Efficient Fail-Stop Signatures from the Factoring Assumption

Atefeh Mashatan and Khaled Ouafi

Ecole Polytechnique Fédérale de Lausanne (EPFL),
CH-1015 Lausanne, Switzerland
`Atefeh.Mashatan@epfl.ch, Khaled.Ouafi@epfl.ch`

Abstract. In this paper, we revisit the construction of fail-stop signatures from the factoring assumption. These signatures were originally proposed to provide information-theoretic-based security against forgeries. In contrast to classical signature schemes, in which signers are protected through a computational conjecture, fail-stop signature schemes protect the signers in an information theoretic sense, i.e., they guarantee that no one, regardless of its computational power, is able to forge a signature that cannot be detected and proven to be a forgery. Such a feature inherently introduced another threat: malicious signers who want to deny a legitimate signature.
Many construction of fail-stop signatures were proposed in the literature, based on the discrete logarithm, the RSA, or the factoring assumptions. Several variants of this latter assumption were used to construct fail-sop signature schemes. Bleumer et al. (EuroCrypt '90) proposed a fail-stop signature scheme based on the difficulty of factoring large integers and Susilo et al. (The Computer Journal, 2000) showed how to construct a fail-stop signature scheme from the so-called "strong factorization" assumption. A later attempt by Schmidt-Samoa (ICICS '04) was to propose a fail-stop signature scheme from the p^2q factoring assumption.
Compared to those proposals, we take a more traditional approach by considering the Rabin function as our starting point. We generalize this function to a new bundling homomorphism while retaining Rabin's efficient reduction to factoring the modulus of the multiplicative group. Moreover, we preserve the efficiency of the Rabin function as our scheme only requires two, very optimized, modular exponentiations for key generation and verification. This improves on older constructions from factoring assumptions which required either two unoptimized or four exponentiations for key generation and either two unoptimized or three modular exponentiations for verifying.

Key words: Fail-stop signature schemes, Digital signatures, Rabin function, Factoring

1 Introduction

Digital signatures are one of the main achievements of public key cryptography: they are the main primitive that ensures authenticity of data transmitted through an insecure channel. In being so, digital signatures were originally designed to be the "digital" equivalent to handwritten signatures. Their underlying mechanism is simple and elegant: to each person is associated a "public" key that can be used to verify any signature this person has produced for a document or certificate. To each such key, that is publicly known, is associated a so-called "private" key that is known to the signer only and is used to sign electronic messages or documents. Naturally, this private key should not be computable from the public key. A computational assumption believed to be "impossible" to solve is usually used to prevent this scenario.

However, when coming to the practical world, the question of forgeries was quickly raised by lawyers and judges. This case was even more critical in countries like Germany in which digital signatures have a legal value. However, cryptographers initially ignored the problem of forgery and repudiation, that is because the whole security of a digital signature scheme stands on the shoulders of a problem that we trust to be "hard" to solve. So, as long as the computational assumption holds, forgeries do not happen, except with very small probability that we also hope to never occur. Nevertheless, from a legal point of view, forgeries do exist and signers have to be given the ability to defend themselves against them.

Fail-stop signatures were designed to address this problem: they provide the signer with a mean to prove that a signature is a forgery and that the cryptographic assumption holding the security of the scheme has been broken. So, in the case of a dispute, the signer can exhibit the evidence and deny any responsibility on it. Furthermore, such a forgery shows that the scheme has become insecure, and thus has to be stopped, hence the denomination "fail-stop". Since they consider the extreme eventuality of the break of a hard cryptographic problem, fail-stop signature schemes should protect the signers against computationally unlimited adversaries. However, granting to signers the ability to prove forgeries is not without risk. Indeed, a malicious signer may try to prove that a signature he has produced is a forgery. This way, he can free himself from any commitment the signed document may induce. Unfortunately, it is theoretically impossible to ensure security against computationally unlimited malicious signers if we assume that the forgers are so. Instead, the security against a malicious signer is only computational, i.e., it assumes that solving the hard problem for this party is infeasible.

1.1 Previous Work

While the most popular and efficient scheme is based on the discrete logarithm problem [19], few convincing schemes from the factoring assumption were proposed in the literature. Surprisingly, the first fail-stop signature scheme was based on the factorization assumption [3]. More precisely, it was originally related to the notion of claw-free pairs of permutations in [3] but then reformulated in [11] to explicitly relate the security of the scheme to the factoring assumption. This scheme, to which we refer as the QR scheme, is based on the hardness of factoring an RSA modulus $n = pq$ with factors being such that $p \equiv 3 \mod 8$ and $q \equiv 7 \mod 8$. Up to date, it is unclear how to compare the difficulty of factoring such moduli compared to the classical RSA ones. Moreover, the scheme seems to be made so that it works so its structure is not very "natural".

