison of the Security of Real World RSA Hash-and-Sign Signatures

Saqib A. Kakvi^(✉) 

Bergische Universität Wuppertal, Wuppertal, Germany
kakvi@uni-wuppertal.de

Abstract. In this modern day and age, where the majority of our communication occurs online, digital signatures are more important than ever before. Of the utmost importance are the standardised signatures that are deployed not only across the Internet, but also in everyday devices, such as debit and credit cards. The development of these signatures began in the 1990s and is still an ongoing process to this day. We will focus on RSA-based hash-and-sign signatures, specifically deterministic hash-and-sign signatures. We will give a survey of all standardised deterministic RSA hash-and-signatures, where we explore the history of each one, from inception, to attacks and finally proofs of security. As the security proofs have also appeared over the span of two decades, their statements are not always compatible with one another. To ensure this, we will consider only deterministic standardised signature schemes included in PKCS, ISO, and ANSI standards, as well as the non-standardised Full-Domain Hash, to provide a complete picture.

Keywords: Digital signatures · Random Oracle Model · RSA · Full Domain Hash · Lossy TrapDoor Permutation · PKCS · ANSI · ISO · Standards

1 Introduction

In the early days of the Internet, several practical signature schemes based on the intractability of the RSA problem appeared, both as industry standards and in academic literature. While there was a range of schemes, they were all based on the “hash-and-sign” paradigm, where any message was converted into a “message representative” in the group \mathbb{Z}_N with the use of a hash function(s) and possibly some padding. This was first suggested by Denning [21] and independently by Gordon [27]. The first scheme to follow this blueprint was an industry standard, namely PKCS#1 v1 Signature Scheme with Appendix, which first appeared at the NIST/OSI Implementors’ Workshop in 1991. This would later appear publicly in 1998 as the PKCS#1 v1.5 Signature Scheme with Appendix RSASSA-PKCS1-v1_5 [40]. This scheme did not initially have a security proof, and indeed the PKCS#1 standard has seen many attacks, mainly in the form of Bleichenbacher’s attacks [8]. In contrast to this, the first scheme proposed

in academic circles was the Full Domain Hash scheme RSA-FDH of Bellare and Rogaway [2], which had an accompanying security proof, albeit a non-tight one.

While initially disparate, academia and standards did eventually converge with the first provably secure standard being the IEEE P1363-2000 Inter Factorization Signature Scheme with Appendix IFSSA [43] and the ISO/IEC 14888-2:2008 RSA Signature scheme (14888-2 RSA Signature) [31], which is a variant of the Probabilistic Signature Scheme RSA-PSS by Bellare and Rogaway [3, 4]. The standardised variant was proven secure by Jonnson [35], the proof itself being based on the proof for the original scheme due to Coron [14]. Ideally, RSASSA-PKCS1-v1.5 would have been replaced with IFSSA, or indeed another provable secure scheme, however, this is not the case. There has been an attempt to replace RSASSA-PKCS1-v1.5 with IFSSA (the scheme is called RSASSA-PSS in the standard), but this has been slow going. IFSSA was suggested as an eventual replacement for RSASSA-PKCS1-v1.5 in PKCS#1 v2.1 [34] in 2003 and was upgraded to a requirement for *new* applications in PKCS#1 v2.2 [49]. However, RSASSA-PKCS1-v1.5 still remains, primarily for the purpose of backwards compatibility.

“Although no attacks are known against RSASSA-PKCS1-v1.5, in the interest of increased robustness, RSASSA-PSS is REQUIRED in new applications. RSASSA-PKCS1-v1.5 is included only for compatibility with existing applications.” [49, Sec. 8]

This is of course not the complete story. After the initial release of the PKCS#1 standard, other standards bodies developed RSA based hash-and-sign signature schemes, with mixed success. The International Organization for Standardization (ISO) developed their own RSA hash-and-sign schemes in the form of the ISO/IEC 9796-2 standard in 1997 [29]. However, this scheme was promptly broken by Coron, Naccache and Stern [16]. The standard was withdrawn and then updated version appeared in 2002 [30], but this was also later broken by Coron, et al. [17]. This standard was also withdrawn and it was then replaced in 2010 [32]. This version remains active and in use and to the best of our knowledge is not vulnerable to known attacks. What is quite interesting is that the vulnerabilities of the ISO/IEC 9796 signatures did not apply to the EMV standard for card payments. The EMV standard uses the ISO/IEC 9796 signatures scheme, but with strictly formatted messages, which meant that the payment ecosystem’s security was not affected by this attack.

In addition to the ISO/IEC 9796 signatures being used by EMV payment cards, the American National Standards Institute (ANSI) developed the X9.31 Standard for use in the banking sector [1]. While parts of the X9.31 Standard have been withdrawn, to the best of our knowledge the X9.31 rDSA signature scheme is still valid. This scheme has also been studied in the cryptographic literature, with a proof for the Rabin-Williams variant by Coron [15], as well as appearing in a survey by Menezes [47]. Since this scheme does follow the construction pattern of the other known standards, we include it in our comparison for completeness.

There have also been some strides made forward in the academic side, not only limited to attacks. Coron presented improved proofs for RSA-FDH [12] and the original RSA-PSS [14], as well as showing that these proofs are indeed optimal. The optimality was revisited by Kakvi and Kiltz [38] and shown to hold only when RSA is a Certified TrapDoor Permutation (CTDP) [5, 6, 25, 39]. We say a trapdoor permutation is *certified*, if there is a polynomial time algorithm that verifies that the public evaluation parameters of the trapdoor permutation are well formed i.e. that they do indeed define a permutation. Kakvi and Kiltz exploited the fact that for small prime RSA exponents e , the RSA function is actually a Lossy TrapDoor Permutation (LTDP) [50] under the Φ -Hiding Assumption [9]. This technique laid the groundwork for future proofs of standardised RSA based hash-and-sign signature schemes.

There was also progress more closely related to the standards themselves. Most notably, Coron presented security proofs for 9796-2 Scheme 1 as well as RSASSA-PKCS1-v1.5 with the restriction that $e = 2$, i.e. the Rabin-Williams variant [15]. The caveat is that the output size of the hash function needed to be 2/3 of the bit length of the modulus N . Much later, Jager, Kakvi and May [33] showed a security proof for RSASSA-PKCS1-v1.5 with the restriction that e be a small prime, using the techniques of Kakvi and Kiltz [38]. This proof also requires a large hash function output, but Jager, Kakvi and May require only 1/2 of the modulus size, compared to Coron's 2/3. There is the additional requirement that the modulus must effectively double in bit length and the modulus must have (at least) 3 prime factors¹. While not explicitly stated, it is clear the proof of Jager, Kakvi and May [38] also applies to the ISO/IEC 9796-2 schemes.

These proofs all crucially consider only the schemes themselves in isolation, and one must ask how close this is to reality. For reasons of economy and efficiency, it is common practice for key material to be shared amongst algorithms. Most notably the EMV standard uses the same RSA key for both signatures and encryption, which was shown to be vulnerable to attack [20]. While strongly suggested against in the PKCS#1 standard, it is common practice to use the same key for both RSASSA-PKCS1-v1.5 and RSASSA-PSS. In contrast to the EMV setting, this was shown to be secure by Kakvi [36], with the same caveats as that of Jager, Kakvi and May [33].

Given that these schemes are widely used in practice and have such a storied history, we feel that it is worth revisiting all the previous schemes and proofs and unifying their notations and security concepts. This will allow for a fair and accurate comparison of the schemes in question. Furthermore, taking a look back at older schemes, their security and how it has developed gives us a good overview of how standardised digital signatures, and the corresponding security proofs, have evolved over the years. We will only look at the deterministic schemes, as they are generally preferable due to the difficulty of generating randomness, especially on constrained devices. Furthermore, there is only one RSA-based ran-

¹ The proof is presented with 3 prime factors, but it works for any number co-prime to the modulus where one can compute e^{th} roots, but requires an additional assumption similar to the 2v3PA.

domised digital signature scheme, RSA-PSS, which requires two hash functions, thus making it difficult to compare with the deterministic schemes that use only one.

Finally, we will compare all the schemes in all aspects, namely modulus size, exponent size, number of prime factors and of course security loss, allowing us to give a more complete comparison of parameters. As a small sample, we show the parameters required for each scheme and what the (approximate) effective security is, which we show in Table 1. All the security proofs consider unforgeability in the Random Oracle Model, which we explain in Sect. 2.2 in Fig. 1. The computational assumptions are detailed in Sect. 2.3.

