Elementary Attestation of Cryptographically Useful Composite Moduli

Rémi Géraud-Stewart[1,2] and David Naccache[2]

[1] QPSI, Qualcomm Technologies Incorporated, USA
rgerauds@qti.qualcomm.com
[2] ÉNS (DI), Information Security Group
CNRS, PSL Research University, 75005, Paris, France
david.naccache@ens.fr

Abstract. This paper describes a non-interactive process allowing a prover to convince a verifier that a modulus n is the product of two primes (p, q) of about the same size. A further heuristic argument conjectures that $p-1$ and $q-1$ have sufficiently large prime factors for cryptographic applications.

The new protocol relies upon elementary number-theoretic properties and can be implemented efficiently using very few operations. This contrasts with state-of-the-art zero-knowledge protocols for RSA modulus proper generation assessment.

The heuristic argument at the end of our construction calls for further cryptanalysis by the community and is, as such, an interesting research question in its own right.

1 Introduction

Several cryptographic protocols rely on the assumption that an integer $n = pq$ is hard to factor. This includes for instance RSA [RSA78], Rabin [Rab79], Paillier [Pai99] or Fiat–Shamir [FS86]. A user generating their own keys can ensure that n is indeed such a product; however this becomes a concern when n is provided by a third-party. This scenario appears e.g. with Fiat–Shamir identification, or in the context of certificate authentication: carelessly using an externally-provided n may compromise security if n is incorrectly generated. Naturally, one cannot ask for the factor(s) p, q to check n.

The state of the art in the matter are the zero-knowledge procotols of Auerbach–Poettering [AP18] and Camenisch–Michels [CM99], which ascertain that a given n is the product of safe primes of prescribed size. While correct and very useful, these protocols are difficult to implement and analyze, and have high computational costs. This motivates the search for simpler and more efficient solutions.

This paper introduces an alternative protocol that nearly achieves the same functionality with fewer operations and communication. The new protocol is also simpler to understand and therefore to implement.

2 Preliminaries & Building Blocks

This paper uses the following notations: $a|b$ denotes the concatenation of the bitstrings a and b. If c is an integer, $\|c\| = \lceil \log_2 c \rceil$ denotes the size of c, i.e. the minimal number of bits needed to write c. We denote by 0^ℓ the ℓ-bit all-zero string. For $A \in \mathbb{N}$ we denote by $[A]$ the set $\{0, 1, \ldots, A-1\}$. If X is a finite set, then $x \xleftarrow{\$} X$ indicates sampling uniformly at random from X.

2.1 Camenisch–Michels proofs

While this paper does not rely on [CM99] we feel that it is important to recall Camenisch and Michels' protocol (hereafter, CM) given that it is currently considered as the state-of-the-art tool for achieving the functionality that we try to approach by our construction.

CM provides provable guarantees that an RSA modulus is well-formed. At its heart is a pseudo-primality proof, combined with zero-knowledge proofs of knowledge of a discrete logarithm, of a conjunction, and for belonging to a range. We recall here the protocol for the sake of completeness, using the following syntax [CS97]:

$$\mathrm{PK}\{(w) : P(x, w)\}$$

refers to a *proof of knowledge* that property P holds, i.e., that the prover (henceforth $\mathcal{P}$) knows a witness w that makes $P(x, w)$ true.

Let $n = pq$ be the number of interest, $\varepsilon > 1$ be a security parameter, ℓ such that $2^\ell > n$, G a cyclic group of prime order $Q > 2^{5+2\varepsilon\ell}$, and $g, h \in G$. There are two main phases to the protocol [CM99, Sec 5.1]:

