Cryptography in Subgroups of $\mathbb{Z}_n^*$

Jens Groth[⋆]

Dept. of Computer Science, UCLA
jg@brics.dk

Abstract. We demonstrate the cryptographic usefulness of a small subgroup of $\mathbb{Z}_n^*$ of hidden order. Cryptographic schemes for integer commitment and digital signatures have been suggested over large subgroups of $\mathbb{Z}_n^*$, by reducing the order of the groups we obtain quite similar but more efficient schemes. The underlying cryptographic assumption resembles the strong RSA assumption.
We analyze a signature scheme known to be secure against known message attack and prove that it is secure against adaptive chosen message attack. This result does not necessarily rely on the use of a small subgroup, but the small subgroup can make the security reduction tighter.
We also investigate the case where $\mathbb{Z}_n^*$ has semi-smooth order. Using a new decisional assumption, related to high residuosity assumptions, we suggest a homomorphic public-key cryptosystem.

Keywords: RSA modulus, digital signature, homomorphic encryption, integer commitment.

Remark. In comparison with the conference version the paper has been changed in three places: In the definition of the strong RSA subgroup assumption we admit $w \in \mathbb{Z}_n^*$. We have changed the proof of Theorem 2. And in light of the analysis by Coron et al. [CJM+11] we have increased the suggested security parameter in Definition 1 from 100 to 160.

1 Introduction

Consider an RSA-modulus $n = pq$, where p and q are large primes. Many cryptographic primitives take place in the multiplicative group $\mathbb{Z}_n^*$ and use the assumption that even if n is public, the order of the group $\varphi(n) = (p-1)(q-1)$ is still unknown. Concrete examples of such primitives are homomorphic integer commitments [FO97,DF02], public key encryption [RSA78,Rab79,Pai99,CF85,KKOT90,NS98] and digital signatures that do not use the random oracle model [BR93] in their security proofs [CS00,CL02,Fis03].

In order to speed up cryptographic computations it is of interest to find as small groups of hidden order as possible. We suggest using a small subgroup of $\mathbb{Z}_n^*$. More precisely, if we have primes $p'|p-1, q'|q-1$, then we look at the unique subgroup $\mathbb{G} \leq \mathbb{Z}_n^*$ of order $p'q'$. We make a strong root assumption for this group, which roughly states that it is hard to find a non-trivial root of a random element in $\mathbb{G}$. We call this the strong RSA subgroup assumption.

Following Cramer and Shoup [CS00] several very similar signature schemes have been suggested. One variation is the following: We publish n and elements $a, g, h \in \mathrm{QR}_n$. To sign a 160-bit message m, select at random a 161-bit randomizer r and a 162-bit prime e. Compute y so $y^e = ag^m h^r \bmod n$. The signature is (y, e, r). A natural question to ask is whether we really need the randomizer r. We analyze this question and show that indeed it is not needed provided we are willing to accept a weaker security reduction.

Restricting ourselves to an even more specialized group, we look at $n = pq = (2p'r_p+1)(2q'r_q+1)$, where r_p, r_q consists of distinct odd prime factors smaller than some low bound B. We can form a new cryptosystem using modular arithmetic in $\mathbb{Z}_n^*$. Let g have order $p'q'r_g$ and h have order $p'q'$.

[⋆] Work done while at Cryptomathic, Denmark and BRICS, Dept. of Computer Science, University of Aarhus, Denmark

Assuming random elements of $\mathbb{G}$ are indistinguishable from random elements of QR_n we can encrypt m as $c = g^m h^r \bmod n$. To decrypt we compute $c^{p'q'} = g^{p'q'm} h^{p'q'r} = (g^{p'q'})^m \bmod n$. Since $g^{p'q'}$ has order $r_g | r_p r_q$, which only has small prime factors, it is now possible to extract $m \bmod r_g$. This cryptosystem is homomorphic, has a low expansion rate $\frac{|c|}{|m|}$ and fast encryption. The decryption process is slow, yet as we shall see, there are applications of this kind of cryptosystem. A nice property of the cryptosystem is that under the strong RSA assumption it serves at the same time as a homomorphic integer commitment scheme. This comes in handy in verifiable encryption where we want to prove that the plaintext satisfies some specified property.

2 Subgroup Assumptions

As mentioned in the introduction, it is of interest to find a small group, where some sort of strong root assumption holds. Obviously, a prerequisite for a strong root assumption is that the order of the group is hidden. Otherwise we have for any $g \in \mathbb{G}$ that $g = g^{1+ord(\mathbb{G})}$ giving us a non-trivial root. We suggest using a subgroup of $\mathbb{Z}_n^*$, where n is some suitable RSA modulus.

A small RSA subgroup of unknown order. Throughout the paper we shall work with RSA moduli on the form $n = pq$, where p, q are primes. We choose these moduli in a manner such that $p = 2p'r_p + 1, q = 2q'r_q + 1$, where p', q' are primes so there is a unique subgroup $\mathbb{G} \leq \mathbb{Z}_n^*$ of order $p'q'$. Let g be a random generator for this group. We call (n, g) an RSA subgroup pair.

Definition 1 (Strong RSA subgroup assumption). *Let K be a key generation algorithm that produces an RSA subgroup pair (n, g). The strong RSA subgroup assumption for this key generation algorithm is that it is infeasible to find $u, w \in \mathbb{Z}_n^*$ and $d, e > 1$ such that $g = uw^e \bmod n$ and $u^d = 1 \bmod n$.*

In comparison with the strong RSA assumption [BP97] we have weakened the assumption by only worrying about non-trivial roots of elements from $\mathbb{G}$. On the other hand, we have strengthened the assumption by generating the RSA modulus in a special way and publicizing a random generator g of a small subgroup of $\mathbb{Z}_n^*$.

We write $\ell_{p'}, \ell_{q'}$ for the bit-length of the primes p', q'. The order of $\mathbb{G}$ then has bit-length $\ell_G = \ell_{p'} + \ell_{q'}$. A possible choice of parameters is $\ell_{p'} = \ell_{q'} = 160$.[1] We shall also use a statistical hiding parameter ℓ_s. Given some number a and a random $|a| + \ell_s$-bit integer r, the idea is that $a + r$ and r should be statistically indistinguishable. A reasonable choice is $\ell_s = 60$.

