Factoring Primes to Factor Moduli: Backdooring and Distributed Generation of Semiprimes*

Giuseppe Vitto

SnT, University of Luxembourg, Esch-sur-Alzette, Luxembourg
name.surname@uni.lu

December 22, 2021

Abstract. We describe a technique to backdoor a prime factor of a composite odd integer N, so that an attacker knowing a possibly secret factor base $\mathcal{B}$, can efficiently retrieve it from N. Such method builds upon Complex Multiplication theory for elliptic curves, by generating primes p associated to $\mathcal{B}$-smooth order elliptic curves over $\mathbb{F}_p$. When such primes p divide an integer N, the latter can be efficiently factored using a generalization of Lenstra's Factorization Method over rings bigger than $\mathbb{Z}_N$, and with no knowledge other than N and $\mathcal{B}$.
We then formalize semiprimality certificates that, based on a result by Goldwasser and Kilian, allow to prove semiprimality of an integer with no need to reveal any of its factors. We show how our prime generation procedure can be used to efficiently produce semiprimality certificates, ultimately allowing us to sketch a multi-party distributed protocol to generate semiprimes with unknown factorisation, particularly relevant in the setting of distributed RSA modulus generation.
We provide and discuss implementations of all proposed protocols and we address security of semiprimality certificates by showing that semiprimes generated within our methods result at least as secure as random semiprimes of same size.

Keywords: elliptic curves · complex-multiplication · backdoor · semiprime · certificate · MPC · RSA · ECM

1 Introduction

In this paper, we will detail a technique to backdoor a prime factor of an odd composite integer N so that third parties knowing some (secret) auxiliary information, i.e., a factor base, can efficiently retrieve it from N.

More precisely, we will use the *Complex Multiplication* theory to build primes p and elliptic curves over $\mathbb{F}_p$, such that the orders of the latter are $\mathcal{B}$-*smooth* (that is, factor as products of elements in $\mathcal{B}$), and can be explicitly represented as a function of the field characteristic, i.e. $\#E(\mathbb{F}_p) = p+1\pm a$ where $4p = a^2+|D|b^2$ with D a negative *discriminant.* over a ring containing

In the following Sections, we will describe a generation procedure for such primes p so that, with no knowledge of any of the factors of N, we will be able to construct a curve equation $E(\mathbb{Z}_N)$ along with multiple random points P on it, so that the order of $E(\mathbb{F}_p)$

* The work of Giuseppe Vitto was supported by the Luxembourg National Research Fund (FNR) project FinCrypt (C17/IS/11684537).

(the modulo p component of $E(\mathbb{Z}_N)$) is $\mathcal{B}$-smooth for a certain factor base $\mathcal{B}$. This will ultimately allow us to efficiently factor N by computing in projective coordinates some multiple $[\prod_{p_i \in \mathcal{B}} p_i] \cdot P$ of the point P, similarly as done in Lenstra's Elliptic-Curve factorization Method (ECM) [30]. While, in ECM, random curve equations $E(\mathbb{Z}_N)$ and points $P \in E(\mathbb{Z}_N)$ are generated until a curve of smooth order is found, in our construction, we will be able to deterministically construct from N a $\mathcal{B}$-smooth order elliptic curve in short Weierstrass form defined over a ring containing $\mathbb{Z}_N$, together with multiple points on it.

Although different backdooring techniques exist in literature within our attack model[1], our methods show that large semiprimes N, employed by the RSA encryption scheme [33], can be maliciously generated to be still vulnerable to a generalization of Lenstra's ECM, an attack that is usually not considered of interest for cryptographically sized N due to its negligible success probability. Furthermore, the insights provided by our prime generation procedure will ultimately allow us to sketch a distributed protocol to generate semiprimes.

Recent protocols for the multi-party generation of semiprimes [10, 11, 34] build on the seminal work of Boneh and Franklin [8], by extending it to different security assumptions and adding several algorithmic optimizations. Informally, in these protocols, parties jointly generate some (random) candidate primes p, q, securely multiply them as $N = p \cdot q$ and run a distributed statistical semiprimality test until they are confident enough that N is a semiprime.

In contrast to these, our sketched multi-party protocol outputs integers which are guaranteed to be a product of two unknown primes (and thus there is no need to run a semiprimality test such as the one detailed in [8]), thanks to *semiprimality certificates* that can be publicly verified.

Semiprimality certificates are based on a generalization of a theorem by Goldwasser and Kilian [25], which provides a sufficient condition for an integer N to have at most m distinct prime factors. Such a condition requires the existence of an elliptic curve over $\mathbb{Z}_N$ so that some points on it have particular small prime order. It follows that when we require an integer N to be the product of two distinct primes, as happens for RSA moduli, such result can be used to ensure semiprimality for N in case we find (or we construct) a semiprimality certificate for it, with the latter consisting in an elliptic curve over $\mathbb{Z}_N$ and some points on it satisfying the theorem assumptions. Remarkably, such certificates prove semiprimality of N with no need to publish nor know any of its factors.

Our semiprime generation protocol uses our prime backdooring procedure to construct elliptic curves with (partially random) orders that satisfy the assumptions of the Goldwasser-Kilian generalized criterion. We then retrieve N from this information, and we test if all conditions of the theorem match: when this happens, N is a semiprime *by construction* and its semiprimality can be publicly verified from the certificate.

We can then apply standard MPC arithmetic to perform all the computations involved (mainly multiplications and additions, but also scalar-point multiplications that we address with an ad-hoc technique) to make this protocol distributed, and we show its correctness and feasibility by implementing it using state-of-the-art MPC libraries.

[1] A standard one consists in fixing half of the bits of p to a certain secret value c and then use Coppersmith's method [15] to efficiently find a small root of the polynomial $f(x) = 2^{\log_2(p)/2}x + c$ modulo a divisor of $N = p \cdot q$.

Lastly, we analyze if and how knowledge of a semiprimality certificate for N generated within our framework may reduce its bit-security with respect to factorization. We investigate some possible attacks based on generic discrete-logarithm finding algorithms in groups of unknown order, and attack variants that require just a few operations to compute a factor of N if any multiple of the order of the curve provided in the certificate is known. In all cases, semiprimality certificates generated according to our protocol are easier to attack using generic factorization algorithms, and thus we don't expect a decrease in terms of bit-security unless other more specific attacks are found.

We were, however, able to factor a 3599-bits semiprime N, generated by Don Reble [21] in 2005 and which remained unfactored until now. Such semiprime comes with a semiprimality certificate that, contrary to ours, includes an extra divisor of the full curve order that makes it vulnerable to one of our attack variants that factors N in just a few operations.

Related Works In a later stage of the (independent) development of the results reported in this paper, we found out that Aikawa *et al.* in [1] use similar observations to factor integers N where one of its primes is of the form $4p = 1 + |D|b^2$. These primes, indeed, correspond to anomalous elliptic curves modulo p for which a multiple of the order, i.e., N, is known, thus allowing an attacker to factor N with just 1 scalar-point multiplication. Aikawa *et al.* work extends the applicability of the methods described by Shirase [37], whose manuscript, in turn, is based on Cheng's [12, 13] ideas, to a bigger class of *discriminants* D.

In addition to this, Aikawa *et al.* generalize their factoring method for anomalous curves to allow p be of the form $4p = a^2 + |D|b^2$, but requiring associated curves to have smooth order rather than a multiple of N. In contrast to our results, however, they do not provide an actual method to construct such vulnerable elliptic curves efficiently, and the probability to randomly hit any of them exponentially decreases with the size of p. Furthermore, their method works in multiple quotients of the ring $\mathbb{Z}_N[x]$ to carry out arithmetic operations on the constructed curve, while we use a more efficient approach based on XZ-arithmetic.

On a more practical side, Sedlacek *et al.* in [36] generated millions of RSA keys using different software libraries and hardware and tested them against Cheng's and Shirase's attacks, looking for prime factors p of the form $4p = 1 + |D|b^2$.

As regards semiprimality certificates, to our knowledge, Don Reble, with his short online post [21], was the first one to publicly propose the idea of proving semiprimality of an odd integer N, that he calls *interesting semiprimes*, using the Goldwasser-Kilian Theorem. Except for another similar certificate generated by Broadhurst [19] soon after, we did not find any formal treatment of these concepts.

1.1 Outline

In Section 2 we provide the necessary theoretical background to characterize orders of some elliptic curves in terms of the field characteristic on which are defined, while in Section 3 we provide the first sketch of our prime generation procedure. In Section 4 we address some technicalities related to the practical implementation of our idea, which we formalize in Section 5 and Section 6, where we detail how it is possible to retrieve a factor of a backdoored integer. In Section 7 we discuss implementations of both our prime generation procedure and factorization attack, and we detail a full example on a 1024-bits modulus. In Section 8 we formalize the concept of a semiprimality certificate for an integer N by generalizing

the Goldwasser-Kilian Theorem, and we show how they can be efficiently computed by properly generating the prime factors of N. In Section 9 we propose a distributed protocol for generating certifiable semiprimes with unknown factorization, and we discuss how we overcome some practical issues to implement it in practice. Lastly, in Section 10, we address the security of semiprimality certificates generated according to our protocol, and we describe how we factored a public 1084-digits semiprime.

2 Preliminaries

2.1 Curves of Prescribed Order

Our construction relies on the theory of Complex Multiplication (CM) to build elliptic curves of a prescribed order. An elliptic curve E is said to have complex multiplication, if its endomorphism ring $End(E)$ is strictly larger than $\mathbb{Z}$[2]: this is always the case for elliptic curves defined over finite fields, where the endomorphism ring is isomorphic to an order in either a quaternion algebra (in this case, the curve is said to be *supersingular*) or a quadratic imaginary field (in this case, instead, the curve is said to be *ordinary*) [38, V - Theorem 3.1].

A field K is said *quadratic field* if $[K : \mathbb{Q}] = 2$, and discriminants of quadratic fields are said *fundamental*. An integer $d \in \mathbb{Z}$ is a fundamental discriminant if it satisfies either $d \equiv 1 \pmod 4$ and d is square-free, or $d = 4D$ with D square-free and $D \equiv 2, 3 \pmod 4$.

Of interest for this paper are ordinary elliptic curves whose endomorphism rings are isomorphic to orders in a quadratic imaginary field.

Definition 1 (Order). *For a quadratic field K, let $\mathcal{O}_K$ denote the ring of integers of K. A subring $\mathcal{O} \subset \mathcal{O}_K$ is said to be an order, if it is a free $\mathbb{Z}$-module of rank 2 containing an integral basis of K.*

In particular, given a quadratic field K of fundamental discriminant d, we will denote it as $K = \mathbb{Q}(\sqrt{d})$, and an integral basis for it is given by $(1, \omega)$ with $\omega = \frac{d+\sqrt{d}}{2}$, thus $\mathcal{O}_K = \mathbb{Z}[\omega]$.

Every order of a quadratic field is associated to a *discriminant*, characterized by its residuosity modulo 4.

Definition 2 (Discriminant). *A $D \in \mathbb{Z}$ is said to be a* discriminant *if D is not a perfect square and $D \equiv 0, 1 \pmod 4$.*

Fundamental discriminants of quadratic fields and discriminants of orders are related by the following result.

Proposition 1 ([14, Proposition 5.1.3]). *If K is a quadratic field of fundamental discriminant d, then every order $\mathcal{O}$ of K has discriminant $D = df^2$, where $f \in \mathbb{N}$ is the* conductor *of $\mathcal{O}$. Conversely, if D is a discriminant, then D can be written uniquely as $D = df^2$, with d a a fundamental discriminant, and there exists a unique order $\mathcal{O}$ of $\mathbb{Q}(\sqrt{d})$ of discriminant D.*

[2] The multiplication by $[n] : E \to E$ map which sends $P \mapsto nP$, is an endomorphism of E for any $n \in \mathbb{Z}$.

The following result from CM-theory, also known as *CM Algorithm*, relates specific short Weierstrass curve equations over $\mathbb{F}_p$ to their cardinality, thus providing a practical method for computing elliptic curves over finite fields of a given order.

Theorem 1 ([9, Theorem 3.6]). *Let D be a square-free negative discriminant not equal to $-3, -4$, and let p be an odd prime so that $4p = a^2 + |D|b^2$ for certain $a, b \in \mathbb{Z}$. Then, the elliptic curve*

$$E(\mathbb{F}_p) \; : \; y^2 = x^3 - 3kc^2x + 2kc^3$$

where

- *j is a root of the Hilbert Class Polynomial $H_D(x) \in \mathbb{Z}[x]$ modulo p,*
- *$k = \frac{j}{j-1728}$,*
- *c is a random non-zero element in $\mathbb{F}_p$*

has either $p + 1 + a$ or $p + 1 - a$ points, depending on the residuosity of c modulo p, and j-invariant equal to j.