1. $\mathcal{P}$ computes

$$c_p \leftarrow g^p h^{r_p} \qquad c'_p \leftarrow g^{\frac{p-1}{2}} h^{r'_p}$$
$$c_q \leftarrow g^q h^{r_q} \qquad c'_q \leftarrow g^{\frac{q-1}{2}} h^{r'_q}$$

   where r_p, r'_p, r_q, r'_q are random integers. The values (c_p, c'_p, c_q, c'_q) are sent to the verifier (henceforth $\mathcal{V}$).
2. Run the protocol

$$\begin{aligned}
&\mathrm{PK}\{(x_1, \ldots, x_{11}) : \\
&\quad c'_p = g^{x_1} h^{x_2} \wedge c'_q = g^{x_3} h^{x_4} \wedge c_p = g^{x_5} h^{x_6} \wedge c_q = g^{x_7} h^{x_8} \\
&\quad \wedge c_p g^{-1} c_p'^{-2} = h^{x_9} \wedge c_q g^{-1} c_q'^{-2} = h^{x_{10}} \wedge g^n c_p^{-1} c_q^{-1} = h^{x_{11}} \\
&\quad \wedge x_1 \in [-2^{\ell'}, 2^{\ell'}] \wedge x_4 \in [-2^{\ell'}, 2^{\ell'}] \\
&\quad \wedge x_1 \in \texttt{pprimes(t)} \wedge x_3 \in \texttt{pprimes(t)} \\
&\quad \wedge x_5 \in \texttt{pprimes(t)} \wedge x_7 \in \texttt{pprimes(t)} \\
&\}
\end{aligned}$$

The pprimes(t) sub-algorithm denotes t rounds of the Lehmann primality test for a committed number [CM99, Sec 4.2] and $\ell' = \varepsilon\ell + 2$. The above protocol is a statistical zero-knowledge proof that $n = pq$ is an RSA modulus where p, q are safe primes.

Remark 1. As noted by [CM99], under some conditions running the protocol of Gennaro et al. [GMR98] after the first phase we can remove the two last pseudo-primality tests and reduce the number of rounds to $t = 1$. However this assumes that n was not adversarially constructed and we cannot rely on this hypothesis here.

Remark 2. As discussed in [CM99], the protocol can be extended to ensure additional properties of p and q, e.g. that $(p + 1)/2$ is prime, or that p and q satisfy some lower bound.

The protocol's cost is dominated by the four pseudo-primality tests, which use $O(t \log n)$ exponentiations and exchange $O(t \log n)$ group elements.

2.2 Goldberg–Reyzin–Sagga–Baldimtsi modulus tests

Our first building block is a very elegant protocol published by Goldberg, Reyzin, Sagga and Baldimtsi (GRSB) [GRSB19]. This protocols allows to verify that n has exactly two prime factors.

A first GRSB protocol checks that a pair (n, e) defines a permutation over $\mathbb{Z}/n\mathbb{Z}$. For typical parameter settings, this proof consists of nine integers, with proof generation and verification requiring both about nine modular exponentiations.

A further protocol in [GRSB19] allows $\mathcal{V}$ to check that n is the product of two distinct primes [GRSB19, Sec. 3.4] in a zero-knowledge fashion. We recall the protocol here:

1. $\mathcal{P}$ and $\mathcal{V}$ agree on a security level κ, integer $m := \lceil 32 \cdot \kappa \cdot \ln 2 \rceil$, and n.
2. $\mathcal{V}$ checks that n is a positive odd integer and that n isn't a prime power, otherwise the protocol stops with a failure.
3. $\mathcal{V}$ chooses m values ρ_i whose Jacobi symbol $(\rho_i|n) = 1$, and sends them to $\mathcal{P}$ as challenge.
4. $\mathcal{P}$ checks for each ρ_i whether it is a quadratic residue modulo n: if it is, $\mathcal{P}$ returns a square root σ_i of ρ_i to $\mathcal{V}$; otherwise $\mathcal{P}$ returns $\sigma_i = 0$.
5. $\mathcal{V}$ checks that there are at least $3m/8$ non-zero σ_i and that they all satisfy $\sigma_i^2 = \rho_i \bmod n$.

As is, this protocol assumes that $\mathcal{V}$ is honest. The honest verifier assumption is removed through a classical derivation of the ρ_is by hashing. Both protocols are very efficient for $\mathcal{P}$ and $\mathcal{V}$.