To compute these values, we assume $q_h = 2^{60}$ hash queries and $q_s = 2^{30}$ signature queries. An effective RSA modulus of k bits means that forging a signature is as hard as solving the corresponding problem for a k bit RSA modulus. The equivalent modulus size is computed by first using the equations of Lenstra [44] to calculate the estimated cost of the NFS for that security level. We then take the cost of the NFS and then find the modulus size to the nearest 10 bits that most closely matches it. To compensate for losses that are constants and for losses caused by running time increases, we simply reduce the modulus size by the binary logarithm of the loss factor. We only provide an approximate equivalent key size as there are several factors, such as time-memory trade-offs, that need to be considered for an exact figure and considering these would detract from the main goal. We believe that these figures are accurate enough to illustrate the security of each scheme. We provide a more comprehensive comparison in Tables 2, 3, 4, 5. We provide a comparison of all schemes in Tables 6.

Table 1. Parameter sizes and security for deterministic RSA based hash-and-sign schemes with a 1024 bit modulus

Scheme	Proof	Assumption	No. prime factors	Exponent e	$ \mathcal{H}(\cdot) $	Equiv. modulus
RSASSA-PKCS1-v1_5	[15]	Factoring	2	2	≥ 623	≈ 560
	[33]	RSA	3	Arbitrary	≥ 512	≈ 280
	[33]	Lossines	3	Prime $\leq 2^{256}$	≥ 512	≈ 511
9796-2 Scheme 1	[15]	Factoring	2	2	≥ 623	≈ 552
	[33]	RSA	3	Arbitrary	≥ 512	≈ 273
	[33]	Lossines	3	Prime $\leq 2^{256}$	≥ 512	≈ 504
X9.31 rDSA	[15]	Factoring	2	2	≥ 623	≈ 544
	[33]	RSA	3	Arbitrary	≥ 512	≈ 265
	[33]	Lossines	3	Prime $\leq 2^{256}$	≥ 512	≈ 497

We now recall the definitions of signature schemes and the relevant computational assumptions. We then present a brief discussion of the known attacks.

After that we present each scheme and recall the security theorem(s) for each scheme. We conclude with an overview of all the schemes.

2 Preliminaries

2.1 Notations and Conventions

We denote our security parameter as $\lambda \in \mathbb{N}$, which determines our key sizes. For all $n \in \mathbb{N}$, we denote by 1^n the n -bit string of all ones and by 0^n the n -bit string of all zeroes. We denote the concatenation of two bitstrings x and y as $x||y$. For any set S , we use $x \leftarrow_R S$ to indicate that we choose x uniformly random from S . All algorithms may be randomised. For any algorithm A , we define $x \leftarrow_{\$} A(a_1, \dots, a_n)$ as the execution of A with inputs a_1, \dots, a_n and fresh randomness and then assigning the output to x . For deterministic algorithms, we drop the $\$$ from the arrow. We denote the set of prime numbers by \mathbb{P} and we denote the subset of κ -bit primes as $\mathbb{P}[\kappa]$. Similarly, we denote the set of κ -bit integers as $\mathbb{Z}[\kappa]$. We denote by \mathbb{Z}_N^* the multiplicative group modulo $N \in \mathbb{N}$. For any $a, b \in \mathbb{Z}$, with $a < b$ we denote the set $\{a, a + 1, \dots, b - 1, b\}$ with $\llbracket a, b \rrbracket$. For any $n \in \mathbb{N}$ and for any $a \in \mathbb{N}[\kappa]$, with $\kappa < n$, we denote by $\langle a \rangle_n$ the binary representation of a padded to n bits, i.e. $\langle a \rangle_n = 0^{n-\kappa}||a$. For any bit string x of sufficient length, we denote by $\text{MSBs}(x, n)$ the n most significant (leading) bits of x and $\text{LSBs}(x, n)$ the n least significant (trailing) bits of x .

2.2 Signature Schemes

We first recall the standard definition of a signature scheme, as well as its security.

Definition 1. A digital signature scheme Sig with message space \mathbb{M} and signature space \mathbb{S} is defined as a triple of probabilistic polynomial time (PPT) algorithms $\text{Sig} = (\text{KeyGen}, \text{Sign}, \text{Verify})$:

- KeyGen takes as an input the unary representation of our security parameter 1^λ and outputs a signing key sk and verification key pk .
- Sign takes as input a signing key sk , message $m \in \mathbb{M}$ and outputs a signature $\sigma \in \mathbb{S}$.
- Verify is a deterministic algorithm, which on input of a public key and a message-signature pair $(m, \sigma) \in \mathbb{M} \times \mathbb{S}$ outputs 1 (accept) or 0 (reject).

We say Sig is correct if for any $\lambda \in \mathbb{N}$, all $(\text{pk}, \text{sk}) \leftarrow_{\$} \text{KeyGen}(1^\lambda)$, all $m \in \mathbb{M}$, and all $\sigma \leftarrow_{\$} \text{Sign}(\text{sk}, m)$ we have that

$$\Pr[\text{Verify}(\text{pk}, m, \sigma) = 1] = 1.$$

For signature security, we consider the standard notion of *UnForgeability under adaptive Chosen Message Attack* [26] in the Random Oracle Model [2] (UF-CMA(ROM)). The security experiment is presented in Fig. 1. It must be

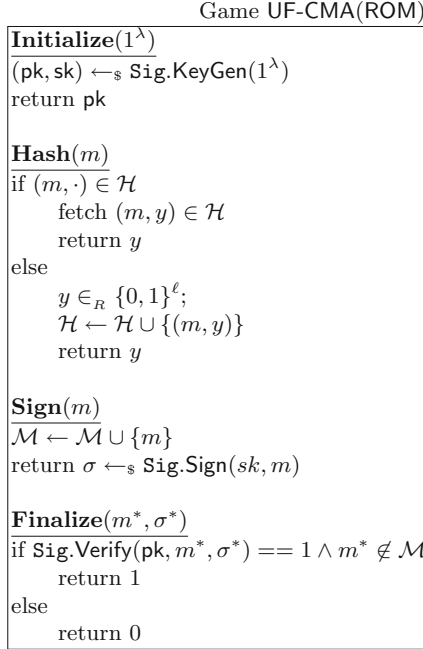


Fig. 1. UF-CMA security game in the Random Oracle Model

noted that all hash function calls are replaced with a call to the random oracle. In the case where we have multiple hash functions, we have multiple oracles. All the security statements we discuss in this paper are with respect to UF-CMA(ROM).

We say that **Sig** is $(t, \varepsilon, q_h, q_s)$ -UF-CMA(ROM) secure if for any forger \mathcal{F} running in time at most t , making at most q_h hash queries and making at most q_s signature queries, we have:

$$\text{Adv}_{\mathcal{F}, \text{Sig}}^{\text{UF-CMA(ROM)}} = \Pr \left[\begin{array}{l} 1 \leftarrow \mathbf{Finalize}(m^*, \sigma^*); \\ (m^*, \sigma^*) \leftarrow \mathcal{F}^{\text{Hash}(\cdot), \text{Sign}(\cdot)}(pk) \\ pk \leftarrow_{\S} \mathbf{Initialize}(1^\lambda) \end{array} \right] \leq \varepsilon$$

2.3 Computational Assumptions

We now recall the computation assumptions that were used in the proofs of the security statements that we will discuss, namely the RSA Assumption k -RSA $[\lambda]$, φ -Hiding Assumption k - Φ HA $[\lambda]$, the Factoring Assumption FACT $[\lambda]$ and the 2 vs 3 Primes Assumption 2v3PA. Note that all assumptions have additional parameters k , which is the number of prime factors in our modulus, and λ , which is the bit-size of the modulus. The number of prime factors does play a role in some of the proofs, so we include it in all the theorem statements for consistency.

We begin by presenting the RSA Assumption, which essentially states that given a modulus N , and exponent e and a random $y \in \mathbb{Z}_N^*$, it is hard to com-

pute an e^{th} root of y modulo N , equivalently, it is hard to find x such that $x^e \bmod N = y$.

