A Provably Secure Nyberg-Rueppel Signature Variant with Applications

Giuseppe Ateniese (ateniese@cs.jhu.edu)
Breno de Medeiros (breno.demedeiros@acm.org)

Abstract

This paper analyzes the modified Nyberg-Rueppel signature scheme (mNR), proving it secure in the Generic Group Model (GM). We also show that the security of the mNR signature is equivalent (in the standard model) to that of a twin signature [32], while achieving computational and bandwidth improvements.

As a provably secure signature scheme, mNR is very efficient. We demonstrate its practical relevance by providing an application to the construction of a provably secure, self-certified, identity-based scheme (SCID). SCID schemes combine some of the best features of both PKI-based schemes (functionally trusted authorities, public keys revocable without the need to change identifier strings) and ID-based ones (lower bandwidth requirements). The new SCID scheme matches the performance achieved by the most efficient ones based on the discrete logarithm, while requiring only standard security assumptions in the Generic Group Model.

Keywords: Generic Group Model, signature schemes, Nyberg-Rueppel variants, self-certified identity-based cryptography

1 Introduction

Several constructions of digital signatures based on arbitrary one-way functions are known [33, 40]. The security of these general constructions is provable in the *standard model,* following directly from the way the schemes employ one-way functions. However, these schemes are computationally expensive. Practical constructions (i.e., comparable in efficiency w/ commonly used methods) based on general one-way functions exist only for one-time signature schemes [6, 18]. In contrast, practical, multi-use signatures employ specific functions (such as integer product or discrete exponentiation) that are conjectured one-way based on specific mathematical assumptions (such as hardness of factoring or of computing the discrete logarithm). Among these signature schemes, few have been proven secure in the standard model, by reduction to the underlying mathematical assumption without having to resort to idealizations of the cryptographic constructs.

The first example of a practical signature scheme that was proven secure in the standard model is provided by the Gennaro-Halevi-Rabin (GHR) signature scheme – though that scheme required a non-standard hash function satisfying a division intractability assumption. The GHR scheme was revised by Naccache, Pointcheval, and Stern in [32], via an ingenious technique, *signature twinning,* that eliminated the use of hashes, and consequently the need for the non-standard requirement of division intractability. The second instance of a standard proof of security for a practical signature scheme is provided by the Cramer-Shoup [15] signature scheme. Both GHR and Cramer-Shoup are reducible to the variant of the RSA assumption known as *Strong* RSA assumption (SRSA). (The SRSA states that given z in $\mathbb{Z}_n^*$, the multiplicative residues modulo n, it is hard to find an integer $e > 1$ and $t \in \mathbb{Z}_n^*$ such that $t^e = 1 \bmod n$. It was introduced in [2].)

Gap Diffie-Hellman (GDH) groups provide another setting where provably secure signatures in the standard model have been constructed. A GDH group is any group where the Computational Diffie-Hellman assumption is believed to hold, but where the Decisional Diffie-Hellman is easy.[1] The Gap Diffie-Hellman problems were formalized in [36], and the first application of GDH groups in cryptography (for one-round tripartite key agreement) appeared in [28], but the existence of groups where the DDH is easy (but the CDH still looks difficult) was noticed earlier: [20, 21]. Recently, it has been shown that some signature schemes in GDH groups are provably secure in the standard model [8, 7] by reduction to the Gap Diffie-Hellman problem (the Computational Diffie-Hellman problem in gap groups) and to the so-called Strong Diffie-Hellman assumption, respectively. It is worth noticing that even after subsequent performance improvements (see [3]), pairing-based cryptosystems are less efficient than the schemes we shall describe later.

Unfortunately, there are still no practical digital signature schemes that have been proven secure by a standard reduction to the discrete logarithm. In particular, popular digital signature schemes such as DSA, ECDSA, Elgamal, Nyberg-Rueppel and Schnorr fall in this category. Instead, their security relies on the *unpredictability* of hash function values, a proof artifice known as the random oracle model (ROM, formally introduced in [4]) that cannot be expressed in terms of computational assumptions on the concrete instantiations of the hash function. The best security results in the ROM are signature schemes with tight reductions to the CDH ([23]) and to the DDH ([30]), both variants of a signature scheme first proposed by Chaum and Pedersen [14], and also considered by Jakobsson and Schnorr [27].

An alternative for obtaining proofs in the discrete logarithm setting is the Generic Group Model. The GM (introduced in [42, 34]) is a restricted model of computation that assumes algorithms do not have access to the representation of group elements and must access all group operations as blackbox function calls. Among the interesting features of the GM is that it is a potentially instantiatable computational model, i.e., provided one realizes the signature within a group for which only generic algorithms are known (e.g., general elliptic curves), the generic model faithfully describes the costs of an optimal attack on the discrete logarithm problem. In this, it differs from the inherently uninstatiatable ROM – in the sense that deterministic hash functions can never computationally realize a random function. Secondly, problems such as the discrete logarithm problem (DLP), the Computational Diffie-Hellman problem (CDH), and even the RSA assumption, the hardness of which underlies the security of many practical (and believed secure) signature schemes, are only provable in the generic model [42, 16]. So a proof that a certain computational problem is hard in the generic model can be seen as an indication that breaking it poses genuine difficulties.

A limitation of the GM (but not the ROM) is its inability to capture decisional problems, as shown in [19]. In our case this is not highly relevant as we consider the hardness of a computational problem instead. In the context of this paper, we believe the application of the GM is appropriate, and perhaps a superior approach to ROM-type proofs. Intuitively, the abstraction of the encoding as an random function is closer to real problem instances, in the sense that a group admits multiple encodings. In contrast, hash functions are deterministic, there are just a few hashing schemes in use, and most protocols do not randomize the hash via, for instance, the use of keys. Finally, it

[1] The CDH assumption states that given g, g', g'', where $g' = g^a$, and $g'' = g^b$ for unknown values a, and b, it is hard to compute $y = g^{ab}$. The decisional version instead provides g, g', and g'' and z and asks if $z = g^{ab}$.

should be noted that both ROM and GM share similar separation results from the standard model: There exist constructions that are provably secure (in the ROM or GM) but do not admit of any secure realization with concrete hash functions, for instance see [13, 17].

Some signature schemes have been shown secure in the GM with additional (or reduced) assumptions; for instance, see [24] which contains a description of several such signature schemes, and a proof for a modified ECDSA (elliptic curve DSA) secure under a minimalist idealized model. Other specific examples include proofs of security in the combined ROM and GM in [41]. Also, in [11] a proof of the security of the PVSSR scheme [38] (Pintsov-Vanstone signature with partial message recovery) is provided in a hybrid model combining the GM and the ideal cipher model.

As can be inferred from table 1, there exist a few other examples of signature schemes provably secure in the same setting (i.e, GM without ROM). For instance, the security of the ECDSA and Abstract DSA signature schemes in the GM is addressed in [10], which also includes interesting remarks about parallels between the ROM and the GM. Another instance is the work on twin signatures [32]. Our work has interesting connections with the twin signature paradigm, and in particular we show a reduction (in the standard model) from our scheme to a twin signature construction in section §5.