Here the Hilbert Class Polynomial $H_D(x)$ is a polynomial with roots exactly the j-invariants of elliptic curves over $\mathbb{C}$ whose endomorphism ring equals an order with discriminant D in a quadratic imaginary field. It can be proven that $H_D(x)$ is irreducible, is defined over $\mathbb{Z}[x]$, and there exist efficient algorithms to compute it: we refer to [9, Section 3.3] for more technical details on this.

We recall that given an elliptic curve $E : y^2 = x^3 + ax + b$, a (quadratic) twist of E over $\mathbb{F}_p$ is given by $\tilde{E} : y^2 = x^3 + c^2ax + c^3b$, where c is a quadratic non-residue modulo p. If $E(\mathbb{F}_p)$ has trace of Frobenius t, then $\tilde{E}(\mathbb{F}_p)$ has trace $-t$ for any chosen quadratic non-residue c, namely $\#E(\mathbb{F}_p) = p + 1 - t$ and $\#\tilde{E}(\mathbb{F}_p) = p + 1 + t$. If instead, c is a square modulo p, then E is isomorphic to $\tilde{E}$ through the change of variable $(x', y') = (x/c, \sqrt{c}y/c^2)$ and hence $\#E(\mathbb{F}_p) = \#\tilde{E}(\mathbb{F}_p)$. It follows that in the statement of Theorem 1, the value c chosen determines one of these two different curve twists.

It is possible to characterise the cases $D = -3, -4$ as well, which correspond to curves with j-invariants $j = 0, 1728$, respectively.

Theorem 2 ([9, Theorem 3.6]). *Let $D \in \{-3, -4\}$ and let p be an odd prime so that $4p = a^2 + |D|b^2$ for some $a, b \in \mathbb{Z}$. Then, the elliptic curve*

$$E(\mathbb{F}_p) : \begin{cases} y^2 = x^3 + c^3 & \text{if } D = -3 \\ y^2 = x^3 + c^2x & \text{if } D = -4 \end{cases}$$

with c a random non-zero element in $\mathbb{F}_p$, has either $p + 1 + a$ or $p + 1 - a$ points, depending on the residuosity of c modulo p, and j-invariant equal to 0 if $D = -3$, or 1728 if $D = -4$.

In the case $D = -4$, we have a more general result that completely characterize the number of points of the curve $E(\mathbb{F}_p) : y^2 = x^3 - kx$, regardless of $-k$ being a square, as required instead by Theorem 2.

We recall that, by Fermat's Theorem on *Sums of Two Squares* [22, Lemma 18.4], an odd prime p can be written as $p = a^2 + b^2$ if and only if $p \equiv 1 \pmod 4$. It follows that any decomposition of a prime p into a sum of two squares $a^2 + b^2$, is in bijection with a decomposition of $4p$ as $(2a)^2 + |D| \cdot b^2$ with $D = -4$: the following result, in fact, generalizes Theorem 2 for the case $D = -4$.

Algorithm 1 Modified Cornacchia Algorithm ([14, Algorithm 1.5.3])

Input: an odd prime p, a negative discriminant D with $|D| < 4p$.
Output: if exists, an integer solution to $x^2 + |D|y^2 = 4p$, otherwise `None`.

1: Compute $k = \left(\frac{D}{p}\right)$. If $k = -1$, return `None`.
2: Compute, e.g. using Shank's algorithm, an x_0 so that $x_0^2 \equiv D \pmod{p}$ and $0 \leq x_0 < p$. If $x_0 \not\equiv D \pmod{2}$, set $x_0 = p - x_0$. Set $a = 2p$, $b = x_0$, $l = \lfloor 2\sqrt{p} \rfloor$.
3: If $b > l$, set $r = a \bmod b$, $a = b$, $b = r$. Go to Step 3.
4: If $|D| \nmid (4p - b^2)$ or $c = (4p - b^2)/|D|$ is not a perfect square, return `None`. Otherwise, return $(x, y) = (b, \sqrt{c})$.

Theorem 3 ([42, Theorem 4.23]). *For an odd prime p, let $k \not\equiv 0 \pmod{p}$ and consider the elliptic curve $E(\mathbb{F}_p) : y^2 = x^3 - kx$. Then*

1. *If $p \equiv 3 \pmod{4}$, E is supersingular and $\#E(\mathbb{F}_p) = p + 1$.*
2. *If $p \equiv 1 \pmod{4}$, let $p = a^2 + b^2$ where a, b are integers with b even and $a + b \equiv 1 \pmod{4}$. Then*

$$\#E(\mathbb{F}_p) = \begin{cases} p + 1 - 2a & \text{if } k \text{ is a fourth power modulo } p \\ p + 1 + 2a & \text{if } k \text{ is a square but not a fourth power modulo } p \\ p + 1 \pm 2b & \text{if } k \text{ is not a square modulo } p \end{cases}$$

2.2 Cornacchia's Algorithm

The above characterizations for the orders of elliptic curves over $\mathbb{F}_p$ require the knowledge of a specific decomposition for $4p$, namely $4p = a^2 + |D|b^2$, with D a negative discriminant and $a, b \in \mathbb{Z}$.

In 1908, Giuseppe Cornacchia proposed an algorithm to solve the Diophantine equation $x^2 + dy^2 = p$ with p prime and $0 < d < p$: a proof of correctness of his algorithm can be found, for example, in [5]. It is possible to slightly modify the original Cornacchia's algorithm to find a solution to our case of interest as well, namely finding solutions to the equation $4p = x^2 + |D|y^2$ for a negative D so that $D \equiv 0, 1 \pmod{4}$. For completeness, we report in Algorithm 1 such modified version of his algorithm, taken from [14, Algorithm 1.5.3].

In the following Sections, we will often refer to a pair (a, b) satisfying $4p = a^2 + |D|b^2$ as the *Cornacchia decomposition* of p for the discriminant D.

3 The Idea

Given a square-free integer N that decomposes as a product of primes of approximately the same size as $N = p_1 \cdot \ldots \cdot p_n$, and an elliptic curve E defined over $\mathbb{Z}_N$, we clearly have that $E(\mathbb{Z}_N) = E(\mathbb{F}_{p_1}) \times \ldots \times E(\mathbb{F}_{p_n})$ as groups. We are interested in generating primes p so that whenever they divide a square-free integer N as above, we can explicitly write a curve equation E over $\mathbb{Z}_N$ with no need to know any factor of N, and so that $\#E(\mathbb{F}_p)$ is smooth or factors in a chosen factor base $\mathcal{B}$. We will do so using the results and the explicit curve equations provided by Theorem 1, Theorem 2, Theorem 3.

It will then follow that, if we are able to get a (non-trivial) point P on one of such $E(\mathbb{Z}_N)$, by computing $\left(\prod_{p_i \in \mathcal{B}} p_i^{\ell}\right) \cdot P$ for some fixed exponent l, we might reveal a factor of N, similarly as happens in Lenstra's elliptic-curve factorization method [30], with probability depending on the size of N and its prime factors. Such probability is high, if N is a cryptographically-sized semiprime, which are of particular interest for this paper: for ease of exposition, from this point on we will therefore assume N to be a semiprime.

Taking into account all the different sub-cases of the above results, we can sketch a procedure to generate such primes p as follows:

1. Set a factor base $\mathcal{B}$ and a discriminant D.
2. Generate a random integer t (i.e., a candidate for the order of $E(\mathbb{F}_p)$) which factors completely in $\mathcal{B}$.
3. Use Algorithm 1 to find a pair (a, b) so that' $4t = a^2 + |D|b^2$. If no solution exists go to step Step 2.
4. Check if $p = \frac{(a\pm 2)^2 + |D|b^2}{4}$ is prime. If yes, output p. If p is not prime and $D \neq -4$, go to Step 2.
5. If $D = -4$ check if $p = \frac{|D|a^2 + (b\pm 2)^2}{4}$ is prime. If yes, output p, otherwise go to Step 2.

We note that Step 4 is justified from the fact that, from Theorem 1, Theorem 2, we require the order of the curve $E(\mathbb{F}_p)$ to be $t = p + 1 \pm a$: thus, if $4t = a^2 + |D|b^2$, then $4p = (a \pm 2)^2 + |D|b^2$. Similarly, Step 5 addresses the simmetry induced for the case $D = -4$, where for a prime $p = a^2 + b^2$ we can express $4p$ as either $(2a)^2 + |D| \cdot b^2$ or $|D| \cdot a^2 + (2b)^2$.

Although this procedure looks quite simple, it has however some not trivial aspects that need to be addressed when implementing it in practice. In particular, the probability that a random $\mathcal{B}$-smooth integer t so that $4t$ admits a Cornacchia decomposition should not be negligible, and such decomposition should allow to generate many candidate primes. Furthermore, when we attack an integer N that has at least one of its factor generated as above, to write a curve equation to work with, we should be able to extract roots of the Hilbert Class Polynomial modulo N, and once we compute the full curve equation $E(\mathbb{Z}_N)$, we should be able to pick, for any choice of the curve twist, random points over it. All these aspects will be addressed in next Section.

4 Addressing Technicalities

4.1 Cornacchia's Decompositions for Random Curve Orders

From the *Sum of Two Squares Theorem* [22, Theorem 18.1], we know that an integer can be written as the sum of two squares if it is not divisible by any factor p_i^k, with p_i prime, $p_i \equiv 3 \pmod 4$ and k odd. For the case $D = -4$, this theorem provides an easy condition to generate random curve candidate orders t, since if $t = a^2 + b^2$ then $4t = (2a)^2 + |D|b^2$. Unfortunately, this does not generalize straightforwardly to other values of D, and elements in the factor base $\mathcal{B}$ needs to be chosen carefully if we wish to decompose with non-negligible probability a random $\mathcal{B}$-smooth t using Algorithm 1.

We will exploit the following observation to generate a factor base which will allow us to efficiently generate curve candidate orders along with their Cornacchia's decompositions.

Observation 41. *If $p_1, \ldots, p_n$ are positive integers that split in $\mathbb{Z}[\sqrt{D}]$ as $p_i = \pi_i \bar{\pi}_i$, then $p_1 \cdot \ldots \cdot p_n$ splits as $(\pi_1 \cdot \ldots \cdot \pi_n) \cdot (\bar{\pi}_1 \cdot \ldots \cdot \bar{\pi}_n) \doteq \pi \cdot \bar{\pi}$.*

Its main insight is that, when we multiply by its conjugate a given $\pi_i = a + b\sqrt{D} \in \mathbb{Z}[\sqrt{D}]$, we get $p_i = \pi_i \cdot \bar{\pi}_i = a^2 + |D|b^2 \in \mathbb{Z}$, which automatically ensures a Cornacchia decomposition for p_i as $4p_i = (2a)^2 + |D|(2b)^2$.

In fact, we can efficiently compute Cornacchia's decompositions for random $\mathcal{B}$-smooth values, without running Algorithm 1 every time. To show this, we define our factor base $\mathcal{B}$ to contain tuples $(p_i, \pi_i) \in \mathbb{Z} \times \mathbb{Z}[\sqrt{D}]$ so that $p_i = \pi_i \cdot \bar{\pi}_i$. If in Step 2 of our sketched prime generation procedure we compute in parallel the products

$$\prod_j p_{i_j} = a^2 + |D|b^2 \,, \qquad \prod_j \pi_{i_j} = a + b\sqrt{D}$$

from the two coordinates of the latter, we immediately get a solution (a, b) to $\prod_j p_{i_j} = x^2 + |D|y^2$.

In our case of interest, however, we look for a $\mathcal{B}$-smooth value t constrained to a Cornacchia decomposition for $4p$: by expanding their relations, if $p = a^2 + |D|b^2$, then $4p = (2a)^2 + |D|(2b)^2$, and from $t = p + 1 \pm 2a$ we then have $t = (a \pm 1)^2 + |D| \cdot b^2$. It follows, in turn, that if the t generated in Step 2 are of the form $t = a^2 + |D|b^2$, the expression for p in Step 4 can be simplified by checking primality of $p = (a \pm 1)^2 + |D|b^2$ (and $p = |D| \cdot a^2 + (b \pm 1)^2$ in Step 5). Since such p, t values are constrained by $t = p + 1 \pm 2a$ or $t = p + 1 \pm 2b$, for p to be prime we necessarily require t to be even.

To address this, we include in the factor base a pair $(e, v) \in \mathbb{Z} \times \mathbb{Z}[\sqrt{D}]$ where e is even and decomposes in $\mathbb{Z}[\sqrt{D}]$ as $v \cdot \bar{v}$. We can set

$$e = (D \bmod 2) + |D| \,, \qquad v = (D \bmod 2) + \sqrt{D}$$

that is, e is equal to $1 + |D|$ if D is odd or to $|D|$ otherwise.

We note that, regardless of the parity of D, when D is negative $e \equiv 0 \pmod 4$ and the minimum value for such e is 4: hence, for all negative discriminants D, by including such tuple in the factor base $\mathcal{B}$, and by requiring a random $\mathcal{B}$-smooth value $\tilde{t}$ to always have (at least) e as a factor, we can efficiently compute, as above, a Cornacchia's decomposition for the generated curve orders candidates.