Later, Susilo et al. proposed another scheme [17], that, following [15], we refer to as the order scheme, based on the difficulty of factoring an RSA modulus $n = pq$ when the algorithm is also given an element from a superset $\mathbb{Z}_P^\star$ of $\mathbb{Z}_n^\star$ (such that P is prime and a that $P-1$ is a multiple of n) of order q. This scheme has been shown to be insecure by Schmidt-Samoa [15] but was also repaired in the same paper by reducing the message space form $\mathbb{Z}_n^\star$ to $\mathbb{Z}_p$. This fix affected the efficiency of the scheme as it made signatures around three times larger than their corresponding messages.

In parallel, Schmidt-Samoa proposed a scheme based on the hardness of factoring RSA moduli of the form $n = p^2q$ [15] (we refer to this one as the p^2q scheme). Although it is probably the most elegant scheme of those mentioned before, this scheme inherently requires larger sizes for the modulus compared to the other two. Recently, Susilo proposed a fail-signature scheme from the factoring and discrete logarithm assumption [16]. However, we note that its security proof does not make any reduction to the factoring problem but rather to the discrete logarithm problem in an RSA group. Therefore, we do not consider it based on a factoring assumption.

1.2 Our Contribution

We revisit the design of fail-stop signatures from the factoring assumption. Our starting point is the Rabin function, $f(x) = x^2 \mod n$, whose invertibility is known to be equivalent to the problem of factoring the integer n. As a consequence of the Chinese Remainder theorem, when $n = pq$, a classical RSA modulus, this function maps four integers to a single one. In this work, we use a generalization of the Rabin function, $f(x) = x^a \mod n$, for a a that divides the order of the group $\mathbb{Z}_n^\star$. In practice, we will construct such moduli by choosing a p that satisfies $p-1 \equiv 0 \mod a$. We remark that this type of RSA moduli, with a publicly known, was already used by Benaloh to define a public key encryption scheme [4].

As we show later, the function $f(x) = x^a \mod n$ enjoys a number of nice properties. At first, using a result from number theory due to Frobenius [6], we prove that this function actually maps a elements of $\mathbb{Z}_n^\star$ to a single a-th residue of the same group. The second property, inherited from the Rabin function is that finding collisions is provably as hard as factoring the modulus with the knowledge of a. The last observation is that both proofs for the results above do not take into consideration the structure of a, but only on its size. Consequently, numbers of the form $2^s + 1$ will be used for a as it allows fast computation of the modular exponentiation in s multiplications using the square-and-multiply method, resulting in a quadratic complexity for computing the function f. All these properties allow us to use this function as a bundling homomorphism and then instantiate Pfitzmann's general construction [12] of fail-stop signatures to yield the most efficient scheme based on a factoring assumption.

The rest of the paper is structured as follows. Section 2 provides the necessary background, definitions of fail-stop signature schemes and the factoring intractability assumption. We then dedicate Section 3 to our instantiation of the fail-stop signature scheme and prove its security. Efficiency analysis is discussed in Section 4.

2 Preliminaries

2.1 Notations and Negligible Probabilities

Throughout this paper, we use the expression $y \leftarrow \mathcal{A}(x)$ to mean that y is assigned the output of algorithm $\mathcal{A}$ running on input x. An algorithm is said to be polynomial if its running time can be expressed as a polynomial in the size of its inputs.

If X denotes a set, $|X|$ denotes its cardinality and $x \in_R X$ expresses that x is chosen from X according to the uniform distribution. If X and Y are sets, then the relative complement of X in Y, denoted $X \setminus Y$, is the set of elements in X, but not in Y.

We also recall the classical notions of negligible function and one-way function.

Definition 1 (Negligible Function). *A function $f : \mathbb{N} \to \mathbb{R}$ is negligible in k if for any positive polynomial function $p(\cdot)$ there exists k_0 such that:*

$$k \geq k_0 \Rightarrow f(k) < \frac{1}{p(k)}.$$

Negligible functions in k are denoted $\mathsf{negl}(k)$.

2.2 Number Theoretic and Factoring Background

We call a prime number p to be a-strong, if $p = 2ap' + 1$ for a prime $p' > 2a$. Note that such prime numbers are called strong by Rivest and Silverman [14] and that for $a = 1$ we get the usual definition of strong primes.

In the following, we consider a probabilistic polynomial-time algorithm Gen that, on input a security parameter 1^k, that picks an integer a and generates a random a-strong prime $p = 2ap' + 1$ and a regular, not a-strong, prime number q, such that $\lfloor \log_2 p \rfloor = \lfloor \log_2 q \rfloor = \ell_{\mathsf{F}}(k)$, where $\ell_{\mathsf{F}}(k)$ denotes a function that represents, for any given security parameter k, the recommended (bit-)size of the RSA modulus $n = pq$. In the end, Gen outputs $n = pq$ and a along with p and q. The following definition formalizes the main assumption that will be used throughout this work.