Definition 2 (RSA Assumption [51]). *The RSA Assumption, denoted by $k\text{-RSA}[\lambda]$, states that given (N, e, x^e) it is hard to compute x , where N is a λ -bit number and is the product of k distinct random prime numbers $p_i \in \mathbb{P}$, for $i \in \llbracket 1, k \rrbracket$, for k constant, $e \in \mathbb{Z}_{\varphi(N)}^*$, and $x \in_{\mathcal{R}} \mathbb{Z}_N$. $k\text{-RSA}[\lambda]$ is said to be (t, ε) -hard, if for all adversaries \mathcal{A} running in time at most t , we have*

$$\text{Adv}_{\mathcal{A}}^{k\text{-RSA}[\lambda]} = \Pr[x = \mathcal{A}(N, e, x^e \bmod N)] \leq \varepsilon.$$

Next, we discuss the φ -Hiding Assumption by Cachin et al. [9], which essentially states that given a modulus N and a sufficiently small exponent e , it is hard to decide if $e|\varphi(N)$ or not. In this case sufficiently small means that $e < N^{\frac{1}{4}}$, as for larger exponents, Kakvi, Kiltz and May. [39] show how to decide this using Coppersmith's method [10]. Note that when $\gcd(e_{\text{los}}, \varphi(N)) = e_{\text{los}}$, the RSA function $x^{e_{\text{los}}} \bmod N$ is exactly e_{los} -to-1, i.e. it is said to be e_{los} -regular lossy as defined by Kakvi and Kiltz [38].

Definition 3 (The φ -Hiding Assumption. [9]). *The φ -Hiding Assumption, denoted by $k\text{-}\Phi\text{HA}[\lambda]$, states that it is hard to distinguish between (N, e_{inj}) and (N, e_{los}) , where N is a λ -bit number and is the product of k distinct random prime numbers $p_i \in \mathbb{P}$, for $i \in \llbracket 1, k \rrbracket$, for k constant, and $e_{\text{inj}}, e_{\text{los}} \in \mathbb{P}$ and and $3 < e_{\text{inj}}, e_{\text{los}} \leq N^{\frac{1}{4}}$, with $\gcd(e_{\text{inj}}, \varphi(N)) = 1$ and $\gcd(e_{\text{los}}, \varphi(N)) = e_{\text{los}}$, where φ is the Euler Totient function. $k\text{-}\Phi\text{HA}[\lambda]$ is said to be (t, ε) -hard, if for all distinguishers \mathcal{D} running in time at most t , we have:*

$$\text{Adv}_{\mathcal{D}}^{k\text{-}\Phi\text{HA}[\lambda]} = \Pr[1 \leftarrow \mathcal{D}(N, e_{\text{inj}})] - \Pr[1 \leftarrow \mathcal{D}(N, e_{\text{los}})] \leq \varepsilon$$

Now we mention the strongest assumption we need, namely the Factoring assumption **Factoring**. While both RSA and ΦHA imply **Factoring**, it is also conjectured that ΦHA is equivalent to **Factoring**. Additionally, the problem of finding quadratic residues modulo N is known to be equivalent to **Factoring**.

Definition 4 (Factoring Assumption). *The Factoring Assumption, denoted by $k\text{-FACT}[\lambda]$, states that given N , which is a λ -bit number and is the product of k distinct random prime numbers $p_i \in \mathbb{P}$, for $i \in \llbracket 1, k \rrbracket$, for k constant, it is hard to compute the factors of N , p_1, \dots, p_k . $k\text{-FACT}[\lambda]$ is said to be (t, ε) -hard, if for all adversaries \mathcal{A} running in time at most t , we have*

$$\text{Adv}_{\mathcal{A}}^{k\text{-FACT}[\lambda]} = \Pr[(p_1, \dots, p_k) = \mathcal{A}(N)] \leq \varepsilon.$$

The final assumption that we need to recall is the 2 vs 3 primes assumption $2v3\text{PA}$. This assumption essentially states that you cannot decide if a given modulus has 2 or 3 prime factors. This assumption has never formally been studied and is simply widely believed to hold. This is needed to bring any proof that requires a 3 prime factor modulus to the standard case of a 2 prime factor modulus.

Definition 5 (2v3PA Assumption). *The 2 vs. 3 Primes Assumption, denoted by $2v3PA[\lambda]$, states that it is hard to distinguish between N_2 and N_3 , where N_2, N_3 are λ -bit numbers, where $N_2 = p_1 p_2$ is the product of 2 distinct random prime numbers $p_1, p_2 \in \mathbb{P}$ and $N_3 = q_1 q_2 q_3$ is the product of 3 distinct random prime numbers $q_1, q_2, q_3 \in \mathbb{P}$. $2v3PA[\lambda]$ is said to be (t, ε) -hard, if for all distinguishers \mathcal{D} running in time at most t , we have:*

$$\text{Adv}_{\mathcal{D}}^{\phi_{\text{HA}}} = \Pr[1 \leftarrow \mathcal{D}(N_2)] - \Pr[1 \leftarrow \mathcal{D}(N_3)] \leq \varepsilon$$

3 Attacks

Here we briefly discuss the known attacks on standardised RSA hash-and-sign signatures. Even before the development of the standardised signatures, there had been general cryptanalysis of the RSA primitive. One of the first general RSA attacks was due to Davida [18], which was later generalised by Desmedt and Odlyzko [22]. Indeed, the most general attacks on RSA-based system are so-called ‘‘Coppersmiths attacks’’ as they are based on the initial results of Coppersmith [10] based on the LLL algorithm [45]. This has since been an active research area and we refer the reader to May’s survey for further details [46].

Following this, attacks were found on more concrete settings by Gordon [27] and DeJonge and Chaum [19]. The latter attacks were extended by Girault and Misarsky [23, 48]. For a more technical overview of the attacks, we refer the reader to the invited survey of Misarsky [48].

Based on this, Denning [21] and Gordon [27] independently suggested what would become the hash-and-sign paradigm. This knowledge was taken on board during the design of the PKCS#1 signatures [42], as well as the ISO 9796 signatures [28]. One would hope that this would result in secure and robust schemes. This, however, proved not to be the case, as demonstrated by Bleichenbacher’s ‘‘million message attack’’ for PKCS#1 v1.5 encryption [7], which used the malleability of RSA to transform a valid ciphertext into a new (possibly invalid) ciphertexts. These new ciphertexts were sent to the server, whose responses could be used to eventually decrypt the original ciphertext. In some circumstances, the attack can also be extended to PKCS#1 v1.5 signatures. Also of note are the attacks on ISO/IEC 9796 by Coron, Naccache and Stern [16] and Coron et al. [17].

4 Full-Domain Hash

We start by looking at the RSA Full-Domain Hash signature scheme RSA-FDH . While it is not included in any of the standards, it still bears investigation, as the proof methodologies developed for it have led to proof methodologies for all the other schemes. RSA-FDH was first introduced by Bellare and Rogaway [2] and while they did have a security proof, it was a non-tight one. This was then improved by Coron [12] who showed a better, but still non-tight proof. Coron’s proof has a security loss that depends on the number of signing queries q_s ,

as opposed to the number of hash queries q_h , which is generally much larger. Additionally, Coron showed that this proof was actually optimal by means of a meta-reduction [12], removing any hope of a tight proof.

However, Kakvi and Kiltz [38] noticed that the meta-reduction crucially requires that the RSA public key (N, e) be a CTDP [5, 6]. We say a TDP is *certified*, if there is a polynomial time algorithm that verifies that the public evaluation parameters of the trapdoor permutation are well formed i.e. that they do indeed define a permutation. This is not the case for RSA if the exponent e is a small prime. In particular, if $e \leq N^{1/4}$ then this defines an LTDP [50] under the φ -Hiding Assumption [9]. Furthermore, Kakvi, Kiltz and May [39] showed that for large prime exponents e , i.e. $e \geq N^{1/4}$, one can check if an RSA key does indeed define a permutation in polynomial time. Additionally, Goldberg et al. showed an efficient non-interactive method to certify an RSA public key [25]. We recall the RSA-FDH scheme in Fig. 2 and then present the proofs.

Scheme RSA-FDH

<p>KeyGen(1^λ) $p, q \in_R \mathbb{P}[\lambda/2]$ $N = pq$ $\varphi(N) = (p-1)(q-1)$ $e \in_R \mathbb{Z}_N^*, \gcd(e, \varphi(N)) = 1$ pick hash function $H : \{0, 1\}^* \rightarrow \{0, 1\}^\lambda$ return $(pk = (N, e, H), sk = (p, q))$</p> <p>Sign$(sk, m)$ $y \leftarrow H(m)$ return $\sigma = y^{1/e} \pmod N$</p> <p>Verify(pk, m, σ) $y' = \sigma^e \pmod N$ $z = H(m)$ if $(z == y')$ return 1 else return 0</p>
--

Fig. 2. RSA Full-Domain Hash RSA-FDH

The original proof of Bellare and Rogaway [2] was simply to guess which message would be forged and program the random oracle to give a solution to the RSA problem. This is achieved by setting the hash value of the selected message to be the RSA target value y . This works with probability $1/q_h$, where q_h is the number of hash function queries made by the adversary.