Lemma 1. *Consider a subgroup pair (n, g) generated in a way such that the strong RSA subgroup assumption holds. Let $g_1, \ldots, g_k$ be randomly chosen generators of $\mathbb{G}$. Give $(n, g_1, \ldots, g_k)$ as input to an adversary $\mathcal{A}$ and let it produce $(y, e, e_1, \ldots, e_k)$ such that $y^e = g_1^{e_1} \cdots g_k^{e_k} \bmod n$. If $e = 0$, then $e_1, \ldots, e_k = 0$. Else we have $e | e_1, \ldots, e | e_k$ and $y = u \prod_{i=1}^k g_i^{e_i/e} \bmod n$, where $u^e = 1 \bmod n$.*

Proof. Pick $\gamma_1, \ldots, \gamma_k \leftarrow \{0,1\}^{\ell_G + \ell_s}$ and set $g_i = g^{\gamma_i} \bmod n$. We give $(n, g_1, \ldots, g_k)$ to $\mathcal{A}$ that with noticeable probability produces $(y, e, e_1, \ldots, e_k)$. We have $y^e = g^{\sum_{i=1}^k \gamma_i e_i} \bmod n$. If $e = 0$ then $g = g^{1 + \sum_{i=1}^k \gamma_i e_i}$. Unless $e_1, \ldots, e_k = 0$ this is likely to be a breach of the strong RSA subgroup assumption.

Assume from now on $e \neq 0$. Let $d = \gcd(e, \sum_{i=1}^k \gamma_i e_i)$ and choose α, β such that $d = \alpha e + \beta \sum_{i=1}^k \gamma_i e_i$. We have $g^d = g^{\alpha e + \beta \sum_{i=1}^k \gamma_i e_i} = (g^\alpha y^\beta)^e \bmod n$. If $p'q' | d$ then $g = g^{1+e}$ and a breach of the strong RSA subgroup assumption has been found. If $1 < \gcd(d, p'q') < p'q'$, then we have $1 < \gcd(g^d - 1, n) < n$ giving us a non-trivial factorization of n, and indirectly a breach of the strong RSA subgroup assumption. Therefore, d is invertible modulo $p'q'$ and we have $g = u(g^\alpha y^\beta)^{e/d} \bmod n$, where $u^d = 1 \bmod n$. Unless $d = \pm e$, this breaks the strong RSA subgroup assumption.

[1] In the conference version of the paper we suggested using $\ell_{p'} = \ell_{q'} = 100$. However, Coron et al. [CJM+11] have shown that a factorization attack with complexity $\tilde{O}(\sqrt{p'})$ exists.

So $e|\sum_{i=1}^{k}\gamma_i e_i$. Write $\gamma_i=\kappa_i p'q'+\lambda_i$. We have $e|p'q'\sum_{i=1}^{k}\kappa_i e_i+\sum_{i=1}^{k}\lambda_i e_i$. Since κ_i is completely hidden to the adversary and randomly chosen this implies $e|e_i$ for all i. We now have $y=u\prod_{i=1}^{k}g_i^{e_i/e} \bmod n$, where $u^e=1 \bmod n$. □

Definition 2 (Decisional RSA subgroup assumption). *Let K be a key generation algorithm that produces an RSA subgroup pair (n,g). The decisional RSA subgroup assumption for this key generation algorithm K is that it is hard to distinguish elements drawn at random from $\mathbb{G}$ and elements drawn at random from QR_n.*

The assumption is related to high-residuosity assumptions made by other authors [GM84,CF85,KKOT90,NS98]. These assumptions are on the form: Given (n,r), where $r|r_pr_q$, it is hard to distinguish a random element and a random element on the form $z^r \bmod n$. In comparison, the decisional RSA subgroup assumption is weaker in the sense that we do not publish $r=r_pr_q$. On the other hand, it is stronger in the sense that we may have a much smaller group $\mathbb{G}$.

Under the decisional RSA subgroup assumption, the strong RSA subgroup assumption implies the standard strong RSA assumption. To see this, consider choosing $r\leftarrow\{0,1\}^{\ell_G+\ell_s}$ at random and feeding g^r to the strong RSA assumption adversary. Under the decisional RSA subgroup assumption this looks like a random element and the SRSA assumption adversary might return $w,e>1$ so $g^r=w^e$. Write $r=\kappa p'q'+\lambda$, then κ is perfectly hidden from the adversary. There is at least 50% chance of $\gcd(e,\kappa p'q'+\lambda)\neq e$. This contradicts Lemma 1, which states $e|r$.

2.1 RSA with Semi-smooth Order.

In Section 7, we restrict the way we generate the RSA subgroup pair. Consider choosing p,q so $p=2p'p_1\cdots p_{t_p}+1, q=2q'q_1\cdots q_{t_q}+1$, where $p_1,\dots,p_{t_p},q_1,\dots,q_{t_q}$ are distinct odd primes smaller than some small bound B. We call (n,g) a semi-smooth RSA subgroup pair.

Define $P_B=\prod_{1<p<B, p \text{ is prime}}p$. Choosing h at random and setting $g=h^{P_B}$ we have overwhelming probability of g generating $\mathbb{G}$. In other words, given n it is easy for anybody to find a generator for $\mathbb{G}$. We can therefore save specifying g and just make n public.

Typical parameters would be $\ell_{p'}=\ell_{q'}=160$ and $B=2^{15}$. Setting $\ell_{p_i}=15$, we choose $t=t_p+t_q$ distinct odd primes $p_1,\dots,p_{t_p},q_1,\dots,q_{t_q}$ such that $p=2p'p_1\cdots p_{t_p}+1, q=2q'q_1\cdots q_{t_q}+1$ are primes.

Lemma 2. *Let n be a semi-smooth RSA subgroup modulus generated with parameters as described above. Pick g at random from QR_n and let d be an arbitrary non-negative integer smaller than t. With probability at least $1-1/p'-1/q'-\frac{(t2^{1-\ell_{p_i}})^{d+1}}{(1-t2^{1-\ell_{p_i}})(d+1)!}$ the order of g is greater than $p'q'2^{(t-d)(\ell_{p_i}-1)}$.*

Proof. Consider a generator h of QR_n. Pick at random $x\in\mathbb{Z}_{p'q'r_pr_q}$. Then $g=h^x$ is uniformly distributed in QR_n. Consider the prime factors $p_1,\dots,p_t$ of r_pr_q. We will consider the probability that $x=0 \bmod p_i$ for more than d of these prime factors. Each event is independent of the others and has at most probability $2^{1-\ell_{p_i}}$ of occurring. Therefore, from Lemma 3 we get a probability lower than $\frac{(t2^{1-\ell_{p_i}})^{d+1}}{(1-t2^{1-\ell_{p_i}})(d+1)!}$. Combine this with the probabilities $1/p'$ and $1/q'$ for respectively $x=0 \bmod p'$ and $x=0 \bmod q'$ to conclude the proof. □

