Weak instances of composite order protocols

Sorina Ionica[1] and Malika Izabachène[2,3]

[1] Microsoft Research**
[2] sorina.ionica@m4x.org
[3] EPF, France
[4] Loria-CNRS, France
malika.izabachene@epf.fr

Abstract. In pairing-based cryptography, the security of protocols using composite order groups relies on the difficulty of factoring a composite number N. Boneh *et al* proposed the Cocks-Pinch method to construct ordinary pairing-friendly elliptic curves having a subgroup of composite order N. Displaying such a curve as a public parameter implies revealing a square root of the complex multiplication discriminant $-D$ modulo N. We exploit this information leak and the structure of the endomorphism ring of the curve to factor the RSA modulus, by computing a square root λ of $-D$ modulo one of its factors. Our attack is based on a generic discrete logarithm algorithm. We recommend that λ should be chosen as a high entropy input parameter when running the Cocks-Pinch algorithm, in order to ensure protection from our attack.

Keywords : composite order group, integer factorization, elliptic curve, endomorphism, Coppersmith's algorithm

1 Introduction

Bilinear groups of composite order have been used as a convenient tool to provide plenty of cryptosystems with advanced functionalities, such as zero-knowledge proof systems [15], group signatures in the standard model [5], traitor tracing with full traceability [8], functional encryption with full security [23, 2] and attribute-based encryption with efficient revocable storage [1].

In the composite order setting, the abelian group G can be expressed as a direct product of m prime order subgroups G_i for $i \in [1, m]$. Assuming the order of the group is hard to factor, such decomposition is non trivial. Since we rely on the hardness of factoring, we need to use large moduli for implementing these protocols. This yields too costly group operations and pairing computation. Freeman [11] has investigated the possibility

** This work was carried out in part while this author was working at the Ecole Normale Supérieure.

to generically convert composite order schemes into primes order ones and has identified two properties of composite order constructions, called *cancelling* and *projecting* shown to be also achievable in the prime order setting. In this work, Freeman provides several examples of protocols that could be transformed, but the conversion does not work mechanically; description and proofs have to be reworked for each scheme. Several papers [22, 30, 29] have further explored the possibility to build a general framework for such conversion. Although a fully generic transformation would be desirable in many aspects, composite order groups remain a very convenient tool to implement more efficient schemes with new functionalities. In addition, they allow to introduce nice abstractions such as cancelling, projecting and parameter hiding which provide a clear understanding of the underlying algebraic structure. It is thus essential to find efficient ways to implement pairings on composite order groups securely.

In order to construct pairing-based schemes with composite order groups, we can either implement the pairing using supersingular curves as in the original construction in [9] or use ordinary curves as suggested by Boneh *et al* [4]. These are particularly recommended if the security relies on the Decisional Diffie-Hellman (DDH) assumption. Moreover, recent results [12, 20, 32, 10] show that the discrete logarithm problem on supersingular curves is weak. In particular, recent announcements (see [14] for instance) show that pairings of Type 1 as defined in [28] are no longer safe. This is a first reason for constructing pairing-friendly ordinary curves. Secondly, note that the use of ordinary curves for implementing composite order group protocols is particularly meaningful if we want to compare performance analysis between composite order schemes and their prime order analogues in Freeman's framework [11].

In this paper, we investigate the security of pairing-based composite order group protocols that use ordinary curves. We identify security weaknesses of instantiations of pairing-based protocols in the composite order setting. Our analysis departs from the complex multiplication (CM) construction of pairing friendly elliptic curves whose number of points is divisible by the product of several large primes. In order to construct pairing over composite order groups from such curves, the Cocks-Pinch method can then be employed [6]. Boneh, Rubin and Silverberg [4] point several security issues that occur in this setting. Following their work, we provide a discrete log based type attack when implementing the protocol using this method. We take the Boneh-Goh-Nissim encryption scheme as master study case.

Description of our attack. Boneh *et al* [4] noticed that when displaying a public ordinary curve, whose number of points divisible by N can be counted in polynomial time, one may easily compute a square root s of the CM discriminant $-D \bmod N$. Assuming that a generator P_i of one of the subgroup G_i is given as a public value, we show how we can recover λ such that $\lambda \equiv s \pmod{p_i}$ with $\lambda < p_i$. To do so, we use an efficiently computable endomorphism ϕ and the fact that an element of order p_i is publicly known. We thus have the equation $\phi(P_1) = \lambda P_1$ with $0 \leq \lambda < p_1$.

We first show how to recover λ in the case where $\lambda \approx \lfloor\sqrt{p_i}\rfloor$. We consider two approaches: the first one consists in viewing the previous equation as a monic polynomial equation of degree one whose root is λ. We then apply Coppersmith's techniques for finding small roots of modular polynomial equation. The second one consists in running the kangoroo method in an interval of small size depending on the desired security level. Once λ is found, we can recover p_i by computing $p_i = \gcd(N, s \pm \lambda)$.

While the case $\lambda \approx \lfloor\sqrt{p_i}\rfloor$ arises with negligible probability, we build on the treatment of these low density instances in order to attack factorization when λ is arbitrary large (i.e. $0 \leq \lambda < p_i$). The idea is to write λ as $x + \lfloor\sqrt{p_i}\rfloor y$ where x and y are of size bounded by $\lfloor\sqrt{p_i}\rfloor$. We then combine the two previous methodologies as follow: we will use a Pollard-rho like algorithm to find x, while running Coppersmith's algorithm to recover y.