The modified Nyberg-Rueppel signature (mNR, proposed in [1]) has provable security in the generic model; the proof we include herein is inspired by techniques in [32]. The signature produces a triplet of signing values (including the message itself) – as opposed to a quartet (as is the case with twin NR) or quintet (twin DSA + message). In fact, the mNR signature has length comparable with that of other common Elgamal signature schemes. Our proof of security of the mNR signature provides support for the claims in [1], where this signature was introduced, but the provided analysis included incorrect arguments.

Table 1 includes signature schemes that have been proven secure in various settings, to facilitate a comparison with the mNR:

Table 1: Proofs of security for signature schemes

Signature	Proof model	assumption	tight/loose
[8, 7]	Std. model	GDH	tight
THIS	GM	DL	tight
[10]	GM	DL	tight
[32]	GM	DL	tight
[30]	ROM	DDH	tight
[23]	ROM	CDH	tight
[41]	ROM+GM	DL	tight
[39]	ROM	DL	loose

The mNR signature has applications to several cryptographic protocols, as for instance to enable the unlinkable, verifiable encryption of a certificate, shown in [1]. In order to further demonstrate the practical significance of the mNR signature scheme, we present an application to the construction of a provably secure self-certified identity-based (SCID) cryptographic scheme. This new SCID scheme matches the best performance among discrete-logarithm based SCID schemes ([25], with modifications by Girault [22] that make it self-certified, and also [37]) while simplifying security assumptions and arguments. In particular, a full proof of security of the scheme is achievable within the Generic Group Model based on standard assumptions such as the hardness of the discrete logarithm and the existence of collision-resistant hash functions, unlike previous schemes which required redundancy functions with specific properties.

Organization of this paper: In the next section, we describe the Nyberg-Rueppel signature in a general setting, following with the definition of the modified Nyberg-Rueppel signature in section §3. Section §4 includes the proof of security of the scheme under the generic model of computation. In section §5, we show that the mNR signature scheme has equivalent security to a twin signature scheme (in the standard model), while being more efficient. In section §6, we use the mNR signature to construct a self-certified public key cryptographic scheme that is provably secure (in the GM), and we illustrate its superior bandwidth characteristics over PKI-based signature schemes.

2 Plain Nyberg-Rueppel

The *plain* version of the Nyberg-Rueppel signature is as follows: Let p be a large prime, and g a generator of the of a q-order subgroup of $\mathbb{Z}_p^*$, where q is also a large prime. To generate such parameters, one may start by generating a suitably large prime q, and then searching for primes of the form $p = uq + 1$, with u small [29].

Let A be the signer, and assume that he chooses a secret key $x \in [1, q-1]$, and computes the public key $y = g^x \bmod p$. Let m be an element of $\mathbb{Z}_p^*$ which one wants to sign. For instance, m could be a short message in binary, which is then interpreted as the expansion of an element of $\mathbb{Z}_p^*$. First, A generates a random value $k \in [1, q-1]$, and computes $r = mg^k \bmod p$. Next, A solves the following equation for $s \in [1, q-1]$:

$$s = -k - x\bar{r} \bmod q, \tag{1}$$

where $\bar{r} = r \bmod q$. The signing values are $(r \bmod p, s \bmod q)$. If a verifier receives the pair (r, s), it should check that r is in the interval $[1, p-1]$, and if so, it may recover the signed message by computing:

$$ry^r g^s = (mg^k)(g^x)^{\bar{r}} g^s = mg^{k + x\bar{r} + s} = m \bmod p. \tag{2}$$

It is clear that in its plain form, NR is vulnerable to *existential forgery* attacks. Namely, one may choose $r \in [1, p-1]$ and $s \in [1, q-1]$ arbitrarily and these values sign the unique message that is obtained by applying the recovery algorithm to (r, s). While this message cannot be chosen in advance by the attacker, it still means that the signature scheme is insecure.

The typical solution for this type of problem is to use a *redundancy function*. Let R be an efficiently computable, one-to-one function from $\{0,1\}^\nu$ to $[1, p-1]$ that is sparse, i.e., the image set of R corresponds to a small fraction of all values in the range. Moreover, we assume that given Z in the image of R, there is an efficient algorithm to compute $R^{-1}(Z)$. Consider the modified version of the signature scheme which, given m, computes $m' = R(m)$ and then signs m' according to the plain NR scheme. In order to recover the signed message, the verifier first recovers m' according to the recovery mechanism of plain NR, and then the actual message m as $R^{-1}(m')$. The security of the modified version depends on it being hard to choose the values r and s such that the output of the recovery algorithm lies in the image of R. In practice, the design of redundancy functions that provide adequate security is a delicate task. The signature schemes that we examine in this paper avoid the issue of redundancy function design and analysis at the expense of losing the message-recovery property.

2.1 Nyberg-Rueppel in general groups

As with other signature schemes based on the discrete logarithm problem, the Nyberg-Rueppel signatures can be used in a variety of groups apart from the multiplicative residues $\mathbb{Z}_p^*$. In particular, there is interest for the implementation of NR signatures in elliptic curves. Therefore, we shall use a more general notation.

Let $\mathcal{G}$ be a cyclic group with generator g. For instance, $\mathcal{G}$ could be a cyclic subgroup of an elliptic curve, of $\mathcal{G}$ could be a subgroup of $\mathbb{Z}_p^*$, the multiplicative residues modulo a prime p, or it could be a subgroup of $\mathbb{Z}_n^*$, the multiplicative residues modulo a composite n. In the first two cases, the group $\mathcal{G}$ has known order, say q. In the latter case, the group has unknown composite order n'. In this latter case, we assume that η is known such that $2^\eta < n' < 2^{\eta+1}$.

It is assumed that there is an efficiently computable function $\rho : \mathcal{G} \longrightarrow \mathbb{Z}$. This is obtained in a natural way – if the elements of $\mathcal{G}$ are presented by their binary encodings, these values may be interpreted as the binary expansion of an integer. If the order of $\mathcal{G}$ is known, then $\rho(\cdot)$ can be considered as having images in the interval $[0, q-1]$ by computing the positive remainder modulo q of the integer values. In the case of unknown order, we assume there is a small value t such that each representation falls within the interval $[-2^\eta t, 2^\eta t - 1]$, where small means polynomial with respect to a security parameter τ.

For instance, if $\mathcal{G}$ is a subgroup of $\mathbb{Z}_n^*$ one may define $\rho(g)$ as the integer in in $[1, n-1]$, which represents the residue g. In that case, t is some value such that $n < t2^\eta$. If $\mathcal{G}$ is a cyclic subgroup of an elliptic curve $\mathcal{E}$ defined over $\mathbb{Z}_p$, and g is a point generating $\mathcal{G}$, one may define $\rho(g)$ as the x-coordinate of the point g, prefixed by a single bit b – this bit indicates the correct choice for $y(g)$ among the two roots y_0 and y_1 of the equation $y^2 = f(x(g)) \bmod p$ that defines the elliptic curve $\mathcal{E}$. The corresponding integer value can then be reduced modulo $q = |\mathcal{G}|$. Finally, the case $\mathcal{G} = \mathbb{Z}_p^*$ is immediate – take the representative in $[1, p-1]$ of each element of $\mathcal{G}$ and reduce it modulo q.