2.3 Girault–Poupard–Stern signatures

The GPS signature scheme was first proposed by Girault in 1991 [Gir91] without a security proof; a first analysis was given in 1998 [PS98, PS99] by Poupard and Stern. This was further refined by all three authors in 2006 [GPS06]. GPS has been standardized as ISO/IEC 9798-5 in 2004.

Algorithms. GPS consists of four algorithms ($\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Sign}, \mathsf{Verify}$) that we now describe.

- $\mathsf{GPS.Setup}(\lambda) \to \mathsf{pp}$: the public parameters consist of integers A, B and a hash function $h : \{0,1\}^* \to [B]$. They are chosen so that a security level λ is achieved.
- $\mathsf{GPS.Keygen}(\mathsf{pp}) \to (\mathsf{sk}, \mathsf{pk})$: The signer chooses two safe primes $P = 2p + 1$, $Q = 2q + 1$, computes $n \leftarrow PQ$, and finds an element $g \in \mathbb{Z}/n\mathbb{Z}$ whose order is divisible by pq.[3] The order of g (and therefore of the subgroup generated by G) needs not be explicitly known.
  The signer's secret key is $\mathsf{sk} := n - \varphi(n)$, while the public key is given by $\mathsf{pk} := (n, g)$.
- $\mathsf{GPS.Sign}(\mathsf{pp}, \mathsf{sk}, m) \to \sigma$:
  1. $r \xleftarrow{\$} [A]$
  2. $x \leftarrow g^r \bmod n$
  3. $c \leftarrow h(m, x)$
  4. $y \leftarrow r + c \cdot \mathsf{sk}$
  5. If $y \geq A$, restart from step 1 with a new value of r.

  The signature is $\sigma \leftarrow (x, c, y)$.
- $\mathsf{GPS.Verify}(\mathsf{pp}, \mathsf{pk}, m, \sigma) \to \{\text{valid}, \text{invalid}\}$: $\mathcal{V}$ checks the ranges:

$$x > 0 \quad \text{and} \quad x \in [n] \quad \text{and} \quad c \in [B] \quad \text{and} \quad y \in [A]$$

  If either of these checks fails the signature is invalid. Otherwise, $\mathcal{V}$ computes :
  1. $\widetilde{c} \leftarrow h(m, x)$
  2. $\widetilde{x} \leftarrow g^{y - n\widetilde{c}} \bmod n$
  3. If $c = \widetilde{c}$ and $\widetilde{x} = x$ then σ is valid otherwise σ is invalid.

Security of GPS signatures. Under the discrete logarithm with short exponent assumption (DL-SEA, see below), and if $\mathsf{sk} \cdot B/A$ and $1/B$ are negligible, GPS signatures are existentially unforgeable under adaptive chosen message attacks in the random oracle model [GPS06, Theorem 7].

DL-SEA is an *ad-hoc* strengthening of the usual discrete logarithm assumption, which formalizes the notion that it should be hard to recover a discrete logarithm, knowing that it is smaller than some known bound:

[3] This is the case if and only if $\gcd(g - 1, n) = \gcd(g + 1, n) = 1$, which happens with high probability.

Definition 1 (DL-SEA, [GPS06]). *For every polynomial Q and every PPT Turing machine $\mathcal{A}$ running on a random tape ω_M, for sufficiently large λ,*

$$\Pr_{\omega_p,\omega_M}\left[\mathcal{A}(n,g,\mathsf{sk},g^x)=x \mid (n,g,\mathsf{sk}) \leftarrow \mathsf{Setup}(\omega_p,\lambda) \wedge x \in [\mathsf{sk}]\}\right] < \frac{1}{Q(\lambda)}$$

where Setup *is a randomized algorithm generating public parameters n, g from a security parameter λ using the random tape ω_p, and $\mathsf{sk} = n - \varphi(n)$.*

In summary, the choice of parameters for GPS to be secure are:

- An integer n which is hard to factor;
- Integers $B < A < n$ such that $2^\lambda B/A$ is negligible;
- A hash function h for which the random oracle model is appropriate.

For instance, at the 128 bit security level, $B \sim 2^{128}$, $A \sim 2^{80+128+128}$ and $n \sim 2^{3072}$, with SHA-3 as h.