Theorem 1 (Bellare-Rogaway [2]). *Assume that $2\text{-RSA}[\lambda]$ is (t', ε') -hard. Then for any (q_h, q_s) , RSA-FDH is $(t, \varepsilon, q_h, q_s)$ -UF-CMA secure in the Random Oracle Model, where*

$$\begin{aligned}\varepsilon' &= \frac{\varepsilon}{q_h} . \\ t' &= t + (q_h + q_s + 1) \cdot \mathcal{O}(\lambda^3).\end{aligned}$$

Coron [12] improved this by embedding the RSA target value y into multiple hash values. The drawback to this was that the reduction cannot simulate a signature for any message whose hash value had y embedded in it. Therefore, the proportion must be chosen carefully. The analysis by Coron showed that an optimal choice yielded a loss of q_s , where q_s is the number of signature queries.

Theorem 2 (Coron [12]). *Assume that $2\text{-RSA}[\lambda]$ is (t', ε') -hard. Then for any (q_h, q_s) , RSA-FDH is $(t, \varepsilon, q_h, q_s)$ -UF-CMA secure in the Random Oracle Model, where*

$$\begin{aligned}\varepsilon' &= \frac{\varepsilon}{q_s} \cdot \left(1 - \frac{1}{q_s + 1}\right)^{q_s + 1} \approx \frac{\varepsilon}{q_s} \\ t' &= t + \mathcal{O}(q_h \cdot \lambda^3).\end{aligned}$$

Kakvi and Kiltz [38] later revisited Coron’s optimality result and observed that this only holds for the case when RSA is a CTDP [5, 6]. This is only possible for RSA if the exponent is a large prime (or a product thereof) [39], or if we provide some additional information [25]. Kakvi and Kiltz leveraged the fact that RSA is a lossy permutation for small prime e and were able to show a tight proof in this case.

Theorem 3 (Kakvi-Kiltz [38]). *Assume $2\text{-}\Phi\text{HA}[\lambda]$ is (t', ε') -hard and gives an η -regular lossy trapdoor function. Then, for any (q_h, q_s) , we have that RSA-FDH is $(t, \varepsilon, q_h, q_s)$ -UF-CMA secure in the Random Oracle Model, where*

$$\begin{aligned}\varepsilon &= \left(\frac{2\eta - 1}{\eta - 1}\right) \cdot \varepsilon' \\ t &= t' + \mathcal{O}(q_h \cdot \lambda^3)\end{aligned}$$

We now compare these security statements and their effect on parameter selection in Table 2, concretely for the case of 1024 bit moduli. We present these in a similar manner to that of Table 1, but we drop some of the less relevant parameters. Here we also provide the size of the modulus used in the reduction, but this does not directly correspond to the security of the scheme, i.e. the scheme is not necessarily as secure as the assumption. The scheme is as secure as the assumption with a modulus of bit length given in the “Equiv. modulus” column. To compute these values we have taken $q_h = 2^{60}$, $q_s = 2^{30}$. As with Table 1, the key sizes are approximate due to the complexity of computing exact key sizes.

Table 2. Security of RSA-FDH with a 1024-bit modulus

Scheme	Proof methodology	Assumption	Exponent e	Equiv. modulus
RSA-FDH	Bellare-Rogaway	2-RSA[1024]	Arbitrary	≈ 250
	Coron	2-RSA[1024]	Arbitrary	≈ 560
	Kakvi-Kiltz	2- Φ HA[1024]	Prime $\leq 2^{256}$	≈ 1023

5 Public-Key Cryptography Standards #1

We will now look at the PKCS#1 signature scheme, which was one of the first standardised hash-and-sign signature schemes. The PKCS#1 version 1 actually predates Full-Domain Hash and hence the idea of a larger hash function was not considered at first. The scheme was first made public with version 1.5 [40], which is why the scheme is most commonly referred to as PKCS#1 v1.5. The standard was updated to version 2.0 [41] in short order to include RSA-OAEP as a replacement for the encryption algorithm due Bleichenbacher’s “million message attack” [7].

While the signatures schemes were not affected by this, the consensus was that RSASSA-PKCS1-v1.5 needed to be replaced with a secure signature scheme. The natural candidate in RSASSA-PSS was added in version 2.1 [34]. However, at this point it was a recommendation and not a requirement. This was further changed in version 2.2 [49], where it was required for all *new* applications. The main reason for this was that a large number of systems had implemented RSASSA-PKCS1-v1.5 in *hardware*, in particular in middleware, which proved problematic to upgrade².

While no real attacks were found against RSASSA-PKCS1-v1.5, the lack of security proof was of some concern. The only known proof was that of Coron [15], but that was for the Rabin-Williams variant and required a larger hash function output than the norm. This has since been improved somewhat by Jager, Kakvi and May [33], who showed a proof for the small-exponent RSA case, but still required a large hash function output, albeit smaller than that of Coron [15]. We now recall (a generalised variant of) RSASSA-PKCS1-v1.5 in Fig. 3 and then we recall the security theorems.

The first statement of security was by Coron [15] for the Rabin-Williams variant, i.e. with $e = 2$, which is secure based on the factoring assumption. Coron’s theorem statements was quite general and in indeed it extended to the ANSI X9.31 rDSA signatures directly. Coron considered signatures of the form $\sigma = (\gamma \cdot H(m) + f(m))^{1/2}$, i.e. scheme with (potentially) message-dependent padding. Coron gave the security of these schemes in Theorem 4.

Theorem 4 (Coron [15]). *Assume that 2-FACT $[\lambda]$ is (t', ε') -hard. Then for any (q_h, q_s) , any partial domain hash signature scheme i.e.,*

² cf. <https://www.ietf.org/mail-archive/web/tls/current/msg19360.html>.

Scheme RSASSA-PKCS1-v1_5

KeyGen($1^\lambda, k, (\lambda_1, \dots, \lambda_k), \ell$)

for $i \in \llbracket 1, k \rrbracket$

$p_i \in_R \mathbb{P}[\lambda_i]$

next i

$N = \prod_{i=1}^k p_i$

$\varphi(N) = \prod_{i=1}^k (p_i - 1)$

$e \in_R \mathbb{Z}_N^*$, $\gcd(e, \varphi(N)) = 1$

pick hash function $H : \mathbb{M} \rightarrow \{0, 1\}^\ell$

Look-up α -bit ID_H for H

$\nu = \lambda - \ell - \alpha - 23$

$\text{PAD} = 0^{15} \| 1^\nu \| 0^8 \| \text{ID}_H$

return $(\text{pk} = (N, e, \text{PAD}, H), \text{sk} = (p, q))$

Sign(sk, m)

$z \leftarrow H(m)$

$y = \text{PAD} \| z$

return $\sigma = y^{1/e} \pmod N$

Verify(pk, m, σ)

$y' = \sigma^e \pmod N$

$z \leftarrow H(m)$

if $(\text{PAD} \| z == y')$

return 1

else

return 0

Fig. 3. RSA PKCS#1 v1.5 Signature RSASSA-PKCS1-v1.5

$\sigma = (\gamma \cdot H(m) + f(m))^{1/2}$, is $(t, \varepsilon, q_h, q_s)$ -UF-CMA secure in the Random Oracle Model, where

$$\varepsilon' = \frac{\varepsilon - 32(q_h + q_s + 1)(\ell - \frac{2\lambda}{3})(\gamma \cdot 2^{\frac{3}{13}(\ell - \frac{2\lambda}{3})})}{8q_s}$$

$$t' = t + \gamma \left(\ell - \frac{2\lambda}{3} \right) (q_h + q_s + 1) \cdot \mathcal{O}(\lambda^3).$$

For RSASSA-PKCS1-v1.5 we can see that $\gamma = 1$ and $f(m) = \text{PAD} \times 2^\ell$. If we set $\ell = \frac{2\lambda}{3} + 1$, we get the following Theorem.