Definition 2 (The Factorization Assumption with a-strong primes). *Let us consider a probabilistic polynomial-time algorithm $\mathcal{A}_{\mathsf{FACT}}$ who takes as input an RSA modulus $n = pq$ and an odd integer a, such that p, q, and ${}^{p-1}/_{2a}$*

are all prime numbers, and outputs p and q . The factoring assumption states that for any such $\mathcal{A}_{\mathsf{FACT}}$, we have

$$\Pr\left[(p,q) \leftarrow \mathcal{A}_{\mathsf{FACT}}(1^k, n, a) \,\middle|\, (n,a,p,q) \leftarrow \mathsf{Gen}(1^k, a)\right] = \mathsf{negl}(k).$$

The probability is taken over the random tapes of Gen *and* $\mathcal{A}_{\mathsf{FACT}}$.

While this type of prime numbers has already been used before by Benaloh to construct a public encryption scheme [4], we refer the reader to the work of Groth [8] which makes a more detailed treatment of the factorization of such numbers by Pollard's rho method [13] and other factoring algorithms such as the general number field sieve.

In the rest of this section, we concentrate on exponentiation to a and a-th residuosity. It is well known that every $x \in \mathbb{Z}_n^\star$ has a unique a-th root if and only if $\mathsf{gcd}(a, \varphi(n)) = 1$, where $\varphi(\cdot)$ denotes the Euler totient function. In fact, this ensures the correctness of the RSA cryptosystem. However, when a and $\varphi(n)$ are not coprime, an element of $\mathbb{Z}_n^\star$ may admit multiple a-th roots as stated by the following theorem.

Theorem 1 (Frobenius [6]). *If a divides the order of a group, then the number of elements in the group whose order divides a is a multiple of a. If the group is cyclic, then this number is exactly a.*

Note that $\mathbb{Z}_n^\star$ is not cyclic when n is an RSA modulus. However, the next theorem proves that the number of a-th roots of any element in $\mathbb{Z}_n^\star$ is *exactly* equal to a in the class of RSA moduli we consider in this paper.

Theorem 2. *Let a be an odd number and $n = pq$ be an RSA modulus such that p is an a-strong prime, and q is a regular prime number that satisfies $\mathsf{gcd}(a, q-1) = 1$. If y is an a-th residue in $\mathbb{Z}_n^\star$, then the equation $g^a = y \mod n$ admits exactly a solutions.*

Proof. Let $\mathsf{CRT} : \mathbb{Z}_n^\star \to \mathbb{Z}_p^\star \times \mathbb{Z}_q^\star$ denote the isomorphism induced by the chinese remainder theorem. By applying CRT, if g is a solution to the equation $g^a = y \mod n$, then $(g_p, g_q) = (g \mod p, g \mod q)$ is a solution to the two equations

$$(g_p)^a = y \mod p, \tag{1}$$
$$(g_q)^a = y \mod q. \tag{2}$$

On one hand, Recalling that $\mathbb{Z}_p^\star$ is cyclic when p is prime, we can apply Thm. 1 to deduce that Eq. (1) has exactly a solutions (remember that a divides $\varphi(p) = ap'$). On another hand, since $\mathsf{gcd}(a, \varphi(q)) = 1$, there must exist a'_q such that $a \cdot a'_q = 1 \pmod{q-1}$ and Eq. (2) can be rewritten as $g_q = y^{a'_q} \mod q$. Hence, Eq. (2) admits one unique solution.

As there exist a tuples of the form (g_p, g_q) that satisfy equations (1) and (2), by applying CRT^{-1}, we deduce that the number of a-th roots of y in $\mathbb{Z}_n^\star$ is exactly a. $\square$

2.3 Fail-stop Signatures

We briefly review the basic definition of fail-stop signature schemes and their security properties. A complete and more formal definition can be found in [5].

The idea of fail-stop signatures is to associate, to each possible message m, a number of signatures s that passes the verification test with the public key. These signatures are called acceptable signatures. However, the signer should not be able to construct more than one signature from the secret key. This one signature is called the valid signature. Of course, the set of acceptable signatures must be small in comparison with the signature space, so that it should be difficult for a computationally bounded signer to find an acceptable signature different from his own.