In order to sign messages, as seen before, it is necessary to use randomness. More precisely, signers must choose random integers in a fixed size interval I. If the order of $\mathcal{G}$ is a known prime q, then it suffices to take the interval $I = [1, q-1]$. Otherwise, if the order of $\mathcal{G}$ is unknown, k can be chosen in the interval $I = [-2^{\epsilon(\eta+\tau)}t, 2^{\epsilon(\eta+\tau)}t - 1]$, where τ is a security parameter, and ϵ is larger than 1 by a *non-negligible* amount.

As before, $y = g^x \in \mathcal{G}$ is the signer's public key, where x is chosen in the interval $[-2^\tau, 2^\tau - 1]$ if $\mathcal{G}$ is unknown, otherwise x is chosen in the interval $[1, q-1]$. The signing space $\mathcal{M}_S$ equals the group $\mathcal{G}$. To sign a message m, the signer computes:

$$r = g^k m \in \mathcal{G}, \text{ and } s = -k - x\rho(r),$$

where if the order of $\mathcal{G}$ is known, then s can be reduced modulo q to arrive at some value in the interval $[1, q-1]$.[2] If on the other hand, the order of $\mathcal{G}$ is unknown, it is straightforward to see that s is contained in the interval $[-2^{\epsilon(\eta+\tau)+1}t, 2^{\epsilon(\eta+\tau)+1}t - 1]$. The signature is the pair (r, s).

[2]In theory s could be 0, but this case is considered a failure and the signing algorithm need to be restarted with a different value for k.

The verification of the signature starts by checking that r is indeed the representation of an element of $\mathcal{G}$ and that s is in the interval I (where I equals $[1, q-1]$ or $[-2^{\epsilon(\eta+\tau)+1}t, 2^{\epsilon(\eta+\tau)+1}t-1]$, accordingly). If these conditions are satisfied, the verifier checks the equation:

$$ry^{\rho(r)}g^s = (mg^k)(g^x)^{\rho(r)}g^s = mg^{k+x\rho(r)+s} = m \in \mathcal{G}. \tag{3}$$

3 Modified Nyberg-Rueppel

We consider a modification of the Nyberg-Rueppel signature that avoids the difficulties of redundancy function design by giving up the message-recovery property. Its performance is slightly worse than Elgamal – which similarly does not offer message-recovery – but more efficient than twin Nyberg-Rueppel signatures, which are provably secure in the same model. While it is true that twin NR provides message recovery, it still requires the transmission of four signing values, while our signature requires only three values (including the message). As the raison d'être for message-recovery is bandwidth savings, our scheme achieves this goal through a different means.

The mNR substitutes a discrete exponentiation for the redundancy function. More explicitly, let g_1 be another generator of the same group of order q, such that the discrete logarithms of g_1 with respect to both g and y are unknown. The message space $\mathcal{M}_S$ is the integer interval I in the previous section, i.e., equal to $[1, q-1]$ in case of known order, or $[-2^\eta, 2^\eta - 1]$ in the case of unknown order. If (r, s) is the signature on a message m, the modified verification equation is as follows:

$$g_1^m = ry^{\rho(r)}g^s. \tag{4}$$

The signing procedure works as before, as the message m in I is first changed into $m' = g_1^m$ as a message in $\mathcal{G}$, and then signed as in the previous algorithm.

DSA style verification: In the case of known order, the signature on message m can be shortened to the pair $(e, s) \in \mathbb{Z}_q^2$, where $e = \rho(r) \bmod q$. The verification algorithm first recovers $r = g_1^m y^{-e} g^{-s} \in \mathcal{G}$, then recomputes $e' = \rho(r) \bmod q$ and checks if $e' = e$. The length of this signature equals that of DSA in the same setting.

4 Proof in the Generic Model

In this section, we consider the security of mNR in the Generic Model. The GM captures algorithms that access group operations (and indeed the group encoding) through black box function calls. This proof is inspired by the techniques found in [32].

To simplify the discussion, we consider initially only the case of groups $\mathcal{G}$ of known prime order q, later discussing the modifications necessary for the proof over groups of unknown (composite) order.

In the GM, the group encoding $\sigma(\cdot) : [0, q-1] \to \mathcal{G}$ represents an *encoding oracle* that implements a homomorphism from $\mathbb{Z}_q^+$ onto $\mathcal{G}$. Moreover, as before, we assume knowledge of a function $\rho(\cdot)$ from $\mathcal{G}$ to $\mathbb{Z}_q$.

In this setting, one describes the public key $y = g^x$ as $\{\sigma(1), \sigma(x)\}$. This notation just means that the homomorphism $\sigma(\cdot)$ maps 1 to g and therefore maps x to y. $\sigma(\cdot)$ is an exponential notation, so x is unrecoverable from $\sigma(x)$. Moreover, we also consider powers of the element $g_1 = g^z$, represented in this notation as $\sigma(z)$.

The group operation oracle $\cdot \oplus \cdot$ takes two encoded group elements $\sigma(v_1)$, $\sigma(v_2)$, and returns the encoded product $\sigma(v_1 + v_2)$. (Since this is exponential notation, the product translate as a sum in the exponents.) Similarly, given $\sigma(v)$ and an integer u, one can implement the square-and-multiply algorithm for exponentiation, using multiple calls to the group operation oracle, to obtain $\sigma(uv)$. One also needs a group inversion oracle $\ominus\sigma(v) \to \sigma(-v)$.

Now, consider the process of verifying a signature (r, s) on a message m using only generic algorithms, where m and s are elements of $\mathbb{Z}_q^+$ and r is in $\mathcal{G}$. Let $e = \rho(r)$ in the following.

1. Obtain $\sigma(zm)$ from $\sigma(z)$ and m by repeated calls to the group operation oracle, as described above. Similarly, obtain $\sigma(xe)$ from $\sigma(x)$ and e, and $\sigma(s)$ from $\sigma(1)$ and s.
2. Obtain $\sigma(xe+s)$ as $\sigma(xe) \oplus \sigma(s)$. Invert this to obtain $\sigma(-xe-s)$ by computing $\ominus\sigma(xe+s)$.
3. Obtain $r' = \sigma(zm - xe - s)$ as $\sigma(zm) \oplus \sigma(-xe - s)$ and check if $\rho(r') = e$.

