Hidden Pairings and Trapdoor DDH Groups

Alexander W. Dent and Steven D. Galbraith

Information Security Group, Royal Holloway,
Egham, Surrey, TW20 0EX, U.K.
a.dent@rhul.ac.uk steven.galbraith@rhul.ac.uk

Abstract. This paper suggests a new building block for cryptographic protocols and gives two instantiations of it. The concept is to generate two descriptions of the same group: a public description that allows a user to compute a restricted set of operations, and a private description that allows a greater set of operations to be computed. We will concentrate on the case where the public description allows a user to perform group operations, and the private description also allows a user to compute a bilinear pairing on the group. A user who has the private information can therefore solve decision Diffie-Hellman problems, and potentially also discrete logarithm problems. Some possible cryptographic applications of this idea are given.

Both of our instantiations are based on elliptic curves. The first relies on the factoring assumption for hiding the pairing. The second relies on the hardness of solving a system of multivariate equations. The second method also gives rise to a practical trapdoor discrete logarithm system, thereby solving an important problem in cryptography. We hope that the paper will stimulate further research on these problems by both cryptographers and computational number theorists.

1 Introduction

Public key cryptography relies on the existence of mathematical objects which can be given a "partial description" in the sense that two users can work with the same mathematical object but one user has more knowledge (and therefore greater computational ability) than the other.

Consider a simple example. Suppose that G is a cyclic group of prime order r generated by an element P and let $P' = [x]P$ for some randomly chosen element $x \in \{1, 2, \ldots, r-1\}$. Hence, P' is also a generator of G. The partial description consists of (G, P, P', r), while the full description consists of (G, P, x, r). From the partial description, one can compute $(Q, [x]Q)$ providing that one knows the discrete logarithm of Q, but it is thought to be infeasible to compute $(Q, [x]Q)$ when one does not know the discrete logarithm of Q to the base P. However, one can always compute $(Q, [x]Q)$ from the full description. The difference between the capabilities of a user with the partial group description and a user with the full group description underpins the security of the Diffie-Hellman key exchange protocol.

The paper provides some new examples and applications of partial descriptions of groups. In particular, we consider groups with a "hidden pairing" in the sense that only the holder of some private information can compute pairings. We give two instantiations of this idea: one based on elliptic curves modulo an RSA modulus N and another based on Frey's idea of 'disguising an elliptic curve'. Our attempt to understand the security of these systems has led to the formulation of a number of interesting mathematical questions.

The aims of the paper are to raise awareness of the idea of partial group descriptions, to give some new building blocks for cryptography, and to state a number of computational questions which deserve further study. We hope that the ANTS community will find the paper a fruitful source of problems for future study and that further research follows from this work.

1.1 Pairings in cryptography

The use of pairings has been something of a minor revolution in public key cryptography. First, they were used to attack the discrete logarithm problem in certain elliptic curve groups [7, 15]. More recently they have been used as a device with which to build cryptographic primitives (see [1] for a survey). Their usefulness in this latter context is derived from their ability to provide "gap groups": groups in which the decisional Diffie-Hellman (DDH) problem is known to be easy, but in which the computational Diffie-Hellman (CDH) problem is assumed to be difficult to solve (these problems are defined in Section 2).

In this paper, we develop the idea of a "trapdoor DDH group". This is a group whose (public) description allows anyone to compute the group operation and for which there is a private trapdoor which allows a user to solve the DDH problem. Our solutions are based on "hidden pairings", which are pairings on an elliptic curve that can only be computed by an entity in possession of the trapdoor information. It is assumed that the pairing is difficult to compute for anybody not in possession of the trapdoor information.

The idea of a hidden pairing suggests three important applications. First, it implies the existence of trapdoor DDH groups, which could be of direct use in the construction of cryptographic algorithms and protocols.

