Security of a Nyberg-Rueppel Signature Variant

Preliminary Version

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

Abstract

This paper describes a discrete-logarithm based signature scheme – the modified Nyberg-Rueppel signature (mNR) – that is secure in the Generic Group Model (GM). As a signature on short messages, the scheme does not use any hashes and therefore it does not require the random oracle model of computation for its analysis, while for long messages it only requires a cryptographic hash function satisfying standard assumptions such as collision-resistance. The new scheme has the property that it can be efficiently verifiably encrypted, such that multiple applications of the verifiable encryption protocol are unlinkable, i.e., not correlatable to each other. This was shown in [AdM03], where the scheme was first introduced and this property was used in the construction of a group signature scheme. The security argument for the mNR we provided in the appendix of that work was incorrect, however, and is now supplanted by the analysis in this paper.

We show that the security of our scheme is equivalent to that of a twin signature [NPS01], while achieving computational and bandwidth improvements.

Keywords: Generic Group Model, signature schemes, Nyberg-Rueppel variants

1 Introduction

Several constructions of digital signatures based on arbitrary one-way functions are known [NY89, Rom90]. 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 [BM94, EGM96]. 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 signature scheme – though that scheme required a non-standard hash function satisfying a division intractability assumption. In [NPS01], the GHR scheme was revised using an ingenious technique (called 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 [CS00] 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 $\mathbf{Z}_n^*$, the multiplicative residues modulo n, it is hard to find an integer $e > 1$ and $t \in \mathbf{Z}_n^*$ such that $t^e = 1 \bmod n$. It was introduced in [BP97].)

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[1] is believed to hold, but where the Decisional Diffie-Hellman is easy. The Gap Diffie-Hellman problems were formalized in [OP01], and the first application of GDH groups in cryptography (for one-round tripartite key agreement) appeared in [Jou00], but the existence of groups where the DDH is easy (but the CDH still looks difficult) was noticed earlier: [FH94, FMR99]. Recently, it has been shown that certain "short signatures" in GDH groups are provably secure in the standard model [BMS03, BB04]. These signatures are very short, and therefore could be useful in bandwidth-constrained settings, but are relatively computationally costly when compared to discrete-logarithm based signature schemes in a similar setting.

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 introduced in [BR93]) 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 ([JG03]) and to the DDH ([KW03]), both variants of a signature scheme first proposed by Chaum and Pedersen [CP92], and also considered by Jakobsson and Schnorr [JS99].

An alternative for obtaining proofs in the discrete logarithm setting is the Generic Group Model. The GM (introduced in [Sho97, Nec94]) 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 in 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 [Sho97, DK02]. 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 has been shown in [Fis00]. 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 the real problem instances, in the sense that a group admits multiple encodings. In contrast, hash functions are deterministic, there are just a few

[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 $z = g^{ab}$.

hashing schemes in use, and most protocols do not randomize the hash by using keys. Finally, 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 [CGH98, Den02].

Some signature schemes have been shown secure in the GM with additional (or reduced) assumptions; for instance, see[Gra02] 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 [SJ99]. Also, in [BJ01] a proof of the security of the PVSSR scheme [PV00] (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 [Bro02], which also includes interesting remarks about parallels between the ROM and the GM. (For more information about the relationship between the ROM and the GM, see also [Den02], which shows that similar separation results between the standard model and the random oracle model can be adapted to separate the GM and the standard model.) Another instance is the work on twin signatures [NPS01]. Our work has interesting connections with the twin signature paradigm, and in particular we shown a reduction (in the standard model) from our scheme to a twin signature construction in section §5.

The mNR signature proposed here relies on hash functions only to provide collision-resistance when the scheme is extended to arbitrary length messages. Instead, we use proof techniques inspired by [NPS01]. In particular, we achieve a signature scheme provably reducible to the discrete logarithm in the generic model that 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 scheme generates a signature of same size as in other common Elgamal signature schemes. This proof of security of the mNR signature provides support for the claims in [AdM03], where this signature was introduced, but the provided analysis included incorrect arguments. We refer the reader to that work for the construction of the efficient, unlinkable verifiable encryption of the mNR signature.

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
[BMS03, BB04]	Std. model	GDH	tight
THIS	GM	DL	tight
[Bro02]	GM	DL	tight
[NPS01]	GM	DL	tight
[KW03]	ROM	DDH	tight
[JG03]	ROM	CDH	tight
[SJ99]	ROM+GM	DL	tight
[PS96]	ROM	DL	loose

Statement of results: This paper contains a proof of security for the modified Nyberg-Rueppel signature in the generic model, without random oracles. We also show a reduction (in the standard model) of our signature scheme to a twin signature variant, while the mNR signature requires fewer exponentiations and are shorter than the equivalent twin construction.