2.4 RSA moduli with a prescribed pattern

To preserve compatibility with the notations of [Joy08] we will temporarily rename the modulus N in this section. We will then revert back to the notation n introduced previously. An RSA modulus generator is a PPT algorithm that outputs p, q such that $N = pq$ is an RSA modulus. There exist several algorithms that output N with additional properties; one such family of algorithms gives *prescribed patterns*: a bitstring (the "pattern") is given as input to the generation algorithm, and will be found in N. We denote $n = \|N\|$.

Different methods are known to achieve this, depending on the pattern length being considered [Len98, LdW05a, Joy08, LdW05b]. To the best of our knowledge, no polynomial-time algorithm capable of imposing a pattern of more than $2n/3$ bits while respecting the constraint $\|q\| \cong \|p\|$ has been described. The leading motivations for such algorithms originates from the desire to compress RSA keys, so that they can be stored on less bits and the from the intention to speed-up modular reduction using "computation-friendly" moduli.

Let $n > n_0$ be integers, $\kappa \lesssim 2n/3$, a predetermined portion N_H of length κ. The following algorithm [Joy08] outputs a pair (p, q) such that $N = pq = N_H \| N_L$ along with N_L, with p of size $n - n_0$ and q of size n_0.

1. Sample p_0 uniformly of length $n - n_0$, and let
$$q_0 = \left\lfloor N_H \frac{2^{n-\kappa}}{p_0} \right\rfloor.$$
2. Define recursively the triples (d_i, u_i, v_i) as:
$$\begin{aligned}(d_0, u_0, v_0) &= (p_0, 0, 1)\\ (d_{-1}, u_{-1}, v_{-1}) &= (q_0, 1, 0)\\ (d_i, u_i, v_i) &= \left(d_{i-2} \bmod d_{i-1}, u_{i-2} - \left\lfloor \frac{d_{i-2}}{d_{i-1}} \right\rfloor u_{i-1}, v_{i-2} - \left\lfloor \frac{d_{i-2}}{d_{i-1}} \right\rfloor v_{i-1}\right)\end{aligned}$$

3. Define recursively the triples (x_i, y_i, z_i) as:

$$(x_0, y_0, z_0) = \left(0, 0, \left(N_H 2^{n-\kappa} \bmod p_0\right) + 2^{n-\kappa-1}\right)$$
$$(x_i, y_i, z_i) = \left(x_{i-1} + \left\lfloor \frac{z_{i-1}}{d_i} \right\rfloor u_i, y_{i-1} + \left\lfloor \frac{z_{i-1}}{d_i} \right\rfloor v_i, z_{i-1} \bmod d_i\right)$$

   such that $|z_i - x_i y_i| < 2^{n-\kappa-1}$. (This value decreases and then increases, so once the condition is reached we can break out of the loop.)
4. Sieve the pairs (x_i, y_i) until both

$$p = p_0 + x_i$$
$$q = q_0 + y_i$$

   are prime. If no such pair is prime, start over from Step 1.
5. Output $N_L \leftarrow N \bmod 2^{n-\kappa}$ and p, q.

Note that the generation process can be repeated until (p, q) satisfies any desired property (e.g., being safe primes).

3 Assembling the puzzle to get an attestation

We decompose our construction into four steps.

Section 3.1	$\xrightarrow{\text{provable}}$	n is the product of exactly two primes p, q
Section 3.2	$\xrightarrow{\text{provable}}$	p, q are of about the same size
Section 3.3	$\xrightarrow{\text{heuristic}}$	p, q cannot be controlled
Section 3.4	$\xrightarrow{\text{trivial}}$	even if the adversary controls p, q they face a slow down

Fig. 1. The general outline of the construction proposed in this paper.

3.1 Checking that n has exactly two prime factors

Our first building block is the GRSB protocol that we run, unmodified, between $\mathcal{P}$ and $\mathcal{V}$.

Note that GRSB does not suffice, in itself, to guarantee n is cryptographically useful. For instance, $n = 3p$ with $p > 3$ will pass this phase but is clearly a bad RSA modulus.