In order to be able to exploit all the 4 orders characterization provided by Theorem 3, we need to write p as a sum of two squares. If t is generated as above as $t = a^2 + |D|b^2$, then $t = a^2 + (2b)^2 = a^2 + \tilde{b}^2$, and p would then be of the form $p = (a \pm 1)^2 + \tilde{b}^2$ or $p = a^2 + (\tilde{b} \pm 1)^2$. Thus, in the case $D = -4$ we proceed as above, ensuring t to be even, but once $t = a^2 + |D|b^2$ is generated, we *absorb* $\sqrt{|D|} = 2$ into the value $\tilde{b} = 2b$ and we check p accordingly.

4.2 Roots of the Hilbert Class Polynomial $H_D(x)$ modulo N

When $D \neq -3, -4$, we need to compute a root of the Hilbert Class Polynomial $H_D(x)$ modulo p, with $p|N$, in order to compute the j-invariant of a curve with desired order (Theorem 1). Since, at this point, no factor of N is known, we lift the problem to $\mathbb{Z}_N$ (using the correspondence given by the Chinese Remainder Theorem) and we compute $H_D(x)$ modulo N instead.

We have different possibilities for computing $H_D(x) \in \mathbb{Z}_N[x]$: it is well known that $H_D(x)$ can be expressed as an irreducible polynomial in $\mathbb{Z}[x]$ and we can thus compute it in $\mathbb{Z}[x]$ first, using for example the techniques outlined in [6], and then reduce modulo N its coefficients. Unfortunately, this approach quickly becomes infeasible as $|D|$ increases, since the sizes of the coefficients of $H_D(x) \in \mathbb{Z}[x]$ would grow exponentially. Another possibility consists in computing $H_D(x)$ directly with the coefficients reduced modulo N: Sutherland in [40] describes a method to compute $H_D(x)$ mod N for any integer N, which runs in $O(|D|^{1+\epsilon})$ time and requires $O(|D|^{1/2+\epsilon} \log N)$ space.

Once $H_D(x) \in \mathbb{Z}_N[x]$ is computed, in general, it is hard to find a root modulo N, without knowing at least a factor of N.

Clearly, when the Hilbert Class Polynomial $H_D(x) \in \mathbb{Z}[x]$ has degree 1, a root modulo N can be trivially computed: there are only 13 negative discriminants D for which this is the case [16, Theorem 7.30], namely

$$D \in \{-3, -4, -7, -8, -11, -12, -16, -19, -27, -28, -43, -67, -163\}$$

When $H_D(x)$ has degree higher than 1, we can get around the problem of explicitly finding a root j modulo N by simply defining our elliptic curve over the ring

$$\mathbb{Z}_N(j) \simeq \mathbb{Z}_N[x]/H_D(x)$$

rather than $\mathbb{Z}_N$. Hence, when using Theorem 1, the coefficients of the resulting Weierstrass curve can be easily computed by inverting $j - 1728$, and all arithmetic operations on the curve can be carried on $\mathbb{Z}_N(j)$ similarly as for $\mathbb{Z}_N$: if some inversions fail, then a root j (or a factor of N) can be explicitly computed.

4.3 Picking Random Points on Curves over $\mathbb{Z}_N(j)$

Once we obtain a short Weierstrass curve equation E over $\mathbb{Z}_N(j) \simeq \mathbb{Z}_N[x]/H_D(x)$, computed according to either Theorem 1, Theorem 2 or Theorem 3, we need at least one point on it to be able to run our attack against the modulus N.

At first glance, this might look hard, since we need to be able to extract square roots modulo N in order to obtain the Y-coordinate corresponding to some (random) X-coordinate on the curve. Indeed, an oracle returning square roots modulo N can be used to efficiently factor N. We can however bypass this problem by adopting, instead, XZ-arithmetic: Bernstein and Lange collected in [7] many efficient XZ-arithmetic formulas, that work in our ring $\mathbb{Z}_N(j)$ as well.

More concretely, for a random X-coordinate $P_X \in \mathbb{Z}_N$, we consider the XZ-point $P = (P_X : 1)$, and we compute $Q = \left(\prod_{p_i \in \mathcal{B}} p_i^{\ell}\right) \cdot P$ using XZ-arithmetic formulas. If $P \in E(\mathbb{Z}_N(j))$, and the order of $E(\mathbb{F}_p)$ is $\mathcal{B}$-smooth, we can then attempt a factorization for N. If, instead, $P \notin E(\mathbb{Z}_N(j))$, the obtained point Q does not help in factoring N: we then need to pick another $P_X \in \mathbb{Z}_N$ and try the above again, until we are confident enough to have picked at least one point on the curve.[3]

[3] We recall that if an X-coordinate does not lie on a curve, it then lies on its quadratic twist. So, for a random $P_X \in \mathbb{Z}_N$, the probability that the XZ-point $P = (P_X : 1) \in E(\mathbb{Z}_N(j))$ is $\frac{1}{2}$.

When a point P lying on $E(\mathbb{Z}_N(j))$ is effectively picked, we need to check if the corresponding XZ-point $Q = (Q_X, Q_Z)$ gives rise to a non-trivial factorization of N. Given the canonical projection $\pi : \mathbb{Z}_N(j) \to \mathbb{F}_p(j) \times \mathbb{F}_q(j)$, we have that if the order of $E(\mathbb{F}_p)$ is $\mathcal{B}$-smooth, then $\pi(Q_Z) = (0, \tilde{Q}_q)$, that is Q is projected to the identity element $(0 : 0)$ (in XZ-coordinates) of $E(\mathbb{F}_p(j))$: if $\tilde{Q} \neq 0$, we can then reveal a factor for N as follows.

From the fact that $\mathbb{Z}_N(j) \simeq \mathbb{Z}_N[x]/H_D(x)$, we can see the Z-coordinate of Q as a polynomial $Q_Z(x) \in \mathbb{Z}_N[x]$ of degree less than $\deg(H_D(x))$: if j is the j-invariant of $E(\mathbb{F}_p)$, it then must be a root of both $H_D(x)$ and $Q_Z(x)$ modulo p, since, as we already saw, $\pi(Q_Z) = (0, \tilde{Q})_q$. In other words the *resultant*[4] over $\mathbb{F}_p$ of these two polynomials satisfies

$$Res(H_D(x), Q_Z(x)) \equiv 0 \pmod{p}$$

If $\tilde{Q}_q \neq 0$, that is Q is not projected to the identity element of $E(\mathbb{F}_q(j))$, we might then reveal a factor for N as

$$\gcd(\, Res(H_D(x), Q_Z(x)),\, N\,) = p$$

Picking points in affine XY-coordinates It is possible to construct XY-points lying on $E(\mathbb{Z}_N(j))$ with no need to extract square roots modulo N. By taking advantage of the free choice of the twisting coefficient and, for $D = -4$, of the curve parameter, we can indeed explicitly write a solution to the curve equation associated to a certain discriminant. The drawback of this approach is that we will then be able to pick points on a particular curve twist only (if $D \neq -4$), and thus, to successfully backdoor semiprimes, we need to generate primes associated to curves isomorphic only to such reachable twist.

On the other hand, the main advantage of using XY-arithmetic in place of XZ, is that we can use standard formulas to compute points additions, rather than *differential addition formulas*, the only available for XZ-points. Indeed, given two points P, Q in XZ-coordinate, we can compute $P + Q$ only if we know the XZ-coordinates of $P, Q, P - Q$ (hence the adjective *differential*), while scalar-point multiplications $[k] \cdot P$, the only relevant for our attacks, can be executed using Montgomery Ladder by only knowing the XZ-coordinates of P. The possibility to execute additions between arbitrary points is of relevance for sketching a distributed protocols to generate semiprimes, based on the prime backdooring idea detailed in previous Sections: we will address this in Section 9.

Assuming $N = p \cdot q$, where p is a backdoored prime, we can construct, for each possible discriminant, points in affine XY-coordinates lying on the curves built according to the above Theorems, as follows.

- $\boldsymbol{D \neq -3, -4}$. We have, for a given k, the curve $E(\mathbb{Z}_N(j)) : y^2 = x^3 - 3kc^2x + 2kc^3$. We generate a random $x \in \mathbb{Z}_N(j)$ and we consider the point $P = (x^2, x^3)$: it lies on E if $2c - 3x^2 = 0$, that is if we set $c = \frac{3}{2}x^2$. It follows that with this approach, only one twist can be selected, namely the one given by the residuosity of $\frac{3}{2}$ modulo p.
- $\boldsymbol{D = -3}$. In this case, we work with the curve $E(\mathbb{Z}_N(j)) : y^2 = x^3 + c^3$. If we set $c = 2$, then the points $P = (1, 3)$ and $Q = (2, 3)$ are both points on E. Again, only one twist can be reached, depending on whether $c = 2$ is or is not a quadratic residue modulo p. These two points seems to be all the non-trivial (i.e. points of order not equal $2, 3, 6$) solution to the Diophantine equation $\frac{x^3+c^3}{y^2} = 1$.

[4] The resultant of two polynomials defined over an integral domain $\mathbb{F}[x]$ is zero if and only if they share a common root in the closure $\overline{\mathbb{F}}$.

– $\boldsymbol{D = -4}$. Here we have $E(\mathbb{Z}_N(j)) : y^2 = x^3 - cx$ for an arbitrary $c \in \mathbb{Z}_N(j)$: we can iteratively hit all the 4 possible orders given by Theorem 3 by picking random $x, w \in \mathbb{Z}_N(j)$ and letting $c = x^2 - xw^2$. It follows that $P = (x, xw)$ is on E.

An alternative approach, described in [1], allows to work with all discriminants and twists by using standard formulas for XY-coordinates. Given an elliptic curve equation $y^2 = f(x)$, with $\deg f = 3$, we pick a random $P_X \in \mathbb{Z}_N$ and we set $\tau = f(P_X)$. We then consider the quotient of the polynomial ring $\mathbb{Z}_N[x, y]$ given by

$$R_{j,\tau} = \mathbb{Z}_N[x, y]/(H_D(x), y^2 - \tau)$$

and the elliptic curve $E(R_{j,\tau}) : y^2 = f(x)$. It follows that $P = (P_X, y) \in E(R)$ and we can perform scalar-point multiplications and additions over $E(R_{j,\tau})$ as usual, by working with the point P. Indeed, if for a certain scalar $k \in \mathbb{Z}_N$ we have $[k] \cdot P = \mathcal{O}$ in $E(R_{j,\tau})$, then $[k] \cdot P = \mathcal{O}$ in $E(\mathbb{Z}_N)$ if $\sqrt{\tau} \in \mathbb{Z}_N$ and j is a root of $H_D(x)$ modulo N. As happens for random points in XZ-coordinates, we cannot be sure that P_X is the X-coordinate of a point in $E(\mathbb{Z}_N)$, and we then need to select multiple points (and thus work on different rings $R_{j,\tau}$) until we are confident enough to have picked one lying on $E(\mathbb{Z}_N)$. Clearly, the arithmetic in $R_{j,\tau}$ is slower than the one we have in $\mathbb{Z}_N(j)$ or $\mathbb{Z}_N$.

5 The Prime Generation Procedure

In light of the considerations outlined in Section 4, we can now formally state our prime generation procedure, previously sketched in Section 3.

Input: a negative discriminant D, the bitsize of the output prime p, the factor base size $n + 1$.
Output: a b-bits prime p, the factor base $\mathcal{B}$.

1. Set $\mathcal{B} = \emptyset$. Generate n tuples (p_i, v_i), where $p_i \in \mathbb{Z}$ are prime odd integers decomposing as $p_i = a_i^2 + |D|b_i^2$ and $v_i = a_i + b_i\sqrt{D} \in \mathbb{Z}[\sqrt{D}]$, and add them to the factor base $\mathcal{B}$. Only if $D = -4$, ensure that for all $(p_i, v_i) \in \mathcal{B}$, $p_i \equiv 1 \pmod 4$.
2. Set $p_0 = (D \pmod 2) + |D|$ and $v_0 = (D \pmod 2) + \sqrt{D}$, and add (p_0, v_0) to $\mathcal{B}$.
3. Randomly pick elements $(p_1, v_1), \ldots, (p_m, v_m)$ from $\mathcal{B}$, for some $m > 0$, so that

$$b - 2 < \log_2 \left(p_0 \cdot \prod_{i=1}^{m} p_i \right) < b$$

   Set $t = p_0 \cdot p_1 \cdot \ldots \cdot p_n$ and $v = v_0 \cdot v_1 \cdot \ldots \cdot v_n = a + b\sqrt{D} \in \mathbb{Z}[\sqrt{D}]$, for some $a, b \in \mathbb{Z}$.
4. If $D \neq -4$, check if $p = (a \pm 1)^2 + |D|b^2$ is prime. If yes, return $(p, \mathcal{B})$. If p is not prime, go to Step 3.
5. If $D = -4$, check if $p = (a \pm 1)^2 + (2b)^2$ or $p = a^2 + (2b \pm 1)^2$ is prime. If yes, return $(p, \mathcal{B})$, otherwise go to Step 3.