In all cases, an adversary with unlimited computational power can always compute the set of acceptable signatures (one way is to try the verification algorithm with every element of the signature space). So, in order to achieve security against such an adversary, we have to consider an information-theoretic security determined by a security parameter $\sigma \in \mathbb{N}$: an adversary with unlimited computational power should not have enough information to distinguish the valid signature in the set of acceptable ones except with a probability upper-bounded by $2^{-\sigma}$ (the probability to guess it successfully). This property is called the *signer's security.*

In parallel, verifiers must be secure against signers who, by computing a valid proof of forgery, manage to disavow one of their own legitimate signatures. The security against these signers can only be computational which means that the probability of a signer disavowing his own signature is negligible in k. This property is known as the *verifier's security.*

Definition 3 (Fail-Stop Signature Schemes). *A fail-stop signature scheme is defined by the following five polynomial-time algorithms:*

– $\mathsf{KeyGen}(1^k, 1^\sigma, 1^N) \longrightarrow (sk, pk)$. *This is a probabilistic polynomial-time, possibly interactive, protocol run by the signer and the verifier (or a trusted center) runs on security parameters k, σ and an integer N representing the number of signatures the signer can release. At the end of the protocol, he signer obtains a private key sk which can be used to sign at most N messages (This type of signature schemes are said to be N-times). The other party obtains the corresponding public key pk. For the sake of simplicity, the input 1^N is omitted when the scheme is meant to be used to sign one message only.*

– $\mathsf{Sign}(sk, i, m) \longrightarrow s$. *Given a message m, a counter $i \in \{1 \ldots N\}$, incremented at each invocation of this algorithm, and the private key sk, this (probabilistic) polynomial-time algorithm creates a valid signature s for the message m. When the scheme is a one-time signature scheme, i.e., $N = 1$, the input i is omitted.*

– $\mathsf{Verify}(pk, m; s) \longrightarrow \{0, 1\}$. *Given a signature s for m and a public key pk,* Verify *is a deterministic polynomial-time algorithm that outputs 1 if the signature is acceptable, otherwise it outputs 0.*

– $\mathsf{ProveForgery}(sk, m, s) \longrightarrow \{\mathsf{pr}, \perp\}$. *On input an acceptable signature s on m, and private key sk, this algorithm outputs a bit-string* pr *or $\perp$ in case of failure.*
– $\mathsf{VerifyProof}(pk, m, s, \mathsf{pr}) \longrightarrow \{0, 1\}$. *A polynomial-time algorithm that takes on input pk, an acceptable signature s of a message m and a proof of forgery* pr *and outputs either 1, meaning that the proof is valid, or 0, meaning that the proof is invalid.*

Additionally, any fail-stop signature scheme should also satisfy two correctness properties: every signature honestly produced using Sign is acceptable and every proof computed using $\mathsf{ProveForgery}$ passes the verification. This is more formally defined hereafter.

Definition 4 (Correctness of a Fail-Stop Signature Scheme). *We say that a fail-stop signature scheme is correct if the two conditions hold:*

1. *Every honestly generated signature is valid, i.e.,*

$$\forall \lambda, N, m \in M : \Pr\left[1 \leftarrow \mathsf{Verify}(pk, m, s) \;\middle|\; \begin{array}{c}(pk, sk) \leftarrow \mathsf{Keygen}(1^{\lambda}, 1^{N}),\\ s \leftarrow \mathsf{Sign}(sk, m)\end{array}\right] = 1.$$

2. *Every honestly generated proof is valid, i.e.,*

$$\forall \lambda, N, m \in M, s \in S :$$
$$\Pr\left[1 \leftarrow \mathsf{VerifyProof}(pk, m, s, pr) \;\middle|\; \begin{array}{c}(pk, sk) \leftarrow \mathsf{Keygen}(1^{\lambda}, 1^{N}),\\ pr \leftarrow \mathsf{ProveForgery}(sk, m, s),\\ pr \neq \perp\end{array}\right] = 1.$$

Security against malicious signers and powerful forgers is formalized as follow.

Definition 5 (Security of a Fail-Stop Signature Scheme). *A fail-stop signature scheme with security parameters k and σ is said to be secure if the two properties hold*

1. ***Signer's security.*** *The probability that a computationally unlimited adversary knowing pk and the signature of N adaptively chosen messages $(m_1, s_1), \ldots, (m_N, s_N)$ outputs a pair (m, s) such that $(m, s) \neq (m_i, s_i), \forall i = 1 \ldots N$, and* $\mathsf{ProveForgery}(sk, m, s) = \perp$ *must be smaller than $2^{-\sigma}$.*
2. ***Verifier's security.*** *The probability that a polynomially bounded malicious signer generates a public key pk and then outputs a pair (m, s) and a proof of forgery* pr *such that* $\mathsf{VerifyProof}(pk, m, s, \mathsf{pr}) = 1$ *must be negligible in k.*

A constraining result for fail-stop signatures is that, in order to be able to sign N messages, the signer has to generate at least $(N+1)(k-1)$ secret random bits. We stress that these bits do not necessarily need to generated during the key generation as the signature algorithm may be probabilistic and update the internal state according to some random bits. Although this result seems a sever limitation against the practical implementation of fail-stop signature schemes, it has been proved in [18] that it is unavoidable if one hopes to achieve security against unbounded adversaries.