Organisation. This paper is organised as follows. Section 1 specifies the attack setting and makes it explicit using the BGN encryption scheme. Section 2 reviews the Cocks-Pinch method and its generalization to the case of a composite modulus. Section 3 reviews some backgrounds on Coppersmith techniques to find small roots of polynomials. Section 4 identifies weak instances and show how to recover part of the factorization of N. Section 5 extends the methodologies of the attack on low density instances to any instances. Section 6 gives the complexity of our attack and compares it to the best known algorithm for factoring.

2 Backgrounds on pairings and composite order protocols

2.1 Composite order schemes and setting of the attack

In this section, we specify the general frame of our attack and review the schemes which work in this setting. We focus on schemes using a pairing

$$e : G \times H \mapsto G_T \tag{1}$$

and such that the order of the groups G, H, G_T is a composite modulus $N = p_1 \dots p_m$, where $p_1, \dots, p_m$ are primes. We denote by G_i (resp. H_i) the subgroup of G (resp. H) of order p_i. We define by $\mathcal{G}$ a generator for a composite order bilinear structure which on input the security parameter τ outputs $(p_1, \dots, p_m, G, H, G_T, e)$. The system that fits to the attack setting is any composite order scheme implemented using an ordinary curve for which:

1. the number of points is known and divisible by a large composite public modulus N.
2. an element of one of the subgroups of G (or H) is displayed as public information when implementing the scheme.

To simplify, we focus on systems implemented using a pairing generator for $m = 2$ and describe the BGN encryption scheme [9] using asymmetric pairings. Our attack works as well for certain instances when $m > 2$. This will be further explained in Section 6.

The security of the BGN encryption scheme relies on the Subgroup Decision problem [9], which is a particular instance of the General Subgroup Decision family of assumptions introduced in [24]. Recall that this assumption has been proven valid in the generic group model assuming that N is a hard to factor modulus (see [18]). It is worth to precise that our attack does not assume that the General Subgroup Decision problem is easy, but uses the fact that an element of one of the subgroups of G or H can be displayed when running the protocol.

When implementing the Boneh, Goh, Nissim encryption scheme [9] using asymmetric pairings, the public key consists in $(N = p_1 p_2, G \cong G_1 \times G_2, H \cong H_1 \times H_2, G_T, P, P_1, Q, Q_1)$, where G_1 and H_1 (resp. G_2 and H_2) are both of order p_1 (resp. p_2) and P, P_1 and Q, Q_1 are random generators of G, G_1 and H, H_1 respectively. To encrypt a message m, one computes $(mP + rP_1, mQ + sQ_1)$, for random scalars $r, s \in \mathbb{Z}_N$. One can thus get for free an element of H_1 from the public key.

2.2 The Cocks-Pinch method

Pairings on elliptic curves, i.e. the Weil and the Tate pairing map to the multiplicative group of an extension field of $\mathbb{F}_q$. This extension field needs to contain the N-th roots of unity. We denote by k the embedding degree with respect to N, i.e. the smallest integer such that the extension field $\mathbb{F}_{q^k}$ contains the N-th roots of unity. Pairings are usually computed by using Miller's algorithm, whose efficiency depends on an efficient arithmetic for

$\mathbb{F}_{q^k}$. Therefore, we use elliptic curves for which the embedding degree k is small.

Writing the Frobenius endomorphism π as an element of the endomorphism ring leads to the following condition

$$\exists\, y \in \mathbb{Z} \quad 4q = t^2 + y^2 D \tag{2}$$

If this condition is satisfied, the CM method outputs a curve whose number of points is $\#E(\mathbb{F}_q) = q+1-t$. Cocks and Pinch [17] proposed an algorithm which, given a prime number N and an integer k, constructs a pairing friendly elliptic curve such that $N|\#E(\mathbb{F}_q)$ and whose embedding degree with respect to N is k. This method was extended by Boneh, Rubin and Silverberg [4] to the case where N is a composite number. We briefly recall the method in Algorithm 1.

Algorithm 1 The Cocks-Pinch algorithm

Require: k, r prime numbers p_i, D and $N|(q^k-1)$, $N = p_1^{\alpha_1} \dots p_r^{\alpha_r}$, and $-D$ a square mod N.

Ensure: q, t such that there is a curve with CM by $-D$ over $\mathbb{F}_q$ with $q+1-t$ points where $N|(q+1-t)$ and $N|(q^k-1)$.

1: Choose an integer X which is a primitive k-th root of unity modulo every $p_i^{\alpha_i}$.
2: Choose any s $\pmod N$ such that $s^2 = -D \pmod N$.
3: Take an integer Y congruent to $\pm(X-1)s^{-1} \pmod N$.
4: **repeat**
5: $\quad q \leftarrow ((X+1)^2 + D(Y+jN)^2)/4$
6: $\quad j \leftarrow j+1$
7: **until** q is prime
8: **return** q and $t = X+1$

One may use the Chinese Remainder Theorem to obtain X of order k modulo all the $p_i^{\alpha_i}$'s, $i \in \{1, \dots, r\}$, in Step 1. Similarly, Step 2 computes $s^2 = -D \pmod N$ by using the Chinese Remainder Theorem. Boneh et al. [4] note that if D is a divisor of k, then one may compute a square root of unity by the formula

$$s = \begin{cases} \sum_{\substack{a=1 \\ (a,2D)=1}}^{2D-1} (\frac{-D}{a}) X^{\frac{ak}{D}} & \text{if } D \equiv 3 \pmod 4, \\ \\ \frac{1}{2} \sum_{\substack{a=1 \\ (a,2D)=1}}^{4D-1} (\frac{-D}{a}) X^{\frac{ak}{4D}} & \text{otherwise} \end{cases} \tag{3}$$

2.3 Implementation and choice of subgroups on ordinary curves

In this section, we are interested in pairing implementation on ordinary curves.