Lemma 3. *Consider n independent Bernoulli-trials with probability p, where $np<1$. The probability of having at least k successes out of n trials is lower than $\frac{(np)^k}{(1-np)k!}$.*

Proof.

$$\begin{aligned}\sum_{i=k}^{n}\binom{n}{i}p^i(1-p)^{n-i}&\le\sum_{i=k}^{n}\binom{n}{i}p^i\le\sum_{i=k}^{n}\frac{n^i}{i!}p^i\le\frac{1}{k!}\sum_{i=k}^{n}(np)^i\\&=\frac{1}{k!}\frac{(np)^k-(np)^{n+1}}{1-np}\le\frac{(np)^k}{(1-np)k!}.\end{aligned}$$

□

3 Factorization Attacks

If we can factor n we know $p-1, q-1$ and it is easy to break the strong RSA subgroup assumption. In the case of a semi-smooth RSA subgroup modulus, the factorization would also tell us the factors p', q' and we can break the decisional RSA subgroup assumption. We do not know of any non-factorization attacks that could be used to break either the strong RSA subgroup assumption or the decisional RSA subgroup assumption, therefore we will focus on the possibility of factoring n.

Pollard's rho method. Consider a semi-smooth RSA subgroup pair (n, g). We can use the following variation of Pollard's ρ-method [Pol75] to factor n. We define f by $f(0) = g$ and $f(i+1) = (f(i)+1)^{P_B} \bmod n$. Intuitively this corresponds to taking a random walk on $\mathbb{G}$ starting in g. Actually, modulo p, it corresponds to taking a random walk on a group of size p', and modulo q, it corresponds to taking a random walk on a group of size q'. We now hope to find points i, j such that $f(i) = f(j) \bmod p$ or $f(i) = f(j) \bmod q$. This would give us $\gcd(n, f(i) - f(j)) > 1$ and most likely a non-trivial factor of n. Using Brent's [Bre80] cycle finding method we expect to find a factorization using $\mathcal{O}(\min(\sqrt{p'}, \sqrt{q'}) \log(P_B)) = \mathcal{O}(2^{\ell_{p'}/2} B)$ modular multiplications.

In case (n, g) is simply a normal RSA subgroup modulus it seems hard to find a function f that always ends up inside $\mathbb{G}$. It therefore seems like Pollard's ρ-method is of little use.

Other factorization methods. Other methods such as the baby-step giant-step algorithm of Shanks [Sha71], Pollard's λ-method [Pol78] or Pollard's $p-1$ method [Pol74] seem to use at least $2^{\ell_{p'}}$ modular multiplications.[2]

While the above mentioned algorithms take advantage of a special structure of the divisors of n, other algorithms such as the elliptic curve method (ECM) [Len87] or the general number field sieve (GNFS) [CP01] do not. We therefore believe that the best one can do here is to run the general number field sieve with heuristic running time $\exp((1.92 + o(1)) \ln(n)^{1/3} \ln\ln(n)^{2/3})$.

Dangers. It is of course important not to give away too much information about the factorization of $p-1$ and $q-1$. An adversary knowing p' could compute $\gcd(n, g^{p'} - 1) = p$. For this reason, we do not release $p'q'$.[3]

Likewise, if we were to release $\sigma | (p-1)(q-1)$ with $|\sigma| > |n|/4$ then we may risk the factorization attack described in [NS98]. Therefore, we must make sure that there is enough entropy in the primes $p_i | (p-1)(q-1)$ that the adversary cannot guess a significant portion of them. Unlike other high-residuosity schemes, we cannot publicize the value $\prod_{p_i | (p-1)/2} p_i \prod_{q_i | (q-1)/2} q_i$.

4 Signature

Cramer and Shoup [CS00] suggest an efficient signature scheme based on the strong RSA assumption where security can be proved in the standard model without using random oracles. Subsequently, Fischlin [Fis02] has proposed efficient schemes for both the case of a statefull signer and a stateless signer. Koprowski [Kop03] points out a minor flaw in the statefull signature scheme and an easy correction of it. Camenisch and Lysyanskaya [CL02] have suggested a variant that is more suitable as a building block in larger protocols such as group signatures. Finally, Zhu [Zhu03] suggests a variation that combines the efficiency of the stateless version of Fischlin's scheme with the suitability of the Camenisch and Lysyanskaya signature scheme. All these signature schemes use safe-prime product moduli. We will suggest similar looking signature schemes for both the statefull and the stateless case and prove security under the strong RSA subgroup assumption. We are not the first to use RSA

[2] Coron et al. [CJM+11] have since the publication of this paper demonstrated that it is possible to factor in time $\tilde{O}(\sqrt{p'})$.

[3] Actually, such a factorization attack is possible on scheme 3 of the Paillier cryptosystem [Pai99], since it uses an element $g = 1 \bmod q$. In a subsequent variant [PP99] this has been corrected and they work in a subgroup of the same nature as we do.

moduli that are not safe-prime products, Damgård and Koprowski [DK02,Kop03] have generalized the Cramer-Shoup signature approach to basing signature schemes on general groups with a strong root assumption. RSA subgroups as we suggest using can be seen as an example of such a group.

Key generation: We generate an RSA subgroup $\mathbb{G}$ and pick $a, g, h \leftarrow \mathbb{G}$. The private key is $p'q'$, the order of $\mathbb{G}$. We select a positive integer t so $t(\ell_e - 1) + 1 > \ell_m$.
Public verification key $vk = (n, a, g, h, t)$. Private signature key $sk = p'q'$.
Signature: To sign a message $m \in \{0,1\}^{\ell_m}$, choose an ℓ_e-bit prime e that has not been used before. Choose at random $r \in \mathbb{Z}_{e^t}$. Compute $y = (ag^m h^r)^{e^{-t} \bmod p'q'} \bmod n$.
The signature on m is (y, e, r).
Verification: Given a purported signature (y, e, r) on $m \in \{0,1\}^{\ell_m}$, check that e is an ℓ_e-bit number and $r \in \mathbb{Z}_{e^t}$. It is not necessary to check specifically that e is a prime. Accept if $y^{e^t} = ag^m h^r \bmod n$.