2.4 The General Construction using Bundling Homomorphisms

A general framework for constructing a fail-stop signature scheme secure for signing one message, on which are built all known secure fail-stop signature schemes, has been proposed by Pfitzmann [12]. It is based on the notion of bundling homomorphism. We note that classical techniques such as Merkle trees [9, 10], top-down authentication trees [7], and one-way accumulators [1, 2], can be used to extend a one-time fail-stop signature scheme to many messages.

Definition 6 (Family of Bundling Homomorphisms). *Let $\imath \in I$ be an indexing for a family of triples $(h_\imath, G_\imath, H_\imath)$ such that for all possible $\imath \in I$, $(G_\imath, +, 0)$ $(H_\imath, \times, 1)$ are Abelian groups and $h_\imath : G_\imath \to H_\imath$. $(h_\imath, G_\imath, H_\imath)$ is called a family of bundling homomorphisms with degree level 2^τ and collision-resistance security of level k if it satisfies:*

1. *For every $\imath \in I$, $h_\imath$ is an homomorphism.*
2. *There exist polynomial-time algorithms for sampling from I, computing $h_\imath$ and the group operations in $G_\imath$ and $H_\imath$, for every $\imath \in I$*
3. *For every $\imath \in I$, every image $y \in \mathsf{Im}(h_\imath)$ has at least 2^τ preimages in $G_\imath$. We call 2^τ the bundling degree of the homomorphism.*
4. *It is computationally infeasible to find collisions, i.e. for any probabilistic polynomial-time algorithm $\tilde{\mathcal{A}}$, we have*

$$\Pr\left[h_\imath(x_1 - x_2) = 1, \quad x_1 \neq x_2 \,\middle|\, \begin{array}{c} \imath \in_R I, \\ (x_1, x_2) \leftarrow \tilde{\mathcal{A}}(\imath) \end{array} \right] = \mathsf{negl}(k)$$

where the probability is taken among the random choice of $\imath$ and the random coins of $\tilde{\mathcal{A}}$.

The basic idea behind Pfitzmann's construction is to use the high bundling degree of a bundling homomorphism to "hide" the signer's secret key, which lies in the space of the G's, even against powerful adversaries who can invert the homomorphism (note that the collision resistance of bundling homomorphisms stated in point 4 of the definition implies resistance against preimage attacks).

The complete description of the construction is described below. For the sake of simplicity, we assume the case in which the key generation is run by the signer in conjonction with a center trusted by the verifiers (and not necessarily by the signer). This easily generalizes to the case of many verifiers [11].

– $\mathsf{KeyGen}(1^k, 1^\sigma)$. As a prekey, the center picks a random index K for the family of bundling homomorphisms $H_\imath$. It sets $h = h_K$, $G = G_K$, $H = H_K$ and also defines the finite message space $M \subset \mathbb{Z}$.
For prekey verification, the signer has to be convinced that h is a group homomorphism with bundling degree 2^τ for an appropriate τ to be later discussed. Once he is convinced, the signer chooses randomly his private key

$$sk = (sk_1, sk_2) \in_R G^2,$$

and computes the public key

$$pk = (pk_1, pk_2) = (h_K(sk_1), h_K(sk_2)) \in H^2.$$

– $\mathsf{Sign}(sk, m)$. For signing a message $m \in M$, compute

$$s = sk_1 + m \cdot sk_2,$$

where $m \cdot sk_2$ denotes the operation of performing m additions of sk_2 in G.

– $\mathsf{Verify}(pk, m, s)$. Check whether:

$$h(s) = pk_1 \times pk_2^m.$$

– $\mathsf{ProveForgery}(sk, m, s')$. Given s' a forgery for m that passes the verification, the signer computes $s = \mathsf{Sign}(sk, m)$ and exhibits

$$(m, s, s').$$

– $\mathsf{VerifyProof}(pk, m, s, \mathsf{pr})$. Given m and $(s, s') \in G^2$, verify that

$$s \neq s' \quad \wedge \quad h(s) = h(s')$$

For the security analysis of this construction, we will denote SK_{pk} the set of secret keys that correspond to the public key pk, i.e,

$$SK_{pk} = \{sk \in SK : h(sk) = pk\}.$$

Since h has a bundling degree of 2^τ, we must have $|SK_{pk}| \geq 2^\tau$. In consequence, given a public key (pk_1, pk_2) there are at least $2^{2\tau}$ private keys that match the signer's public key. Each of these private keys produces a signature that passes the verification. These keys can be computed by an adversary with unlimited computational power, but he cannot know which of these 2^{2^τ} keys is the signer's key that he used to sign the message. However, because of the equation $sk_1 + m \times sk_2 = s$, it is sufficient to find one key to determine the other, thus the number of possible private keys is reduced to 2^τ.