Let $\mathcal{A}$ be a conjectural, efficient forging algorithm. As a generic algorithm, it works as follows: It maintains a list of linear polynomials $\{F_i\}$, where $F_i = \alpha_i + \beta_i X + \gamma_i Z$, and the coefficients lie in $\mathbb{Z}_q$. The list is initiated as $\{F_1 = 1, F_2 = X, F_3 = Z\}$. The algorithm also maintains a list $\{\sigma_i\}$ of encodings, initiated as $\{\sigma_1 = \sigma(1), \sigma_2 = \sigma(x), \sigma_3 = \sigma(z)\}$. At the k-th time the algorithm queries the oracle, it provides the indices i, j and a bit b, and the oracle responds with either $\sigma_k = \sigma_i \oplus \sigma_j$ or $\sigma_i \oplus (\ominus\sigma_j)$, according to the case $b = 0$ or $b = 1$, respectively. The algorithm adds σ_k and $F_k = F_i \pm F_j \bmod q$ to each of the respective lists, with the $+$ sign being chosen if $b = 0$. (So it is the same sign as in the definition of σ_k in terms of σ_i and σ_j.) All the F_i's computed by $\mathcal{A}$ are degree-1 polynomials in the variables X and Z, with coefficients in $\mathbb{Z}_q$. Without loss of generality, we may assume that the F_i are distinct polynomials with coefficients in $\mathbb{Z}_q$.

If, during the execution of the protocol, it happens that $F_i(x, z) = F_j(x, z) \bmod q$, with $i \neq j$, it follows that $F = F_i - F_j$ is a non-zero polynomial, with $F(x, z) = 0 \bmod q$. Let $F = a + bX + cZ$, for some coefficients a, b, and c in $\mathbb{Z}_q$, not all of which equal $0 \bmod q$. Then we conclude that $a + bx + cz = 0 \bmod q$, or equivalently that $1 = g^a y^b g_1^c$, with not all of a, b, c equal to 0. Such execution sequences are labeled *unsafe* (following GM terminology), and have negligible probability of occurrence if the discrete logarithm problem is hard in $\mathcal{G}$. (More exactly, the hardness of computing representations of 1, but the two problems are equivalent. To see this, note that an algorithm that computes such representations could be coaxed to compute the discrete logarithm of y w.r.t. g by feeding it with a known power of g for g_1.) In the following analysis, we shall assume that $\mathcal{A}$ only generates safe sequences.

Consider now a forging algorithm that produces a modified Nyberg-Rueppel signatures (m, r, s) on some message m, after u queries to the group operation oracle. Note that in this case, the verification equation implies that $r = \rho(\sigma(zm) \oplus \sigma(-x\rho(r)) \oplus \sigma(-s))$. Let $e = \rho(r)$ and $P = mZ - eX - s$. If P is not in the list of oracle queries performed by the algorithm, augment the list by adding $F_{u+1} = P$ at the end, and increment the number of queries $u \leftarrow u + 1$.

Let F_j be the unique appearance of the polynomial P in the list, without loss of generality. There is a possibility that a polynomial F_i, with $i \neq j$, satisfies $F_i(x, z) = P(x, z)$. As before, if $F = F_i - P$ is written as $ax + bz + c = 0$, this implies a non-trivial relation $1 = g^a y^b g_1^c$, in violation of the discrete logarithm hardness in $\mathcal{G}$. We conclude that this event happens only with negligible probability, and therefore that we may assume that there exists no F_i, $i \neq j$, such that

$F_i(x, z) = P(x, z) \bmod q$. This implies that the group operation oracle may return a random value for σ_j, because F_j represents a query for a new encoding when the encoding oracle is called at step j. The probability that $\rho(\sigma_j)$ equals $e = \rho(r)$ is therefore, no more than $1/q$, as (almost) all values are now equally likely. We conclude that there is no such efficient, generic forging algorithm $\mathcal{A}$.

Fact 1 *There is no efficient, generic algorithm that can compute a message m and a modified Nyberg-Rueppel signature (r, s) on it with non-negligible probability of access. This is true even if $\mathcal{A}$ has oracle access to a signing oracle, as long as $\mathcal{A}$ never queries the oracle on message m.*

The above discussion already proves the first part of the statement, i.e., the case of passive adversaries. To take in account active attacks, consider the following simulation, that proves the encoding oracle can simulate signatures without knowledge of the signing key. Suppose an active attacker requests a signature (r, s) on a message m of choice. The simulator chooses e and s as random elements of $\mathbb{Z}_q$ and also a random encoding σ_k satisfying $e = \rho(\sigma_k)$. Later, when the attacker requests the value $\sigma(zm) \oplus \sigma(-ex) \oplus \sigma(-s) = \sigma(zm - ex - s)$, the simulator may return σ_k and pass the verification algorithm. The only risk is that the value of $\sigma(zm - ex - s)$ becomes defined through queries made to the oracle after it has chosen e and s and therefore has committed to the signature forgery query, but before the actual forging query is performed. This corresponds to the probability that the simulator chooses query polynomial $P = mZ - eX - s$ that independently appears in the attacker's list or that leads to an unsafe sequence, events of negligible probability when the simulator choice is made at random.

4.1 The case of unknown order

In the case of unknown order, the algorithm computes the sequence $\{F_i\}_{i=1,\ldots,u}$ as polynomials with integer coefficients. Most of the proof is similar to the known order case, but it is necessary to extend the notion of unsafe sequences to include the case where $F_i = F_j \bmod n'$, where n' is the unknown order, but $F_i \neq F_j$ as polynomials with integer coefficients. Such sequences occur with negligible probability: If the forging algorithm could, with non-negligible probability, find such pairs F_i, F_j, it would with non-negligible probability extract a multiple of the unknown order as the greatest common divisor of the coefficients of the difference polynomial $F = F_i - F_j$. Since the number of steps taken by the efficient algorithm $\mathcal{A}$ is at most a polynomial $p(\tau)$ in the security parameter, the length of the coefficients of F is at most equal to $2^{p(\tau)}$. Intuitively, several (polynomially many, in inverse relation with the non-negligible probability of the sequences) applications of this method would eventually produce several multiples of the unknown order (and in a bounded range), eventually allowing the order of $\mathcal{G}$ to be computed exactly, as there is only a polynomial number of pairs F_i, F_j.

In the case of unknown order, learning the order is equivalent to factoring, and the above efficient algorithm contradicts the hardness of factoring assumption. Therefore, the expanded definition of unsafe sequences still includes only a negligible fraction of protocol executions, and can be eliminated from the analysis. The rest of the proof works formally as in the case of known order, however it accomplishes a reduction to the Strong RSA assumption instead.

4.2 Security against one-more forgery attacks

We now consider the security of the mNR signature scheme against the following type of attack. Let m be a message, and assume that the attacker already knows several signatures $(r_1, s_1), \ldots, (r_k, s_k)$

on m. The attacker succeeds if he is able to construct a new signature pair on the same message m, distinct from all previously seen signature pairs on m. This property will be needed later in applications of the mNR signature scheme to the construction of self-certified, identity-based cryptographic schemes.

It is not difficult to extend the security arguments given in section §4 to cover also this case. First, note that new signatures on the same message can also be simulated by an encoding oracle without access to the signing key. When the active attacker requests a new signature (r, s) on the message m, the simulator chooses e and s at random as before – with high probability they will differ from any previously seen pairs. The simulator then chooses a random encoding σ_{k+1} satisfying $e = \rho(\sigma_{k+1})$, and returns (σ_{k+1}, s) as the requested signature pair. When the attacker requests the value $\sigma(zm - ex - s)$, the simulator may return σ_{k+1} and pass the verification algorithm. Again, the probability that the value of $\sigma(zm - ex - s)$ cannot be chosen anew for having been defined through previous queries is negligible.