For a stateless signature scheme it would be reasonable to choose $\ell_m = 160, \ell_e = 161$ and $t = 1$. We can use the method from [CS00] to pick the primes e, this way it is still unlikely that we run into a collision where we use the same prime in two different signatures. For a statefull signature scheme we can pick $\ell_m = 160, \ell_e = 28$ and $t = 6$ and keep track of the last prime we used. Whenever we wish to sign, we pick the subsequent prime and use that in the signature. For so small primes, the exponentiation is the dominant computational cost.

Theorem 1. *If the strong RSA subgroup assumption holds for the key generation algorithm, then the signature scheme described above is secure against existential forgery under adaptive chosen message attack.*

Proof. There are three cases to consider. The first case is where the adversary forges a signature using a prime e that it has not seen before. The second case is where the adversary reuses a prime, i.e., $e = e_i$, where e_i is the prime from query i but $r \neq r_i$. The third case is where the adversary reuses both e_i and r_i for some i.

Case 1: $e \neq e_i$. With non-negligible probability, we can guess the number k of signing queries the adversary is going to make. Choose according to the signature algorithm distinct ℓ_e-bit primes $e_1, \ldots, e_k$. Set $E = \prod_{j=1}^{k} e_j^t$. Given random elements $\alpha, \gamma, \eta \in \mathbb{G}$ we set $a = \alpha^E, g = \gamma^E, h = \eta^E$. We give (n, a, g, h) to the adversary. We can answer the ith query since we know e_i^t-roots of a, g, h. Consider now the adversary's signature (y, e, r). We have $y^{e^t} = ag^m h^r = \alpha^E \gamma^{Em} \eta^{Er}$ so by Lemma 1 we have $e^t | E$. This means, $e = e_i$ for some i, i.e., the first case only occurs with negligible probability.

Case 2: $e = e_i, r \neq r_i$. Consider next the case of an adversary that reuses e_i. We guess the query i, where the adversary is going to make the forgery. We pick r_i at random and set up $a = \alpha^E h^{-r_i}, g = \gamma^E, h = \eta^{E/e_i^t}$. We can easily answer queries $j \neq i$, and for query i we return the answer (y_i, e_i, r_i), where $y_i = \alpha^{E/e_i^t} \gamma^{m_i E/e_i^t}$. Consider now the adversary's signature (y, e_i, r) on message m. We have $(y/y_i)^{e_i^t} = g^{m-m_i} h^{r-r_i} = \gamma^{(m-m_i)E} \eta^{(r-r_i)E/e_i^t}$. By Lemma 1, we have $e_i^t | (r - r_i)E/e_i^t$. Since e_i does not divide E/e_i^t and $|r - r_i| < e_i^t$ this means $r = r_i$, so the second case occurs with negligible probability.

Case 3: $e = e_i, r = r_i$. Consider finally the case where the adversary reuses both e_i and r_i. We make the following setup. Pick at random $r_m \in \mathbb{Z}_{e_i^t + 2^{\ell_m}}$. Set $a = \alpha^E g^{-r_m}, g = \gamma^{E/e_i^t}, h = g\eta^E$. On query m_i we pick $r_i = r_m - m_i$, which enables us to compute y_i. r_i is uniformly distributed over $\mathbb{Z}_{e_i^t + 2^{\ell_m}} - m_i$ and has more than 50% chance of being inside $\mathbb{Z}_{e_i^t}$. Conditioned on $r_i \in \mathbb{Z}_{e_i^t}$, we have a correctly distributed signature. Suppose now the adversary forms a new signature (y, e_i, r_i) on message m. We get $(y/y_i)^{e_i^t} = g^{m-m_i} = \gamma^{(m-m_i)E/e_i^t}$. By Lemma 1 we have $e_i^t | (m - m_i)E/e_i^t$ so $m = m_i$. □

To form a signature we make an exponentiation with $e^{-t} \bmod p'q'$. In comparison, the other schemes use an exponent of size ℓ_n. Especially for the statefull signature scheme, we obtain a significant reduction in computation.

Strong signature. A signature scheme is strong if it impossible to form a new signature on a message m, even if we have already seen many signatures on this message under the chosen message attack. If we ensure that no ℓ_e-bit primes divide $\varphi(n)$, then it is impossible to find a non-trivial u such that $u^e = 1$, where e is an ℓ_e-bit prime. Generating the modulus like this makes the signature scheme strong, since this way the adversary can only use y belonging to $\mathbb{G}$ because $((ag^m h^r)^{e^{-t}} y^{-1})^{e^t} = 1$.

Applications. An advantage of the signature scheme is that it allows us to sign a committed message without knowing the content. The receiver creates a commitment $c = ug^m h^r \bmod n$ and proves knowledge of an opening $(m, (u, e, r))$ of c. We then choose a prime e and return (y, e) where $y = (ac)^{e^{-t} \bmod p'q'} \bmod n$. The receiver now has a signature $(y, e, r \bmod e^t)$ on m.

This kind of committed signature can be set up in a safe-prime product modulus as suggested in [CL02]. To hide m this requires a large r. We gain an advantage by working in a small group and thus needing a much shorter r. One application of this is to speed up group signatures such as [CG04].

5 Simplified Signature

It is well known that if a signature scheme secure against known message attack suffices, then we can drop the r in the scheme described in the previous section. I.e., a stateless signature can look like (y, e), where $y^e = ag^m \bmod n$. The public key is also shorter since we do not need h any more. We shall investigate whether this signature scheme is actually secure against adaptive chosen message attack.

Key generation: We generate an RSA subgroup $\mathbb{G}$ and pick $a, g \leftarrow \mathbb{G}$.
Public verification key $vk = (n, a, g)$. Private signature key $sk = p'q'$.

Signature: To sign a message $m \in \{0,1\}^{\ell_m}$ choose a random ℓ_e-bit prime e. Compute $y = (ag^m)^{e^{-1} \bmod p'q'} \bmod n$.
The signature on m is (y, e).

Verification: Given a purported signature (y, e) on $m \in \{0,1\}^{\ell_m}$ check that e is an ℓ_e-bit number. It is not necessary to check specifically that e is a prime. Accept if $y^e = ag^m \bmod n$.

In practice, there may be more convenient ways to choose the prime e than completely at random. Consider for instance the method of Cramer and Shoup for generating 161-bit primes [CS00]. It is important for the proof of Theorem 2 that the primes have a distribution that is somewhat close to uniform though.

Choosing parameters for the signature scheme is not straightforward. We do certainly need $\ell_e > \ell_m$, as well as ℓ_n to be large enough to make factoring n hard. We also want the group $\mathbb{G}$ to be large enough to make it hard to break the strong RSA subgroup assumption. To simplify notation we will assume p', q' both are $\ell_{p'}$-bit primes, i.e., $\ell_G = 2\ell_{p'}$. On the other hand, for reasons that will become apparent in the proof of Theorem 2 we must be able to factor $\ell_e + 2\ell_{p'}$-bit numbers.