We note that, in Step 1, one can either use Algorithm 1 over some $p_i \in \mathbb{Z}$ in order to compute the corresponding v_i, or we can generate random $a_i, b_i \in \mathbb{Z}$, set $p_i = a_i^2 + |D|b_i^2$ and $v_i = a_i + b_i\sqrt{D}$, and test if such p_i are primes. Although primality for elements in $\mathcal{B}$ is, in

general, not required, we experimentally observed that backdoored primes p are generated faster when $\mathcal{B}$ contains only odd primes and the even element p_0, probably because this choice reduces the probability that p is divisible by small factors, e.g. $2, 3$: we therefore generate $\mathcal{B}$ to contain, besides $p_0 = (D \bmod 2) + |D|$, only odd primes which factors in $\mathbb{Z}[\sqrt{D}]$.[5]

Since the factor base $\mathcal{B}$, once generated, can be used to generate multiple primes p, the running time of the above algorithm is dominated by the search for primes in Step 4. Assuming that the computed values p behave like random b-bits odd integers, we then expect to find a prime after $O(b)$ loops.

6 Factoring Backdoored Integers

Let p be a prime generated with respect to a factor base $\mathcal{B}$ and a negative discriminant D, using the prime generation procedure outlined in Section 5. If p is a factor of an integer N, we say that N is *backdoored* since, by knowing $\mathcal{B}$ and D, it would be then possible to recover the (secret) factor p from N. We note that in case of semiprimes $N = p \cdot q$, as in the case of RSA moduli, we will obtain a full factorization for N, regardless of the choice of the other prime q.

To recover p from $\mathcal{B}$, D and $N = p \cdot q$, we run the following attack.

Input: a backdoored N, a negative discriminant D, the factor base $\mathcal{B} = \{p_0, \ldots, p_n\}$.
Output: a factor p of N.

1. If $D \neq -3, -4$:
   (a) Compute the Hilbert Class Polynomial $H_D(x) \in \mathbb{Z}[x]$ and let $\mathbb{Z}_N(j) = \mathbb{Z}_N[x]/H_D(x)$;
   (b) Compute $k = \frac{j}{j-1728} \in \mathbb{Z}_N(j)$;
   (c) Pick a random $c \in \mathbb{Z}_N$ and consider the curve $E : y^2 = x^3 - 3kc^2x + 2kc^3$ over $\mathbb{Z}_N(j)$.
2. If $D = -3$, pick a random $c \in \mathbb{Z}_N$ and consider the curve $E : y^2 = x^3 + c^3$ over $\mathbb{Z}_N$.
3. If $D = -4$, pick a random $c \in \mathbb{Z}_N$ and consider the curve $E : y^2 = x^3 - cx$ over $\mathbb{Z}_N$.
4. Pick a random $P_X \in \mathbb{Z}_N$ and set the XZ-point $P = (P_X : 1)$. For $i \in [0, n]$ compute, using XZ-arithmetic formulas over the curve $E(\mathbb{Z}_N(j))$, the point
$$Q = \left(\prod_{p_i \in \mathcal{B}} p_i^{\ell_i} \right) \cdot P$$
where $\ell_i = \lfloor \log_{p_i} N \rfloor$ or is a fixed constant.
5. Let $Q = (Q_X : Q_Z)$ and consider Q_Z as a polynomial residue in $\mathbb{Z}_N[x]/H_D(x)$. If $\deg H_D = 1$ and $p = \gcd(Q_Z, N) \neq 1, N$, output p. If $\deg H_D > 1$ and $p = \gcd(Res(H_D, Q_Z), N) \neq 1, N$, output p.
6. If $D = -3$ go to Step 2; if $D = -4$ go to Step 3, otherwise go to Step 1c.

[5] This fact should explain the title of this paper.

We note that, if the factor base $\mathcal{B}$ used to backdoor such semiprimes N contains all primes less than a certain bound B, then anyone would be able to factor such N, if correctly guesses the discriminant D used. Indeed, as we already discussed in Subsection 4.2, D cannot be too big, because, otherwise, the computation of $H_D(x) \in \mathbb{Z}_N(x)$ and the corresponding induced arithmetic in $\mathbb{Z}_N(j)$, would result too expensive to carry out. If, instead, the factor base $\mathcal{B}$ is partially secret and contains, for example, many public small elements and few secret big factors, when backdoored $\mathcal{B}$-smooth curve orders are generated to be multiple of at least one of such secret big factors, then N can be easily factored only by those who fully know $\mathcal{B}$.

Question 1. For a given discriminant D, are backdoored integers N distinguishible from non-backdoored ones when the factor base is (partially) secret?

7 Implementation

We implemented both the prime generation procedure from Section 5 and the factorization attack detailed in Section 6 in SageMath [41]. Our implementations is available on GitHub at

`https://github.com/cryptolu/primes-backdoor`

The first script `gen_prime.sage` generates, for a given input negative discriminant D and bitsize b, a random b-bits prime p admitting a Cornacchia's decomposition with respect to D, and so that one among the curve orders built according to either Theorem 1, Theorem 2, Theorem 3 is $\mathcal{B}$-smooth, with $\mathcal{B}$ containing the first suitable primes up to a certain (input) bound. Optionally, this script can output *safe primes* rather than just primes.

The second script is `attack.sage` and takes as input an integer N and a discriminant D. It attempts the attack from Section 6 on N, by building a proper elliptic curve $E(\mathbb{Z}_N)$ and by computing $[\prod_{p_i \in \mathcal{B}} p_i^{\ell}] \cdot P$ for a random point $P = (P_X : 1)$ in XZ-coordinates. In order to compute scalar-point multiplications, the implemented XZ-arithmetic uses Montgomery Ladder [31], differential doubling `xECDBL` and differential addition `xECADD` formulas reported, respectively, in [26, Algortihm 3, A.3, A.5]

We employed these two scripts to backdoor and later factor the modulus we report in next Section to show an attack example.

7.1 A Full Attack Example

Suppose we have just generated the following 1024-bits RSA modulus

$$\begin{aligned} N \;= {} & 10821805528622065889230568696679645000565609385356188057330375777987085 \\ & 25027157508967173806014163322826354167515058707101795774644011936428240 \\ & 27551121223839066820321283484303809645541513752756237536860056248506557 \\ & 92830578845228701699881266834878524228204494605987230242486665425532888 \\ & 73831841082047820147452 21 \end{aligned}$$

using some shady closed-source software found online. We suspect it has been somehow backdoored and we would like to check it against the attack outlined in Section 6. We start iterating through all possible discriminants up to a certain bound, and we are now working with $D = -107$. We then compute the Hilbert Class Polynomial

$$H_{-107}(x) = x^3 + 129783279616000 \cdot x^2 - 6764523159552000000 \cdot x + 337618789203968000000000$$

we set $\mathbb{Z}_N(j) \simeq \mathbb{Z}_N[x]/H_D(x)$, and we obtain $k = \frac{j}{j-1728} \in \mathbb{Z}_N(j)$ as

$$\begin{aligned} k \;=\; & 472613932478766669340966844279313193169990661224983092468863164165826307613699313280380573797884742870676468100220081538362394298484445303207371515261890383308740374138417052518390519213493529749497051238720669707580258579708435533425339388316585678359757629037199917351893853149116839732775799298262143392652583537491239223892487049209941342381846842377666674250844657890259491423980136846866193858910647641442977366590032461132868140632160013934433342109560442476654225465545173024567492854420947803357476124489494165071306819849051585970711489381922793715839533605339477741539037810199585641157784517018132244846020921251340899301433133091422967299422144510775713942007364855038198294294074452975232079006272402865814000258622429373692473727820051692220181932959624105835644432609167110193260413277951994679542200751242365012754312396131057298816375017611381308687252198337528312773336198453215573257308017661226329313052120 \cdot j^2 \;+ \\ & 258353749123922389248704920994134238184684237766667425084465789025949142398013684686619385891064764144297736659003246113286814063216001393443334210956044247665422546554517302456749285442094780335747612448949416507130681984905158597071148938192279371583953360533947774153903781019958564115778451701813224484602125134089930143313309142296729942214451077571394200736485503819829429407445297523207900627240286581400025862242937369247372782005169222018193295962410583564443260916711019326041327795199467954220075124236501275431239613105729881637501761138130868725219833752831277333619845321557325730801766122633293130521 \cdot j \;+ \\ & 921251340899301433133091422967299422144510775713942007364855038198294294074452975232079006272402865814000258622429373692473727820051692220181932959624105835644432609167110193260413277951994679542200751242365012754312396131057298816375017611381308687252198337528312773336198453215573257308017661226329313052120 \end{aligned}$$

We pick a random $c \in \mathbb{Z}_N$, and we define the curve $E(\mathbb{Z}_N(j)) : y^2 = x^3 + Ax + B$, where $A = -3kc^2$ and $B = 2kc^3$. In our case, these values result to be

$$\begin{aligned} A \;=\; & 327995130978423695949911778178828621193335624428619226678494051131870788874045524570484841135760473494416159360628197206582602436582918492949905396314203212027061815963064338782746734177580622582323886810815056929636602581034529682795514610361665304507462995342415407193855712062892768333900714820992270292189 \cdot j^2 \;+ \\ & 107249083075978240751276263919220465878752147961845607664173475142638397088074991995772702203857717228151158940364308079702814510166667790343423279033188777287190801104917972949118291714832737203762143779784374706244554915429186905254336517475532965059602383225069319036437993060566379889493546762895764180 14 \cdot j \;+ \\ & 819160971922140650359819037400073641565902668643711131669462520070780364452384201748612105417995406358643088745166654606993288705239340463153087566698150625199250395141425416658589321540290967944489201432328127690468661842228330448470250580291070716536018499660018728202197972870044690373459887404123397542 1 \end{aligned}$$

$$
\begin{aligned}
B \;=\; & 37643281679908451518787240246983699788548039697396258693595972085905148\\
& 99938443403690095427983393285685315480827331067412102780873968392844583\\
& 97608707532457609382704422614143853293195820528050043962557041892207826\\
& 70921663270503590660363838807542796580830434422155103586357147978131797\\
& 490769219422563399764351 \cdot j^2 \;+\\
& 32626534476473067853864460804039952271142555656166723801964494031024106\\
& 90220980303345333246188191866771155608362353346142265495937928857247431\\
& 31088094681213733581292166083595187928876253687304909544272446933953143\\
& 70483605838297461241956398349370087815706020355607878266998770014036572\\
& 0355713277916839557377 \cdot j \;+\\
& 20187952415027906389928913590443320149447964629255548845322188250621303\\
& 98301801288156033427731216315886543490632862896772443034233202131475860\\
& 43433031814108287949381589352802976017014977188595963194192471898706544\\
& 71654604057276712064018623814379425617988738224774811104681656315004660\\
& 54560752894547005649069 9
\end{aligned}
$$

We then generate a random XZ-point P, such as

$$
\begin{aligned}
P \;=\; & (82133862752128935234225899250164185444197374702059983503603347370808 76\\
& 75343837772874769173897288321433887175513664433507107489728296742138840\\
& 55186744682533073101226469679418098641072829349126102914037432335818914\\
& 18089798305128319217013949654585856460461999703884598446829478890812625\\
& 78425826039835717706169 26 : 1)
\end{aligned}
$$

and we compute, using XZ-arithmetic, the point $Q = \left(\prod_{p_i \in \mathcal{B}} p_i^{\ell_i}\right) \cdot P$ (we set $\ell_i = 2$ for all i), using a factor base $\mathcal{B}$ containing all primes p_i of the form $p_i = a_i^2 + |D| b_i^2$, with $0 \leq a_i, b_i < 2^8$ for all $i > 0$, and the element $p_0 = 1 + |D| = 108$, for a total of 3468 elements. The point we obtained is

$$
\begin{aligned}
Q \;=\; & (33348654677483343436398943001604402358868875186728538744594056968719 29\\
& 01187112782588071993988090051581869480356207729617515633762706996270205\\
& 01762228657451819904121885847689656173650676028922225633544880373520 54\\
& 05855804928111263260919442420055411327737604112320289061101226982531565\\
& 28214638996568895564046 99 \cdot j^2 \;+\\
& 76899064963804553947100734765597160240602793157812297374592707576165068\\
& 33008583862582611334448071824599063655204175095973123384801205838140434\\
& 81344311982524250512505605731260851632209268820236257688149220279076838\\
& 91511832006389277862362492860443942877643243130905343548520132640854211\\
& 28950995465463715347443 5 \cdot j \;+\\
& 43417397680346045496842107591102055998528639529347969490726893067887581\\
& 31453510690971259655887610317060352371378239469718379470199395383106806\\
& 75972092802552957068857743927536560104426449684823455184876742782247286\\
& 83524283318084102266565399244576196892371920484484129148865961043236956\\
& 46824168404593031366713 4 \;:
\end{aligned}
$$

$$
\begin{aligned}
&6182115395610171116512648228653716984299018809622163408323488022374796373632812995398780367381750565749075802819168683372166826519855312796113\\
&2031704294095209463756691728653769822746319710521198516381541615057312198579604410190485869167369178476177586812373538524310238956155534423748\\
&88382248599669119196652 8\cdot j^2 \;+\\
&6402009202641366184445716387547812503451800342832070085814782172104565818099915491272074968012291963648399121125602177743509824254475666648162\\
&3813285991729671734877269407128296465725650516056477191180920231907561220531683357964007760952960102102932026611149676923234595697323298973117\\
&308194412478184695042548\cdot j \;+\\
&6284204000085662559475394410903505368358675525673969202322254516593231907285845733613656164239022079675483953309265982187813651134963341392690\\
&7088739990567263600206279729444443226317924679240553488041770400714347415353020883868051179887770286381243799587295814437072069872663260291675\\
&3943955125963075063372 94)
\end{aligned}
$$