A signer is able to prove a forgery on a message m' only if his signature differs from the forged signature. To measure the probability that an adversary outputs the signer's signature we must analyze the number of pairwise different signatures that can be produced using the 2^τ private keys. This number is related to the number of possible secret keys that produce a valid signature. Some easy implications show that this number is upper-bounded by the size of the set

$$T = \{d \in G : h(d) = 1 \wedge (m' - m)d = 0\}.$$

In order to prove security, we have to consider the worst case with respect to the choice of the message, i.e., we consider the signature that can be produced by the maximum number of candidate secret keys,

$$T_{max} = \max_{m' \in M \setminus \{0\}} |\{d \in G : h(d) = 1 \wedge m'd = 0\}|.$$

A more detailled proof and analysis of this construction can be found in [11, 12].

Theorem 3 (Security of the General Construction [11, 12]). *Let a fail-stop signature scheme following the general construction above with security parameters k, τ. We do have*

1. *The scheme provides a level of security k for the verifiers.*
2. *If 2^τ is chosen such that $T_{max} \leq 2^{\tau-\sigma}$, then this scheme provides a level of security of σ for the signer.*

3 New Fail-Stop Signatures From Factoring

3.1 The Construction

Let us consider the set of RSA moduli $n = pq$ for which p is an a-strong prime number and $p - 1$ is co-prime with q. We define the following group homomorphism

$$\begin{aligned} H_a : \mathbb{Z}_n^\star &\longrightarrow \mathbb{Z}_n^\star \\ x &\longmapsto x^a \mod n. \end{aligned}$$

And using Pfitzmann's general construction, the following fail-stop signature scheme comes naturally:

KeyGen. On input σ, k the center chooses a σ-bit odd integer a and two equally sized primes p, q such that $\,^{p\,-\,1}\!/_{2a}$ is also a prime number and that $\mathsf{gcd}(q-1, a) = 1$. The prekey of the scheme then consists of $n = pq$ and a.

To verify that the prekey is correctly generated, the signer has to be provided with a zero-knowledge proof that a indeed divides $\varphi(n)$ (such a proof can easily be constructed from general zero-knowledge proofs). From that point, he generates the key material as follow

$$(sk_1, sk_2) \in_R \mathbb{Z}_n^{\star 2}, \qquad (pk_1, pk_2) = (sk_1^a \mod n, sk_2^a \mod n).$$

Sign. The message space is defined to be $M = \{1, \ldots, \varphi(n) - 1\} \setminus \{x > 1 | \mathsf{gcd}(a, x) \neq 1\}$. To sign a message $m \in M$, the signer computes

$$s = sk_1 \times sk_2^m \mod n$$

Verify. An element $s \in \mathbb{Z}_n^\star$ is an acceptable signature on $m \in M$ if and only if the following equality holds

$$s^a = pk_1 \times pk_2^m \mod n.$$

ProveForgery. On the happening of a forgery $(m, s^\star)$, the proof consists of, as states the general construction, the signer's signature s on the message m using his secret key (sk_1, sk_2).

VerifyProof. Given two signatures s and $s^\star$ on the same message m, the proof of forgery is valid if both signatures are different and $s^a = s^{\star a} \mod n$ (It will be shown in the next section that this equation leads to the factorization of n).

3.2 Security Analysis

To prove that h_n, described above, is a family of bundling homomorphisms, we first need to prove this lemma.

Lemma 1. *Let $n = pq$ and a be an odd number such that p, q are prime number and a divides ${}^{p-1}/_{2}$, ${}^{p-1}/_{2a}$ is prime and $\mathsf{gcd}(q-1, a) = 1$. If there exists a successful polynomial-time algorithm $\tilde{\mathcal{A}}$ who succeeds in finding x_1 and x_2 such that $h_n(x_1) = h_n(x_2)$ and $x_1 \neq x_2$, then there exists a successful polynomial-time algorithm against the factorization problem of Def. 2.*

Proof. We construct the algorithm against factoring as follows. First, it calls upon $\tilde{\mathcal{A}}$ and gets x_1 and x_2 such that

$$x_1^a = x_2^a \pmod{n}.$$

Since p and q are of the same size, it must be that $q > a$. Hence, $a \mod q-1 = a$. So,

$$x_1^a = x_2^a \pmod{q}.$$

Since $\mathsf{gcd}(a, q-1) = 1$, a is invertible modulo $q-1$ and it results that

$$x_1 = x_2 + k \cdot q, \quad \text{for } 1 \leq k < p$$

Hence, we conclude that $\mathsf{gcd}(x_1 - x_2, q) = q$.