On an ordinary curve, if the embedding degree k is greater than 1, it is most efficient to choose to implement the pairing in (1) by choosing

$$G = E[N] \cap \mathrm{Ker}(\pi - [1]) \text{ and } H = E[N] \cap \mathrm{Ker}(\pi - [q]) \qquad (4)$$

Indeed, it was shown that when pairing elements in these subgroups, many operations in Miller's algorithm can be done in subfields $\mathbb{F}_{q^k}$, which critically reduces the cost of the pairing computation. When implementing composite order protocols, if we need to ensure that the DDH assumption holds in both G and H, then we need to choose these subgroups such that there are no *distortion maps* for points in these subgroups. Given a point $P \in E[N]$, a distortion map for P is a map ϕ such that for a point P, $\phi(P) \notin \langle P \rangle$. The following result was given by Verheul [34], whose purpose was to investigate the existence of distortion maps for points of order N.

Theorem 1. *Let E be an ordinary curve defined over $\mathbb{F}_q$ and let P be a point over E of prime order $N \neq q$. Suppose the embedding degree k is greater than 1 and denote by Q a point defined over $\mathbb{F}_{q^k}$, such that $\pi(Q) = qQ$. Then P and Q are eigenvectors of any other endomorphism of E.*

Hence, if the security of the protocol relies on the DDH assumption, then one needs to choose G and H as in (4).

3 Coppersmith's method

In 1996, Coppersmith [7] introduced lattice-based reduction techniques to find small roots of polynomials. These techniques have been reformulated in a simplified manner [16] and also generalized to more variables and become a powerful cryptanalytic toolbox. Let $N = p_1 p_2$ be a public RSA modulus, whose factorization is secret. In this paper, we will use Coppersmith's technique to find small roots of a polynomial equation $F(t) = 0 \mod p_1$ where F is a monic polynomial of degree 1 and p_1 is unknown. Our attack relies on the following results.

Lemma 1. *[16] Let $F \in \mathbb{Z}[t]$ be a polynomial of degree d. Let $X = p_1^{h/d}$ and let b_F be the vector given by the coefficients of $F(tX)$. Assume that $F(t_0) = 0 \bmod p_1^h$ where $|t_0| \leq X$ and that $\|b_F\| \leq \frac{p_1^h}{\sqrt{d+1}}$. Then $F(t_0) = 0$ holds over the integers.*

Theorem 2. *[3] Let $N = p_1 p_2$ with $p_1 < p_2 < 2p_1$. Let $0 < \epsilon < 1/4$ and let $F(t)$ as above. Then, if t_0 is a small root of F satisfying $|t_0| \leq \frac{1}{2\sqrt{2}} N^{1/4-\epsilon}$, then one can recover p_1 in polynomial time in $\log N$ and $1/\epsilon$.*

In the following, we explain how this technique works for $F(t) = t + a$, when we want to solve $F(t) = 0 \bmod p_1$. First, we collect a set of small polynomials defined by $g_{u,v}(t) = N^{h-u} F(t)^u t^v$ where $0 \leq u \leq h$ and $v \geq 0$. The parameter h will be determined later.

We represent polynomials as row vectors and use a lattice reduction algorithm to find small vectors in a lattice. Let $k = 2h$ and $X = \lfloor N^{1/4-\epsilon} \rfloor$. We construct the $k+1$-dimensional lattice L spanned by the vectors of coefficients of the following polynomials:

- $g_{u,0}(tX)$ for $u = 0, \ldots, h$
- $g_{h,v}(tX)$ for $v = 1, \ldots, k-h$

The determinant of the lattice is then given by $\det(L) = N^{h(h+1)/2} X^{h(2h+1)}$. The condition of Lemma 1 ensures that if the first vector b_1 of the LLL reduced basis satisfies $\|b_1\| \leq p_1^h / \sqrt{k+1}$, the corresponding polynomial will have a root over $\mathbb{Z}$.

Since $\|b_1\| \leq 2^{\frac{k}{4}} \det(L)^{1/(k+1)}$, it is sufficient to have the following condition

$$2^{k/4} \det(L)^{1/k+1} < p_1^h / \sqrt{k+1}. \tag{1}$$

By using the LLL algorithm and root finding algorithms for polynomials over the integers, one can find $|t_0| \leq X$ in polynomial time in $(\log\ N, \frac{1}{\epsilon})$.

Now, by substituting the bound for X in (1), we obtain that if h satisfies $\frac{1}{(4(2h+1))} < \epsilon$, then we are guaranteed to find t_0.

4 Finding weak instances

Boneh *et al* [4] note that in the case of a RSA modulus N, a square root $s \pmod N$ is leaked during the Cocks-Pinch construction. This is a potential security concern, since computing all square roots modulo a composite is as hard as factoring. In this section, we show that, under

certain conditions, when a square root of $s \pmod N$ is revealed from q, N and E, one may use this information to factor N. As explained before, we present our attack in the case where $N = p_1 p_2$.