Second, it could be used to give trapdoor DL groups: groups in which the discrete logarithm (DL) problem is easy to solve for anyone in possession of the trapdoor information, but the DL problem is difficult for anybody not in possession of the trapdoor information. The development of good trapdoor DL groups is a major problem in cryptography. A partial solution to this problem (due to Paillier [18]) provides a trapdoor DL subgroup of $(\mathbb{Z}/N^2\mathbb{Z})^*$. One problem with Paillier's solution is that the trapdoor DL group is only a subgroup of the whole group and it is required to 'blind' the trapdoor DL group by elements of $(\mathbb{Z}/N\mathbb{Z})^*$. Some related approaches are Naccache-Stern [16] and Okamoto-Uchiyama [17]. There are several other trapdoor DL proposals in the literature [9, 19, 22, 10] but none of these seem to be practical.

A third application is in the provable security of cryptographic protocols. There is a class of cryptosystems whose security is proved relative to a 'gap

assumption', namely that a certain computational problem should be hard even when an oracle for the corresponding decision problem is provided. Such proofs arise when one needs the decision oracle as part of the simulation for the security reduction. We stress that the decision oracle is needed only for the simulation, and not for the protocol itself. One problem with current instantiations of gap groups is that the algorithm to solve the DDH problem is available to anyone with the group description. The hidden DDH groups considered in this paper would be ideal for instantiating such cryptosystems; no individual user need be provided with the private information, but the security of the system follows from the existence of the hidden DDH oracle.

2 Problem definitions

In this section we define the relevant computational problems and we will formally define trapdoor groups. We use multiplicative notation for groups in this section.

We informally define the DL, CDH and DDH problems. The discrete logarithm problem (DL) in a group G is, given two elements $g, h \in G$, to find the integer a, if it exists, such that $h = g^a$. The computational Diffie-Hellman problem (CDH) in a group G is, given a triple of elements (g, g^a, g^b) in G, to compute the element g^{ab}. The decision Diffie-Hellman problem (DDH) is, given a quadruple of elements (g, g^a, g^b, g^c), to determine whether $g^c = g^{ab}$.

A trapdoor DDH group is defined as follows.

Definition 1 (Trapdoor DDH Group). *A trapdoor DDH group is defined by*

- *a polynomial-time group generator Gen which takes a security parameter 1^k as input, and outputs a triple (G, g, τ) where G is a group description (including a description, or partial description, of the group operation), g is the generator of a cyclic subgroup of G and τ is some trapdoor information;*
- *and a polynomial-time algorithm $\mathcal{DDH}$ which takes as input the group description G, the generator g, the trapdoor information τ and a triple (g^a, g^b, g^c), and outputs 1 if $g^c = g^{ab}$ and 0 otherwise.*

We require that the DDH problem is hard on the group G for any polynomial-time attacker who does not know the trapdoor information τ. Formally, we define the group generator Gen' as the algorithm that computes $(G, g, \tau) = Gen(1^k)$ and outputs (G, g), and insist that the DDH problem is hard for Gen'.

We shall instantiate a trapdoor DDH group using hidden pairings on an elliptic curve in Sections 3 and 4. Here the trapdoor information τ allows the computation of a pairing, but it is difficult to compute the pairing without knowing τ.

Note that the above definition says nothing about the ability to randomly sample group elements. One property of our examples is that it seems to be hard, given only the public key, for a user to randomly sample from $\langle g \rangle$ without

simply computing g^a for random $a \in \{1, \ldots, r\}$. As a result, it seems to be difficult to hash onto $\langle g \rangle$, without using the trivial construction $H(m) = g^{h(m)}$ where $h : \{0,1\}^* \to \mathbb{Z}/r\mathbb{Z}$.

We may go further with one of our constructions and conjecture the existence of trapdoor DL groups: groups in which the discrete logarithm problem is easy to solve for anyone who knows the trapdoor information τ, but difficult for anyone who does not know the trapdoor information.