Consider a rigorous factorization algorithm such as the class-group-relations method. Lenstra and Pomerance [LP92] prove that it takes time $L(2^\ell) = \exp((1 + o(1))\sqrt{\ln(2^\ell) \ln\ln(2^\ell)})$ to factor an ℓ-bit number. We want $L(\ell) < \ell_n^d$ for some degree d, i.e., a running time that is polynomial in the security parameter. This is satisfied if ℓ is chosen such that $\ell \ln(2) \ln(\ell \ln(2)) \leq (d \ln(\ell_n)/(1+o(1)))^2$. With this choice of ℓ we also have $\ell \leq \frac{\ln^2(\ell_n)}{\ln(2)} \frac{d^2}{(1+o(1))^2 \ln(\ell \ln(2))}$. Letting $\ell = \ell_e + 2\ell_{p'}$ we have an upper bound on the length of $2\ell_{p'} = \ell - \ell_e$.

For the strong RSA subgroup assumption to hold, we need that it is hard to guess the order of the group $\mathbb{G}$. Known algorithms that compute this order use at least time $2^{\ell_{p'}/2}$. We therefore want $2^{\ell_{p'}/2}$ to be superpolynomial in the security parameter. Suppose we choose the parameters so $\ell/4 \leq \ell_{p'}$, then we want to choose ℓ as large as possible so $2^{\ell/8}$ is superpolynomial. To see whether there is room for that consider choosing ℓ so $\ell = \frac{\ln^2(\ell_n)}{\ln(2)} \frac{d^2}{(1+o(1))^2 \ln(\ell \ln(2))}$. We then have

$$2^{\ell/8} = \ell_n^{\frac{\ln(\ell_n)}{\ln(\ell \ln(2))} \frac{d^2}{8(1+o(1))^2}} \geq \ell_n^{\frac{\ln(\ell_n)}{\ln((d \ln(\ell_n)/(1+o(1)))^2)} \frac{d^2}{8(1+o(1))^2}}.$$

This is a superpolynomial function of ℓ_n. So we do have reasonable hope to have wriggle-room for choosing $\ell_e, \ell_{p'}$ so that the strong RSA subgroup assumption holds and at the same time, it takes polynomial time to factor $\ell_e + 2\ell_{p'}$-bit numbers.

Theorem 2. *If the strong RSA subgroup assumption holds for the key generation algorithm and factoring of $\ell_e+2\ell_{p'}$-bit numbers can be done in polynomial time then the signature scheme described above is a strong signature scheme secure against existential forgery under adaptive chosen message attack.*

Proof. We consider two cases. In the first case the adversary forges a signature using a prime e that it has not seen before in an adaptive chosen message attack. In the second case the adversary reuses a prime e_i that it has received in an answer to query i.

Case 1: $e \neq e_i$. Consider first a variation where we choose α, γ at random from $\mathbb{G}$. We guess the number of signature queries the adversary will make and choose at random corresponding primes $e_1, \ldots, e_k$. Let $E = \prod_{i=1}^{k} e_i$. Then $a = \alpha^E, g = \gamma^E$ look like random elements from $\mathbb{G}$ and we can answer the k queries. After having asked the queries the adversary must produce a message m and a signature (y, e) so $y^e = ag^m$. I.e., $y^e = \alpha^E \gamma^{mE}$, which by Lemma 1 implies that $e|E$. Since e must be an ℓ_e-bit number this means $e = e_i$ for some i. Case 1 occurs with negligible probability, a successful forger must reuse a prime e_i from one of the oracle queries.

Case 2: $e = e_i$. Consider a different way to set up the signature scheme. We choose γ at random from $\mathbb{G}$, and $r \leftarrow \{0,1\}^{\ell_e+2\ell_{p'}}$. We guess the number of signing queries k that the adversary will make and an index i for which it will make a forgery. Set $E = \prod_{i \neq j} e_j$. We set $a = \gamma^{Er}, g = \gamma^E$ and give the public key (n, a, g) to the adversary.

It is easy to answer signature queries $j \neq i$ by returning $y = \gamma^{(r+m_j)E/e_j}$ together with e_j. Remaining is the question of answering query i. Suppose signature query i ask for a signature on m_i. Write $r = sp'q' + t$, where $t < p'q'$, then s is statistically hidden to the adversary. This means it must choose m_i independently of s. For any ℓ_e bit prime e and message m_i there is at least one value of s for which $e|sp'q'+t+m_i$. There can be at most $1 + \lfloor 2\ell_{p'}/\ell_e \rfloor$ ℓ_e-bit primes dividing $r+m_i$. By assumption we can factor $r+m_i$ in polynomial time. With some luck it contains an ℓ_e-bit prime factor e_i, if not we give up in the simulation. If there are more we choose one of them at random. We can now return $y_i = \gamma^{E(r+m_i)/e_i}$.

With the method presented above a given ℓ_e-bit prime has at least $2^{2-\ell_e}/(1 + \lfloor 2\ell_{p'}/\ell_e \rfloor)$ of being chosen. In the real signature scheme, the distribution of primes may be different. Consider for instance the method of Cramer and Shoup [CS00] for picking primes, this distribution is very different from what we have. However, we can consider our distribution of primes as a weighted sum of two distributions: The correct distribution and a residual distribution. We include in the residual distribution all the cases where we simply do not find any prime-factor of $r + m_i$. I.e., we have $\text{Dist}_{\text{our}} = w\text{Dist}_{\text{correct}} + (1-w)\text{Dist}_{\text{residual}}$. In [CS00] they suggest using 161-bit primes and get a distribution where none of the possible primes has more than probability 2^{-144} of being chosen. In our distribution each prime, and thus each of those primes, has at least probability 2^{-161} of being chosen. Thus, w can be chosen to be at least 2^{-17}.

With probability w, we end up in a case where we give the adversary a signature that is statistically indistinguishable from a real signature. Consider now a signature (y, e_i) on message m produced by this adversary. We have $y^{e_i} = ag^m$ so $(y/y_i)^{e_i} = g^{m-m_i} = \gamma^{E(m-m_i)}$. By Lemma 1 it must be the case that $e_i|(m-m_i)E$. However, e_i is a prime and $e_i > |m - m_i|$ and e_i does not divide E. Therefore, $m = m_i$. We can therefore not produce a signature on a new message if w is non-negligible.