Theorem 5 (Coron [15]). Assume that $2\text{-FACT}[\lambda]$ is (t', ε') -hard. Then for any (q_h, q_s) , RSASSA-PKCS1-v1.5 with $e=2$ is $(t, \varepsilon, q_h, q_s)$ -UF-CMA secure in the Random Oracle Model, where

$$\varepsilon' = \frac{\varepsilon - 32(q_h + q_s + 1)(2^{\frac{3}{13}})}{8q_s} \approx \frac{\varepsilon}{8q_s} - \frac{4q_h}{q_s}$$

$$t' = t + \mathcal{O}(q_h \cdot \lambda^3).$$

For many years, this remained the only known proof for RSASSA-PKCS1-v1.5, or any variant thereof, until the proof of Jager, Kakvi and May [33]. The main

technical hurdle was that the fixed padding and relatively small size of the hash function meant that signatures could not be simulated in polynomial time. Jager, Kakvi and May overcame this with their novel Encode algorithm to allow simulation of signatures in polynomial time. Using this, combined with the proof techniques for RSA-FDH of Coron [12, 13] and Kakvi and Kiltz [37, 38], they presented theorems with similar bounds.

The drawback, however, is that the Encode algorithm requires the bit size of the modulus to be doubled, by multiplying it with a prime of the same size. That is to say, if we wish to reduce the security of $2\text{-}\Phi\text{HA}[\lambda]$ or $2\text{-RSA}[\lambda]$, we need a 2λ -bit modulus in our key, with the additional λ -bits made up by a third prime factor. Therefore they proved security for keys where $N = pqr$, i.e. is the product of three primes. Under the assumption that 3-prime moduli are indistinguishable from 2-prime moduli, these results can be brought back to the case $N = pq$. We now present the results of Jager, Kakvi and May [33].

Theorem 6 (Jager-Kakvi-May [33]). *Assume that $2\text{-RSA}[\lambda]$ is (t', ε') -hard. Then for any (q_h, q_s) , RSASSA-PKCS1-v1.5 is $(t, \varepsilon, q_h, q_s)$ -UF-CMA secure in the Random Oracle Model, where*

$$\begin{aligned}\varepsilon' &= \frac{\varepsilon}{q_s} \cdot \left(1 - \frac{1}{q_s + 1}\right)^{q_s + 1} \approx \frac{\varepsilon}{q_s} \cdot \exp(-1) \\ t' &= t + \mathcal{O}(q_h \cdot \lambda^4).\end{aligned}$$

Theorem 7 (Jager-Kakvi-May [33]). *Assume $2\text{-}\Phi\text{HA}[\lambda]$ is (t', ε') -hard and gives an η -regular lossy trapdoor function. Then, for any (q_h, q_s) , RSASSA-PKCS1-v1.5 is $(t, \varepsilon, q_h, q_s)$ -UF-CMA secure in the Random Oracle Model, where*

$$\begin{aligned}\varepsilon &= \left(\frac{2\eta - 1}{\eta - 1}\right) \cdot \varepsilon' \\ t &= t' + \mathcal{O}(q_h \cdot \lambda^4)\end{aligned}$$

We now compare these security statements and their effect on parameter selection in Table 3, concretely for the case of 1024 bit moduli. We present these in a similar manner to that of Table 1. Here we also provide the size of the modulus used in the reduction, but this does not directly correspond to the security of the scheme, i.e. the scheme is not necessarily as secure as the assumption. The scheme is as secure as the assumption with a modulus of bit length given in the ‘‘Equiv. modulus’’ column. To compute these values we have taken $q_h = 2^{60}$, $q_s = 2^{30}$. As with Table 1, the key sizes are approximate due to the complexity of computing exact key sizes.

6 International Organization for Standardization 9796-2

We will now look at the ISO/IEC 9796-2:2010 signature scheme, which is one of the most widely deployed signature schemes. This scheme is used in the EMV

Table 3. Parameter sizes and security for RSASSA-PKCS1-v1.5 with a 1024 bit modulus

Scheme	Proof methodology	Assumption	No. prime factors	Exponent e	$ \mathbb{H}(\cdot) $	Equiv. modulus
RSASSA-PKCS1-v1.5	Coron	2-FACT[1024]	2	2	≥ 623	≈ 560
	Jager-Kakvi-May	2-RSA[512]	3	Arbitrary	≥ 512	≈ 280
		+ 2v3PA[1024]	2	Arbitrary	≥ 512	≈ 280
	Jager-Kakvi-May	2- Φ HA[512]	3	Prime $\leq 2^{256}$	≥ 512	≈ 511
		+ 2v3PA[1024]	2	Prime $\leq 2^{256}$	≥ 512	≈ 511

payment system for so-called “chip and pin” cards. According to EMVCo. There are around 9,893,000,000 (9.8 billion) EMV cards in circulation as of Q4 2019, making up 63.8% of cards issued globally³. Despite this huge usage, the signatures are known to be vulnerable to some attacks and indeed the ISO/IEC 9796 standard has had several iterations of breaks and fixes. It is worth noting that while the scheme has been broken, the EMV implementation is not vulnerable. The reason for this is that the attacks require signatures on some specially crafted messages, which exploit the multiplicative property of RSA, but are incompatible with the EMV standard. Messages in the EMV standard have a very fixed format and include some identifiers and serial numbers, as well as the date. Thus, it is very unlikely that an honest EMV endpoint would sign the messages required for the attack to be successful. An equivalent way of looking at this would be to say that the attacks only work on larger message spaces than the messages space used by the EMV protocol.

The very first version, the ISO/IEC 9796-1 was very quickly broken and is no longer in use, so we will not discuss in great detail, but instead refer the reader to the articles by Coppersmith, Halevi and Jutla [11] and Girault and Misarsky [24] for further details. It is for this reason that the ISO/IEC 9796-1 standard was replaced by the first version of the ISO/IEC 9796-2 standard in 1997 [29]. This version was also vulnerable to attack, specifically the attacks due to Coron, Naccache and Stern [16]. The standard was then further updated in 2002 to combat these attacks [30], but eventually would fall to the attacks of Coron et al. [17]. Finally, the standard was updated to its current form, that is the ISO/IEC 9796-2:2010 [32], which is what we will focus on, particularly Scheme 1.

The ISO/IEC 9796-2 Scheme 1 is a deterministic scheme with message recovery. In particular it has two modes, namely full message recovery, which works for messages that are sufficiently small, and partially message recovery for larger messages. When signing, the starting bits of the message representative indicate if we are using full or partial recovery. For partial recovery, the message representative begins with 0x6A, and for full message recovery it begins with 0x4A. This is then followed by the message portion that is recoverable, padded up with zeros, if needed, which is followed by the hash of the complete message. The signatures then end with 0xBC. We recall the (generalised) scheme with

³ cf. <https://www.emvco.com/about/deployment-statistics>.

partial recovery 9796-2 Scheme 1 (PR) in Fig. 4a and the (generalised) scheme with full recovery 9796-2 Scheme 1 (FR) in Fig. 4b.

Scheme 9796-2 Scheme 1 (PR)	Scheme 9796-2 Scheme 1 (FR)
<pre> KeyGen($1^\lambda, k, (\lambda_1, \dots, \lambda_k), \ell$) for $i \in \llbracket 1, k \rrbracket$ $p_i \in_R \mathbb{P}[\lambda_i]$ next i $N = \prod_{i=1}^k p_i$ $\varphi(N) = \prod_{i=1}^k (p_i - 1)$ $e \in_R \mathbb{Z}_N^*$, $\gcd(e, \varphi(N)) = 1$ pick hash function $H : \{0, 1\}^* \rightarrow \{0, 1\}^\ell$ $\text{PAD}_L = 1101010, \text{PAD}_R = 10111100$ return $(\text{pk} = (N, e, \text{PAD}_L, \text{PAD}_R, H), \text{sk} = (p, q))$ Sign(sk, m) $z \leftarrow H(m)$ $\nu = \lambda - \ell - 16$ $m_1 = \text{MSBs}(m, \nu)$ $y = \text{PAD}_L m_1 z \text{PAD}_R$ return $\sigma = y^{1/e} \bmod N$ Verify(pk, m_2, σ) $y' = \sigma^e \bmod N$ interpret $y = \text{PAD}_L m_1 z \text{PAD}_R$ if $(H(m_1) m_2) == z$ return 1 else return 0 </pre>	<pre> KeyGen($1^\lambda, k, (\lambda_1, \dots, \lambda_k), \ell$) for $i \in \llbracket 1, k \rrbracket$ $p_i \in_R \mathbb{P}[\lambda_i]$ next i $N = \prod_{i=1}^k p_i$ $\varphi(N) = \prod_{i=1}^k (p_i - 1)$ $e \in_R \mathbb{Z}_N^*$, $\gcd(e, \varphi(N)) = 1$ pick hash function $H : \{0, 1\}^* \rightarrow \{0, 1\}^\ell$ $\text{PAD}_L = 1101010, \text{PAD}_R = 10111100$ return $(\text{pk} = (N, e, \text{PAD}_L, \text{PAD}_R, H), \text{sk} = (p, q))$ Sign(sk, m) $z \leftarrow H(m)$ $\nu = \lambda - \ell - 16$ $m_1 = \text{MSBs}(m, \nu)$ $y = \text{PAD}_L m_1 z \text{PAD}_R$ return $\sigma = y^{1/e} \bmod N$ Verify(pk, m_2, σ) $y' = \sigma^e \bmod N$ interpret $y = \text{PAD}_L m_1 z \text{PAD}_R$ if $(H(m_1) m_2) == z$ return 1 else return 0 </pre>