We let $Q = (Q_X, Q_Z)$ and we consider Q_Z as an element in $\mathbb{Z}_N[x]/H_D(x)$, rather than $\mathbb{Z}_N(j)$. We then compute the resultant $r = Res(Q_Z, H_D(x))$, corresponding to

$$
\begin{aligned}
r \;=\; &8093552098721637783664885262489676950910732180486184398613457277816113052910181368256682771271632250041863572397842181395379909775550322540693\\
&8890096385721376531555513857135772622868558730773132076148750842435806479124176665460164828297949244069504173394101556250503142950724934058250\\
&88628395154536375860 3886
\end{aligned}
$$

Unexpectedly, by computing $p = gcd(r, N)$ we obtain

$$
\begin{aligned}
p \;=\; &1006989116827285328968241453344410115896197116072181096042529972750004085615539991040809791080655609508252915420491222187265608651502003491652\\
&4535013055607
\end{aligned}
$$

which confirms our suspicions! The attack leads to full factorization of N as

$$
\begin{aligned}
N \;=\; &1006989116827285328968241453344410115896197116072181096042529972750004085615539991040809791080655609508252915420491222187265608651502003491652\\
&4535013055607 \cdot 9163585010630240914315854502079840960495506763984904313022218586074134346260658498746192771596180175665775150505169330955024838\\
&8469175825632468393477290 83
\end{aligned}
$$

We note that both factors of N are *safe primes*.

8 Certifiable Semiprimes

We can use the prime backdooring procedure from Section 5 to sketch a *multi-party computation* (MPC) protocol which outputs semiprimes of unknown factorization, a particularly useful application in the setting of distributed generation of RSA moduli. We will take

advantage of a generalization of a Theorem by Goldwasser and Kilian, which provides a criterion for an integer N to be semiprime given partial knowledge of the order of an elliptic curve modulo N. When we port this idea into the MPC setting, parties will jointly construct a curve modulo N by backdooring part of its order according to this theorem, and where N, once revealed, can be *certified* to be semiprime, thanks to a semiprimality proof that can be publicly checked without knowing any of the factors of N.

8.1 Preliminaries

In this Section we provide the theoretical results to formalize semiprimality *certificates* for odd integers. We start by stating Goldwasser-Kilian Theorem, originally thought as a tool to prove primality of a certain integer N.

Theorem 4 (Goldwasser-Kilian [25]). *Let $N > 1$ and let E be an elliptic curve defined over $\mathbb{Z}_N$. Suppose there exist distinct primes $p_1, \ldots, p_k$ and finite points $P_1, \ldots, P_k \in E(\mathbb{Z}_N)$ such that $[p_i] \cdot P_i = \mathcal{O}$ for all $1 \leq i \leq k$ and $\prod_{i=1}^{k} p_i > (\sqrt[4]{N} + 1)^2$. Then N is prime.*

Proof. See [42, Theorem 7.3]. □

We can slightly restate Goldwasser-Kilian Theorem to provide a condition that, when true, will ensure N has at most a certain number of distinct prime factors.

Theorem 5. *Let $N > 1$ and let E be an elliptic curve defined over $\mathbb{Z}_N$. Suppose there exist distinct primes $p_1, \ldots, p_k$ and finite points $P_1, \ldots, P_k \in E(\mathbb{Z}_N)$ such that $p_i \cdot P_i = \mathcal{O}$ for all $1 \leq i \leq k$ and $\prod_{i=1}^{k} p_i > \left(\sqrt[2m]{N} + 1\right)^2$. Then N can have at most $m - 1$ distinct prime factors.*

Proof. We will adjust to our needs the proof of Theorem 4, that is [42, Theorem 7.3]. Let q be a prime factor of N so that $N = q^s \cdot d$ for some $s > 0$ and $\gcd(q, d) = 1$. Since P_i is a finite point in $E(\mathbb{Z}_N) \simeq E(\mathbb{Z}_{q^s}) \times E(\mathbb{Z}_d)$ it is a finite point modulo q^s and, in turn, modulo q as well. Then $p_i \cdot P_i \pmod{q} = \mathcal{O} \in E(\mathbb{F}_q)$, which implies that $P_i \pmod{q}$ has order p_i in $E(\mathbb{F}_q)$ for all $1 \leq i \leq k$. It follows that $\prod_{i=1}^{k} p_i \mid \#E(\mathbb{F}_q)$ and by Hasse bound

$$\left(\sqrt[2m]{N} + 1\right)^2 < \prod_{i=1}^{k} p_i \leq \#E(\mathbb{F}_q) < q + 1 + 2\sqrt{q} = (\sqrt{q} + 1)^2$$

So $q > \sqrt[m]{N}$ for any prime q dividing N, hence N cannot have more than $m - 1$ distinct prime factors. □

Corollary 1. *Let $N > 1$ be a composite non-square integer and let E be an elliptic curve defined over $\mathbb{Z}_N$. Suppose there exist a prime $s > \left(\sqrt[6]{N} + 1\right)^2$ and a finite point $P \in E(\mathbb{Z}_N)$ so that $[s] \cdot P = \mathcal{O}$ in $E(\mathbb{Z}_N)$. Then N is semiprime.*

We note that in Corollary 1 we require s to be prime: its primality can be certified using Theorem 4, factoring the orders of elliptic curves built over $\mathbb{Z}_s$ and explicitly construct, when possible, points of prime order satisfying the assumption of Theorem 4. It follows that

Theorem 4 can be used, in turn, to prove primality of the orders of the points employed to prove primality of a certain integer: this process ultimately results in a *chain* of primality proofs which reduces the problem of certifying primality of s to certifying primality of (much) smaller integers. Such certificate chains are commonly known as Atkin-Goldwasser-Kilian-Morain primality certificates [4, 25].

8.2 Certificates for Semiprimes for which a Factorization in Known

From Corollary 1 follows that, if we know (or *we construct!*) a tuple $(N, E(\mathbb{Z}_N), P, s)$ whose elements satisfy its assumptions, then such tuple is, in fact, a *semiprimality certificate* for N.

A naive approach to compute such certificates when the factors of $N = p \cdot q$ are known, consists in generating a random curve $E(\mathbb{Z}_N)$ for which we can factor its order, in turn obtained by computing the orders of the two curves $E(\mathbb{F}_p)$, $E(\mathbb{F}_q)$ using, for example, the Schoof–Elkies–Atkin (SEA) algorithm [2,3,23,35]. Indeed, if $|E(\mathbb{Z}_N)|$ is divisible by a prime $s > (\sqrt[6]{N} + 1)^2$, or by multiple primes p_i so that $\prod_i p_i > (\sqrt[6]{N} + 1)^2$ (Theorem 5), we can pick random points $Q \in E(\mathbb{Z}_N)$[6] and check if $P = [|E(\mathbb{Z}_N)|/s] \cdot Q \neq \mathcal{O}$. When this holds, it follows that $[s] \cdot P = \mathcal{O} \in E(\mathbb{Z}_N)$ and thus $(N, E(\mathbb{Z}_N), P, s)$ is a semiprimality certificate for N.

This approach has two main drawbacks: the first is that partial factoring the order $|E(\mathbb{Z}_N)|$ until enough divisors $\{p_i\}_i$ so that $\prod_i p_i > (\sqrt[6]{N}+1)^2$ are found, is often expensive or even impractical depending on the size of N. Secondly, if a certain p_i divides $|E(\mathbb{F}_p)|$ but not $|E(\mathbb{F}_q)|$, it cannot belong to a semiprimality certificate since any point P_i of order p_i will leak a factor of N, similarly as we describe in Section 6: from the condition $[p_i] \cdot P_i = \mathcal{O} \in E(\mathbb{Z}_N)$, we must have P_i to correspond to the identity element of $E(\mathbb{F}_q)$ when its projective coordinates are reduced modulo q, and thus the Z-coordinate of P_i shares a factor with N.

In other words, each element p_i published in a semiprimality certificate should divide *both* orders $|E(\mathbb{F}_p)|$ and $|E(\mathbb{F}_q)|$: on this regards, we note that to reduce certificate sizes, we simply avoid publishing multiple points P_i, each of size at least $2 \log N$, along with their order p_i, and we publish instead a single point P of order $s > (\sqrt[6]{N} + 1)^2$ with s prime, as done in the formulation of Corollary 1. Indeed, if s is prime, by Cauchy's Theorem we know that both curves $E(\mathbb{F}_p)$ and $E(\mathbb{F}_q)$ have a point of order s, and thus, by Chinese Remainder Theorem, it exists a point P in $E(\mathbb{Z}_N)$ of order s. In fact, $E(\mathbb{Z}_N)$ will contain a subgroup isomorphic to $\mathbb{Z}_s \times \mathbb{Z}_s$ and, to avoid leaking factors of N as showed above, we look for points P of order s in correspondence to elements $(a, b) \in \mathbb{Z}_s \times \mathbb{Z}_s$ with $a, b \neq 0$.

How can we then efficiently generate big certifiable semiprimes? In [19,21], we found concrete instances of semiprimality certificates presumably generated according to observations similar to the above[7]. In a private conversation, Reble confirmed us that his certificate [21] for a semiprime N of 1084 digits was generated from its prime factors, and added, in regards to its generation: "*I knew the factors. I sought a pair of primes such that the Goldwasser-Kilian test almost worked for that product*". Shortly after [21] become public, Broadhurst generated a semiprimality certificate for a 5061-digits integer [19] consisting of a single elliptic curve point P and a 1690-digits prime s, for which a primality proof is known. We

[6] Equivalently, we can work, thanks to the Chinese Remainder Theorem, over $E(\mathbb{F}_p) \times E(\mathbb{F}_q)$.

[7] We were not able to find more details about how these certificates were generated, except a short discussion on their validity with respect to (a generalization of) Goldwasser-Kilian Theorem.

note that both [19, 21] employ the elliptic curve $E(\mathbb{Z}_N) : y^2 = x^3 + A \cdot x$, whose order is characterized by Theorem 3.

In a more general fashion, we can efficiently compute semiprimality certificates for an integer N by generating both its prime factors according to the backdooring procedure of Section 5, and ensuring that the candidate curve orders are divisible by a prime $s > (\sqrt[6]{N} + 1)^2$ which admits a Cornacchia decomposition for the employed D. This can be achieved by slightly change our prime generation procedure, adding such prime s to the factor base $\mathcal{B}$ (Step 1), and requiring t to always be a multiple of s and the even element p_0 (Step 3). From the full knowledge of $|E(\mathbb{F}_p)|$ and $|E(\mathbb{F}_q)|$, where the curve E is retrieved from N as in Section 6, we can then randomly pick and re-scale points in $E(\mathbb{Z}_N)$ as above, until we find one of order s (and Z-coordinate not equal to 0 when seen modulo p and modulo q).

9 Distributed Computation of Certifiable Semiprimes

In this Section we investigate how we can possible have a multi-party computation protocol to jointly compute an integer N of unknown factorization and a semiprimality certificate for it. Our goal is to port our prime generation procedure, slightly restated as described at the end of Subsection 8.2, in a distributed setting, so that parties generate two candidate prime curve orders t_p, t_q for $E(\mathbb{F}_p)$ and $E(\mathbb{F}_q)$, respectively, both divisible by a public prime $s > (\sqrt[6]{N}+1)^2$. Parties then opens $N = p \cdot q$ and multiple points $Q_i = [\frac{t_p t_q}{s^2}] \cdot P_i$ for random $P_i \in E(\mathbb{Z}_N)$. If for some i, we have $Q_i \neq \mathcal{O}$ and $[s] \cdot Q_i = \mathcal{O}$, we then output N and the semiprimality certificate (N, E, s, Q_i).