3.2 Checking for factor sizes

The second building block checks that the two factors of n have the appropriate size. We achieve this by requesting that $\mathcal{P}$ provides a valid GPS signature of some agreed-upon message (e.g. n) to $\mathcal{V}$.

The key observation here is that GPS signatures have a size that depends on the factors of n. More precisely, if $\sigma = (x, c, y)$ is the GPS signature and we are working modulo n, the size of y essentially reveals the size of n's factors (up to a relatively small constant).

The following lemma makes this statement more precise and more general.

Lemma 1. *Let p, q be two positive integers. Let $u = \|pq\|$, $v = \|p + q\|$, and $w = \|p - q\|$. Let $\Delta > 0$, if $v \geq \log_2(2^{2\Delta-1} + 2^u)$, then $w \leq \Delta$.*

Proof. Without loss of generality, assume $p \leq q$ and let $\delta = q - p$. Then

$$\begin{aligned} w = \|\delta\| = \frac{1}{2}\|\delta^2\| &= \frac{1}{2}\|(p+q)^2 - 4pq\| \\ &\leq \frac{1}{2}\left\|2^{v+1} - 2^{u+1}\right\| = \frac{1}{2}\left[1 + \log_2\left|2^v - 2^u\right|\right] \\ &\leq \frac{1}{2}\left[1 + 2\Delta - 1\right] = \Delta. \end{aligned}$$

□

A valid GPS signature comprises $y = r + c \cdot \mathsf{sk}$ which is of size $\max(\|A\|, \|B\| + \|\mathsf{sk}\|)$. With A, B being a public parameters and sk being $p + q - 1$ we see that Lemma 1 can be used to set a threshold so that with typical GPS parameters, pq is large enough and at most a discrepancy between p and q of about 200 bits is possible.

This is unfortunately not enough: even if we know that $n = pq$ with p and q of similar size, partial Pohlig–Hellman factorization [vOW96] can exploit the smoothness of $p - 1$ or $q - 1$ to factor n. Thus we need an additional building block fill that gap.

3.3 Checking that n is a cryptographically useful modulus

What follows is only *conjectured* to be secure. The intuition is to restrict $\mathcal{P}$ to use only certain moduli, obtained through a verifiable procedure that (hopefully!) makes it hard to obtain smooth $p - 1$ and/or $q - 1$ or otherwise purposely weaken n. Indeed, from a practical standpoint [vOW96] fails whenever $(p - 1)/2$ and $(q - 1)/2$ have each a large factor: $(p - 1)/2$ and $(q - 1)/2$ being primes is ideal but not strictly necessary to resist this attack. This does not imply that methods other than [vOW96] would fail to factor n but we know of no such strategies and encourage the community to further scrutinize our proposal.

For preserving compatibility with Joye's notations we switch again to the notation N for the modulus.

We now want to ensure that the factors of N (which we know to be exactly two primes of comparable sizes), are not easy to factor. A complete but inefficient

solution consists in plugging-in the CM sub-protocol for safe-primality testing [CM99]. Doing so has a sizable cost, so we take an alternative, cheaper route.

To illustrate the difficulty, consider the following procedure which generates a couple (p, q) of primes whose product features a prescribed bit pattern N_H:

1. Form a string $N' := N_H|\rho$ where ρ is a random bitstring.
2. Generate a prime p and compute $q \leftarrow \lfloor x/p \rfloor$
3. Increment q until the result is prime.

At the end of this procedure, p and q are prime and their product $N = pq$ features N_H in its MSBs. Informally, because q is constrained we expect q to be hard to control, and therefore it would be hard to ensure that $q - 1$ is abnormally smooth using this procedure. A malicious generator could however manipulate p freely, so this approach is unsatisfactory. The above algorithm stops to work as N_H grows beyond $n/2$ bits while keeping the sizes of p and q balanced.