(a) ISO/IEC RSA Signature with Partial Message Recovery 9796-2 Scheme 1 (PR)

(b) ISO/IEC RSA Signature with Full Message Recovery 9796-2 Scheme 1 (FR)

Fig. 4. The two versions of the ISO/IEC 9796-2 Scheme 1

While the two schemes are distinct, the proofs work identically for both. Therefore, we will simply present the theorems for the scheme as whole and not for each case individually. We first present the Rabin-Williams proof by Coron [15]. Recall that Theorem 4 was stated for schemes of the form $\sigma = (\gamma \cdot H(m) + f(m))^{1/2}$. Here we can see that for 9796-2 Scheme 1, we have $\gamma = 2^8$ and $f(m) = \text{PAD}_L || \text{MSBs}(m, \nu) \times 2^{\ell+8} + \text{PAD}_R$. If we set $l = \frac{2\lambda}{3} + 1$, we see that we get the following Theorem.

Theorem 8 (Coron [15]). *Assume that 2-FACT $[\lambda]$ is (t', ε') -hard. Then for any (q_h, q_s) , 9796-2 Scheme 1 is $(t, \varepsilon, q_h, q_s)$ -UF-CMA secure in the Random Oracle Model, where*

$$\varepsilon' = \frac{\varepsilon - 32(q_h + q_s + 1) \cdot 2^8 \cdot 2^{\frac{3}{13}}}{8q_s} \approx \frac{\varepsilon}{8q_s} - \frac{1024 \cdot q_h}{q_s}$$

$$t' = t + 2^8 \cdot \mathcal{O}(q_h \cdot \lambda^3).$$

Although Jager, Kakvi and May did not explicitly prove the security of 9796-2 Scheme 1, the scheme fits almost perfectly into their setting. If we use

the repeated Encode method of Kakvi [36], then we can adapt the proof accordingly and we get similar bounds, with the similar 3-prime requirement as in Theorems 6 and 7. We now present the Theorems for 9796-2 Scheme 1.

Theorem 9 (Jager-Kakvi-May [33]). *Assume that 2-RSA[λ] is (t′, ε′)-hard. Then for any (q_h, q_s), 9796-2 Scheme 1 is (t, ε, q_h, q_s)-UF-CMA secure in the Random Oracle Model, where*

$$\begin{aligned} \varepsilon' &= \frac{\varepsilon}{q_s} \cdot \left(1 - \frac{1}{q_s + 1}\right)^{q_s + 1} \approx \frac{\varepsilon}{q_s} \cdot \exp(-1) \\ t' &= t + 2^7 \cdot \mathcal{O}(q_h \cdot \lambda^4). \end{aligned}$$

Theorem 10 (Jager-Kakvi-May [33]). *Assume 2-ΦHA[λ] is (t′, ε′)-hard and gives an η-regular lossy trapdoor function. Then, for any (q_h, q_s), 9796-2 Scheme 1 is (t, ε, q_h, q_s)-UF-CMA secure in the Random Oracle Model, where*

$$\begin{aligned} \varepsilon &= \left(\frac{2\eta - 1}{\eta - 1}\right) \cdot \varepsilon' \\ t &= t' + 2^7 \cdot \mathcal{O}(q_h \cdot \lambda^4) \end{aligned}$$

We now compare these security results and their effect on parameter selection in Table 3, concretely for the case of 1024 bit moduli. We present these in a similar manner to that of Table 1. Here we also provide the size of the modulus used in the reduction, but this does not directly correspond to the security of the scheme, i.e. the scheme is not necessarily as secure as the assumption. The scheme is as secure as the assumption with a modulus of bit length given in the “Equiv. modulus” column. To compute these values we have taken q_h = 2⁶⁰, q_s = 2³⁰. As with Table 1, the key sizes are approximate due to the complexity of computing exact key sizes.

Table 4. Parameter sizes and security for 9796-2 Scheme 1 with a 1024 bit modulus

Scheme	Proof methodology	Assumption	No. prime factors	Exponent e	H(·)	Equiv. modulus
ISO 9769-2 Scheme 1	Coron	2-FACT[1024]	2	2	≥ 623	≈ 552
	Jager-Kakvi-May	2-RSA[512]	3	Arbitrary	≥ 512	≈ 273
		+ 2v3PA[1024]	2	Arbitrary	≥ 512	≈ 273
	Jager-Kakvi-May	2-ΦHA[512]	3	Prime ≤ 2 ²⁵⁶	≥ 512	≈ 504
+ 2v3PA[1024]		2	Prime ≤ 2 ²⁵⁶	≥ 512	≈ 504	

7 American National Standards Institute X9.31 Signatures

We now look at the final deterministic hash-and-sign signature on our list, namely, the ANSI X9.31 signatures [1]. While these signatures were standardised at a similar time to the others, and follows a similar construction philosophy, there is scant mention of them in the academic literature. To the best of our knowledge, the signature were only ever investigated by Coron [15] and Menezes [47]. While parts of the standard have been withdrawn, specifically those related to the generation of random numbers, to the best of our knowledge the signature is still valid. We now recall the (generalised) scheme in Fig. 5.

Scheme X9.31 rDSA

<p>KeyGen$(1^\lambda, k, (\lambda_1, \dots, \lambda_k), \ell)$</p> <p>for $i \in \llbracket 1, k \rrbracket$</p> <p style="padding-left: 20px;">$p_i \in_R \mathbb{P}[\lambda_i]$</p> <p>next i</p> <p>$N = \prod_{i=1}^k p_i$</p> <p>$\varphi(N) = \prod_{i=1}^k (p_i - 1)$</p> <p>$e \in_R \mathbb{Z}_N^*, \gcd(e, \varphi(N)) = 1$</p> <p>pick hash function $H : \mathbb{M} \rightarrow \{0, 1\}^\ell$</p> <p>Look-up 16-bit ID_H for H</p> <p>$\nu = (\lambda - \ell - 24)/4$</p> <p>$PAD = 0110 \parallel (1011)^\nu \parallel 1010$</p> <p>return $(pk = (N, e, PAD, ID_H, H), sk = (p, q))$</p> <p>Sign$(sk, m)$</p> <p>$z \leftarrow H(m)$</p> <p>$y = PAD \parallel z \parallel ID_H$</p> <p>return $\sigma = y^{1/e} \pmod N$</p> <p>Verify(pk, m, σ)</p> <p>$y' = \sigma^e \pmod N$</p> <p>$z \leftarrow H(m)$</p> <p>if $(PAD \parallel z == y')$</p> <p style="padding-left: 20px;">return 1</p> <p>else</p> <p style="padding-left: 20px;">return 0</p>

Fig. 5. American National Standards Institute X9.31 rDSA

As we can see the scheme is very similar to RSASSA-PKCS1-v1_5, with two small differences. Firstly, the padding string is 0x6B...BA and not 0x0F...F0, but this makes no difference for the proofs, as they are for both for arbitrary padding. Secondly, the (fixed-length) hash function identifier is after the hash as opposed to before it. Although both proofs can deal with this, it does affect

the security differently. In the case of Jager, Kakvi and May [33], if we adapt the proof using the repeated Encode sampling method of Kakvi [36], we get this loss appearing only in the runtime of our reduction. On the other hand, in the proof of Coron [15] this factor only appears in the both the running time and the success probability, as if we express X9.31 rDSA in terms of Theorem 4, we have $\gamma = 2^{16}$ (and $f(m) = \text{PAD} \times 2^{\ell+16} + \text{ID}_{\mathbb{H}}$). As in the previous proofs, we set $\ell = \frac{2\lambda}{3} + 1$. We now present the theorems for the security of X9.31 rDSA.