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, and an analysis of its relative performance with respect to the one attainable using the twin signature paradigm.

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 $\mathbf{Z}_p^*$, 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 [JPV00].

Let U 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 $\mathbf{Z}_p^*$ 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 $\mathbf{Z}_p^*$. First, U generates a random value $k \in [1, q-1]$, and computes $r = mg^k \bmod p$. Next, U 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 may recover the signed message by computing:

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

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 very small fraction of all values in the range. In particular that implies that $\nu < \log q + \kappa$, where κ is a security parameter. 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 $\mathbf{Z}_p^*$. 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 $\mathbf{Z}_p^*$, the multiplicative residues modulo a prime p, or it could be a subgroup of $\mathbf{Z}_n^*$, 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 $\rho : \mathcal{G} \longrightarrow \mathbf{Z}$. 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 $\mathbf{Z}_n^*$ 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 $\mathbf{Z}_p$, 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 $\mathcal{G} = \mathbf{Z}_p^*$ 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).

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:

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

[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.

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 the only examples of signatures provable 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.

An advantage of the mNR signature is that since it does not involve hash functions, and therefore can be verifiably encrypted efficiently, serving as a building block for other protocols, for instance the unlinkable verifiable encryption of signatures in [AdM03].

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:

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

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.

4 Proof in the Generic Model

In this section, we consider the security of mNR in the Generic Model. The GM considers algorithms that access group operations (and indeed the group encoding) through black box function calls. This proof is inspired by the techniques found in [NPS01].

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 prime order.

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

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 $\mathbf{Z}_q^+$ and r is in $\mathcal{G}$:

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 $\mathbf{Z}_q$. 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 $\mathbf{Z}_q$.

If, during the execution of the protocol, it happens that $F_i = F_j$ with $i \neq j$, it follows that $F = F_i - F_j$ is a non-zero polynomial, with $F(1, x, z) = 0 \bmod q$. Let $F = a + bX + cZ$, for some coefficients a, b, and c in $\mathbf{Z}_q$, 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 $r = \rho(\sigma(zm) \oplus \sigma(-xr) \oplus \sigma(-s))$. Let $P = mZ - rX - s$. 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 (since $\{F_j\}_{j=1,\dots,u}$ is a safe sequence). There is a possibility that a polynomial F_i, with $i \neq j$, satisfies $F_i(x, z) = P(x, z)$. As before, if $F_i = P$ 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 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 r and s at random and also a random encoding σ_k satisfying $r = \rho(\sigma_k)$. Later, when the verification algorithm requests the value $\sigma(zm - rx - s)$, 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 r and s and therefore has committed to the signature forgery query, but before the actual forging query is performed. This last event has probability smaller than $1/q$ for each query.

4.1 The case of unknown order

In the case of unknown order, the algorithm computes the sequence $\{F_i\}_{i=1,\dots,u}$ 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. This intuition can be formalized by looking at the distribution of common factors among randomly generated integers.

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.

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.

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 or unknown composite order, 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, $m, s \in [1, q-1], r \in \mathcal{G},\ g_1^m = r y^{\rho(r)} g^s \in \mathcal{G}$.

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. According to the twin signature paradigm [NPS01], 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 (that can be batched), compared with the verification of the twin signature, that requires four exponentiations (batching also possible).

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 Conclusions

This is a preliminary version, dedicated to expounding the security of a variant of the Nyberg-Rueppel signature scheme. The mNR signature presents a more efficient alternative to the twin signature generic DSA or twin generic NR, and has applications to the construction of complex cryptographic protocols. All comments will be appreciated.

References

[AdM03] Giuseppe Ateniese and Breno de Medeiros. Efficient group signatures without trapdoors. In *Proceedings of Advances in Cryptology – ASIACRYPT 2003*, 2003. Revised version: `http://eprint.iacr.org/2002/173`.

[BB04] Dan Boneh and X. Boyen. Short signatures without random oracles. In *Proceedings of Advances in Cryptology – Eurocrypt '04*, 2004.

[BJ01] Daniel R. L. Brown and Don B. Johnson. Formal security proofs for signatures with partial message recovery. In *Topics in Cryptogology: Proceedings of the RSA Conference, Cryptographers' Track (CT-RSA) 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 126–142, 2001.

[BM94] Daniel Bleichenbacher and Ueli Maurer. Directed acyclic graphs, one-way functions and digital signatures. In *Proceedings of Advances in Cryptology – CRYPTO'94*, volume 963 of *LNCS*, pages 75–82, 1994.

[BMS03] Dan Boneh, Ilya Mironov, and Victor Shoup. A secure signature scheme from bilinear maps. In *Topics in Cryptology – CT-RSA 2003: The Cryptographers' Track at the RSA Conference 2003*, volume 2612 of *Lecture Notes in Computer Science*, pages 98–110. Springer-Verlag, 2003.