Definition 2 (Trapdoor DL Group). *A trapdoor DL group is defined by*

- *a polynomial-time group generator Gen which takes a security parameter 1^k as input, and outputs a triple (G, g, τ) where G is a group description (including a description, or partial description, of the group operation), g is the generator of a cyclic subgroup of G and τ is some trapdoor information;*
- *and a polynomial-time algorithm $\mathcal{DL}$ which takes as input the group description G, the generator g, the trapdoor information τ and a group element g^a, and outputs a.*

We require that the DL problem is hard on the group G for any polynomial-time attacker who does not know the trapdoor information τ. Formally, we define the group generator Gen' as the algorithm that computes $(G, g, \tau) = Gen(1^k)$ and outputs (G, g), and insist that the DL problem is hard for Gen'.

Applications of such groups are given in Section 6.

Relation between Trapdoor DDH Groups and the Gap Diffie-Hellman Problem

Many security proofs make use of the Gap Diffie-Hellman (GDH) assumption: that the CDH problem remains hard even when the attacker has access to an oracle that correctly solves the DDH problem. There has been some debate about the security of schemes whose security proof relies on the GDH assumption, but which are implemented upon groups for which the both the CDH and DDH problems are believed to be hard. Specifically, it is not known whether the GDH problem is hard in these groups.

Let $\mathcal{U}$ be the set of all groups in which the CDH problem is difficult to solve and let $\mathcal{V} \subseteq \mathcal{U}$ be the set of all groups in which the CDH problem is difficult to solve but there exists an efficient algorithm that solves the DDH problem with probability 1. Obviously, the set $\mathcal{W}$ of all groups in which the GDH problem is hard satisfies $\mathcal{V} \subseteq \mathcal{W} \subseteq \mathcal{U}$. However, we do not know whether $\mathcal{W} = \mathcal{V}$, $\mathcal{W} = \mathcal{U}$ or whether the inclusions are strict.

We suggest that trapdoor DDH groups are likely to be good examples of groups in which both the GDH and DDH problems are hard, i.e. that these groups lie in the gap $\mathcal{W} \setminus \mathcal{U}$. We cannot be sure that this is the case: it is certainly true that the GDH problem is hard for any group in which the CDH problem remains hard even when the attacker is given the trapdoor information τ that would allow them to compute the solutions of DDH problems. However,

it does not seem likely that this holds for all trapdoor DDH groups, and we do not know if the GDH problem is hard on these groups or not. Nevertheless, it appears intuitively likely that trapdoor DDH groups are more suitable for implementing cryptosystems whose security depends on the GDH problem, than "merely" using groups in which the CDH problem is thought to be hard.

Regardless of the status of the GDH problem on general trapdoor DDH groups, we note that almost all security proofs that reduce the security of a cryptosystem to the GDH problem can be easily adapted to reduce the security of the cryptosystem to the CDH problem on any trapdoor DDH group.

3 Hidden pairings based on factoring

We give two methods for obtaining hidden pairings. The first, presented in this section, uses elliptic curves over RSA moduli and its security depends on the hardness of the factoring problem. The advantages of this approach are that it is relatively practical and efficient, and that the security is well understood.

The second proposal is motivated by Frey's idea of "disguising" an elliptic curve [6]. The advantage of this approach is that it may lead to a relatively efficient trapdoor discrete logarithm system (thereby solving an important problem in cryptography). The disadvantage is that the public key is very large and that the security is less easy to assess. We discuss this proposal in the next section.

Let p_1 and p_2 be primes of at least 512 bits in length that are congruent to 3 modulo 4. Suppose there are large primes $r_j \mid (p_j + 1)$ for $j = 1, 2$. The primes r_j should be at least 160-bit integers.