Let us begin by explaining how to recover s. An elliptic curve over $\mathbb{F}_q$ will have $\#E(\mathbb{F}_q) = q + 1 - t$. Since the curve is public, an attacker may use Schoof's point counting algorithm to get t. Thus he may compute Y in Step 3 of the Cocks-Pinch algorithm by factoring $t^2 - 4q$ (which is feasible for cryptographic sizes). If $(N, Y) = 1$, he will then get $s = (X - 1)Y^{-1} \pmod N$, where $X = t - 1$. Note that if $k = 1$, we have $X = 1 \pmod N$ and $Y = 0 \pmod N$. Hence s may not be recovered by displaying E.

In the remainder of this paper, the embedding degree k is greater than 1. We denote by ϕ the endomorphism whose characteristic equation is $\phi^2 + D = 0$. For $P_1 \in G_1$, we have $\phi(P_1) = \lambda_1 P_1$, with $\lambda_1 < p_1$. Note that λ_1 verifies

$$\lambda_1 = \pm s \pmod{p_1}.$$

Hence if one recovers $\lambda_1 \pmod{p_1}$, one may compute p_1 as $\lambda_1 = \gcd(N, p_1 \pm s)$.

4.1 First approach: finding small root via Coppersmith's method

If $\lambda_1 \leq N^{1/4}$, one may apply the technique presented in Section 3 to find λ_1 as a root of the polynomial $F(t) = t \pm s \pmod{p_1}$.

Algorithm 2 The attack

Require: An elliptic curve E defined over $\mathbb{F}_q$, $N | \#E(\mathbb{F}_q)$.
Ensure: A factor p_1 of N.
1: Compute $t = q + 1 - \#E(\mathbb{F}_q)$, get Y such that $Y^2 D = t^2 - 4q$.
2: Compute $s = (t - 2)Y^{-1}$.
3: Use Coppersmith's algorithm to find λ_1 as a small root of the polynomial $F(t) = t - s \pmod{p_1}$.
4: Compute $p_1 \leftarrow \gcd(N, s \pm \lambda_1)$.

We give a toy example computed with MAGMA.

Example 1. Let $p_1 = 1073741827$ and $p_2 = 1074790447$. We take $N = p_1 p_2$ and $D = 3$. With $\lambda_1 \equiv 32768 \pmod{p_1}$ and $\lambda_2 \equiv 547381745 \pmod{p_2}$, the CRT theorem gives

$$s = 7943732666174021566464 \pmod N.$$

Using the Cocks-Pinch method, we obtain the following curve with embedding degree 2 with respect to N:

$$y^2 = x^3 + 13 \text{ defined over } \mathbb{F}_q,$$

with $q = 1140730183325927132841992508979589859787$. Our implementation of Coppersmith's method using the fplll library [31] for lattice reduction recovers λ_1 as a square root of the polynomial $F(t) = t - 7943732666174021566464 \pmod{p}$ in approximatively 2 seconds, on a Intel Core i3-3227U at 1.90 GHz.

4.2 Second approach: discrete logs in an interval of small size

Note that if D is small, the endomorphism ϕ can be computed very efficiently, by using Vélu's formulae [33]. Details on this computation and its complexity are given in Section 6. At this point, we assume that if P_1 is public, the attacker may easily compute $\phi(P_1)$. Obviously, p_1 is not known to the attacker, but its size $2^{f(\tau)}$ depends on the desired level of security of the cryptosystem. The attacker may then run the kangoroo method in the interval $[2^{\frac{f(\tau)}{2}}, 2^{\frac{f(\tau)}{2}} + w]$ in order to compute λ_1. The size w of the weak interval depends on the security level τ.

Algorithm 3 The attack

Require: An elliptic curve E defined over $\mathbb{F}_q$, $N | \#E(\mathbb{F}_q)$, (G, G_1, P, P_1).
Ensure: A factor p_1 of N.
1: Compute $t = q + 1 - \#E(\mathbb{F}_q)$, get Y such that $Y^2 D = t^2 - 4q$.
2: Compute $s = (t-2)Y^{-1}$.
3: Compute $\phi(P_1) = \lambda_1 P_1$.
4: Apply the kangoroo method in the interval $\left[2^{\frac{f(\tau)}{2}}, 2^{\frac{f(\tau)}{2}} + w\right]$ and compute λ_1.
5: Compute $p_1 \leftarrow \gcd(N, s \pm \lambda_1)$.

We consider again the curve given in Example 1.

Example 2. The endomorphism ϕ corresponding to $\sqrt{-3}$ is given by the equation

$$\phi(x, y) = \left(\frac{x^3 + 52}{x^2}, \frac{x^3 y + 1140730183325927132841992508979589859683 y}{x^3} \right).$$

Given a point P_1 in a subgroup of order p_1, the attacker may then compute $\phi(P_1)$ with a few operation in the finite field $\mathbb{F}_q$. Using the kangoroo algorithm, the attacker finds $\lambda_1 = 32768$ as the discrete log of $\phi(P_1)$ with respect to P_1.

Remark 1. As explained in Section 2.2, if D is a divisor of k, then a square root s of $-D \pmod N$ is given by equation (3). This value should obviously be used when running the Cocks-Pinch algorithm. Note that s is a sum of a small number of powers of X. If $X \pmod{p_1}$ is small, then $s \pmod{p_1}$ may also be small. Hence the output of the Cocks-Pinch algorithm will be a curve vulnerable to our attack.

Example 3. [4, Example 6.2] Assume $k = D = 3$ and $p_1 = p_2 = 1 \pmod 3$. Then one must choose X such that $X^2+X+1 \equiv 0 \pmod N$. Assume $X \bmod p_1$ is small. Using equation (3), one may compute $s \equiv 2X+1 \pmod N$, which verifies $s^2 \equiv -3 \pmod N$. Hence $\lambda_1 \equiv s \bmod p_1$ is small and the Cocks-Pinch construction using these parameters will be vulnerable to our attack.