Therefore, it is sufficient to consider passive attacks. The same proof method for the basic security proof can be repeated. One needs only to consider safe sequences, and such that there are no repeated polynomials in the list. As before, unless the encoding query $\sigma(mz - ex - s)$ becomes defined by an earlier query (an event with negligible probability), the encoding oracle can return a random value, which will match e with probability at most $1/q$.

5 Reduction to twin signatures

Now we prove a tight reduction (in the standard model) from the modified NR signature to twin plain NR signatures, as further evidence of the security of the scheme. This shows that the security of mNR and twin NR are comparable, while mNR is more efficient by requiring fewer exponentiations. We consider only the case of known order.

Let $\mathcal{A}$ be an efficient signature forging algorithm that has non-negligible probability of failure when fed as input a quadruple $(\mathcal{G}, g_1, g, y)$, where $\mathcal{G}$ is a group of prime order q, and g_1, g, and y are distinct generators of $\mathcal{G}$. A simulator feeds the algorithm $\mathcal{A}$ with the values $\mathcal{G}$, g, and y of another partys public key. (The simulator does not know the associated secret key.) Moreover, let the simulator choose some random integer z and compute $g_1 = g^z$ for input to $\mathcal{A}$. After execution, with non-negligible probability the algorithm $\mathcal{A}$ produces an output $m, (r, s)$ of a message and a signature pair. That is, $m, s \in [1, q-1], r \in \mathcal{G},\ g_1^m = ry^{\rho(r)}g^s \in \mathcal{G}$.

The simulator keeps $\mathcal{G}$, g, y, and g_1 fixed and repeatedly calls the forging algorithm $\mathcal{A}$ until it succeeds at least twice in obtaining signed messages $m_1, (r_1, s_1)$, and $m_2, (r_2, s_2)$. If the algorithm $\mathcal{A}$ needs β steps to arrive at a forgery in average, then the expected number of steps before two signatures are generated is no more than 2β.

The simulator then uses the knowledge of the trapdoor z to transform each signature (r_i, s_i) into a signature pair (r_i, s_i') on a common message m (chosen arbitrarily): $g_1^m = r_i y^{r_i} g^{s_i'}$, where $s_i' = s_i + z(m - m_i)$. Now, let $M = g_1^m$. This implies that the simulator is able to use the forging algorithm for the modified NR to compute two regular Nyberg-Rueppel signatures (without redundancy) on the same message M. The twin signature paradigm in [32] suggests that this is a secure signature scheme in the generic model.

The reduction is tight (a work factor expansion of 2 implies the loss of a single security bit), while the verification of the mNR requires three exponentiations, compared with the verification of the twin signature, that requires four exponentiations. With both signatures, multi-exponentiation techniques ([43]) can significantly decrease the total cost of verification.

This takes care of passive attacks. To deal with active attacks, start with an oracle for the twin NR signature and use it to construct an oracle for the mNR by substituting request for signature in messages m for requests for signatures on messages $M = g_1^m$. When the twin NR signing oracle returns two signatures on M, choose one arbitrarily and return it, discarding the other. The rest of the proof is the same as in the passive case.

6 Applications of mNR

Since the mNR signature admits a tight reduction to the discrete logarithm problem in generic groups, it reaches comparable provable security with shorter keys than those of signature schemes whose security proof depends on forking lemma arguments. This makes mNR an attractive candidate in any applications where provable security is desired, and there is a premium in maintaining short signature length. For instance, certain secure network services can suffer from performance deterioration when a longer signature must be fragmented and sent along multiple network packets.

Therefore, we compare the bandwidth efficiency of the mNR signature with other signatures that achieve tight reductions. It could be said that the *short signatures* [9], based on elliptic curves with pairings, achieve tight security (ROM reduction to the Gap Diffie-Hellman problem) and shorter signatures. However, it is difficult to compare the bit-by-bit security of the two signatures as they are reducible to distinct computational assumptions. Moreover, Gap Diffie-Hellman elliptic curves (GDH-EC) admit non-generic algorithms to compute the discrete logarithm (Menezes-Okamoto-Vanstone reduction [31]), and therefore the discrete logarithm problem (DLP) in GDH-EC (which upper-bounds the security of the Gap Diffie-Hellman problem in the same groups) is (possibly) asymptotically weaker than the DLP in elliptic curves for which only generic algorithms are known – an instantiation scenario where the GM-style proof is particularly compelling. In addition, if instantiated over the same class of elliptic curves, mNR signatures are computationally more efficient to verify than short signatures – due primarily to the fact that pairing computations are significantly slower than exponentiations in these groups [3]. Finally, as we shall soon demonstrate, while short signatures are about 50% shorter than mNR signatures when considered in isolation, within certain usage contexts where both signatures and certificate length must be jointly considered, the mNR signature actually provides for shorter authenticity proofs than those based on short signatures!

Indeed, since mNR signature schemes are of Elgamal-type, they lend themselves naturally to sound constructions of self-certified, identity-based public key schemes. SCIDs are alternatives to PKI-style certification of public keys which preserve trust assumptions typical of PKI schemes. If one evaluates mNR within an SCID infrastructure, the length of a full authenticity proof (assuming only one certification authority) equals the length of the signature + the length of the so-called *public reconstruction data* (more on this in the next section). Concretely, if 160-bit elliptic curves are considered, with point-compression representation, the length of an SCID-mNR authenticity proof is 482 bits, while the PKI-based short signature scheme takes 483 bits. (These numbers assume that both the certificate signature – public reconstruction data for mNR – and the message signature are of mNR type in the first case and of short type in the second.) However, for multi-level hierarchies of certification authorities, if a certificate chain contains more than one certificate, the SCID-mNR authenticity proof results in further savings, converging asymptotically to 50% of the length that the same proof would have if constructed by using short signatures within a PKI-style infrastructure.

The above results assumed that the security of the mNR signature in a generic EC compares with that of a short signature on a GDH-EC of same key length. If one is willing to view the generic model proof as a hardness argument that has heuristic value even in non-generic cryptographic groups, and wishes for the shortest authenticity proofs, it is possible to instantiate the SCID-mNR within the GDH-EC to shrink certificate lengths (replacing them with shorter public reconstruction data) and use the short signature scheme to sign messages. This hybridization is feasible because the SCID-mNR construction allows one to derive Elgamal-type public keys, which are also the key type used by short signature schemes. In that case a certificate chain of length 1 would result of an authenticity proof of only 322 bits long.

Other Elgamal-type signature scheme whose security are tightly reducible (in the ROM) to the security of the discrete logarithm are the schemes by Goh and Jarecki [23], and Katz and Wang [30]. These signature schemes are formally similar to short signatures in the sense that they are related to the Chaum-Pedersen signature scheme [14]. Since we do not know how to build SCID schemes from Chaum-Pedersen-style signatures, the same arguments for the smaller length of the SCID-mNR signature scheme against PKI+short signatures are still valid with respect to these other schemes, only more so because the latter are not length-optimized.