The whole procedure is sketched as follows, where we employed the notation $[\,\cdot\,]$ to denote secret values.

Input. The bitsize `bits`, a negative discriminant D.
Output. A semiprimality certificate for a `bits`-bits integer N.

1. Parties publicly agree on two integers s_a, s_b so that $s = s_a^2 + |D| \cdot s_b^2$ is a prime of `bits`/3 bits.
2. Each party i randomly picks 4 values $[a_{i,1}], [b_{i,1}], [a_{i,2}], [b_{i,2}]$ of `bits`/12 bits each.
3. Parties jointly compute the 4 values:
$$[a_1] = \sum_i [a_{1,i}] \,, \qquad [b_1] = \sum_i [b_{1,i}] \,, \qquad [a_2] = \sum_i [a_{2,i}] \,, \qquad [b_2] = \sum_i [b_{2,i}]$$
4. Parties compute the 4 values
$$\begin{aligned} [p_a] &= s_a \cdot [a_1] + D \cdot s_b \cdot [b_1] \\ [p_b] &= s_a \cdot [b_1] + s_b \cdot [a_1] \\ [q_a] &= s_a \cdot [a_2] + D \cdot s_b \cdot [b_2] \\ [q_b] &= s_a \cdot [b_2] + s_b \cdot [a_2] \end{aligned}$$
5. If $D \neq -4$, for each choice of $v_1, v_2 \in [-1, 1]$, parties jointly open the 4 values
$$N_{v_1,v_2,0} = \left(([p_a] + v_1)^2 + |D| \cdot [p_b]^2\right) \cdot \left(([q_a] + v_2)^2 + |D| \cdot [q_b]^2\right)$$

6. For each choice of $v_1, v_2 \in [-1, 1]$, parties may[8] further open the values

$$N_{v_1,v_2,0} = \left([p_a]^2 + (2 \cdot [p_b] + v_1))^2\right) \cdot \left([q_a]^2 + (2 \cdot [q_b] + v_2)^2\right)$$

$$N_{v_1,v_2,1} = \left(([p_a] + v_1)^2 + |D| \cdot [p_b]^2\right) \cdot \left([q_a]^2 + (2 \cdot [q_b] + v_2)^2\right)$$

$$N_{v_1,v_2,1} = \left([p_a]^2 + (2 \cdot [p_b] + v_1))^2\right) \cdot \left(([q_a] + v_2)^2 + |D| \cdot [q_b]^2\right)$$

7. For each opened value N_{v_1,v_2,v_3}:
   7.1. Let $N_{v_1,v_2,v_3} = N$. If either N is a perfect square, has small prime factors or $s < (\sqrt[6]{N} + 1)^2$, parties skip to the next choice of N_{v_1,v_2,v_3}.
   7.2. Depending on the discriminant D, parties publicly agree on one or multiple curves $\{E_\ell(\mathbb{Z}_N[x]/H_D(x))\}$ along with a point $P_\ell \in E_\ell$ on it[9], until are confident enough to have picked a curve representative for each reachable order (cf. Theorem 1, Theorem 2, Theorem 3).
   7.3. For each $P_\ell \in E_\ell$, parties jointly compute and open the point

$$Q_\ell = \left[([a_1]^2 + |D| \cdot [b_1]^2) \cdot ([a_2]^2 + |D| \cdot [b_2]^2)\right] \cdot P_\ell$$

   If $Q_\ell \neq \mathcal{O}$ and $[s] \cdot Q_\ell = \mathcal{O} \in E_\ell$, return the semiprimality certificate $\{N, E_\ell, Q_\ell, s\}$.
8. Go to Step 2.

9.1 Practical Considerations

Although there exist many protocols that, depending on the most suitable security scenario, can efficiently perform in MPC the standard arithmetic operations needed in the first part of our sketched protocol, execution of Step 7 is not trivial and may represent the main bottleneck for implementing it in full. Difficulties mainly reside in the opening of the points Q_ℓ, which involves a scalar-point multiplication $[k] \cdot P_\ell$, for a secret k shared among parties.

Although protocols like [24,39] allow distributed computation of $[k] \cdot P_\ell$ when P_ℓ belongs to a curve defined over a finite field of known characteristic, in our case we work, instead, over a ring (either $\mathbb{Z}_N$ or $\mathbb{Z}_N(j) \simeq \mathbb{Z}_N[x]/H_D(x)$) and the order of P_ℓ is (*and should remain!*) unknown. In other words, these protocols cannot be employed, at least not straightforwardly, and we are not aware of designs that can be used in our case of interest (and which are eventually able to work with XZ-coordinates).

We can however bypass this limitation: instead of directly opening the point $Q_\ell = [k] \cdot P_\ell$, where $k = ([a_1]^2 + |D| \cdot [b_1]^2) \cdot ([a_2]^2 + |D| \cdot [b_2]^2)$, we additively secret share among all parties the secret $[k]$ over a field $\mathbb{F}_r$, with r a public prime much greater than the expected value for k.

Once $[k]$ is additively shared, each party i possesses a share $k_i \in \mathbb{F}_r$ satisfying $(\sum_i k_i) \pmod r = k$. Thus, for each publicly agreed point P_ℓ, party i locally computes $\tilde{P}_\ell = [k_i] \cdot P_\ell$ and publishes this point to other parties. Assuming n parties are involved in the distributed

[8] Different trade-offs are possible depending on the employed MPC arithmetic protocol, the overall network communication and the number of openings parties execute out of the 16 available.

[9] It may be needed to test multiple random points P_ℓ, depending on the curve arithmetic used, before being sure to have picked at least one point on E_ℓ.

computation, they can then compute in clear $\tilde{Q}_\ell = \sum_{i=1}^{n} \tilde{P}_\ell$ and the set of elliptic curve points

$$\mathcal{Q}_\ell = \{\ \tilde{Q}_\ell,\ \ \tilde{Q}_\ell - [r] \cdot P_\ell,\ \ \tilde{Q}_\ell - [2 \cdot r] \cdot P_\ell,\ \ \ldots,\ \ \tilde{Q}_\ell - [n \cdot r] \cdot P_\ell\ \ \}$$

It follows that the point $Q_\ell = [k] \cdot P_\ell$ is in $\mathcal{Q}_\ell$, and parties can then check the condition $[s] \cdot Q_\ell = \mathcal{O} \in E_\ell$ at the end of Step 7, by checking if $\mathcal{O}$ is in the the set of points $\{\ [s] \cdot \tilde{Q}\ \mid\ \tilde{Q} \in \mathcal{Q}_\ell\ \}$.

There is one last subtlety we need to address: the public computation of the point $\tilde{Q}_\ell = \sum_{i=1}^{n} \tilde{P}_\ell$ cannot be done in XZ-coordinates, since addition formulas are defined in terms of *differential addition* and to compute any sum $P + Q$ we need the X-coordinate of the point $P - Q$, which in our case is unknown.

We can address this problem mainly in two ways. The first approach consists in working over the rings $R_{j,\tau}$ we defined at the end of Section 4.3, that, regardless of the discriminant D chosen, allows us to pick random points $P_\ell \in E(R_{j,\tau})$ in XY-coordinates and thus use standard formulas for points addition. Unfortunately, although this approach allows us to generalize the construction to all possible curve twists and negative discriminant D, in practice it results expensive in terms of ring arithmetic and parties need to work on a different ring $R_{j,\tau}$ for each choice of the point P_ℓ in Step 7.

Alternatively, we can explicitly construct random XY-points P_ℓ with coordinates in $\mathbb{Z}_N$ as we detail in Section 4.3: although the arithmetic would result faster, within this approach we will be able to work only with one curve twist for each factor of N when $D \neq -4$, and thus we need, on average, four times as many iterations of the protocol before returning a semiprime certificate. However the case $D = -4$ suits best our needs: first, the corresponding Hilbert Class Polynomial has degree 1, thus a root j for $H_D(x)$ can be easily obtained modulo N; secondly, in Section 4.3, we show how it is possible to explicitly construct points in XY-coordinates over $\mathbb{Z}_N$ for all the curve orders characterised by Theorem 3. It follows that when $D = -4$, we can work exclusively over $\mathbb{Z}_N$ and perform point additions with standard affine XY-coordinates formulas.

Assuming the inputs from parties to be random, the algorithm terminates when: i) N is a product of two primes; ii) the random curve modulo N selected matches the curve twists with orders divisible by s; iii) the elliptic curve arithmetic does not fail modulo N. If at each round of the protocol, in Step 5 and Step 6 m different integers are opened in total, then a semiprime is returned in approximately $\frac{1}{m} \cdot (\frac{\texttt{bits}}{2})^2 = O(\texttt{bits}^2)$ rounds execution.

9.2 Implementation

To show its practicality, we implemented the protocol outlined in Section 9 for the case $D = -4$ in SageMath [41] and MP-SPDZ [27], a multi-protocol framework for multi-party computations based on SPDZ [17, 18]. Our implementation can be found at

`https://github.com/cryptolu/semiprimes`

and further algorithmic optimizations are left as future work.

In order to allow parties to open points Q_ℓ in Step 7, we additively secret share the value $k = ([a_1]^2 + |D| \cdot [b_1]^2) \cdot ([a_2]^2 + |D| \cdot [b_2]^2)$ over a prime order field $\mathbb{F}_r$ of bit-size double as N, we compute the set $\mathcal{Q}_\ell$, and we check the value of $[s] \cdot Q_\ell$ as detailed in previous Section.

The SageMath script `generateN.sage` automates the execution of the MP-SPDZ multi-party computation script `semiprimes.mpc`, which performs the required distributed arithmetic operations over parties' secret values, the retrieval of the opened moduli and parties' additive shares, the elliptic curve arithmetic and the final checks on the returned semiprimality certificate.

As a proof of concept, we report a 128-bits semiprime generated, using our implementation, in $1300 \approx \frac{64^2}{4}$ rounds (to reduce communication, we didn't implement Step 6). The generation involved two parties both running on a standard desktop, who communicated using the MP-SPDZ `semi` protocol for semi-honest Oblvious Transfer based computations modulo a prime. The semiprimality certificate returned is

$$\begin{aligned} N = \;& 46121327044532730890860996866516955982927 33 \\ E(\mathbb{Z}_N) \;:\; & y^2 = x^3 + 20921147449173724872153659295373299245369 03 \cdot x \\ Q = (\;& 35792898069199412144322564728728619314297 11 : \\ & 14154437122626629949263497606251710686648 53) \\ s = \;& 538606233865081 \end{aligned}$$

where, in fact, $N = 556886529430039208669 \cdot 8281997248476609399457$.

10 Security of Semiprimes Certificates

In this Section we briefly investigate security of semiprimality certificates for integers N generated according to Subsection 8.2 or Section 9, that is containing elliptic curves whose orders modulo each prime factor of N are characterized by either Theorem 1, Theorem 2 or Theorem 3[10]. We will detail and compare three kind of attacks: a generic point order-finding algorithm based on Baby-step Giant-step algorithm combined with a *twisting attack*, a brute-forcing approach exploiting some leakage provided by semiprimality certificates, and factorization.

10.1 Baby-step Giant-step and the Twisting Attack

Since semiprimality certificates (N, E, Q, s) are generated using a variant of the prime generation procedure of Section 5, our factorization attack detailed in Section 6 will retrieve a factor of $N = p \cdot q$ with high probability, as soon as we compute (a multiple of) the order of a random point P in either $E(\mathbb{F}_p)$ or $E(\mathbb{F}_q)$. While in Section 6, we assume the order of at least one of the curves $E(\mathbb{F}_p)$, $E(\mathbb{F}_q)$ to be $\mathcal{B}$-smooth for a certain factor base $\mathcal{B}$, in the case of semiprimality certificates we have that $|E(\mathbb{F}_p)| = s \cdot k_p$ and $|E(\mathbb{F}_q)| = s \cdot k_q$ for certain random integers k_p, k_q.

Thus, as we already discussed in Subsection 4.3, given a curve $E(\mathbb{Z}_N) = E(\mathbb{F}_p) \times E(\mathbb{F}_q)$ and a random $P \in E(\mathbb{Z}_N)$ (in XZ-coordinates) on it, the point $\tilde{P} = [s] \cdot P \in E(\mathbb{Z}_N)$ would be such that $[k_p] \cdot \tilde{P} = (\mathcal{O}_p, \tilde{P}_q)$ and $[k_q] \cdot \tilde{P} = (\tilde{P}_p, \mathcal{O}_q)$, where $\mathcal{O}_p$ and $\mathcal{O}_q$ represents the identity elements of $E(\mathbb{F}_p)$ and $E(\mathbb{F}_q)$, respectively. In other words, if we are able to find a

[10] There might exists, indeed, other curve equations for which the order can be characterised in a way that allow efficient generation of semiprimality certificates.

multiple of either k_p or k_q, we will be able to factor N as long as $\tilde{P}_q$ or $\tilde{P}_p$, respectively, are non-trivial.