More weak instances using endomorphisms

Let $D \equiv 3 \bmod 4$. Consider $\tilde{\phi}$ the endomorphism of equation

$$X^2 + X + \frac{1+D}{4} = 0 \tag{5}$$

The solutions of this equation are $\frac{-1\pm\sqrt{-D}}{2}$. This implies that if s is such that $s^2 \equiv -D \bmod N$, then $2^{-1}(-1+s) \bmod N$ is a solution of equation (5). Consider $\lambda_1 \equiv 2^{-1}(-1+s) \bmod p_1$. We have $\tilde{\phi}(P_1) = \lambda_1 P_1$. Obviously, λ_1 verifies

$$\lambda_1^2 + \lambda_1 + \frac{1+D}{4} = zp_1,$$

with $z \in \mathbb{Z}$. If z is small (i.e. a couple of digits), then $\lambda_1 \approx \sqrt{p_1}$ and the attacker will recover it by running the Pollard kangoroo method in the interval $\left[2^{\frac{f(\tau)}{2}}, 2^{\frac{f(\tau)}{2}} + w\right]$, where $f(\tau)$ is a function depending on the level of security of the cryptosystem.

5 Extending the attack to large values of λ_1

In this section, we show that by combining the two techniques presented in Section 4, one may recover λ_1, when this is arbitrarily large.

The setting is the following: we are given an asymmetric pairing group structure (G, H, G_T, P, Q) where P and Q have order N such that $N = p_1p_2$ is an RSA modulus and divides $\#E(\mathbb{F}_q)$. We assume that $E(\mathbb{F}_q)$ is constructed using Algorithm 1 and that its embedding with respect

to N is k. We further assume that s can be recovered as explained in Section 4. Note that although the construction of $E(\mathbb{F}_q)$ uses the fact that the factorization of N is known, we assume that when encrypting, the sender does not know the factorization of N. Let P_1 be an element of order p_1 which is part of the public key.

Now, assume that $p_1 < p_2$ and hence that $p_1 < N^{\frac{1}{2}}$. Let $w = \lfloor N^{\frac{1}{4}} \rfloor$ and write $\lambda_1 = x + \lfloor N^{\frac{1}{4}} \rfloor y$, with $0 \leq x, y < w$. Note that x and y are unique with this property.

Our main idea is the following:

1. we first search for $x \in [0, w]$ using a variant of the Pollard-rho algorithm for an interval [27]. Assuming the departure point is known, we define a pseudo-random walk in the interval, which is similar to the pseudo-random walk in the Pollard-rho algorithm. We will call this a kangoroo walk and describe it below. In order to find x, we use the kangoroo walk in a Floyd cycle finding algorithm. More precisely, we consider two kangoroos, both starting at P_1. At each step, we denote by u and v the distance from the starting point for the first and the second kangoroo, respectively. Both kangoroos follow the same path, with the only difference that the second one is twice faster and takes two steps at a time.
2. We set $y' = |u - v| \in [0, w]$. At each iteration, a collision takes places if $y = y'$. Note that if $y = y'$, then there exists x such that $|x| \leq w$ and $x + y' \lfloor \sqrt{p_1} \rfloor - s = 0 \pmod{p_1}$. In other words, the polynomial $F(X) = X + y' \lfloor N^{\frac{1}{4}} \rfloor - s$ has a small root. Thus, we can find it using Coppersmith's method for finding small roots of polynomials over $\mathbb{Z}$. Assuming that Coppersmith's algorithm will return a root x' for a given y', we verify that $\lambda_1 = x' + y'[N^{1/4}]$ is not an invalid solution by checking whether $(x' + y' \lfloor N^{1/4} \rfloor) P_1 \neq \phi(P_1)$. Heuristically, this almost nevers happens (the probability for a polynomial with coefficients of size N to have a root of size bounded by $\lfloor N^{1/4} \rfloor$ is negligible). Hence the verification test may be avoided most of the times.
   By the birthday paradox, the expected number of iterations is $\sqrt{\frac{\pi}{2}} \lfloor N^{\frac{1}{4}} \rfloor$.

The method is described in Algorithm 4. In the following, we denote by Coppersmith(a_0, a_1, N) the function which returns the root found by the method from Section 3 to the polynomial $F(t) = t + a_0 \lfloor N^{1/4} \rfloor - a_1$.

Computing random walks

To define the kangoroo pseudo-random walk, we need to associate to each group element a step size deterministically. To do so, we first partition G

into n_S disjoint sets S_i such that $G = S_0 \cup S_1 \ldots \cup S_{n_S-1}$, by using a selection function $S : G \to \{0, \ldots, n_S - 1\}$.

In order to ensure that the output of the function is an element whose discrete logarithm is in the interval $[0, w]$, each element of the group is represented by a couple (R, f), where f is the discrete logarithm of R in base P_1. The rho walk g is defined as follows:

$$(R'_{i+1}, f'_{i+1}) = g((R_i, f_i)) = \begin{cases} (2 \cdot R_i, 2 \cdot f_i) & \text{if } S(x_i) = 0 \\ (R_i + Q_i, f_i + \ell_i) & \text{if } S(x_i) = j, \end{cases}$$

where $\ell_i = \log_{P_1} Q_i$. If $f'_{i+1} > w$, then we consider k and f such that $f'_{i+1} = wk + f$ and $f \leq w$. Finally, we compute $(R_{i+1}, f_{i+1}) = (R'_{i+1} - kw \cdot P, f)$. Note that the values of k are either 1 or 2 and can be precomputed. S is modelled as a random function, but in practice one usually takes a hash function $H : \mathbb{F}_q \to \{0, \ldots, n_S - 1\}$, applied to the x-coordinate of points on the elliptic curve. For our implementation on toy examples, we set $n_S = 3$, Q_1 as P_1 and $Q_2 = \frac{w}{2} P_1$. However, note that a common recommended choice is $n_S = 20$.