Strong signature. We still need to argue that the signature scheme is strong. Consider the adversary's signature (y, e_i) on $m = m_i$, where the signature oracle returned (y_i, e_i). We then have $y^{e_i} = y_i^{e_i} = ag^m$. This means $y = uy_i$, where $u^{e_i} = 1$. However, with overwhelming probability $\gcd(e_i, p'q'r_pr_q) = 1$ so $u = 1$. □

In the proof we need to factor $r+m_i$. We discussed the class-group-relations method earlier since this has a rigorously proved run-time. Other possible choices include the GNFS, which is not relevant for practical parameters but gives good asymptotics, and the QS [CP01], which works better than the class-group-relations method in practice. The best option would probably be to use the ECM, which has a heuristic run-time of $L(p)^{\sqrt{2}+o(1)}$, with p being the smallest prime factor.

If we use the ECM we can also consider tackling the original safe-prime setting of this type of signature schemes, where $p = 2p'+1, q = 2q'+1$. In this case r is so large that we cannot reasonably hope to factor $r+m_i$. However, all we need is an ℓ_e-bit prime factor. As long as ℓ_e is small enough, it is feasible to get out such a small prime factor using the ECM.

Applications. Consider a tag-based simulation sound trapdoor commitment scheme as defined by MacKenzie and Yang [MY04]. It takes as input a message and a tag and forms a commitment. With the trapdoor, it is possible to open the commitment with this tag to any message. The hiding property is defined as usual, however, the binding property is strengthened in the following way: Even if we have seen arbitrary trapdoor openings of commitments with various tags, it is still hard to open a commitment to two different messages using a tag for which no commitment has been equivocated.

[MY04] construct a simulation sound trapdoor commitment scheme based on the Cramer-Shoup signature scheme. Essentially, a commitment to message m using tag tag is a simulated honest verifier zero-knowledge argument of knowledge of a signature on tag using challenge m. We can simplify this trapdoor simulation sound commitment scheme by instead simulating an honest verifier zero-knowledge argument of a signature on tag using challenge m, where we use the simplified signature scheme. I.e., we pick a prime e, pick at random r and set $c = r^e(ag^{tag})^{-m} \bmod n$. The commitment is (c, e, tag), while the opening is (r, m). A double opening would give us $(r/r')^e = (ag^{tag})^{m'-m}$. Since $\gcd(e, m'-m) = 1$, this gives us an e-root of ag^{tag}, i.e., a signature on tag.

[MY04] use 5 exponentiations to form their simulation sound trapdoor commitment and remark that using the Fischlin signature scheme it can be reduced to 4 exponentiations. In comparison, we only use 3 exponentiations.

6 Commitment

Homomorphic integer commitments based on the strong RSA assumption were first suggested by Fujisaki and Okamoto [FO97]. Later Damgård and Fujisaki [DF02] corrected a flaw in the security proof of the former paper and generalized the commitment scheme to abelian groups satisfying some specific assumptions. In this section, we suggest a similar integer commitment scheme based on the strong RSA subgroup assumption.

Key generation: We generate an RSA subgroup $\mathbb{G}$ and choose at random two generators g, h. The public key is $pk = (n, g, h)$.

Commitment: To commit to integer m using randomizer (u, e, r), where $u^e = 1 \bmod n, e > 0$ and $r \in \mathbb{Z}$ we compute

$$c = commit_{(n,g,h)}(m; (u, e, r)) = ug^m h^r \bmod n.$$

When making a commitment from scratch we choose $r \leftarrow \{0,1\}^{\ell_G+\ell_s}$ and use the randomizer $(1, 1, r)$.

Opening: To open commitment c we reveal $(m, (u, e, r))$ such that $c = ug^m h^r \bmod n$, where $u^e = 1 \bmod n, e > 0$.

Theorem 3. *The commitment scheme is statistically hiding and if the strong RSA subgroup assumption holds for the key generation algorithm then it is computationally binding.*

Proof. It is easy to see that the commitment is statistically hiding since h^r is almost uniformly distributed on $\mathbb{G}$.

To see that the commitment scheme is binding consider a commitment c and two openings $(m,(u,e,r))$ and $(m',(u',e',r'))$ produced by the adversary. We have $c = ug^m h^r = u'g^{m'}h^{r'}, u^e = 1, (u')^{e'} = 1$. We must have $\gcd(e,p'q') = \gcd(e',p'q') = 1$, since otherwise we can as in the proof of Lemma 1 break the strong RSA subgroup assumption. This means $u,u' \in \mathbb{Z}_n^*/\mathbb{G}$ and therefore $u = u'$. We then have $1^0 = g^{m-m'}h^{r-r'}$. By Lemma 1 we get $m = m'$. □

The commitment scheme has several nice properties. It is homomorphic in the sense that for all $(m,(u,e,r)),(m,(u',e',r'))$ we have $commit_{(n,g,h)}(m+m';(uu',ee',r+r')) = commit_{(n,g,h)}(m;(u,e,r))commit_{(n,g,h)}(m';(u',e',r'))$. It is a trapdoor commitment scheme, if we know both $p'q'$ and x such that $g = h^x$ and an opening $(m,(u,e,r))$ of c, then we can open c to m' by revealing $(m',(u,e,r'))$, where r' is picked at random from $\{0,1\}^{\ell_G+\ell_s}$ such that $r' = (m-m')x + r \bmod p'q'$. Finally, it has the following root extraction property: Consider an adversary that produces $(c,m,(u,e,r),d)$ so $c^e = ug^m h^r, u^d = 1$, then we can find a valid opening of c. Notably, we have $c^{de} = g^{dm}h^{dr}$ so from Lemma 1 we get $e|m, e|r$ and $c = (u')g^{\frac{m}{e}}h^{\frac{r}{e}}$, where $(u')^{ed} = 1$. The homomorphic property combined with the root extraction property means that we can form efficient honest verifier zero-knowledge arguments (Σ-protocols [CDS94]) for many interesting properties of the message inside the commitment.

The commitment schemes of [FO97,DF02] pick the randomness from $\{0,1\}^{\ell_n+\ell_s}$ while we pick the randomness from $\{0,1\}^{\ell_G+\ell_s}$. This means that we have a much shorter exponentiation when computing the commitment.