To find such multiple, we can apply the Baby-step Giant-step algorithm to find the order of $\tilde{P} = [s] \cdot P$, where $P \neq Q$ is a random point in $E(\mathbb{Z}_N)$ expressed in XZ-coordinates (as usual, we compute the orders of different points $\tilde{P}$, until we are confident enough to have picked one lying on $E(\mathbb{Z}_N)$).

Assuming $p \approx q$, we have $\ln(k_p) \approx \ln(k_q) \approx \frac{\ln N}{6}$: since $\tilde{P}$ generates a group of order at most $k_p k_q$, we have that a successful execution of the Baby-step Giant-step algorithm would require on average $\exp\left(\frac{\ln N}{6}\right)$ scalar-point multiplications and $\exp\left(\frac{\ln N}{6}\right)$ space to return a (big) divisor k of $k_p k_q$ which, for simplicity, we can safely assume to be $k = k_p k_q$.

However, when we try to compute, for multiple $P \in E(\mathbb{Z}_N) = E(\mathbb{F}_p) \times E(\mathbb{F}_q)$, the point $[k \cdot s] \cdot P$, we obtain $[k \cdot s] \cdot P = (\mathcal{O}_p, \mathcal{O}_q)$, which does not help us in factoring N, unless we factor k itself. Indeed, by factoring k, we will be able to find a multiple $\tilde{k}$ of k_p which is not a multiple of k_q, or vice-versa, and so that $[\tilde{k} \cdot s] \cdot P \neq (\mathcal{O}_p, \mathcal{O}_q)$.

Although factoring k is asymptotically easier than running the Baby-step Giant-step algorithm to find it, we now show a technique which finds with overwhelming probability a factor of N knowing only such k and none of its factors. We call this method *"twisting attack"*. Let $c \in \mathbb{Z}_N$ a random integer so that its Jacobi symbol $\left(\frac{c}{N}\right) = -1$: since $N = p \cdot q$ is a semiprime, it follows that c is a quadratic residue in $\mathbb{F}_p$ and not in $\mathbb{F}_q$, or the opposite. Hence, the quadratic twist $\tilde{E}(\mathbb{Z}_N) = \tilde{E}(\mathbb{F}_p) \times \tilde{E}(\mathbb{F}_q)$ of $E(\mathbb{Z}_N)$ through c would be isomorphic to either $E(\mathbb{F}_p) \times \tilde{E}(\mathbb{F}_q)$ or $\tilde{E}(\mathbb{F}_p) \times E(\mathbb{F}_q)$. It follows that, given its size, s will not divide $|\tilde{E}(\mathbb{F}_p)|$ and $|\tilde{E}(\mathbb{F}_q)|$ with overwhelming probability and thus, when we pick a random point $P \in \tilde{E}(\mathbb{Z}_N)$, we will have that $[s \cdot k] \cdot P$ is not equal to $\mathcal{O} \in \tilde{E}(\mathbb{Z}_N)$ but decomposes either as $(\mathcal{O}_p, \cdot)$ or $(\cdot, \mathcal{O}_q)$. We can then easily retrieve a non-trivial factor of N by applying the greatest common divisor to the Z-coordinate of $[s \cdot k] \cdot P$.

As a side note, we observe that the twisting attack can be applied to target either *pseudo super anomalous curves* or *super anomalous curves* [28], that is elliptic curves $E(\mathbb{Z}_N)$ with order equal N (super anomalous curves further require that if $N = \prod_i p_i$, then each curve $E(\mathbb{F}_{p_i})$ has order p_i, that is *anomalous*). For these curves, indeed, the ring characteristic trivially reveals the full curve order modulo N and hence, by working with twists of $E(\mathbb{Z}_N)$ as above, we can factor N, and directly work over the corresponding sub-curves.

Another example of applicability of this method is given by [32], where a class of parameters for Demytko's Elliptic Curve Cryptosystem [20] is shown to be weak, allowing an attacker to ultimately factor a public RSA modulus N. Here, the authors recover the order of an elliptic curve modulo N, which is factored with Lenstra's ECM to ultimately factor N and break the cryptosystem. With a twisting attack, instead, the knowledge of the curve order would suffice to factor N, allowing the attacks by [32] to remain feasible even with bigger key-sizes where the ECM approach becomes unpractical.

10.2 Curve Order Leakage Exploitation

Even though the methods of Subsection 10.1 will be used in Subsection 10.4 to quickly factor Reble's semiprimality certificate [21], much better asymptotic alternatives to the Baby-step Giant-step approach are possible for recovering the curve order.

We note, indeed, that a semiprimality certificate (N, E, Q, s) leaks $\frac{11}{12}$ of the bits of the order of the curve $E(\mathbb{Z}_N)$. This immediately follows from Hasse's bound [38, V.I - Theorem 1.1]: by denoting with $\#E(\mathbb{Z}_N)$ the order of $E(\mathbb{Z}_N) \simeq E(\mathbb{F}_p) \times E(\mathbb{F}_q)$, we have that

$$|\#E(\mathbb{Z}_N) - N - 4\sqrt{N} - p - q - 1| \leq 2p\sqrt{q} + 2q\sqrt{p} + 2(\sqrt{p} + \sqrt{q})$$

Since $s \approx \sqrt[3]{N}$, and s^2 divides $\#E(\mathbb{Z}_N)$ by construction, we obtain

$$\left| \frac{\#E(\mathbb{Z}_N)}{s^2} - \frac{N - 4\sqrt{N}}{s^2} \right| \leq \frac{2p\sqrt{q} + 2q\sqrt{p}}{s^2} \approx \sqrt[12]{N}$$

Thus, using the above notation, we have $k_p k_q = \frac{N-4\sqrt{N}}{s^2} + t$ with $|t| \approx \sqrt[12]{N}$.

In this setting, such t can be found, to the best of our knowledge, either memory-less using a brute-forcing approach or with a time-space trade-off using Baby-step Giant-Step. In the case of brute-force, each guess of t is verified by picking multiple points $P \in E(\mathbb{Z}_N)$ and checking if $\left[N - 4\sqrt{N} + s^2 \cdot t\right] \cdot P = \mathcal{O} \in E(\mathbb{Z}_N)$. To use Baby-step Giant-step, instead, we set $R = [4\sqrt{N} - N] \cdot P$ and $Q = [s^2] \cdot P$, constrained by the relation $[t] \cdot Q = R$, and we then search for the discrete-logarithm of R in base Q. Once t is correctly retrieved, we can compute the full order of $E(\mathbb{Z}_N)$ and factor N, similarly as we do in Subsection 10.1.

It follows that the average time complexity of brute-force consists of $\exp\left(\frac{\ln N}{12}\right)$ guesses for t, while the Baby-step Giant-step would require $\exp\left(\frac{\ln N}{24}\right)$ time given $\exp\left(\frac{\ln N}{24}\right)$ space.

10.3 Comparison to Factorization

How do these two attacks compare with respect to directly factoring N? The fastest known general-purpose factoring algorithm is the General Number Field Sieve (GNFS) [29] which allows to factor an integer N with complexity

$$\exp\left(\left(\sqrt[3]{\frac{64}{9}} + o(1)\right)(\ln N)^{\frac{1}{3}}(\ln \ln N)^{\frac{2}{3}}\right)$$

A semiprimality certificate $(N, E(\mathbb{Z}_N), Q, s)$ generated as in Subsection 8.2 or Section 9, will have $s \approx \frac{\ln N}{3}$.

The Baby-step Giant-step attack outlined in Subsection 10.1 would require, on average, $\exp\left(\frac{\ln N}{6}\right)$ time (assuming constant table look-ups) to find the order of the point $[s] \cdot P$, where P is randomly picked from $E(\mathbb{Z}_N)$. Instead, the curve-order finding method of Subsection 10.2 which exploits the leakage provided by the certificate requires a time-space trade-off of $\exp\left(\frac{\ln N}{12}\right)$.

It follows that factoring N using the GNFS factorization algorithm is easier in terms of asymptotic time complexity as long as $N > \approx 2^{3550}$ (under the unrealistic assumption to have access to $\approx 2^{148}$ memory units with constant-time look-ups). Thus, at least with respect to the two attacks outlined above, the structure induced by our semiprimality certificate generation methods does not seem to decrease the bit-security of returned semiprimes when these are of cryptographic size.

10.4 Cryptanalysis of Reble's Semiprimality Certificate

As we already briefly discussed in Subsection 8.2, in 2005 Don Reble published a semiprimality certificate for a 1084-digits integer N [21]. His certificate consists of a tuple (N, A, s, t, Z) where: i) s is a prime greater than $(\sqrt[6]{N}+1)^2$; ii) t is an integer of size approximately $\sqrt[3]{N}$; iii) $A \in \mathbb{Z}_N$ and $E(\mathbb{Z}_N) : y^2 = x^3 + A \cdot x$; iv) $Z \in E(\mathbb{Z}_N)$ is so that $[2st] \cdot Z = \mathcal{O} \in E(\mathbb{Z}_N)$; v) $[2t] \cdot Z$ and $[st] \cdot Z$ are both not equal to $\mathcal{O}$.

It follows that $Q = [2t] \cdot Z$ is such that $[s] \cdot Q = \mathcal{O}$, and thus N is semiprime. For completeness, we report the relevant values of such certificate:

$$
\begin{aligned}
N = {} & 2354024638195369096484615970884339820364456147543619758078791036601683 \\
& 5054945730773425013209134823767913836517543105239536719778522630598306 0 \\
& 9131177801517018473419551782083685970482736851488324669809678264241294 2 \\
& 6918703837554365381987202581644055207294439281283465989299148386103333 1 \\
& 1926647139217361844392966569416841949144589355450831214521115967827260 9 \\
& 6361025012304288075013742142879482094899227940491745687352778980891233 2 \\
& 8514098559487995775109530064742516289155842487937324115166995479992403 8 \\
& 4456822944006777458824969177192912226967635528307876492581854466547687 5 \\
& 5654506771225332408011916919938505370692668148114213031389780777114780 1 \\
& 7700487114651351601764737051295848373315140397997090803794315079856953 5 \\
& 4618849164417252142747095137525007700363418270382794214457630912235836 9 \\
& 4564915884274677107758842804083947544941515945116919834042566389996135 6 \\
& 7014727022803472837915664138948795302353410201541057681970330841514731 7 \\
& 9374226307186150307934745502893756679402305608524968489170554185641700 2 \\
& 5029457397529187683879234264025678162912225114657588289449733450131843 6 \\
& 3023296235457948241
\end{aligned}
$$

$$
\begin{aligned}
A = {} & 5421560585366176057536517656540827541348280606463315054449398318430183 3 \\
& 6351362100165987609387223118116471828178358048136250081057773954633314 4 \\
& 4356348914569702071674858855538156212752894234186869653495014172927640 2 \\
& 5522504282905939725077989940637806955237317867431515138191012067767092 8 \\
& 1022382132576840497457739776448272069645008166677142960459788718162659 4 \\
& 3662227970282066248326568102180300234905324601952927033377106949538307 1 \\
& 7161684810215998702451351843048244252118076821703304366830007840621982 0 \\
& 8430854404238247676742946705423758189631678146072276526459754971947491 6 \\
& 8363368466966961270309831192271695961672667120555322877445589836857374 9 \\
& 4094413400553867690822718050450411564283982459498248178309690662041620 2 \\
& 3068382112282498719629195618182388184559051503256367664066579952982875 6 \\
& 7195324927191929693117348419003384059344367533130406039881394710806359 5 \\
& 7294738052140582500620095277455640448894314742558729947025206994334790 1 \\
& 3172787548986648323838337027155387658865693901039506782734835488084522 2 \\
& 9757153752432699362340557028635941494226068304457007072144813757748466 1 \\
& 026699816478612597
\end{aligned}
$$

$$s = 28928434216210803690615246524734063145625591441015675162985567756555278$$
$$47475254643269219998751288122386460675073169360611359943088760646061779$$
$$63915153101486156446026954889034721931204780489345443377835050099341337$$
$$67946616448576209133289712559206785258051616195330150089251887798277052$$
$$46804480305755386032561335776618914909668933096505500715518252514951552$$
$$5889989$$

$$t = 70323610117989232939059249931645122359843269434492413277191595877834782$$
$$37835918161785248366689469809564684201210711877500244334224388017710878$$
$$52206517746924932705142830765836952506541400693749501498564556129745067$$
$$76852952721623063218610555085789467640877803479093810669778258714223797$$
$$26559797542141718817576541825421567562917573366046873030803485890583206$$
$$21801$$