In the following, we denote by Randomwalk(Q, u, w) the procedure that updates the tuple (Q, u) to (Q', u') such that $(Q', u') = g(Q, u)$, where g is our kangoroo walk defined above.

Algorithm 4 The attack

Require: An elliptic curve E defined over $\mathbb{F}_q$, $N | \#E(\mathbb{F}_q)$, (G, G_1, P, P_1), a parameter s, an endomorphism ϕ, an integer w
Ensure: λ_1 such that $\phi(P_1) = \lambda_1 P_1$.
1: $(Q_1, u) \leftarrow (P_1, 1)$, $(Q_2, v) \leftarrow (P_1, 1)$
2: Randomwalk(Q_2, v, w)
3: $x \leftarrow$ Coppersmith$(u - v, s, N)$
4: **while** $(x + (u - v)[N^{1/4}])P_1 \neq \phi(P_1)$ **do**
5: Randomwalk(Q_1, u, w)
6: Randomwalk(Q_2, v, w)
7: Randomwalk(Q_2, v, w)
8: $x \leftarrow$ Coppersmith$(u - v, s, N)$
9: **end while**
10: **return** $x + (u - v)\lfloor N^{1/4} \rfloor$

Example 4. We consider the curve given in 1. We consider ϕ the endomorphism verifying the equation $\phi^2 + \phi + 1 = 0$. Then

$$s = 895256486372398168$$

is a root $\pmod N$ of this polynomial. We have $\lfloor N^{\frac{1}{4}} \rfloor = 32775$ and we search for $x, y \in [0, 32775[$. For $y = 16381$, Coppersmith's method finds $x = 22$ a small root of the polynomial $F(X) = X + 32775y - s \pmod{p_1}$. One easily checks that one has $\phi(P_1) = \lambda_1 P_1$, with $\lambda_1 = x + 32775y \bmod p_1$.

6 Complexity of our attack

The fastest algorithm for factoring is the number field sieve algorithm (NFS), whose complexity is given by

$$L[N] = e^{1.923(\log N)^{1/3}(\log\log N)^{2/3}}$$

However, in order to find small factors of a composite number, it is more efficient to use the ECM method. The expected time required in order to find a factor p of a composite number N is then

$$E[N, p] = (\log N)^2 e^{\sqrt{2\log p \log\log p}}$$

We use formulae in Lenstra's paper [21] to compute the size of N and of each of its factors, for 80 bit and 128 bit security levels. We also took into account that the current record for factoring with ECM is finding a factor of 79 digits (see [35]). As a consequence, we take all factors of N of size at least 300 bits, in order to prevent from attacks with ECM. Our computations, for composite numbers with 2, 3 and 4 factors, are summarized in Table 1. The column DLP in this table gives the size of y such that $\lambda_1 = x + y\lfloor N^{\frac{1}{4}} \rfloor$.

Table 1. Composite number size (in bits) for a fixed security level

	80 bit security level			128 bit security level		
Number of prime factors	RSA		DLP	RSA		DLP
	$\log p_i$	$\log N$	$\log y$	$\log p_i$	$\log N$	$\log y$
2	512	1024	256	1612	3244	806
3	341	1024	170	1045	3244	520
4	300	1200	150	811	3244	400

For cryptographic sizes, computing the number of points on the elliptic curve (i.e. $\#E(\mathbb{F}_q) = q+1-t$) by using the SEA algorithm has logarithmic complexity in q and is performed with MAGMA 2.15-15 within seconds on a Intel Core i3-3227U Processor (3M Cache, 1.90 GHz). Moreover, the quantity $t^2 - 4q = DY^2$ usually has many small factors and its factorization can be computed in negligible time on the same processor.

Computing an endomorphism. The kernel of the endomorphism verifying $\phi^2 + D = 0$ is of size D. If r is a prime number, the points of order D are defined over an extension field of degree at most $2\text{ord}_r(q)$, where $\text{ord}_r(q)$ is the order of q in $\mathbb{F}_r$ (see [19]). If D is small (i.e. one or two digits), the extension field over which the points of order D are defined has degree smaller than $2D$. Hence one may apply Vélu's formulae to get an isogeny of degree D in $M(D)$ operations over $\mathbb{F}_q$, where $M(D)$ is the cost of a multiplication over an extension field of degree D.

If D is larger, it is too expensive to use this method because the points of order D will be defined over an extension field of large degree. One may then write $\phi = \frac{-t+\pi}{Y}$. We compute $Y^{-1} \pmod N$ and get $\phi(P_1) = Y^{-1}(-t+\pi)P_1$. Since $|t| < 2\sqrt{q}$ and $Y^{-1} < N$, this costs $O(\log N + \log q)$ operations over $\mathbb{F}_q$. Since the curve is constructed with the Cocks-Pinch method, we have that $\log q \sim 2\log N$. Thus we conclude that computing $\phi(P_1)$ costs $3\log N$ operations in $\mathbb{F}_q$.