Let $N = p_1 p_2$ and let $E : y^2 = x^3 + x$ be an elliptic curve over $\mathbb{Z}/N\mathbb{Z}$. It is known that E is a supersingular curve (with embedding degree 2) over $\mathbb{F}_{p_j}$, and so $\#E(\mathbb{Z}/N\mathbb{Z}) = (p_1 + 1)(p_2 + 1)$. Let $P = (x_P, y_P) \in E(\mathbb{Z}/N\mathbb{Z})$ be a point of order $r_1 r_2$. For information about elliptic curves over rings see [13, 14, 8].

The public key is (N, E, P). From the public key one can compute $[a]P \in E(\mathbb{Z}/N\mathbb{Z})$ efficiently. The private key is (p_1, p_2, r_1, r_2). Using the private key one can solve the DDH problem as follows. Note that, by the Chinese remainder theorem, a quadruple (P, P_1, P_2, P_3) in $E(\mathbb{Z}/N\mathbb{Z})$ is a valid DDH tuple if and only if the elements reduce modulo p_1 and p_2 to valid DDH tuples in $E(\mathbb{F}_{p_1})$ and $E(\mathbb{F}_{p_2})$. One may solve DDH problems in $E(\mathbb{F}_{p_j})$ using the modified Weil or Tate pairing in the usual way [15, 7]. Some other applications of pairings may also be possible in this setting.

One can obviously use other supersingular curves (and therefore have different congruence restrictions on the primes p_j) but there seems to be no reason to use embedding degree larger than 2 in this situation. One could also use ordinary curves with low embedding degree (even embedding degree 1), although not all DDH problems are necessarily easy in this setting.

This system has two potentially useful features. First, if one uses the techniques of Demytko [5] (i.e., working with x-coordinates only) or KMOV [12] then one can hash onto the group. Second, one can delegate the ability to compute Weil pairings to a third party, without necessarily revealing the factorisation of

the modulus. This can be done by revealing the order $r_1 r_2$ of the point P. We refer to [8] for a security analysis of this situation.

Trapdoor discrete logarithms: Given the private key one can reduce the discrete logarithm problem from the elliptic curve to discrete logarithm problems in $\mathbb{F}^*_{p_j^2}$ and then attempt to solve these using an index calculus algorithm (this is just the MOV attack). Since p_j^2 is of the same size as N this will be no easier than factoring N. Hence, it seems that the trapdoor discrete logarithm application is not possible with this system.

Security: The most obvious way to attempt to solve the DDH problem without knowing the trapdoor τ is to attempt to solve the discrete logarithm or computational Diffie-Hellman problem in $\langle P \rangle$. Hence, we must ensure that the base point P has order at least 160-bits. However, a more subtle way might be to try and compute the pairing without knowing τ. As mentioned in [8], there is no known way to compute pairings without knowing the order $r_1 r_2$ of the point P. If $r_j > \sqrt{p_j}$ then knowledge of $r_1 r_2$ is sufficient to factor N. Hence, with current knowledge, we know of no way to solve the DDH problem in $E(\mathbb{Z}/N\mathbb{Z})$ without factoring.

4 Hidden pairings using a disguised elliptic curve

We follow Frey's idea of disguising an elliptic curve [6]. Essentially, we take the Weil restriction of a supersingular elliptic curve E with respect to $\mathbb{F}_{q^m}/\mathbb{F}_q$ and blind the equations by applying an invertible change of variable. One can then publish a list of multivariate polynomial equations which perform the group operation on "blinded" points. The hope is that a user who is only given the blinded group law can perform point multiplication, but cannot compute pairings on the curve.

In this section we first describe how to obtain systems of multivariate polynomials which represent the group law. We then explain a partial linearisation technique to lower the degree of these polynomials. Later we discuss the "blinding" process, and discuss several strategies to attack this proposal.