Joye's protocol described in Section 2.4 [Joy08, 4.1, 4.2] enables us to fix $\frac{2}{3}$ of N's bits to a prescribed pattern, with p and q being generated *simultaneously*. We *conjecture* that this causes $p - 1$ and $q - 1$ to "essentially" behave like large random numbers, which are likely to have a large prime factor as expected by the asymptotic distribution of factors in random integers[4]. This assumption is made explicit below.

Note that from $\mathcal{V}$'s viewpoint, checking that N features the prescribed pattern is cost-less and that only the LSBs of N need to be transmitted to $\mathcal{V}$. For the above to work it is crucial to enforce that N_H is beyond the control of $\mathcal{P}$. For instance set N_H as equal to the digits of $\pi = 3.14159\ldots$ or $e = 2.71828\ldots$.

One interesting question is whether RSA moduli indistinguishable from those generated by Joye's algorithm can be purposely crafted to be more vulnerable than moduli formed by multiplying two random equal-size primes. The case is clearly different with moduli generated by other bit prescription methods because, unlike Joye's algorithm, alternative methods pick p at random first and only then generate q as a function of p and N_H and. As we have already explained, in such a scenario p can be chosen to be weak (e.g. $p - 1$ can be purposely selected to be smooth). Therefore the conjecture upon which the heuristic part of our construction relies is:

> **Conjecture:** *Given a challenge N_H of size $\frac{2n}{3}$, it is hard to generate a vulnerable N featuring the MSB pattern N_H such that N has exactly two prime factors of roughly equal size.*

The above calls for a precise definition of the term "vulnerable". Evidently, nothing can be doing against a dishonest $\mathcal{P}$ could publish his random tape, use a very short (i.e. exhaustible) random tape or run the proof with some public test values for p, q. To capture simply the requirement we construe the term "vulnerable" as follows:

[4] In particular, the Golomb–Dickman constant $\lambda \approx 0.624$ asymptotically governs the relative size of the largest prime factor of an integer [KP76, Dic30, Gol64]

A modulus of size n is vulnerable if its factoring is asymptotically easier than the factorization of a modulus generated by picking two random $\frac{n}{2}$-bit primes.

3.4 Optional security measures

In this section we describe three optional security measures. All are heuristic and incur additional computational cost for $\mathcal{P}$ and/or $\mathcal{V}$.

The rationale behind these measures is that if a cryptanalysis whose work factor is ω_1 is found, the proposed countermeasures will multiply ω_1 by a constant factor ω_2 that may put $\omega_1 \times \omega_2$ out of practical reach.

1. The first countermeasure consists in using an n larger than required. This reinforces the argument of Section 3.3, as larger smooth numbers are scarcer. We recommend to use moduli whose size is larger by 62% than normal for any desired security level. This is meant to account for the fact that the largest prime factor of a random ℓ-bit number is expected to be roughly 0.62ℓ-bits long [KP76].
2. The second countermeasure will put a burden on generation but leave verification unchanged: we require from n an additional *short* redundancy, e.g. $\mathrm{SHA}(n) \bmod 2^{24} = 0$. It is reasonable to assume that there is no efficient algorithm allowing to achieve this property along with a prescribed pattern. Thus, to obtain n the generation procedure should be run on average 2^{24} times. This places no extra burden on $\mathcal{V}$ and because moduli are usually generated once for all in a device's lifetime, slowing the modulus generation process down on one occasion may pay back in case of an attack.
3. The third countermeasure is applicable when $\mathcal{V}$ witnesses the generation of n. It consists in performing a cut-&-choose protocol to ascertain the freshness and the conformity of the generated moduli:
   - $\mathcal{P}$ and $\mathcal{V}$ generate a common secret key u using e.g. Diffie-Hellman.
   - $\mathcal{P}$ picks t random w_is and computes $v_i = \mathrm{hash}(w_i, u)$ for $0 \leq i \leq t-1$.
   - $\mathcal{P}$ uses v_i as a random tape to generate the modulus n_i
   - $\mathcal{V}$ picks a random index j and sends it to $\mathcal{P}$
   - $\mathcal{P}$ reveals to $\mathcal{V}$:
   $$w_0, w_1, \ldots, w_{j-1}, w_{j+1}, \ldots, w_{t-1}$$
   - $\mathcal{V}$ re-generates the corresponding $t-1$ moduli
   $$n_0, n_1, \ldots, n_{j-1}, n_{j+1}, \ldots, n_{t-1}$$
   and checks that all the above $t-1$ moduli were properly generated.
   - $\mathcal{V}$ tests using the first two phases of the protocol proposed in this paper that n_j has two large factors of equal size and if so, he signs n_j to certify it.