7 Encryption

Recall that a semi-smooth RSA subgroup modulus $n = (2p'r_p+1)(2q'r_q+1)$ has B-smooth r_p, r_q. Suppose we have $h \in \mathbb{G}$ and g has order $p'q'r_g$. Given $c = g^m h^r$ we can compute $c^{p'q'} = g^{p'q'm}h^{p'q'r} = (g^{p'q'})^{m \bmod r_g}$. Since r_g is B-smooth, we can from this compute $m \bmod r_g$. This is the main idea in the following cryptosystem.

Key generation: Generate an RSA subgroup modulus $n = pq = (2p'r_p+1)(2q'r_q+1)$, where r_p, r_q are B-smooth and all prime factors are distinct. Select $g \leftarrow \mathrm{QR}_n$ and $h \leftarrow \mathbb{G}$.
The public key is (n,g,h). The secret key is the factorization of $\varphi(n)$.

Encryption: We wish to encrypt a message $m \in \{0,1\}^{\ell_m}$ using randomness $(u,r) \in \{-1,1\} \times \mathbb{Z}$. The ciphertext is

$$c = E_{(n,g,h)}(m;(u,r)) = ug^m h^r \bmod n.$$

We usually choose $u = 1$ and $r \leftarrow \{0,1\}^{\ell_G+\ell_s}$.

Decryption: Given a ciphertext $c \in \mathbb{Z}_n^*$ we compute $C_p = c^{p'} = (g^{p'})^{m_p} \bmod p$. Since the order of $g^{p'}$ in $\mathbb{Z}_p^*$ is smooth, we can now find $m_p \bmod p_i$ for all $p_i|r_p, p_i|ord(g)$. Similarly, we can find $m_q \bmod q_i$ for $q_i|r_q, q_i|ord(g)$. Using the Chinese remainder theorem, we end up with $m \bmod \gcd(r_p r_q, ord(g))$. If $m \in \{0,1\}^{\ell_m}$ we output m, otherwise we output `invalid`.

Theorem 4. *If the decisional RSA subgroup assumption holds for the key generation algorithm then the cryptosystem is semantically secure against chosen plaintext attack.*

Proof. By the decisional RSA subgroup assumption, we can replace g in the public key with a randomly chosen element from $\mathbb{G}$ without the adversary noticing it. This leaves us with a statistically hiding commitment, which of course does not allow the adversary to distinguish plaintexts. □

It is worthwhile to observe that given a semi-smooth RSA subgroup modulus n an adversary can only produce trivial (u,e) so $u^e = 1, e > 1$. It is with overwhelming probability the case that $u = \pm 1$. To see this first note as in the proof of Lemma 1 that if $\gcd(e,p'q') > 1$, then we can break the strong RSA subgroup assumption. If there is a prime $p_i < B$ so $p_i|\gcd(e, ord(u))$ then we can find s so $U = u^{e/p_i^s} \neq 1 \bmod n$ and $U^{p_i} = 1 \bmod n$. This means $U = 1 \bmod p, U \neq 1 \bmod q$ or the other way around. I.e., $1 < \gcd(n, U-1) < n$ gives us a factorization of n.

The cryptosystem looks just like the integer commitment scheme, where we always choose $u = \pm 1$ and $e = 2$. As we argued above it is not possible for an adversary to find $u \neq \pm 1$ so this is not a problem. Since we cannot distinguish between a random g from QR_n and a random g from $\mathbb{G}$ we actually have all the nice properties of the commitment scheme we presented before. In particular, the cryptosystem is homomorphic as long as we are careful to avoid overflows where the messages are longer than ℓ_m bits. It also has the root extraction property that is useful in zero-knowledge arguments.

Let us consider the length of the messages ℓ_m. The ciphertext has length ℓ_n, however, ℓ_G bits are used for the randomization. Suppose d is chosen such that there is negligible probability that more than d of the primes p_i, q_i do not divide the order of g. We are then left with $\ell_m \leq (t-d)(\ell_{p_i} - 1)$.

In comparison with other cryptosystems such as [Pai99,NS98,OU98] the present scheme offers a better expansion rate. Generalized Paillier encryption [DJ01] has expansion rate $|c|/|m| = 1 + 1/s$, where s is some small positive integer. Their scheme, however, requires a modulus of size n^{s+1}. Okamoto-Uchiyama encryption uses a modulus n of about the same size as we do, however, the expansion rate is around 3. Our cryptosystem has an expansion rate as low as $\ell_n/\ell_m = \ell_n/((t-d)(\ell_{p_i} - 1))$. With the parameters $\ell_n = 1280, \ell_{p'} = \ell_{q'} = 160, B = 2^{15}, t = 64, d = 7$ we get from Lemma 2 that the order of g has bit-length no smaller than $320 + (64-7)(15-1) = 1118$ with probability higher than $1 - 2^{-80}$, giving us an expansion rate of $1280/798 \approx 1.6$.

Applications. Strengthening the decisional RSA subgroup assumption a little, we could get away with picking g of full order $p'q'r_pr_q$. This way, we can increase the message space $\{0,1\}^{\ell_m}$ slightly. According to Lemma 2 a random g does have high order so the difference is not that big though.

The reason we prefer a random g is that part of the public key can be picked by coin-flipping. This property can be useful. Consider as an example the universally composable commitment scheme of Damgård and Nielsen [DN02,Nie03]. In their scheme, they first carry out a 2-move coin-flipping protocol to determine the key for what they call a mixed-commitment scheme. If a corrupt party is making a commitment, the coin-flipping protocol makes the key be a so-called X-key. The setup is such that a simulator knows the corresponding secret key, and thus can extract what the corrupt party committed to. On the other hand, if an honest party is making a commitment we can tweak the coin-flipping protocol to produce a so-called E-key. A commitment under an E-key is equivocable. The simulator can therefore make the commitment now, and later when learning the real value it can equivocate the commitment to this value.

Damgård and Nielsen suggest universally composable commitments based on the subgroup-p assumption [OU98] and based on the decisional composite residuosity assumption [Pai99]. Our cryptosystem provides an efficient alternative to these variations. We generate a (n,h) as in the key generation of the cryptosystem. The corresponding trapdoor is the factorization of $\varphi(n)$. Running a coin-flipping protocol we get a random element g. Using this g we can commit to $m \in \{0,1\}^{\ell_m}$ as $g^m h^r$. If g is random, then it is a ciphertext and we can extract m with our knowledge of the factorization. On the other hand, we could also select $x \leftarrow \{0,1\}^{\ell_G + \ell_s}, g = h^x$, which would make g an E-key. With this g we have set up the statistically hiding commitment scheme and with knowledge of the trapdoor x we can form commitments that can be opened to our liking.