So it is sufficient for $\mathcal{A}_{\mathsf{FACT}}$ to compute $q = \mathsf{gcd}(x_1 - x_2, n)$ and then deduce $p = {}^{n}/_{q}$. Note that $\mathcal{A}_{\mathsf{FACT}}$ runs in polynomial-time if $\tilde{\mathcal{A}}$ does so. Moreover, it has the same success probability of $\tilde{\mathcal{A}}$. □

The following theorem proves that h_n, described above, is a family of bundling homomorphisms of degree 2^σ:

Theorem 4. *Under the Factorization Assumption of Def. 2, the construction above is a family of bundling homomorphisms with bundling degree 2^σ .*

Proof. It is trivial to see that h is an homomorphism.

Regarding its bundling degree, we recall Thm. 2 which states that the number of solutions of the equation $x^a = y \mod n$ is a when y is an a-th residue of $\mathbb{Z}_n^\star$. Setting $y = 1$ trivially leads the kernel of the homomorphism. The kernel is thus of size a which is lower-bounded by 2^σ. □

At this point, we state the following theorem, which asserts the security of our construction.

Theorem 5. *The fail-stop signature scheme defined in Sec. 3.1 is secure under the assumption that factoring integers $n = pq$ with the knowledge of a divisor of $\varphi(n)$ is hard.*

Proof. As it has already been show in Thm. 4 that h_n is a family of bundling homomorphisms, we only need to prove security for the signer. For that sake, and according to Thm. 3, we analyse the size of the following set (Here the neutral element is 1):

$$\begin{aligned} T_{max} &= \max_{m' \in M \setminus \{1\}} \left| \{d \in \mathbb{Z}_n^\star : h_n(d) = 1 \wedge d^{m'} = 1\} \right| \\ &= \max_{m' \in M \setminus \{1\}} \left| \{d \in \mathbb{Z}_n^\star : d^a = 1 \wedge d^{m'} = 1\} \right| \\ &= \max_{m' \in M \setminus \{1\}} \left| \{d \in \mathbb{Z}_n^\star : \mathsf{ord}_{\mathbb{Z}_n^\star}(d) | a \wedge \mathsf{ord}_{\mathbb{Z}_n^\star}(d) | m'\} \right| \\ &= \max_{m' \in M \setminus \{1\}} \left| \{d \in \mathbb{Z}_n^\star : \mathsf{ord}_{\mathbb{Z}_n^\star}(d) | \mathsf{gcd}(a, m')\} \right| \\ &= 1 \end{aligned}$$

since $\mathsf{gcd}(m', a) = 1$ by definition of M.

Hence we conclude that taking a bundling homomorphism with degree 2^σ is sufficient to achieve security against powerful adversaries. □

4 Efficiency Analysis

Among the five algorithms of the scheme, the key generation is certainly the most expensive operation to perform. Generating strong primes of our specified form is done in time $O\left(\ell_{\mathsf{F}}^4(\lambda) + (1 + \sigma + \ell_{\mathsf{F}}(\lambda))^4\right)$. However, this is not a critical issue as it is run only once. Note that this complexity is in fact very close to the key generation of RSA with strong primes [14].

Using some simple optimization techniques, we can drastically reduce the complexity of modular exponentiations to the power of a by choosing values for a with very low (or very high) Hamming weight. Indeed, the security proof of our scheme is based the *size* of a and not its Hamming weight. According to the state of the art of factoring algorithms such that ECM and NFS, assuming low Hamming weight for a does not help the adversary factoring. This way, one can use the classical square-and-multiply technique and set $a = 2^\alpha \pm 1$, for some $\alpha \geq \sigma$. This trick allows us to reduce the complexity of all exponentiations to a from $O(\ell_{\mathsf{F}}^3(\lambda))$ to $O(\sigma \ell_{\mathsf{F}}^2(\lambda))$. For practical applications in which the one-time key has to be regenerated at each signature, such a feature is clearly a benefit. Verification also benefit from this optimisations as only one modular exponentiation (to m) remains to be done.

For typical values $k = \sigma = 80$, we need to generate a 80-bit odd a, e.g., $a = 2^{80} + 1$, and a 1024-bit RSA modulus composed of one a-strong prime number. Key generation is then performed using $40 + 1 = 41$ modular multiplications. Verification is also performed using 41 modular multiplications and one exponentiation.

As our scheme has its operations performed in a smaller group with optimized exponents, it clearly outperforms the p^2q scheme in terms of efficiency and signature length. Since this latter is more efficient than the order scheme [15], we omit

Table 1. Comparaison of efficiency parameters of the most efficient factorization-based fail-stop signature schemes. For better readability and easier notations, we let k denote the length of the RSA moduli and define $\rho = \tau - \sigma$. Note that we set $a = 2^\sigma + 1$ in our scheme so that it does not need to be explicitly included in the public-key.