Let $E : y^2 + y = x^3 + 1$ over $\mathbb{F}_{q^m}$ where $q = 2^s$ (one could also work with elliptic curves over fields of characteristic greater than 2). Then E is supersingular with embedding degree 2. Suppose there is a large prime $r \mid (q^m + 1)$, some examples are given in the following table (note that the roles of s and m may be interchanged). Let $P = (x_P, y_P) \in E(\mathbb{F}_{q^m})$ have order r.

s	1	19	23	23	17	31	31	41
m	167	19	17	13	11	11	7	5
$\log_2(r)$	166	200	171	204	161	215	157	160

Choose a vector space basis for $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$ (for example, this could arise from a polynomial representation of $\mathbb{F}_{q^m}$). Represent every element $x \in \mathbb{F}_{q^m}$ as an m-tuple $(x_0, \ldots, x_{m-1})$ of elements in $\mathbb{F}_q$ in the obvious way, i.e. let $(x_0, \ldots, x_{m-1})$

be the coefficients of the representation of x over the basis. Hence a point $P = (x_P, y_P)$ is represented as a $2m$-tuple.

One can write down formulae for the addition rule on $2m$-tuples. Let $P = (x, y)$ and $P' = (x', y')$ be points represented as $2m$-tuples. The sum $P + P'$ is a point whose coordinates are given by ratios of polynomials of degree at most 3.

In the formulation so far, one can perform divisions. However, we will be blinding the equations by a transformation which does not respect the algebraic structure and so it will not be possible to perform divisions. Hence, we are required to work with projective points, in other words, $3m$-tuples. An alternative would be to publish an inversion rule (consisting of a product of conjugates divided by the norm) but this would lead to very high degree formulae.

A naive implementation of this idea gives rise to polynomials of degree 6 in the $3m$ variables. The expressions are relatively sparse and the storage is quite feasible. Later we will perform a linear change of variable which destroys the sparseness and thus increases the storage requirement. To obtain a slightly more practical system for $m \geq 13$ we propose publishing a partial linearisation of the system. The two representations are logically equivalent, but with the linearised system we have a lower degree and a larger number of variables. This latter system does not grow in size quite as badly as m increases.

First we consider the doubling formulae. The doubling formulae in projective coordinates for our curve are

$$[2](x : y : z) = (x^4 z^2 : x^6 + x^3 z^3 + y z^5 + z^6 : z^6)$$

which have degree 6. It is straightforward to expand this to obtain $3m$ polynomial equations in the $3m$ variables.

We now describe the partial linearisation, which exploits the fact that squaring is a linear operation in characteristic 2. Let $(x : y : z)$ be represented by a $3m$-tuple

$$(x_0, \ldots, x_{m-1}, y_0, \ldots, y_{m-1}, z_0, \ldots, z_{m-1})$$

of elements of $\mathbb{F}_q$. Then there is some linear map $S \in \mathrm{GL}_{3m}(\mathbb{F}_q)$ such that the value $(x^2 : y^2 : z^2)$ is represented by

$$(x_0^2, \ldots, x_{m-1}^2, y_0^2, \ldots, y_{m-1}^2, z_0^2, \ldots, z_{m-1}^2)S.$$

We henceforth assume that this matrix S is available, the storage requirements for S are $(3m)^2$ elements of $\mathbb{F}_q$, which will turn out to be small compared with the doubling formulae.

We introduce $5m$ new variables $x_{s,j}, y_{s,j}, z_{s,j}, x_{ss,j}$ and $z_{ss,j}$ for $0 \leq j < m$. When computing the group law these variables will be initialised as $x_{s,j} = x_j^2, x_{ss,j} = x_j^4$ etc. We then partially linearise the group law formulae by replacing all powers x_j by appropriate products of $x_j, x_{s,j}$ and $x_{ss,j}$. For example, a term such as $x_0^4 z_1^2$ becomes $x_{ss,0} z_{s,1}$ and is quadratic. Other terms, such as those coming from the computation of $x^3 y^3 = x^2 \cdot y^2 \cdot x \cdot y$ are of degree 4. Hence the total degree is reduced from six to four.