   We see that this protocol reduces the cheating odds to $\frac{1}{t}$ where by "cheating" we mean generating factors so that n_j is easy to factor. It has the advantage of ascertaining, in addition, the freshness of n_j. Indeed, even if $\mathcal{P}$ would use

a fixed random tape then the randomness injected by the $\mathcal{V}$ into the protocol would ascertain that no entities other than $\mathcal{P}$ and $\mathcal{V}$ would be able to factor n_j.

4 Efficiency

The global computational cost of our protocol is dominated by Section 3.3, as other phases essentially have the cost of a few full-sized modular multiplications. While the complexity of Joye's algorithm does not follow a simple expression, we can consider that around $1/\eta^2$ primality tests are performed, where η is the probability that an integer of size n is prime — by the prime number theorem η is of the order of $1/\ln(2^n) = 1/n\ln 2$. Thus the overall algorithmic complexity is essentially $O(n^2)$, where n is the modulus' size.

If the optional measures of Section 3.4 are implemented, the impact on total complexity is a slowdown by a factor of about $4.25 \simeq (1+\lambda)^3$ for the GPS phase that dominates the slowdown in the other phases.

Note that if n is used to sign a message m, the protocol can made even more efficient. Instead of just signing n during the GPS phase, use GPS to sign $n|m$. This achieves both the goal of attesting the sizes of p, q and signing m. In such a case, the attestation of n comes at the minimal price of phases 3.1 and 3.3 only.

5 Conclusion

In this paper we introduced a cheap non-interactive process for proving that $n = pq$ is a product of two equal-size primes. The process is completed with a heuristic trick conjectured to be sufficient to ascertain that n is cryptographically useful.

This raises a number of interesting research questions. For instance, speeding-up or simplifying [CM99] by hybridizing it with the techniques. The same seems also applicable to [GMR98] although we haven't investigated this avenue.

More importantly, we invite the community to find attacks on the heuristic phase of our protocol. A successful attack consists in exhibiting a modulus n having exactly two equal-size prime factors and featuring $\frac{2}{3}\|n\|$ prescribed LSBs[5]. This n must be easier to factor than an $n = pq$ where p, q are randomly generated primes. Running the proposed protocol as is, but with a known[6] random tape is tantamount to just revealing the key and is hence not considered as usefully "attacking" the proposed construction.

Acknowledgements

The authors are grateful to Arjen Lenstra for his pertinent remarks on an earlier version of this article. During the preparation of this paper Rémi became the

[5] e.g. the binary digits of $\pi = 3.14159265\ldots$

[6] Partially or entirely.

happy father of the newly born baby Esther. To mark the event, the authors decided to kindly ask the community to refer to moduli described in this paper as "Esther moduli".

References

AP18. Benedikt Auerbach and Bertram Poettering. Hashing solutions instead of generating problems: On the interactive certification of RSA moduli. In Michel Abdalla and Ricardo Dahab, editors, *Public-Key Cryptography - PKC 2018 - 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, March 25-29, 2018, Proceedings, Part II*, volume 10770 of *Lecture Notes in Computer Science*, pages 403–430. Springer, 2018.

CM99. Jan Camenisch and Markus Michels. Proving in zero-knowledge that a number is the product of two safe primes. In Jacques Stern, editor, *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 of *Lecture Notes in Computer Science*, pages 107–122. Springer, 1999.

CS97. Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups (extended abstract). In Burton S. Kaliski Jr., editor, *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, volume 1294 of *Lecture Notes in Computer Science*, pages 410–424. Springer, 1997.