Unfortunately, this certificate is vulnerable to the twisting attack outlined in Subsection 10.1, since $k = 4 \cdot t$, due to its size, seems to correspond to $|E(\mathbb{Z}_N)|/s^2$. In fact, if we attempt to compute $[k \cdot s] \cdot P$ for some random point P on the curve twist $\in \tilde{E}(\mathbb{Z}_n)$ given by a c so that $\left(\frac{c}{N}\right) = -1$, we quickly obtain the factors p, q of N as:

$$p = 11720990044176604485656482285369732810060846386219272085536405872537609$$
$$48011511585395454559143032247641663618272332076279747485555807859184694$$
$$06861654147386441510559363392715817728672245699385222089747094454406981$$
$$21693586713136011742194739741899723138624842602143272810739524721078379$$
$$48499063449221486753727501531543239917595871282902624150630909694396150$$
$$92699624981254513801117217086814181188742680766873054766809210661150719$$
$$42491901610394578884716528402831014158285785393084402661789426579443047$$
$$02370750987373948019507352285634402921138312 61$$

$$q = 20083837878225401365247454652399288511943664897737051292330162045380798$$
$$47047967553624738053694816692363354958717219555398238278456569714076351$$
$$99873386941295582613528886860458159891305742279812772280504959629204641$$
$$12804540856578175224873773598186424150500091924032169570267927231676027$$
$$46296336758526213630623424953236880409237682617595943424384640820844771$$
$$70767532502558171140259680357885849656918196880508244291275258005435663$$
$$46284978929191779811241094865466976201603650414220042746338725436438268$$
$$06034913779684361274029326305796599198109018 1$$

By computing Cornacchia's decomposition for p, q with respect to $D = -4$ (since E has j-invariant equal to 1728) we can quickly identify, using Theorem 3, the correct orders of $Z \in E(\mathbb{F}_p)$ and $Z \in E(\mathbb{F}_q)$, and thus further factor t as $t = t_p \cdot t_q$ with

$$t_p = 20258597400353665832991488672810613896938092755725221711196244287139044$$
$$03793176567743497667059136838707693549460158024833330378138966164902722$$
$$91024111634551302604335595267654027056 9$$

$$t_q = 34712970857874668936188691097154723180647484712027869327858850223000688$$
$$837691708183343393107050884105199661059646641809694147278828721532212756$$
$$323691882175782629768021563640343372 9$$

11 Conclusions

In this paper we described a technique to construct primes p for which we can explicitly construct an elliptic curve $E(\mathbb{F}_p)$ such that its order completely factors over a certain factor base. We showed how this method could be used to backdoor one or more factors of an integer N so that an attacker can quickly factor it. To show the practicality of our methods, we implemented our procedures, and we detailed a complete attack example on a 1024-bits RSA modulus, previously backdoored with our implementation and factored in just a few seconds.

We then formalized semiprimality certificates that, based on a result by Goldwasser and Kilian, allow proving semiprimality of an integer with no need to share any of its factors, and we discussed how such certificates can be possibly computed. We described how our prime backdooring procedure could be used to construct prime factors of a semiprime N so that it is easy to compute semiprimality certificates for it, and we ported this construction in the MPC setting by sketching a protocol that allows distributed generation of certifiable semiprimes.

Lastly, we analyzed the security of semiprimality certificates, and we provided different attacks. We concluded that, compared to generic factorization algorithms, the expected bit-security of semiprimes built according to our protocols is the same as the security provided by random semiprimes of the same size.

12 Acknowledgements

The authors are grateful to Aleksei Udovenko, Alex Biryukov, Alice Silverberg, Cyprien Delpech de Saint Guilhem, Dan Boneh, Nadia Heninger and Younes Talibi Alaoui for the valuable discussions, their insights and meaningful suggestions during the writing of the first draft of this paper.

References

1. Aikawa, Y., Nuida, K., Shirase, M.: Elliptic curve method using complex multiplication method. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **102**(1), 74–80 (2019). https://doi.org/10.1587/transfun.E102.A.74 3, 11
2. Atkin, A.O.L.: The number of points on an elliptic curve modulo a prime (i). Draft (1988) 18
3. Atkin, A.O.L.: The number of points on an elliptic curve modulo a prime (ii). Draft (1999) 18
4. Atkin, A.O.L., Morain, F.: Elliptic curves and primality proving. Mathematics of computation **61**(203), 29–68 (1993). https://doi.org/10.1090/S0025-5718-1993-1199989-X 18
5. Basilla, J.M.: On the solution of $x^2 + dy^2 = m$. Proceedings of the Japan Academy, Series A, Mathematical Sciences **80**(5), 40 – 41 (2004). https://doi.org/10.3792/pjaa.80.40 6
6. Belding, J., Bröker, R.M., Enge, A., Lauter, K.: Computing hilbert class polynomials. In: van der Poorten, A.J., Stein, A. (eds.) Algorithmic Number Theory. pp. 282–295. Springer Berlin Heidelberg, Berlin, Heidelberg (2008). https://doi.org/10.1007/978-3-540-79456-1_19 9

7. Bernstein, D.J., Lange, T.: XZ coordinates for short Weierstrass curves. https://hyperelliptic.org/EFD/g1p/auto-shortw-xz.html 9
8. Boneh, D., Franklin, M.K.: Efficient generation of shared RSA keys (extended abstract). In: Kaliski Jr., B.S. (ed.) CRYPTO'97. LNCS, vol. 1294, pp. 425–439. Springer, Heidelberg (Aug 1997). https://doi.org/10.1007/BFb0052253 2
9. Broker, R.M.: Constructing elliptic curves of prescribed order. Ph.D. thesis (2006), https://www.math.leidenuniv.nl/scripties/Broker.pdf 5
10. Chen, M., Hazay, C., Ishai, Y., Kashnikov, Y., Micciancio, D., Riviere, T., Shelat, A., Venkitasubramaniam, M., Wang, R.: Diogenes: Lightweight scalable rsa modulus generation with a dishonest majority. In: 2021 2021 IEEE Symposium on Security and Privacy (SP). pp. 590–607. IEEE Computer Society, Los Alamitos, CA, USA (5 2021). https://doi.org/10.1109/SP40001.2021.00025 2
11. Chen, M., Cohen, R., Doerner, J., Kondi, Y., Lee, E., Rosefield, S., shelat, a.: Multiparty generation of an RSA modulus. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 64–93. Springer, Heidelberg (Aug 2020). https://doi.org/10.1007/978-3-030-56877-1_3 2
12. Cheng, Q.: A new class of unsafe primes. Cryptology ePrint Archive, Report 2002/109 (2002), https://eprint.iacr.org/2002/109 3
13. Cheng, Q.: A new special-purpose factorization algorithm (2002) 3
14. Cohen, H.: A Course in Computational Algebraic Number Theory, Graduate Texts in Mathematics Ser., vol. 138. Springer Berlin / Heidelberg (2000). https://doi.org/10.1007/978-3-662-02945-9 4, 6
15. Coppersmith, D.: Finding a small root of a univariate modular equation. In: Maurer, U.M. (ed.) EUROCRYPT'96. LNCS, vol. 1070, pp. 155–165. Springer, Heidelberg (May 1996). https://doi.org/10.1007/3-540-68339-9_14 2
16. Cox, D.A.: Primes of the Form $x^2 + ny^2$: Fermat, Class Field Theory, and Complex Multiplication: Second Edition (01 2013). https://doi.org/10.1002/9781118400722 9
17. Damgård, I., Keller, M., Larraia, E., Pastro, V., Scholl, P., Smart, N.P.: Practical covertly secure MPC for dishonest majority - or: Breaking the SPDZ limits. In: Crampton, J., Jajodia, S., Mayes, K. (eds.) ESORICS 2013. LNCS, vol. 8134, pp. 1–18. Springer, Heidelberg (Sep 2013). https://doi.org/10.1007/978-3-642-40203-6_1 21
18. Damgård, I., Pastro, V., Smart, N.P., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 643–662. Springer, Heidelberg (Aug 2012). https://doi.org/10.1007/978-3-642-32009-5_38 21
19. David Broadhurst: A 5061-digit semiprime (2005), https://web.archive.org/web/20190827000752/http://physics.open.ac.uk/~dbroadhu/cert/semgpch.gp 3, 18, 19
20. Demytko, N.: A new elliptic curve based analogue of RSA. In: Helleseth, T. (ed.) EUROCRYPT'93. LNCS, vol. 765, pp. 40–49. Springer, Heidelberg (May 1994). https://doi.org/10.1007/3-540-48285-7_4 23
21. Don Reble: Interesting Semiprimes (2005), http://www.graysage.com/djr/isp.txt 3, 18, 19, 23, 25
22. Dudley, U.: Elementary Number Theory: Second Edition. Dover Books on Mathematics, Dover Publications (1978) 5, 7
23. Elkies, N.D.: Elliptic and modular curves over finite fields and related computational issues. In: Buell, D.A., J. T. Teitelbaum, e. (eds.) Computational Perspectives on Number Theory: Proceedings of a Conference in Honor of A. O. L. Atkin. vol. 7, pp. 21–76. American Mathematical Society (1998). https://doi.org/10.1090/amsip/007/03 18
24. Falk, B.H., Noble, D.: Secure computation over lattices and elliptic curves. Cryptology ePrint Archive, Report 2020/926 (2020), https://eprint.iacr.org/2020/926 20

25. Goldwasser, S., Kilian, J.: Almost all primes can be quickly certified. In: Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing. p. 316–329. STOC '86, Association for Computing Machinery, New York, NY, USA (1986). https://doi.org/10.1145/12130.12162 2, 17, 18
26. Izu, T., Takagi, T.: Fast elliptic curve multiplications resistant against side channel attacks. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **88-A**(1), 161–171 (2005). https://doi.org/10.1093/ietfec/E88-A.1.161 13
27. Keller, M.: MP-SPDZ: A versatile framework for multi-party computation. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (2020). https://doi.org/10.1145/3372297.3417872 21
28. Kunihiro, N., Koyama, K.: Two discrete log algorithms for super-anomalous elliptic curves and their applications. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **83**, 10–16 (2000) 23
29. Lenstra, A.K., Lenstra, H.W.: The Development of the Number Field Sieve. Lecture Notes in Mathematics, Springer (1993). https://doi.org/10.1007/BFb0091534 24
30. Lenstra, H.W.: Factoring integers with elliptic curves. Annals of Mathematics **126**(3), 649–673 (1987). https://doi.org/10.2307/1971363 2, 7
31. Montgomery, P.L.: Speeding the Pollard and elliptic curve methods of factorization. Mathematics of Computation **48**, 243–264 (1987). https://doi.org/10.1090/S0025-5718-1987-0866113-7 13
32. Nitaj, A., Fouotsa, E.: A new attack on rsa and demytko's elliptic curve cryptosystem. Journal of Discrete Mathematical Sciences and Cryptography **22**(3), 391–409 (2019). https://doi.org/10.1080/09720529.2019.1587827 23
33. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the Association for Computing Machinery **21**(2), 120–126 (1978). https://doi.org/10.1145/359340.359342 2
34. Delpech de Saint Guilhem, C., Makri, E., Rotaru, D., Tanguy, T.: The return of eratosthenes: Secure generation of rsa moduli using distributed sieving. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. p. 594–609. CCS '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3460120.3484754 2
35. Schoof, R.: Counting points on elliptic curves over finite fields. Journal de Théorie des Nombres de Bordeaux **7**(1), 219–254 (1995). https://doi.org/doi.org/10.5802/jtnb.142 18
36. Sedlacek, V., Klinec, D., Sys, M., Svenda, P., Matyas., V.: I want to break square-free: The $4p-1$ factorization method and its rsa backdoor viability. In: Proceedings of the 16th International Joint Conference on e-Business and Telecommunications (ICETE 2019) - Volume 2: SECRYPT,. pp. 25–36. INSTICC, SciTePress (2019). https://doi.org/10.5220/0007786600250036 3
37. Shirase, M.: Condition on composite numbers easily factored with elliptic curve method. Cryptology ePrint Archive, Report 2017/403 (2017), `https://eprint.iacr.org/2017/403` 3
38. Silverman, J.: The Arithmetic of Elliptic Curves, vol. 106 (1 2009). https://doi.org/10.1007/978-0-387-09494-6 4, 24
39. Smart, N.P., Talibi Alaoui, Y.: Distributing any elliptic curve based protocol. In: Albrecht, M. (ed.) 17th IMA International Conference on Cryptography and Coding. LNCS, vol. 11929, pp. 342–366. Springer, Heidelberg (Dec 2019). https://doi.org/10.1007/978-3-030-35199-1_17 20
40. Sutherland, A.V.: Computing hilbert class polynomials with the chinese remainder theorem. Mathematics of Computation **80**(273), 501–538 (2011). https://doi.org/10.1090/S0025-5718-2010-02373-7 9
41. The Sage Developers: SageMath, the Sage Mathematics Software System (Version 9.4) (2021), `https://www.sagemath.org` 13, 21
42. Washington, L.C.: Elliptic Curves: Number Theory and Cryptography, Second Edition. Chapman and Hall/CRC, 2 edn. (2008) 6, 17