The total number of possible monomials in a homogeneous polynomial of degree d in n variables is

$$\binom{n+d-1}{d} \leq \frac{(n+d-1)^d}{d!}.$$

Our polynomials are not homogeneous, but the memory cost is dominated by the polynomials of highest degree. Hence, the total storage requirement for the doubling formulae is roughly $3(8m+3)^4/24$ elements of $\mathbb{F}_q$.

The storage can be further reduced if m and s are coprime. In this case one can choose the basis for $\mathbb{F}_{q^m}/\mathbb{F}_q$ to be simply a basis for $\mathbb{F}_{2^m}/\mathbb{F}_2$. In this case the total requirement is for $\approx 8^3 m^4$ elements of $\mathbb{F}_2$.

Now we consider addition. A general addition formula will have large degree and may lead to excessively large public keys. For many (but not all) applications it is sufficient to publish formulae for addition of the fixed base point P.

The formulae for addition of the fixed base point P to an arbitrary point $(x:y:z)$ can be written projectively as $(x:y:z)+(x_P:y_P:1)=(x':y':z')$ where

$$\begin{aligned}
x' &= z(x-x_Pz)(y-y_Pz)^2-(x-x_Pz)^4\\
y' &= z(y-y_Pz)^3+(y-y_Pz)x(x-x_Pz)^2+(y_P+1)(x-x_Pz)^3z\\
z' &= z(x-x_Pz)^3\,.
\end{aligned}$$

Note that this formula gives a correct result only if $(x:y:z)$ does not represent P or the identity.

As above, we can exploit the linearity of squaring and use the $5m$ additional variables. We write x_s for the element of $\mathbb{F}_{q^m}$ corresponding to the m-tuple $(x_{s,0},\ldots,x_{s,m-1})$ and similarly for x_{ss}, y_s, z_s and z_{ss}. One has

$$\begin{aligned}
x' &= xy_sz+x_Py_sz_s+y_P^2xzz_s+x_{ss}+x_P(y_P+1)z_{ss}\\
y' &= yy_sz+y_P^2yzz_s+y_Py_sz_s+z_{ss}+xx_sy\\
&\qquad +x_P^2xyz_s+xx_sz+x_P^2xzz_s+x_P(y_P+1)x_sz_s\\
z' &= xx_sz+x_P^2xzz_s+x_Px_sz_s+x_P^3z_{ss}.
\end{aligned}$$

It follows that the formulae have degree 3 in the $8m$ variables so require storage bounded by $3(3m+2)^3/6$ elements of $\mathbb{F}_q$ (in this case we cannot reduce to polynomials over $\mathbb{F}_2$). Hence, the total storage for the group description is roughly $8^3m^4+3^3m^3s/2$ bits.

If general addition is required then the terms $x-x_Pz$ become quadratic, but the idea of pre-computing squares and fourth powers still applies. This leads to a degree 6 system in $16m$ variables. It seems unlikely that this can be considered practical without further tricks being developed.

Now choose an invertible change of variables U on the $3m$ variables in $\mathbb{F}_q$ that define a projective point. In general, this could be non-linear, but we suggest choosing a linear transformation $U \in \mathrm{GL}_{3m}(\mathbb{F}_q)$ since linear maps do not increase the degree of the defining equations. In the security analysis below we argue that

linear maps provide sufficient security. Note that to achieve the public key sizes given above we must impose that U is defined over $\mathbb{F}_2$ and that U maps the $2m$-dimensional subspace corresponding to the x and z variables onto itself. We then 'blind' the addition formulae for the curve by applying the change of variable U to all variables. The action of U on the linearised variables $x_{s,j}$ is determined by the action on all the x_j, and similarly for $y_{s,j}$ etc.

The blinded variables will be stored as an $8m$-tuple and we suggest that the $2m$ blinded variables corresponding to the subspace of x and z variables be listed first. If this is done, one can easily recognise if a given element is the identity, since it will be a point of the form $(0 : y : 0)$.