Computing discrete logarithms. The Pollard kangoroo method described in Section 5 needs $\sqrt{\frac{\pi}{2}}w$ group operations and negligible memory to compute the value of y in a fixed interval of length w.

Finding small roots via Coppersmith's method. The complexity of Coppersmith's algorithm depends mainly on the running time of the LLL algorithm. We briefly recall that the complexity for Coppersmith's lattice-based algorithm is upper bounded by $0\left(d^5(d+\beta)\beta\right)$ if one uses the Nguyen-Stehlé L^2 algorithm [25] and by $O(d^{5+\epsilon}\beta + d^{\omega+1+\epsilon})\beta)$ if one uses Novocin *et al*'s L^1 algorithm [26], where d is the dimension of the lattice, β is the maximal bit-size of an entry in the input basis and ω is the matrix multiplication exponent.

Altogether, this gives an asymptotical running time of $O(h^7(\log N)^2)$ using L^2 and $O(h^6(\log N)^{1+\epsilon})$ using $\tilde{L}^1$ for our $(2h+1)$-dimensional lattice with coefficient size bounded by $N^{\frac{5h}{4}}$.

The table 2 shows how the search space varies depending on the choice of h and the number of factors of N. The formulae for ϵ are taken from [13].

With these parameters in mind, we conclude that the complexity of our algorithm is $0(h^7(\log N)^2 p_1^{1/4})$ using L^2 and $0(h^6(\log N)^{(1+\epsilon)}p_1^{1/4})$ using $\tilde{L}^1$. Using values in Table 1, a simple calculation shows that the complexity of our attack is comparable to the one of the NFS algorithm for factoring RSA moduli with 2 factors, at the 80 bit security level. Although this estimation is rough, we stress here that our attack will be faster, if one runs the Pollard-rho algorithm within a smaller search space for y. In

Table 2. Choice of parameters

Number of prime factors	ϵ/h formula	h	size of p_1	size of X
2	$\epsilon = \frac{1}{4h}$	15	2^{512}	2^{240}
2	$\epsilon = \frac{1}{4h}$	25	2^{512}	2^{245}
3	$\epsilon = \frac{1}{4h}$	25	2^{342}	2^{114}
4	$\epsilon = \frac{1}{4h}$	25	2^{300}	2^{75}

other words, the smaller λ_1 is, the higher the chances to recover it are. We conclude that λ should be chosen as an input parameter with maximal entropy, when running the Cocks-Pinch algorithm. At the 128 security level, our algorithm is slower when compared to the NFS method.

If the RSA modulus has three factors or more, we compare the complexity of our algorithm to that of the ECM method. Our calculation shows that the two methods have comparable complexities for the 80 bit security level. However, if the number of factors of N is greater than 2, our attack can only recover particular instances for λ_1, because the bound X used by Coppersmith's algorithm is very small. For example, for a composite number with 3 factors, our attack works only if λ_1 has approximatively 284 bits. Note that for a fixed security level, the higher the number m of factors of N is, the more efficient the Pollard-rho search for collision of our algorithm is. On the other hand, if $m > 2$, Coppersmith's method for finding roots of polynomials $\pmod{p_1}$ only allows to recover x in a small size search space.

7 Conclusion

Computing pairing friendly ordinary curves for implementing composite order group protocols can be done by using the Cocks-Pinch method. When running a protocol using a curve constructed with this method, one reveals a square root of the CM discriminant $-D$ modulo the composite modulus $N = p_1 p_2$. We show that if the square root λ of $-D \pmod{p_1}$ does not have maximal entropy, one may run a Pollard-rho type algorithm in order to find this value and thus to factorize N. Our conclusion is that extra precautions should be taken when choosing the RSA modulus N. More precisely, λ should be given as an input parameter of the Cocks-Pinch algorithm.

8 Acknowledgements

We are grateful to Nicolas Gama for many helpful discussions on Coppersmith's method and its implementation.