Dic30. Karl Dickman. On the frequency of numbers containing prime factors of a certain relative magnitude. *ArMAF*, 22(10):A–10, 1930.

FS86. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.

Gir91. Marc Girault. Self-certified public keys. In Donald W. Davies, editor, *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, volume 547 of *Lecture Notes in Computer Science*, pages 490–497. Springer, 1991.

GMR98. Rosario Gennaro, Daniele Micciancio, and Tal Rabin. An efficient non-interactive statistical zero-knowledge proof system for quasi-safe prime products. In Li Gong and Michael K. Reiter, editors, *CCS '98, Proceedings of the 5th ACM Conference on Computer and Communications Security, San Francisco, CA, USA, November 3-5, 1998*, pages 67–72. ACM, 1998.

Gol64. Solomon W. Golomb. Random permutations. *Bull. Amer. Math. Soc*, 70:747, 1964.

GPS06. Marc Girault, Guillaume Poupard, and Jacques Stern. On the fly authentication and signature schemes based on groups of unknown order. *J. Cryptol.*, 19(4):463–487, 2006.

GRSB19. Sharon Goldberg, Leonid Reyzin, Omar Sagga, and Foteini Baldimtsi. Efficient noninteractive certification of RSA moduli and beyond. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology - ASIACRYPT*

2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part III, volume 11923 of *Lecture Notes in Computer Science*, pages 700–727. Springer, 2019.

Joy08. Marc Joye. RSA moduli with a predetermined portion: Techniques and applications. In Liqun Chen, Yi Mu, and Willy Susilo, editors, *Information Security Practice and Experience, 4th International Conference, ISPEC 2008, Sydney, Australia, April 21-23, 2008, Proceedings*, volume 4991 of *Lecture Notes in Computer Science*, pages 116–130. Springer, 2008.

KP76. Donald E. Knuth and Luis Trabb Pardo. Analysis of a simple factorization algorithm. *Theor. Comput. Sci.*, 3(3):321–348, 1976.

LdW05a. Arjen Lenstra and Benne de Weger. On the possibility of constructing meaningful hash collisions for public keys. In Colin Boyd and Juan Manuel González Nieto, editors, *Information Security and Privacy*, pages 267–279, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

LdW05b. Arjen K. Lenstra and Benjamin M. M. de Weger. Twin RSA. In Ed Dawson and Serge Vaudenay, editors, *Progress in Cryptology - Mycrypt 2005, First International Conference on Cryptology in Malaysia, Kuala Lumpur, Malaysia, September 28-30, 2005, Proceedings*, volume 3715 of *Lecture Notes in Computer Science*, pages 222–228. Springer, 2005.

Len98. Arjen K. Lenstra. Generating RSA moduli with a predetermined portion. In Kazuo Ohta and Dingyi Pei, editors, *Advances in Cryptology - ASIACRYPT '98, International Conference on the Theory and Applications of Cryptology and Information Security, Beijing, China, October 18-22, 1998, Proceedings*, volume 1514 of *Lecture Notes in Computer Science*, pages 1–10. Springer, 1998.

Pai99. Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 1999.

PS98. Guillaume Poupard and Jacques Stern. Security analysis of a practical "on the fly" authentication and signature generation. In Kaisa Nyberg, editor, *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, volume 1403 of *Lecture Notes in Computer Science*, pages 422–436. Springer, 1998.

PS99. Guillaume Poupard and Jacques Stern. On the fly signatures based on factoring. In Juzar Motiwalla and Gene Tsudik, editors, *CCS '99, Proceedings of the 6th ACM Conference on Computer and Communications Security, Singapore, November 1-4, 1999*, pages 37–45. ACM, 1999.

Rab79. Michael O. Rabin. Digitalized signatures and public key functions as intractable as intractable as factorization. *MIT Laboratory of Computer Sciences*, 21, 1979.

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

vOW96. Paul C. van Oorschot and Michael J. Wiener. On diffie-hellman key agreement with short exponents. In Ueli M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application*

of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding, volume 1070 of *Lecture Notes in Computer Science*, pages 332–343. Springer, 1996.