	Size of sk	Size of pk	Size of m	Size of s	Sign (# mult.)	Verify (# mult.)
QR Scheme	$2(\rho + \sigma + k)$	$2k$	ρ	$\rho + \sigma + k$	ρ	$< 2\rho + \sigma$
Our scheme	$2k$	$2k$	σ	$2k$	σ	$< 2\sigma$

the order scheme from the comparison. Regarding the QR scheme, we provide Table 1 to put a clear comparison with our scheme. It turns out that in term of size of the keys and the signatures, our scheme behaves better than the QR scheme. Regarding the efficiency of the other parameters and algorithms, having a direct comparison is more difficult as the parameter ρ of the QR scheme can be arbitrarily chosen whereas setting σ to a too large value in our scheme would require to increase the modulus size to guard against $p-1$ factoring methods. However, for practical values of comparison $\rho = \sigma = 160$, we can still use 1024-bit moduli. In such a scenario, our scheme still outperforms the QR one.

References

1. Niko Bari and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In Walter Fumy, editor, *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding*, volume 1233 of *Lecture Notes in Computer Science*, pages 480–494. Springer, 1997.
2. Josh Cohen Benaloh and Michael de Mare. One-way accumulators: A decentralized alternative to digital sinatures (extended abstract). In Tor Helleseth, editor, *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*, volume 765 of *Lecture Notes in Computer Science*, pages 274–285. Springer, 1994.
3. Gerrit Bleumer, Birgit Pfitzmann, and Michael Waidner. A remark on a signature scheme where forgery can be proved. In Ivan Damgård, editor, *Advances in Cryptology - EUROCRYPT '90, Workshop on the Theory and Application of of Cryptographic Techniques, Aarhus, Denmark, May 21-24, 1990, Proceedings*, volume 473 of *Lecture Notes in Computer Science*, pages 441–445. Springer, 1991.
4. Josh Benaloh Clarkson. Dense probabilistic encryption. In *Workshop on Selected Areas of Cryptography*, pages 120–128, 1994.
5. Ivan Damgård, Torben P. Pedersen, and Birgit Pfitzmann. On the existence of statistically hiding bit commitment schemes and fail-stop signatures. *J. Cryptology*, 10(3):163–194, 1997.
6. Georg Frobenius. *ber einen Fundamentalsatz der Gruppentheorie, II.* Sitzungsberichte der Preussischen Akademie Weissenstein, 1907.

7. Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.
8. Jens Groth. Cryptography in subgroups of z_n. In Joe Kilian, editor, *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, volume 3378 of *Lecture Notes in Computer Science*, pages 50–65. Springer, 2005.
9. Ralph C. Merkle. Protocols for public key cryptosystems. In *IEEE Symposium on Security and Privacy*, pages 122–134, 1980.
10. Ralph C. Merkle. A certified digital signature. In Gilles Brassard, editor, *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 218–238. Springer, 1990.
11. Torben P. Pedersen and Birgit Pfitzmann. Fail-stop signatures. *SIAM J. Comput.*, 26(2):291–330, 1997.
12. Birgit Pfitzmann. *Digital Signature Schemes, General Framework and Fail-Stop Signatures*, volume 1100 of *Lecture Notes in Computer Science*. Springer, 1996.
13. John M. Pollard. A monte carlo method for factorization. *BIT Numerical Mathematics*, 15(3):331–334, 1975.
14. Ron Rivest and Robert Silverman. Are 'strong' primes needed for RSA? Cryptology ePrint Archive, Report 2001/007, 2001. http://eprint.iacr.org/.
15. Katja Schmidt-Samoa. Factorization-based fail-stop signatures revisited. In Javier Lopez, Sihan Qing, and Eiji Okamoto, editors, *Information and Communications Security, 6th International Conference, ICICS 2004, Malaga, Spain, October 27-29, 2004, Proceedings*, volume 3269 of *Lecture Notes in Computer Science*, pages 118–131. Springer, 2004.
16. Willy Susilo. Short fail-stop signature scheme based on factorization and discrete logarithm assumptions. *Theor. Comput. Sci.*, 410(8-10):736–744, 2009.
17. Willy Susilo, Reihaneh Safavi-Naini, Marc Gysin, and Jennifer Seberry. A new and efficient fail-stop signature scheme. *Comput. J.*, 43(5):430–437, 2000.
18. Eugène van Heijst, Torben P. Pedersen, and Birgit Pfitzmann. New constructions of fail-stop signatures and lower bounds (extended abstract). In Ernest F. Brickell, editor, *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, volume 740 of *Lecture Notes in Computer Science*, pages 15–30. Springer, 1993.
19. Eugène van Heyst and Torben P. Pedersen. How to make efficient fail-stop signatures. In Rainer A. Rueppel, editor, *Advances in Cryptology - EUROCRYPT '92, Workshop on the Theory and Application of of Cryptographic Techniques, Balatonfüred, Hungary, May 24-28, 1992, Proceedings*, volume 658 of *Lecture Notes in Computer Science*, pages 366–377. Springer, 1993.