References

1. H. S. A. Sahai and B. Waters. Dynamic credentials and ciphertext delegation for attribute-based encryption. In *Proc. of* Crypto'12, volume 7417 of *LNCS*, pages 199–217. IACR, Springer-Verlag, 2012.
2. V. I. A.d. Caro and G. Persiano. Fully secure anonymous hibe and secret-key anonymous ibe with short ciphertexts. In *Proc. of* Pairing'10, volume 6487 of *LNCS*, pages 347–366. IACR, Springer-Verlag, 2012.
3. D. Boneh, G. Durfee, and N. A. Howgrave-Graham. Factoring $n = p^r q$ for large r. In *Proc. of Crypto '99*, volume 1666 of *LNCS*. IACR, Springer-Verlag, 1999.
4. D. Boneh, K. Rubin, and A. Silverberg. Finding composite order ordinary elliptic curves using the Cocks-Pinch method. Cryptology ePrint Archive, Report 2009/533, 2009. `http://eprint.iacr.org/2009/533`.
5. X. Boyen and B. Waters. Full-domain subgroup hiding and constant-size group signatures. In *Proc. of* PKC'07, volume 4450 of *LNCS*, pages 1–15, 2007.
6. C. Cocks and R. Pinch. Id-based cryptosystems based on the weil pairing. Unpublished manuscript, 2001.
7. D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *Journal of Cryptology*, 10(4):233–260, 1997. Revised version of two articles from Eurocrypt '96.
8. A. S. D. Boneh and B. Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In *Proc. of Eurocrypt 2006*, volume 4004 of *LNCS*, pages 573–592. IACR, Springer-Verlag, 2006.
9. E.-J. G. D. Boneh and K. Nissim. Evaluating 2-dnf formulas on ciphertexts. In *Proceedings of TCC 2005*, volume 3378 of *LNCS*, pages 325–341. Springer, 2005.
10. G. M. F. Göloglu, R. Granger and J. Zumbrägel. On the function field sieve and the impact of higher splitting probabilities. In *Proc. of* Crypto'13, volume 8043 of *LNCS*, pages 109–128. IACR, Springer-Verlag, 2013.
11. D. M. Freeman. Converting pairing-based cryptosystems from composite-order. In *Proc. of* Eurocrypt'10, volume 6110 of *LNCS*, pages 44–61. IACR, Springer-Verlag, 2010.
12. T. O. G. Adj, A. Menezes and F. Rodrǵuez-Henríquez. Weakness of $\mathbb{F}_{3^{6\cdot 509}}$ for discrete logarithm cryptography. Cryptology ePrint Archive, Report 2013/446, 2013. `http://eprint.iacr.org/2013/446`.
13. S. Galbraith. *Mathematics of Public Key Cryptography, Chapter 20*. Cambridge University Press, April 2012.
14. R. Granger, T. Kleinjung, and J. Zumbrägel. Breaking 128-bit secure supersingular binary curves (or how to solve discrete logarithms in $\mathbb{F}_{2^{4\cdot 1223}}$ and $\mathbb{F}_{2^{12\cdot 367}}$). `http://eprint.iacr.org/2014/119`.
15. J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *Proc. of* Eurocrypt'08, volume 4965 of *LNCS*, pages 415–432. IACR, Springer-Verlag, 2008.
16. N. Howgrave-Graham. Finding small roots of univariate modular equations revisited. In *Proceedings of Cryptography and Coding*, volume 1355 of *LNCS*, page 131âĂŞ142. IACR, Springer-Verlag, 1997.
17. G. S. I. F. Blake and N. P. Smart. *Advances in Elliptic Curve Cryptography*, volume 317. Cambridge University Press, London Mathematical Society Lecture Note Series, 2005.
18. A. S. J. Katz and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *Proc. of* Eurocrypt'08, volume 4965 of *LNCS*, pages 146–162. IACR, Springer-Verlag, 2008.

19. D. S. J. Miret, R. Moreno, J. Tena, and M. Valls. Computing the height of volcanoes of ℓ-isogenies of elliptic curves over finite fields. *In Applied Mathematics and Computation*, 196(1):67–76, 2008.
20. A. Joux. Faster index calculus for the medium prime case: application to 1175-bit and 1425-bit finite fields. In *Proc. of* Eurocrypt'13, volume 7881 of *LNCS*, pages 177–193. IACR, Springer-Verlag, 2013.
21. A. K. Lenstra. Unbelievable security, matching aes security using public key systems. In *Proc. of* Asiacrypt 2001, volume 3796 of *LNCS*, pages 13–36. IACR, Springer-Verlag, 2005.
22. A. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In *Proc. of* Eurocrypt'12, volume 7237 of *LNCS*, pages 318–335. IACR, Springer-Verlag, 2012.
23. A. Lewko and B. Waters. New techniques for dual system encryption and fully secure hibe with short ciphertexts. In *Proc. of* TCC'10, volume 5978 of *LNCS*, pages 455–479. IACR, Springer-Verlag, 2010.
24. B. W. M. Bellare and S. Yilek. Identity-based encryption secure against selective opening attack. In *Proc. of* TCC'2011, volume 6597 of *LNCS*, pages 235–252. IACR, Springer-Verlag, 2011.
25. P. Q. Nguyen and D. Stehlé. An LLL algorithm with quadratic complexity. *SIAM J. Comput.*, 39(3):874–903, 2009.
26. A. Novocin, D. Stehlé, and G. Villard. An LLL-reduction algorithm with quasi-linear time complexity. In *Proc. STOC '11*. ACM, 2011.
27. J. M. Pollard. Kangoroos, monopoly and discrete logarithms. *In* Journal of Cryptology, 13:437–447, 2000.
28. K. P. S.D. Galbraith and N. Smart. Pairings for cryptographers. Journal of Applied Mathematics, 156, numb.3113-3121, 2008.
29. J. Seo. On the (im)possibility of projecting property in prime-order setting. In *Proc. of* Asiacrypt'12, volume 7658 of *LNCS*, pages 61–79. IACR, Springer-Verlag, 2012.
30. J. Seo and J. Cheon. Beyond the limitation of prime-order bilinear groups, and round optimal blind signatures. In *Proc. of* TCC'12, volume 7194 of *LNCS*, pages 133–150. IACR, Springer-Verlag, 2012.
31. D. Stehlé. The fplll library. `http://perso.ens-lyon.fr/damien.stehle/fplll/index.html`.
32. N. S. T. Hayashi, T. Shimoyama and T. Takagi. Breaking pairing-based cryptosystems using η_t pairing over gf(3^{97}). In *Proc. of Asiacrypt'12*, volume 7658 of *LNCS*, pages 46–60. IACR, Springer-Verlag, 2012.
33. J. Vélu. In *Comptes Rendus De Academie Des Sciences Paris*, Serie I-Mathematique, Serie A., 1971.
34. E. R. Verheul. Evidence that xtr is more secure that supersingular elliptic curve cryptosystems. In *Proc. of* Eurocrypt'01, volume 2045 of *LNCS*, pages 195–201. IACR, Springer-Verlag, 2001.
35. P. Zimmermann. Top 50 ecm records. `http://www.loria.fr/~zimmerma/records/top50.html`.