In the following subsections, we review SCID schemes, and provide the construction of the SCID-mNR signature. Our goals are twofold: First, to demonstrate the potential for applications of the mNR signature, as SCID schemes are more efficient alternatives to PKI-certification in a myriad of contexts, such as proxy signatures and for delegation purposes (an extensive list of applications of SCID schemes is provided in [37]). Second, to substantiate our claims of bandwidth efficiency provided by the mNR signature scheme.

6.1 SCID schemes

If one considers only signature schemes and key exchange protocols, the difference between ID-based and PKI-based infrastructures is less pronounced. Indeed, starting from a PKI-based infrastructure, one may obtain an ID-based signature scheme (or key agreement protocol) by augmenting all signatures (key transfer messages) with the signer's (user's) certificate. In this way, verification of signatures (or authenticity of key transfer messages) requires knowledge only of system-wide parameters and the user's ID. Therefore, as far as signature schemes or key agreement protocols are concerned, identity-based constructions do not provide advantages in terms of key management over PKI schemes. In these cases, the main incentives to use ID-based schemes are potential bandwidth savings and related optimizations. Any such benefits have to be weighed against the necessity to change trust assumptions (i.e., to tolerate automatic key-escrow) when adopting an identity-based infrastructure.

On the other hand, it is possible to construct self-certified schemes that preserve some of the bandwidth savings of identity-based schemes while maintaining the same trust assumptions that underlie PKI schemes. In SCID schemes, explicit public keys are present, but not explicit certificates. Moreover, the public keys are not explicitly distributed, but instead reconstructible from public data (which essentially replace certificates). This public data includes system data (the trusted authority's parameters), users' unique names, and additional per-user public data (reconstruction public data). Certification of the keys is implicit by the fact that they are derivable from authentic user public data.

In SCID schemes, users generate their private keys themselves (without these becoming known to the trusted authority) and the trusted authority is involved in the computation of the public reconstruction data. The inputs to the algorithm that creates the reconstruction data are the user's public key[3], identification information (in the case of self-certified identity-based schemes), and the private key of the authority. Informally, the security requirements of the scheme include that it be computationally infeasible (without knowledge of the authority's private key) to compute the private key of a user from the knowledge of (all) the public data about the user, as well as to generate matching user identity and reconstruction public data for which a corresponding private key can also be computed.

6.2 Previous work on SCID schemes

Self-certified public keys were introduced by M. Girault [22], where two constructions of SCID schemes are provided, one based on RSA and one based on Elgamal-type public keys. The latter can be seen as an improvement (in the sense of being self-certified and therefore requiring less trust on the authority) of C. G. Günther's implicitly certified scheme [25]. A similar scheme, but less efficient than [22], was proposed by K. Nyberg and R. Rueppel in [35].

In Girault's Elgamal-type scheme, the trusted authority T has public key $y = g^x$ (for private key x). A prospective user A with identity string I generates a random k in $\mathbb{Z}_q^*$ (where q is the order of the group generated by g) and sends $u = g^k$ to T, who computes $r = u^{k'}$ for some random value k', and solves the following equation for $\bar{s}$

$$xr + k'\bar{s} = I \mod q. \tag{5}$$

The values $(P, \bar{s})$ are returned to the user, who computes $s = \bar{s}k^{-1}$. The values P, s and I now satisfy the equation:

$$y^r r^s = g^I. \tag{6}$$

The well-known existential forgery of the Elgamal signature scheme ([5]) implies that the identity string I must contain redundancy. The most straightforward way to achieve this is to compute I as the hash of the user's unique name, which provides heuristic security. In any case, Girault's work preceded the development of ROM model-proof techniques for the Elgamal signature scheme [39], and therefore, the first scheme to adopt a provably secure signature underlying the method for generation of the reconstruction data is found in [37], which we now describe.

The scheme uses a (weak) blind Schnorr-type signature (introduced in [26]). The authority T chooses $k \in \mathbb{Z}_q^*$ and computes $\bar{r} = g^k$. The user A, upon receiving $\bar{r}$ from T, chooses a random value k' (in $\mathbb{Z}_q^*$) and computes $r = \bar{r}g^{k'}$, returning this to T. The reconstruction data can then be computed by T as:

$$\bar{s} = xh(ID_A, r) + k \mod p. \tag{7}$$

The value $\bar{s}$ is returned to the user, who can compute $s = \bar{s} + k' \bmod q$. The tuple (r, s) is a signature on the user's identity. The corresponding public key is:

$$y_A = g^s = y^{h(ID_A, r)}r.$$

Since the underlying signature scheme is of Elgamal-type, the process for issuing a key involves an unforgeable signature from the authority. The user's private key is protected from the authority as it does not know the value k' generated by the user.

[3]In some schemes, the authority should not learn the user's public key before the computation of the reconstruction data, and in that case the user provides the authority with a blinded version instead. This is the case with our scheme.

The scheme we describe improves in this scheme by using shorter keys for the same level of provable security. Indeed, the proof of security of the Elgamal signature requires forking lemma arguments that result in a loose reduction to the discrete logarithm problem. Our scheme is provable in the generic model, in practice a tight reduction to the discrete logarithm problem. If both schemes are implemented in certain classes of elliptic curves – those for which the only method to compute discrete logarithms are by generic algorithms – this implies that our proof can guarantee comparable security at key lengths which are only half as long. That is, the loose reduction in the security proof of Elgamal signatures requires a doubling of the keylength to achieve comparable security with the discrete logarithm problem.

6.3 Construction of the mNR-based SCID scheme

In this section, we use the same notation as in §2.1. We assume that the group $\mathcal{G}$ has known order q, and we are particularly interested in the case where $\mathcal{G}$ is an elliptic curve.

We first describe a simpler version of the scheme, which does not provide secret key privacy, but that does already generate the secret key in a probabilistic fashion. Let x be the trusted party $\mathcal{T}$'s secret key, and y the public key, i.e., $y = g^x$ (in $\mathcal{G}$). Consider the following protocol used by a trusted third party $\mathcal{T}$ to generate multiple secret keys corresponding to a single identity:

$$\begin{array}{ll} \mathcal{A} \longrightarrow \mathcal{T}: & g_1^{\mathcal{H}(ID_A)}; \\ \mathcal{A} \longleftarrow \mathcal{T}: & (r, s) \ \in \mathcal{G} \times \mathbb{Z}_q. \end{array}$$

The pair (r, s) is computed such that $g_1^{\mathcal{H}(ID_A)} = r y^{\rho(r)} g^s$. Concretely, $\mathcal{T}$ generates a random $k \in \mathbb{Z}_q^*$ and computes:

$$\begin{array}{ll} r = g_1^{\mathcal{H}(ID_A)} g^k & \text{in } \mathcal{G} \\ s = -k - \rho(r)x & \bmod q \end{array}$$

Notice that the pair (r, s) is the mNR signature scheme applied to the message $\mathcal{H}(ID_A)$. The idea is to use the value s in the signature as the private key of the user. A third party may reconstruct the user's public key $g' = g^s$ from the user's identity ID_A and the public reconstruction data r by computing $g' = g_1^{\mathcal{H}(ID_A)} y^{-\rho(r)} r^{-1}$.