Theorem 11 (Coron [15]). *Assume that 2-FACT $[\lambda]$ is (t', ε') -hard. Then for any (q_h, q_s) , X9.31 rDSA is $(t, \varepsilon, q_h, q_s)$ -UF-CMA secure in the Random Oracle Model, where*

$$\begin{aligned} \varepsilon' &= \frac{\varepsilon - 32(q_h + q_s + 1) \cdot 2^{16} \cdot 2^{\frac{3}{13}}}{8q_s} \approx \frac{\varepsilon}{8q_s} - \frac{2^{18} \cdot q_h}{q_s} \\ t' &= t + 2^{16} \cdot \mathcal{O}(q_h \cdot \lambda^3). \end{aligned}$$

Theorem 12 (Jager-Kakvi-May [33]). *Assume that 2-RSA $[\lambda]$ is (t', ε') -hard. Then for any (q_h, q_s) , X9.31 rDSA is $(t, \varepsilon, q_h, q_s)$ -UF-CMA secure in the Random Oracle Model, where*

$$\begin{aligned} \varepsilon' &= \frac{\varepsilon}{q_s} \cdot \left(1 - \frac{1}{q_s + 1}\right)^{q_s+1} \approx \frac{\varepsilon}{q_s} \cdot \exp(-1) \\ t' &= t + 2^{15} \cdot \mathcal{O}(q_h \cdot \lambda^4). \end{aligned}$$

Theorem 13 (Jager-Kakvi-May [33]). *Assume 2- Φ HA $[\lambda]$ is (t', ε') -hard and gives an η -regular lossy trapdoor function. Then, for any (q_h, q_s) , X9.31 rDSA is $(t, \varepsilon, q_h, q_s)$ -UF-CMA secure in the Random Oracle Model, where*

$$\begin{aligned} \varepsilon &= \left(\frac{2\eta - 1}{\eta - 1}\right) \cdot \varepsilon' \\ t &= t' + 2^{15} \cdot \mathcal{O}(q_h \cdot \lambda^4) \end{aligned}$$

We now compare these proofs and their effect on parameter selection in Table 5, concretely for the case of 1024 bit moduli. We present these in a similar manner the that of Table 1. Here we also provide the size of the modulus used in the reduction, but this does not directly correspond to the security of the scheme, i.e. the scheme is not necessarily as secure as the assumption. The scheme is as secure as the assumption with a modulus of bit length given in the ‘‘Equiv. modulus’’ column. To compute these values we have taken $q_h = 2^{60}$, $q_s = 2^{30}$. As with Table 1, the key sizes are approximate due to the complexity of computing exact key sizes.

8 Comparison

Having now examined all the schemes, we now present a complete comparison. Unlike in the case of Tables 2, 3, 4, 5, we do not take any concrete figures, but

Table 5. Parameter sizes and security for X9.31 rDSA with a 1024 bit modulus

Scheme	Proof methodology	Assumption	No. prime factors	Exponent e	$ \mathbb{H}(\cdot) $	Equiv. modulus
X9.31 rDSA	Coron	2-FACT[1024]	2	2	≥ 623	≈ 544
	Jager-Kakvi-May	2-RSA[512]	3	Arbitrary	≥ 512	≈ 265
		+ 2v3PA[1024]	2	Arbitrary	≥ 512	≈ 265
	Jager-Kakvi-May	2- ϕ HA[512]	3	Prime $\leq 2^{256}$	≥ 512	≈ 497
		+ 2v3PA[1024]	2	Prime $\leq 2^{256}$	≥ 512	≈ 497

Table 6. Comparison of the security of the RSA hash-and-sign signatures

Scheme	Proof methodology	Assumption	No. prime factors	Exponent e	$ \mathbb{H}(\cdot) $	Security loss
RSA-FDH	Bellare-Rogaway	2-RSA[λ]	2	Arbitrary	λ	q_h
	Coron	2-RSA[λ]	2	Arbitrary	λ	q_s
	Kakvi-Kiltz	2- ϕ HA[λ]	2	Prime $\leq 2^{\lambda/4}$	λ	$o(1)$
RSASSA-PKCS1-v1.5	Coron	2-FACT[λ]	2	2	$\geq 2\lambda/3$	q_s
	Jager-Kakvi-May	2-RSA[$\lambda/2$]	3	Arbitrary	$\geq \lambda/2$	q_s
		+ 2v3PA[λ]	2	Arbitrary	$\geq \lambda/2$	q_s
	Jager-Kakvi-May	2- ϕ HA[$\lambda/2$]	3	Prime $\leq 2^{\lambda/4}$	$\geq \lambda/2$	$o(1)$
		+ 2v3PA[λ]	2	Prime $\leq 2^{\lambda/4}$	$\geq \lambda/2$	$o(1)$
ISO 9769-2 Scheme 1	Coron	2-FACT[λ]	2	2	$\geq 2\lambda/3$	q_s
	Jager-Kakvi-May	2-RSA[$\lambda/2$]	3	Arbitrary	$\geq \lambda/2$	q_s
		+ 2v3PA[λ]	2	Arbitrary	$\geq \lambda/2$	q_s
	Jager-Kakvi-May	2- ϕ HA[$\lambda/2$]	3	Prime $\leq 2^{\lambda/4}$	$\geq \lambda/2$	$o(1)$
		+ 2v3PA[λ]	2	Prime $\leq 2^{\lambda/4}$	$\geq \lambda/2$	$o(1)$
X9.31 rDSA	Coron	2-FACT[λ]	2	2	$\geq 2\lambda/3$	q_s
	Jager-Kakvi-May	2-RSA[$\lambda/2$]	3	Arbitrary	$\geq \lambda/2$	q_s
		+ 2v3PA[λ]	2	Arbitrary	$\geq \lambda/2$	q_s
	Jager-Kakvi-May	2- ϕ HA[$\lambda/2$]	3	Prime $\leq 2^{\lambda/4}$	$\geq \lambda/2$	$o(1)$
		+ 2v3PA[λ]	2	Prime $\leq 2^{\lambda/4}$	$\geq \lambda/2$	$o(1)$

we instead use the parameters, to allow for a more general comparison. We first compare all our signatures in Table 6.

As we can see from the tables above, there is a wide variety of schemes and proofs, each with advantages and disadvantages, with no clear best or worst scheme. While it would be ideal to be able to state with certainty that one scheme is superior to others, the variety of parameter choices mean that one would have to select the scheme best suited for their purposes. This decision would be based on the specific use case and the factors therein e.g. hardware or communication constraints. For example, if we have a device with constrained storage, we would

want to keep the key size as low as possible, which would mean we would have to avoid any schemes proven using the Jager-Kakvi-May methodology. On the other hand if storage is not an issue, but we have computational constraints, then one might consider picking the scheme that requires the smallest hash function. In which case, the schemes proven with the Jager-Kakvi-May methodology would be good candidates. Furthermore, in a system where we do not expect a large number of signatures, a loss of q_s might lead to acceptable parameters. It remains an open question to get a tight, parameter preserving proof for a deterministic standardised signature.

Acknowledgements. The authors would like to thank the anonymous reviewers of SSR 2020 for their insightful comments. We would also like to thank Cathy Meadows and Ruqayya Shaheed for their editorial comments.

References

1. ANSI: Digital signatures using reversible public key cryptography for the financial services industry (rDSA). Technical report X9.31, American National Standards Institute, New York, New York, USA (1998)
2. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: Denning, D.E., Pyle, R., Ganesan, R., Sandhu, R.S., Ashby, V. (eds.) ACM CCS 93, pp. 62–73. ACM Press (1993). <https://doi.org/10.1145/168588.168596>
3. Bellare, M., Rogaway, P.: The exact security of digital signatures-how to sign with RSA and Rabin. In: Maurer, U. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 399–416. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68339-9_34
4. Bellare, M., Rogaway, P.: PSS: provably secure encoding method for digital signatures. Submission to IEEE P1363 Working Group (1998)
5. Bellare, M., Yung, M.: Certifying cryptographic tools: the case of trapdoor permutations. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 442–460. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-48071-4_31
6. Bellare, M., Yung, M.: Certifying permutations: noninteractive zero-knowledge based on any trapdoor permutation. *J. Cryptol.* **9**(3), 149–166 (1996). <https://doi.org/10.1007/BF00208000>
7. Bleichenbacher, D.: Generating ElGamal signatures without knowing the secret key. In: Maurer, U. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 10–18. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68339-9_2
8. Bleichenbacher, D.: Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 1–12. Springer, Heidelberg (1998). <https://doi.org/10.1007/BFb0055716>
9. Cachin, C., Micali, S., Stadler, M.: Computationally private information retrieval with polylogarithmic communication. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 402–414. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_28
10. Coppersmith, D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *J. Cryptol.* **10**(4), 233–260 (1997). <https://doi.org/10.1007/s001459900030>
11. Coppersmith, D., Halevi, S., Jutla, C.: ISO 9796–1 and the new forgery strategy (working draft). Submission to IEEE P1363 Working Group (1999)