[BP97] N. Baric and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *Advances in Cryptology: Proceedings of Eurocrypt'97*, Lecture Notes in Computer Science, pages 480–494, 1997.

[BR93] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proc. of the* 1^{st} *ACM CCS*, 1993.

[Bro02] Daniel R. L. Brown. Generic groups, collision resistance, and ecdsa. E-print Archive no. 2002/026, 2002. `http://eprint.iacr.org/2002/026/`.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. In *Proceedings of the 30th annual ACM symposium on Theory of computing (STOC'98)*, pages 209–218. ACM Press, 1998.

[CP92] David Chaum and Torben Pryds Pedersen. Wallet databases with observers. In *Advances in Cryptology: Proceedings of CRYPTO'92*, volume 740 of *Lecture Notes in Computer Science*, pages 89–105. Springer-Verlag, 1992.

[CS00] R. Cramer and V. Shoup. Signature schemes based on the strong rsa assumption. *ACM Transactions on Information and System Security*, 2000.

[Den02] Alexander W. Dent. Adapting the weaknesses of the random oracle model to the generic group model. In *Advances in Cryptology: Proceedings of ASIACRYPT 2002*, Lecture Notes In Computer Science, pages 100–109, 2002.

[DK02] Ivan Damgård and Maciej Koprowski. Generic lower bounds for root extraction and signature schemes in general groups. In *Advances in Cryptology: Proceedings of Eurocrypt 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 256–271. Springer-Verlag, 2002.

[EGM96] Shimon Even, Oded Goldreich, and Silvio Micali. On-line/off-line digital signatures. *Journal of Cryptology*, 9:35–67, 1996.

[FH94] G. Frey and H.G.Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Mathematics of Computation*, 62:865–874, 1994.

[Fis00] Marc Fischlin. A note on security proofs in the generic model. In *Advances in Cryptology: Proceedings of ASIACRYPT'00*, number 1976 in Lecture Notes in Computer Science, pages 458–ff. Springer-Verlag, 2000.

[FMR99] G. Frey, M. Müller, and H. G. Rück. The tate-pairing and the discrete logarithm applied to elliptic curve cryptosystems. *IEEE Transactions on Information Theory*, 45:1717–1719, 1999.

[Gra02] Louis Granboulan. 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.

[JG03] Stanislaw Jarecki and Eu-Jin Goh. A signature scheme as secure as the Diffie-Hellman problem. In *Advances in Cryptology: Proceedings of EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 401–415. Springer-Verlag, 2003.

[Jou00] Antoine Joux. A one round protocol for tripartite Diffie-Hellman. In *Proceedings of the 4th International Symposium on Algorithmic Number Theory (ANTS-IV)*, volume 1838 of *Lecture Notes in Computer Science*, pages 385–394, 2000.

[JPV00] Marc Joye, Pascal Paillier, and Serge Vaudenay. Efficient generation of prime numbers. In *Proceedings of the 2nd Annual Workshop on Cryptographic Hardware and Embedded Systems (CHES 2000)*, volume 1965 of *Lecture Notes in Computer Science*, pages 340+. Springer-Verlage, 2000.

[JS99] Markus Jakobsson and Claus-Peter Schnorr. Efficient oblivious proofs of correct exponentiation. In *Proceedings of the IFIP Conference on Communications and Multimedia Security*, volume 152, pages 71–86. Kluwer, 1999.

[KW03] Jonathan Katz and Nan Wang. Efficiency improvements for signature schemes with tight security reductions. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (ACM CCS'03)*, pages 155–164. ACM Press, 2003.

[Nec94] V. I. Nechaev. Complexity of a determinate algorithm for the discrete logarithm. *Mathematical Notes*, 55:165–172, 1994.

[NPS01] David Naccache, David Pointcheval, and Jacques Stern. 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.

[NY89] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing (STOC)*, pages 33–43. ACM Press, 1989.

[OP01] Tatsuaki Okamoto and David Pointcheval. 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)*, number 1992 in Lecture Notes in Computer Science, pages 104–ff. Springer-Verlag, 2001.

[PS96] D. Pointcheval and J. Stern. Security proofs for signature schemes. In *Proceedings of Advances in Cryptology – Eurocrypt '96*, 1996.

[PV00] L. A. Pintsov and S. A. Vanstone. Postal revenue collection in the digital age. In *Proceedings of Financial Cryptography*, Lecture Notes in Computer Science. Springer-Verlag, 2000.

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing*, pages 387–394. ACM Press, 1990.

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology: Proceedings of Eurocrypt'97*, Lecture Notes in Computer Science, pages 256–266. Springer-Verlag, 1997. Revised version: `http://www.shoup.net/papers/`.

[SJ99] Claus Peter Schnorr and Markus Jakobsson. 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.