Projective representations of points are not unique. One way to test whether two points Q_1 and Q_2 represent the same group element is to compute $Q_1 - Q_2$ and test if it is the identity, but this computation will not be possible in general with our partial group law (see Section 5 for further details). Similarly, it seems to be impossible to associate a canonical representative for a point in this setting.

A simpler blinding would be to keep the x, y and z coordinate variables separate from each other. The private transformation would then be three elements of $\mathrm{GL}_m(\mathbb{F}_q)$. This now enables a user to write down a representative for the identity element (just take all $x_j = z_j = 0$). From a performance point of view, there seems to be no significant penalty from using the more general blinding transformation. Hence we recommend the more general blinding.

The public representation of the group consists of the blinded matrix S, the doubling and addition formulae, the blinded point P and the order of P. The values of m and s are implicit in the public key. A user can efficiently compute $[a]P$ using the addition formula and the double and add algorithm in the usual way.

The private key is the inverse transformation to U (and also the original equation of the curve and point P). A user with the private key can translate blinded $3m$-tuples back to the usual representation of points in $E(\mathbb{F}_{q^m})$. Pairings can then be computed easily in the usual way. Hence the DDH problem can be solved in the group.

Compared with the first proposal, the public key for this scheme is very large. For transformations defined over $\mathbb{F}_2$ with $m = 5$ the public key is less than 2 kilobytes. For $m = 7$ and 11 the values are 6 and 31 kilobytes respectively. The value $m = 167$ gives a totally unfeasible public group description of 1.5 gigabytes!

We note that it might be interesting to use the above techniques to give a blinded description of a pairing computation algorithm. We suspect that the memory requirements will be huge, so we do not pursue this idea further.

Trapdoor DLOG: If we apply the Weil pairing to the (unblinded) elliptic curve group, then we map the elliptic curve into $\mathbb{F}_{q^{2m}}$, where $q^m \approx 2^{160}$. A 320-bit discrete logarithm computation in a characteristic 2 finite field is quite feasible. The current world record for the solution of a characteristic 2 discrete logarithm problem is over 600 bits [20, 11]. We are told [21] that the relation

finding and linear algebra computations for a 320-bit discrete logarithm would take less than a week.

Once the linear algebra stage of the index calculus algorithm has been completed we may store the reduced matrix and solve individual discrete logarithm problems in a matter of seconds. Hence, this method does give a completely practical trapdoor discrete logarithm system, thereby solving an important problem in cryptography.

Security: There are a number of ways to attack this system. As before, one can just try to solve the discrete logarithm problem in the group using the baby-step-giant-step method (see the next subsection for details). We normally impose the restriction that the group order be at least 160 bits to thwart such an attack. It follows that we should have $ms > 160$.

Another attack would be to try to compute a pairing using the blinded description of the group operation. To run Miller's algorithm one needs to obtain functions, defined over $\mathbb{F}_{q^m}$, corresponding to the straight lines in the elliptic curve addition rule. It seems hard to achieve this without being able to invert the blinding.

The other obvious attack is to try to find the invertible transformation U. The number of elements of $\mathrm{GL}_n(\mathbb{F}_q)$ is

$$(q^n-1)(q^n-q)(q^n-q^2)\cdots(q^n-q^{n-1}) = \prod_{i=1}^{n} q^{n-i}(q^i-1) > q^{n(n-1)}.$$

Since $n \geq m$ and we already assume q^m is large enough to prevent brute-force attacks, it is impossible to try all possible values for U. The smallest case we consider is when $m = 5$, when we choose matrices U defined over $\mathbb{F}_2$, and when U maps the set of x and z variables to itself. Even in this case there are roughly $2^{5m^2} = 2^{125}$ possible values for U, which means the system resists brute-force attacks.