12. Coron, J.-S.: On the exact security of full domain hash. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 229–235. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44598-6_14
13. Coron, J.S.: Optimal security proofs for PSS and other signature schemes. Cryptology ePrint Archive, Report 2001/062 (2001). <http://eprint.iacr.org/2001/062>
14. Coron, J.-S.: Optimal security proofs for PSS and other signature schemes. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 272–287. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-46035-7_18
15. Coron, J.-S.: Security proof for partial-domain hash signature schemes. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 613–626. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_39
16. Coron, J.-S., Naccache, D., Stern, J.P.: On the security of RSA padding. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 1–18. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_1
17. Coron, J.S., Naccache, D., Tibouchi, M., Weinmann, R.P.: Practical cryptanalysis of ISO 9796–2 and EMV signatures. *J. Cryptol.* **29**(3), 632–656 (2016). <https://doi.org/10.1007/s00145-015-9205-5>
18. Davida, G.I.: Chosen signature cryptanalysis of the RSA (MIT) public key cryptosystem. University of Wisconsin, Milwaukee, Technical report (1982)
19. de Jonge, W., Chaum, D.: Attacks on some RSA signatures. In: Williams, H.C. (ed.) CRYPTO 1985. LNCS, vol. 218, pp. 18–27. Springer, Heidelberg (1986). https://doi.org/10.1007/3-540-39799-X_3
20. Degabriele, J.P., Lehmann, A., Paterson, K.G., Smart, N.P., Strefer, M.: On the joint security of encryption and signature in EMV. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 116–135. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27954-6_8
21. Denning, D.E.: Digital signatures with RSA and other public-key cryptosystems. *Commun. ACM* **27**(4), 388–392 (1984). <https://doi.org/10.1145/358027.358052>
22. Desmedt, Y., Odlyzko, A.M.: A chosen text attack on the RSA cryptosystem and some discrete logarithm schemes. In: Williams, H.C. (ed.) CRYPTO 1985. LNCS, vol. 218, pp. 516–522. Springer, Heidelberg (1986). https://doi.org/10.1007/3-540-39799-X_40
23. Girault, M., Misarsky, J.-F.: Selective forgery of RSA signatures using redundancy. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 495–507. Springer, Heidelberg (1997). https://doi.org/10.1007/3-540-69053-0_34
24. Girault, M., Misarsky, J.-F.: Cryptanalysis of countermeasures proposed for repairing ISO 9796-1. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 81–90. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45539-6_6
25. Goldberg, S., Reyzin, L., Sagga, O., Baldimtsi, F.: Efficient noninteractive certification of RSA moduli and beyond. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019. LNCS, vol. 11923, pp. 700–727. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34618-8_24
26. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.* **17**(2), 281–308 (1988)
27. Gordon, J.A.: How to forge RSA key certificates. *Electron. Lett.* **21**(9), 377–379 (1985). <https://doi.org/10.1049/el:19850269>
28. Guillou, L.C., Quisquater, J.-J., Walker, M., Landrock, P., Shaer, C.: Precautions taken against various potential attacks. In: Damgård, I.B. (ed.) EUROCRYPT 1990. LNCS, vol. 473, pp. 465–473. Springer, Heidelberg (1991). https://doi.org/10.1007/3-540-46877-3_42

29. ISO: Information technology - security techniques - digital signature schemes giving message recovery - part 2: Mechanisms using a hash-function. ISO 9796-2:1997, International Organization for Standardization, Geneva, Switzerland (1997). <https://www.iso.org/standard/28232.html> (WITHDRAWN)
30. ISO: Information technology - security techniques - digital signature schemes giving message recovery - part 2: Integer factorization based mechanisms. ISO 9796-2:2002, International Organization for Standardization, Geneva, Switzerland (2002). <https://www.iso.org/standard/35455.html> (WITHDRAWN)
31. ISO: Information technology - security techniques - digital signatures with appendix - part 2: Integer factorization based mechanisms. ISO 14888-2:2008, International Organization for Standardization, Geneva, Switzerland (2008). <https://www.iso.org/standard/44227.html>
32. ISO: Information technology - security techniques - digital signature schemes giving message recovery - part 2: Integer factorization based mechanisms. ISO 9796-2:2010, International Organization for Standardization, Geneva, Switzerland (2010). <https://www.iso.org/standard/54788.html>
33. Jager, T., Kakvi, S.A., May, A.: On the security of the PKCS#1 v1.5 signature scheme. In: Lie, D., Mannan, M., Backes, M., Wang, X. (eds.) ACM CCS 2018, pp. 1195–1208. ACM Press (2018). <https://doi.org/10.1145/3243734.3243798>
34. Jonsson, J., Kaliski, B.: Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1. RFC 3447 (Informational), February 2003. Obsoleted by RFC 8017. <https://doi.org/10.17487/RFC3447>, <https://www.rfc-editor.org/rfc/rfc3447.txt>
35. Jonsson, J.: Security proofs for the RSA-PSS signature scheme and its variants. Cryptology ePrint Archive, Report 2001/053 (2001). <http://eprint.iacr.org/2001/053>
36. Kakvi, S.A.: On the security of RSA-PSS in the wild. In: Mehrnezhad, M., van der Merwe, T., Hao, F. (eds.) Proceedings of the 5th ACM Workshop on Security Standardisation Research Workshop, London, UK, 11 November 2019, pp. 23–34. ACM (2019). <https://doi.org/10.1145/3338500.3360333>
37. Kakvi, S.A., Kiltz, E.: Optimal security proofs for full domain hash, revisited. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 537–553. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_32
38. Kakvi, S.A., Kiltz, E.: Optimal security proofs for full domain hash, revisited. *J. Cryptol.* **31**(1), 276–306 (2018). <https://doi.org/10.1007/s00145-017-9257-9>
39. Kakvi, S.A., Kiltz, E., May, A.: Certifying RSA. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 404–414. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34961-4_25
40. Kaliski, B.: PKCS #1: RSA Encryption Version 1.5. RFC 2313 (Informational), March 1998. 10.17487/RFC2313, obsoleted by RFC 2437. <https://www.rfc-editor.org/rfc/rfc2313.txt>
41. Kaliski, B., Staddon, J.: PKCS #1: RSA Cryptography Specifications Version 2.0. RFC 2437 (Informational), October 1998. 10.17487/RFC2437, obsoleted by RFC 3447. <https://www.rfc-editor.org/rfc/rfc2437.txt>
42. Kaliski, B.: From PKC to PKI: Reflections on standardizing the RSA algorithm (2019). <https://youtu.be/sqsDKjPaJVg>
43. Kaliski, B. (ed.): IEEE standard specifications for public-key cryptography. IEEE Std 1363-2000, pp. 1–228, August 2000. <https://doi.org/10.1109/IEEESTD.2000.92292>, <https://ieeexplore.ieee.org/servlet/opac?punumber=7168>

44. Lenstra, A.K.: Unbelievable security *matching AES security using public key systems*. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 67–86. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_5
45. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. *Math. Ann.* **261**(4), 515–534 (1982). <https://doi.org/10.1007/BF01457454>
46. May, A.: Using LLL-reduction for solving RSA and factorization problems. In: Nguyen, P., Vallée, B. (eds.) *The LLL Algorithm. Information Security and Cryptography*, pp. 315–348. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-02295-1_10
47. Menezes, A.: Evaluation of security level of cryptography: RSA signature schemes (2002). <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.612.1271&rep=rep1&type=pdf>
48. Misarsky, J.-F.Ç.: A multiplicative attack using LLL algorithm on RSA signatures with redundancy. In: Kaliski, B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 221–234. Springer, Heidelberg (1997). <https://doi.org/10.1007/BFb0052238>
49. Moriarty, K. (ed.), Kaliski, B., Jonsson, J., Rusch, A.: PKCS #1: RSA Cryptography Specifications Version 2.2. RFC 8017 (Informational), November 2016. 10.17487/RFC8017, <https://www.rfc-editor.org/rfc/rfc8017.txt>
50. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC, pp. 187–196. ACM Press (2008). <https://doi.org/10.1145/1374376.1374406>
51. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems. *Commun. Assoc. Comput. Mach.* **21**(2), 120–126 (1978)