Notice, we only use the decryption property in the simulation in the security proof. In a real run of the universally composable commitment protocol we never decrypt anything. Therefore, it does not hurt us that the decryption process is slow.

Consider further the universally composable threshold cryptosystem of Damgård and Nielsen [DN03]. Here the sender encrypts his message and at the same time makes a universally composable commitment to it. He also proves that the two messages are identical.

The cryptosystem itself needs to be a threshold cryptosystem. They suggest using a variation over the Paillier cryptosystem, which gives us a message space on the form $\mathbb{Z}_n$, with known n. However, the UC commitment scheme does not need to be a threshold scheme. Actually, it is only used in the security proof where the simulator can extract the message from the UC commitment rather than the ciphertext itself. Using our universally composable commitment scheme, we have the additional advantage that it serves as an integer commitment. This means, it is easy to make an efficient zero-knowledge argument of the ciphertext and the commitment containing the same message, even though the message spaces are different.

8 Acknowledgment

We thank Jean-Sebastien Coron, Antoine Joux, Avradip Mandal, David Naccache and Mehdi Tibouchi for sharing an early manuscript of their factorization method [CJM$^+$11].

References

[BP97] Niko Bari and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *EUROCRYPT*, volume 1233 of *Lecture Notes in Computer Science*, pages 480–494, 1997.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS*, pages 62–73, 1993.

[Bre80] Richard P. Brent. An improved Monte Carlo factorization algorithm. *BIT*, 20:176–184, 1980.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO*, volume 893 of *Lecture Notes in Computer Science*, pages 174–187, 1994.

[CF85] Josh D. Cohen and Michael J. Fischer. A robust and verifiable cryptographically secure election scheme. In *FOCS*, pages 372–382, 1985.

[CG04] Jan Camenisch and Jens Groth. Group signatures: Better efficiency and new theoretical aspects. In *SCN*, volume 3352 of *Lecture Notes in Computer Science*, pages 120–133, 2004.

[CJM$^+$11] Jean-Sébastien Coron, Antoine Joux, Avradip Mandal, David Naccache, and Mehdi Tibouchi. Cryptanalysis of the rsa subgroup assumption from tcc 2005. In *Public Key Cryptography*, volume 6571 of *Lecture Notes in Computer Science*, pages 147–155, 2011.

[CL02] Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In *SCN*, volume 2576 of *Lecture Notes in Computer Science*, pages 268–289, 2002.

[CP01] Richard Crandall and Carl Pomerance. *Prime Numbers - a Computational Perspective*. Springer Verlag, 2001.

[CS00] Ronald Cramer and Victor Shoup. Signature schemes based on the strong RSA assumption. *ACM Transactions on Information and System Security (TISSEC)*, 3(3):161–185, 2000.

[DF02] Ivan Damgård and Eiichiro Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 125–142, 2002.

[DJ01] Ivan Damgård and Mads J. Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In *PKC*, volume 1992 of *Lecture Notes in Computer Science*, 2001.

[DK02] Ivan Damgård and Maciej Koprowski. Generic lower bounds for root extraction and signature schemes in general groups. In *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 256–271, 2002.

[DN02] Ivan Damgård and Jesper Buus Nielsen. Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 581–596, 2002.

[DN03] Ivan Damgård and Jesper Buus Nielsen. Universally composable efficient multiparty computation from threshold homomorphic encryption. In *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 247–264, 2003.

[Fis02] Marc Fischlin. On the impossibility of constructing non-interactive statistically-secret protocols from any trapdoor one-way function. In *CT-RSA*, volume 2271 of *Lecture Notes in Computer Science*, pages 79–95, 2002.

[Fis03] Marc Fischlin. The Cramer-Shoup strong-RSA signature scheme revisited. In *PKC*, volume 2567 of *Lecture Notes in Computer Science*, pages 116–129, 2003.

[FO97] Eiichiro Fujisaki and Tatsuaki Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 16–30, 1997.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[KKOT90] Kaoru Kurosawa, Yutaka Katayama, Wakaha Ogata, and Shigeo Tsujii. General public key residue cryptosystems and mental poker protocols. In *EUROCRYPT*, volume 473 of *Lecture Notes in Computer Science*, pages 374–388, 1990.

[Kop03] Maciej Koprowski. Cryptographic protocols based on root extracting. Dissertation Series DS-03-11, BRICS, 2003. PhD thesis. xii+138 pp.

[Len87] Hendrik W. Lenstra. Factoring integers with elliptic curves. *Ann. of Math.*, 126:649–673, 1987.

[LP92] Hendrik W. Lenstra and Carl Pomerance. A rigourous time bound for factoring integers. *J. Amer. Math. Soc.*, 5:483–516, 1992.

[MY04] Philip D. MacKenzie and Ke Yang. On simulation-sound trapdoor commitments. In *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 382–400, 2004.

[Nie03] Jesper Buus Nielsen. On protocol security in the cryptographic model. Dissertation Series DS-03-8, BRICS, 2003. PhD thesis. xiv+341 pp.

[NS98] David Naccache and Jacques Stern. A new public key cryptosystem based on higher residues. In *ACM CCS*, pages 59–66, 1998.

[OU98] Tatsuaki Okamoto and Shigenori Uchiyama. A new public-key cryptosystem as secure as factoring. In *EUROCRYPT*, volume 1403 of *Lecture Notes in Computer Science*, pages 308–318, 1998.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite residuosity classes. In *EUROCRYPT*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–239, 1999.

[Pol74] John M. Pollard. Theorems of factorization and primality testing. *Proc. Cambridge Phil. Soc.*, 76:521–528, 1974.

[Pol75] John M. Pollard. A Monte Carlo method for factorization. *BIT*, 15:331–334, 1975.

[Pol78] John M. Pollard. Monte carlo methods for index computation (mod p). *Math.Comp.*, 32(143):918–924, 1978.

[PP99] Pascal Paillier and David Pointcheval. Efficient public-key cryptosystems provably secure against active adversaries. In *ASIACRYPT*, volume 1716 of *Lecture Notes in Computer Science*, pages 165–179, 1999.

[Rab79] Michael O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[Sha71] Daniel Shanks. Class number, a theory of factorization, and genera. In *1969 Number Theory Institute (Proc. Sympos. Pure Math., Vol. XX, State Univ. New York, Stony Brook, N.Y., 1969)*, pages 415–440. Amer. Math. Soc., Providence, R.I., 1971.

[Zhu03] Huafei Zhu. A formal proof of Zhu's signature scheme. Cryptology ePrint Archive, Report 2003/155, 2003. http://eprint.iacr.org/.