However, this naive public key-generation method permits the authority to learn the private key of the user. We now consider how to provide private-key privacy during the key generation process. To guarantee that $\mathcal{T}$ does not have any useful information about $\mathcal{A}$'s secret, we could blind the key generation procedure as follows:

$$\begin{array}{ll} \mathcal{A} \longrightarrow \mathcal{T}: & \tilde{g} = g^\alpha \text{ in } \mathcal{G}; P \\ \mathcal{A} \longleftarrow \mathcal{T}: & r = g_1^{\mathcal{H}(ID_A)} g^{k+\alpha} \text{ in } \mathcal{G}, \quad \bar{s} = -k - \rho(r)x \bmod q \\ & [\text{where } k \in_R \mathbb{Z}_q^*] \end{array}$$

The value P is a proof of knowledge of the logarithm of $\tilde{g} = g^\alpha$ on basis g. This proof can be carried out in the generic model by having the user sign a challenge message from the authority using the mNR signature with public key $\tilde{g}$. It is necessary to include P to avoid impersonation attacks; without that proof, the user could choose $\tilde{g} = g^{\alpha - H(ID_A) + H(ID_B)}$ to obtain a certificate on another user's identity ID_B.

At this point, A computes the value s as:

$$s = \bar{s} - \alpha \bmod q,$$

and verifies that the pair (r, s) is a Nyberg-Rueppel signature on $\mathcal{H}(ID_A)$ under $\mathcal{T}$'s public key y. The verification succeeds:

$$ry^{\rho(r)}g^s = rg^{x\rho(r)+\bar{s}-\alpha} = g_1^{\mathcal{H}(ID_A)}g^{k+\alpha}g^{x\rho(r)-k-x\rho(r)-\alpha} = g_1^{\mathcal{H}(ID_A)}$$

The scheme above is simpler than a *blind* version of the Nyberg-Rueppel signature (that, to the best of our knowledge, was first introduced in [12])[4]. This is because the blind signature must hide from the signer the message to be signed, as well as all of the signature, while in this case the encoded identity ID_A can be revealed to the signer, who ignores only the value s, half of the actual signature.

The public key of the user $\mathcal{A}$ is g^s. Once r is known, anyone can retrieve the public value corresponding to the identity ID_A by computing:

$$\frac{g_1^{\mathcal{H}(ID_A)}}{ry^{\rho(r)}} = g^s$$

In addition, no one can retrieve the secret value s, not even $\mathcal{T}$ who has access to privileged information, such as the values x, k and g^α. Clearly, if $\mathcal{T}$ could retrieve s, then he could compute arbitrary discrete logarithms. This is simple to show: If an efficient algorithm that outputs s exists then it can be used to solve any instance of the discrete logarithm problem given that, from s, the value α can be extracted which was arbitrarily chosen.

The certificates contain $\mathcal{T}$'s mNR signature on the identity, and hence are in principle unforgeable based on the security of the mNR signature scheme. If the system supports revocation, then multiple certificates associated with the same identity string ID will be present, and these are none other than multiple mNR signatures on the same message $\mathcal{H}(ID)$. This implies that a stronger notion of unforgeability is needed, one that defines forgery as either the production of a signature on an arbitrary message (without knowledge of the secret key), or the production of a new signature on a message when other signatures on the same message are known. The mNR signature is secure in this stronger sense, as we have shown in section §4.2. Note that this requirement is not particular of our construction but typical of SCID schemes. However, we are not aware that it has been explicitly recognized elsewhere.

Notice that we have only described the process to generate public keys. Since these are Elgamal type public keys, they can subsequently be used to implement all variety of cryptographic services, such as digital signatures, encryption or key agreement. The case of digital signatures and key agreement protocols are more interesting as applications because the initiator of the transaction can forward the value r (half the mNR certificate), providing the implicit certification of the public keys used and eliminating the need for certificates.

As noted in [37], whenever one has a hierarchical structure of self-certified keys, long certificate chains can be verified with less than linear work factor on the certificate length, by pre-computation of a single formula that collapses the intermediate recovery steps. The formula can then be efficiently evaluated using multi-exponentiation techniques [43]. This results in further efficiency gains over PKI-based schemes.

[4] It cannot be a fully blind version as the TTP must control which identity is signing.

7 Conclusions

This paper provides a proof of security for the modified Nyberg-Rueppel signature in the Generic Group Model. The proof is tight, closely relating the security of the signature scheme with that of the discrete logarithm. This result makes the signature attractive for applications where provable security is desired and there is a premium in maintaining short signature length. The mNR signature compares favorably with the twin signature paradigm for Elgamal signature types in terms of both bandwidth and computation, while providing comparable security in the Generic Model.

The possibility of using the mNR signature scheme to construct an SCID-style public key infrastructure leverages the potential bandwidth savings provided by the basic mNR scheme by permitting the use of very short certificates.

References

[1] Ateniese, G., and de Medeiros, B. Efficient group signatures without trapdoors. In *Proceedings of Advances in Cryptology – ASIACRYPT 2003* (2003). Revised version: `http://eprint.iacr.org/2002/173`.

[2] Baric, N., and Pfitzmann, B. Collision-free accumulators and fail-stop signature schemes without trees. In *Advances in Cryptology: Proceedings of EUROCRYPT'97* (1997), Lecture Notes in Computer Science, pp. 480–494.

[3] Barreto, P. S. L. M., Kim, H. Y., Lynn, B., and Scott, M. Efficient algorithms for pairing-based cryptosystems. In *Advances in Cryptology: Proceedings of CRYPTO 2002* (2002), vol. 2442 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 354–368.

[4] Bellare, M., and Rogaway, P. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the* 1^{st} *ACM CCS* (1993), ACM Press.

[5] Bleichenbacher, D. Generating Elgamal signatures without knowledge of the secret key. In *Advances in Cryptology: Proceedings of EUROCRYPT 1996* (1996), vol. 1070 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 10–18.

[6] Bleichenbacher, D., and Maurer, U. Directed acyclic graphs, one-way functions and digital signatures. In *Proceedings of Advances in Cryptology – CRYPTO'94* (1994), vol. 963 of *LNCS*, pp. 75–82.

[7] Boneh, D., and Boyen, X. Short signatures without random oracles. In *Proceedings of Advances in Cryptology – EUROCRYPT '04* (2004).

[8] Boneh, D., Mironov, I., and Shoup, V. A secure signature scheme from bilinear maps. In *Topics in Cryptology – CT-RSA 2003: The Cryptographers' Track at the RSA Conference 2003* (2003), vol. 2612 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 98–110.

[9] Boneh, D., Shacham, H., and Lynn, B. Short signatures from the Weil pairing. In *Advances in Cryptology: Proceedings of ASIACRYPT 2001* (2001), vol. 2248 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 514–532.

[10] Brown, D. R. L. Generic groups, collision resistance, and ECDSA. E-print Archive no. 2002/026, 2002. `http://eprint.iacr.org/2002/026/`.

[11] Brown, D. R. L., and Johnson, D. B. Formal security proofs for signatures with partial message recovery. In *Topics in Cryptogology: Proceedings of the RSA Conference, Cryptographers' Track (CT-RSA) 2001* (2001), vol. 2020 of *Lecture Notes in Computer Science*, pp. 126–142.

[12] Camenisch, J., Piveteau, J.-M., and Stadler, M. Blind signatures based on the discrete logarithm problem. In *Advances in Cryptology: Proceedings of EUROCRYPT 1994* (1994), vol. 950 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 428–432.

[13] Canetti, R., Goldreich, O., and Halevi, S. The random oracle methodology, revisited. In *Proceedings of the 30th annual ACM symposium on Theory of computing (STOC'98)* (1998), ACM Press, pp. 209–218.

[14] Chaum, D., and Pedersen, T. P. Wallet databases with observers. In *Advances in Cryptology: Proceedings of CRYPTO'92* (1992), vol. 740 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 89–105.

[15] Cramer, R., and Shoup, V. Signature schemes based on the strong RSA assumption. *ACM Transactions on Information and System Security 3(3)* (2000), 161–185.

[16] Damgård, I., and Koprowski, M. Generic lower bounds for root extraction and signature schemes in general groups. In *Advances in Cryptology: Proceedings of Eurocrypt 2002* (2002), vol. 2332 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 256–271.

[17] Dent, A. W. Adapting the weaknesses of the random oracle model to the generic group model. In *Advances in Cryptology: Proceedings of ASIACRYPT 2002* (2002), Lecture Notes In Computer Science, pp. 100–109.

[18] Even, S., Goldreich, O., and Micali, S. On-line/off-line digital signatures. *Journal of Cryptology 9* (1996), 35–67.

[19] Fischlin, M. A note on security proofs in the generic model. In *Advances in Cryptology: Proceedings of ASIACRYPT'00* (2000), no. 1976 in Lecture Notes in Computer Science, Springer-Verlag, pp. 458–ff.

[20] Frey, G., and H.G.Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Mathematics of Computation 62* (1994), 865–874.

[21] Frey, G., Müller, M., and Rück, H. G. The Tate-pairing and the discrete logarithm applied to elliptic curve cryptosystems. *IEEE Transactions on Information Theory 45* (1999), 1717–1719.

[22] Girault, M. Self-certified public keys. In *Advances in Cryptology: Proceedings of EUROCRYPT'91* (1991), vol. 547 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 490–497.

[23] Goh, E.-J., and Jarecki, S. A signature scheme as secure as the Diffie-Hellman problem. In *Advances in Cryptology: Proceedings of EUROCRYPT 2003* (2003), vol. 2656 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 401–415.

[24] Granboulan, L. PECDSA. How to build a DL-based digital signature scheme with the best proven security. NESSIE Tech. Report no. NES/DOC/ENS/WP5/022/1, 2002.

[25] Günther, C. G. An identity-based key-exchange protocol. In *Advances in Cryptology: Proceedings of EUROCRYPT'89* (1990), vol. 434 of *Lecture Notes in Computer Science*, pp. 29–37.

[26] Horster, P., Michels, M., and Petersen, H. Hidden signature schemes based on the discrete logarithm problem and related concepts. In *Proceedings of the First Conference on Communications and Multimedia Security* (1995), Chapman & Hall, pp. 162–177.

[27] Jakobsson, M., and Schnorr, C.-P. Efficient oblivious proofs of correct exponentiation. In *Proceedings of the IFIP Conference on Communications and Multimedia Security* (1999), vol. 152, Kluwer, pp. 71–86.

[28] Joux, A. A one round protocol for tripartite Diffie-Hellman. In *Proceedings of the 4th International Symposium on Algorithmic Number Theory (ANTS-IV)* (2000), vol. 1838 of *Lecture Notes in Computer Science*, pp. 385–394.

[29] Joye, M., Paillier, P., and Vaudenay, S. Efficient generation of prime numbers. In *Proceedings of the 2nd Annual Workshop on Cryptographic Hardware and Embedded Systems (CHES 2000)* (2000), vol. 1965 of *Lecture Notes in Computer Science*, Springer-Verlage, pp. 340+.

[30] Katz, J., and Wang, N. Efficiency improvements for signature schemes with tight security reductions. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (ACM CCS'03)* (2003), ACM Press, pp. 155–164.

[31] Menezes, A., Okamoto, T., and Vanstone, S. Reducing elliptic curve logarithms in a finite field. *IEEE Transactions on Information Theory IT-39*, 5 (1993), 1639–1646.

[32] Naccache, D., Pointcheval, D., and Stern, J. Twin signatures: an alternative to the hash-and-sign paradigm. In *Proceedings of the 8th ACM Conference on Computer and Communications Security (ACM CCS)* (2001), pp. 20–27.

[33] Naor, M., and Yung, M. Universal one-way hash functions and their cryptographic applications. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing (STOC)* (1989), ACM Press, pp. 33–43.

[34] Nechaev, V. I. Complexity of a determinate algorithm for the discrete logarithm. *Mathematical Notes 55* (1994), 165–172.

[35] Nyberg, K., and Rueppel, R. A new signature scheme based on the DSA giving message recovery. In *Proceedings of the First ACM Conference on Computer and Communications Security (ACM CCS 1993)* (1993), ACM Press, pp. 58–61.

[36] Okamoto, T., and Pointcheval, D. The gap-problems: A new class of problems for the security of cryptographic schemes. In *Public Key Cryptography: 4th International Workshop on Practice and Theory in Public Key Cryptosystems (PKC'01)* (2001), no. 1992 in Lecture Notes in Computer Science, Springer-Verlag, pp. 104–ff.

[37] Petersen, H., and Horster, P. Self-certified keys – concepts and applications. In *Proceedings of the Third Conference on Communications and Multimedia Security* (1997), Chapman & Hall.

[38] Pintsov, L. A., and Vanstone, S. A. Postal revenue collection in the digital age. In *Proceedings of Financial Cryptography* (2000), Lecture Notes in Computer Science, Springer-Verlag.

[39] Pointcheval, D., and Stern, J. Security proofs for signature schemes. In *Proceedings of Advances in Cryptology – Eurocrypt '96* (1996).

[40] Rompel, J. One-way functions are necessary and sufficient for secure signatures. In *Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing* (1990), ACM Press, pp. 387–394.

[41] Schnorr, C. P., and Jakobsson, M. Security of discrete log cryptosystems in the random oracle + generic model. In *Conference on The Mathematics of Public-Key Cryptography* (The Fields Institute, Toronto, Canada, 1999).

[42] Shoup, V. Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology: Proceedings of Eurocrypt'97* (1997), Lecture Notes in Computer Science, Springer-Verlag, pp. 256–266. Revised version: `http://www.shoup.net/papers/`.

[43] Yen, S.-M., Laih, C.-S., and Lenstra, A. K. Multi-exponentiation. *IEE Proceedings in Computers and Digital Techniques 141*, 6 (November 1994), 325–326.