The most plausible way to find U is to reduce the problem to solving a system of multivariate polynomial equations. We discuss such an attack here. We assume that an adversary knows not just the public key but also the original system of equations defining the group operation. This is because there are good implementation reasons for certain choices of polynomial basis etc, so one may as well assume that an adversary can simulate the key generation process up to the choice of U. Hence, the security relies on the hardness of computing the transformation U given the system of equations defining the addition rule and the point P.

The obvious attack is to represent the coefficients of the transformation U as unknowns and to obtain a system of equations among these variables. One natural way to obtain equations is by matching known points in the domain and image. We are given an explicit point P in the image, but we do not have the representation of P in the domain. Hence, as long as the original point P remains private, this attack seems to be hard.

Instead, we may obtain equations in the variables of U by relating the addition rule on the original curve with the published addition rule. More precisely,

one writes down the addition formulae on the original curve as a $3m$-tuple of multivariate polynomials, and then transforms using the 'generic' matrix U to get an enormous $3m$-tuple of multivariate polynomials. By equating coefficients with the published addition rule one obtains a system of multivariate equations in the unknown entries of U. The degree of the system depends on the degree of the original system of equations, so that from the doubling formulae one gets a degree 4 system, while from the addition by P one gets a degree 3 system.

One could apply Gröbner basis or linearisation techniques to find a solution of this system and hence deduce the matrix U. Linearisation is the most natural approach, since the system is already partially linearised. However, the required number of equations to solve the degree 3 system in $(3m)^2$ variables will be roughly $9^6 m^6/6$, whereas we only start with about $O(m^3)$ equations. Linearising the degree 4 equations coming from doubling is even worse, since we need $O(m^8)$ relations, when we only have $O(m^4)$.

We suggest that parameters satisfying $ms > 160$ with sufficiently large m are secure against multivariate attacks on this system. We recognise that further research into the problem of recovering U is needed before we can have confidence in the the security of this system. In particular, it is important to determine which values for m are secure: we expect that $m = 3$ is too small for security and that $m \geq 11$ is OK. We hope that our work motivates others to consider this problem.

The simpler formulation (blinding the representations of x, y and z using three elements of $\mathrm{GL}_m(\mathbb{F}_2)$) may also be secure. Further research would clarify this.

One could use a more general initial curve equation, for example $y^2 + Ay = x^3 + Bx + C$ for some $A, B, C \in \mathbb{F}_{q^m}$. All such non-singular equations are isomorphic over $\overline{\mathbb{F}}_2$ to $y^2 + y = x^3 + 1$. An isomorphism of a Weierstrass equation of this form is a linear map. But since the coefficients of the isomorphism may lie in an extension field, this linear change of variable is not necessarily already included in the above analysis. Instead, one would have to perform the linearisation or Gröbner basis methods over small degree extensions of $\mathbb{F}_q$. We do not discuss this idea further as we do not expect it to significantly add to the security.

5 Partial group descriptions

In the previous section, as a way of minimising the group description, we suggested publishing just the operations of doubling and addition by a fixed point P. Note that both these operations are unary, whereas a general group description requires a binary operation. We now make some comments about this idea.

Let G be a group written additively and suppose the published description of G comprises just an element P and unary operations $f(Q) = Q + Q$, $g(Q) = Q + P$. In general, there seems to be no way to obtain the sum $Q_1 + Q_2$ of two general points from the operations f and g.

Using the double-and-add algorithm one can compute $[a]P$ for any positive integer a. But it does not seem to be possible to compute $[a]Q$ for a randomly chosen element $Q \in G$, unless we know a positive integer b such that $Q = [b]P$.

To summarise, the description of the group G is sufficient for some computations, but it does not satisfy the usual computational definition of a group law.

A natural question is whether the discrete logarithm problem is harder in such a group than in a generic group. We first consider the case where a binary predicate is available which determines whether two given inputs represent the same group element.

The baby-step-giant-step algorithm can be implemented in such a group. This algorithm attempts to solve the discrete logarithm of a point Q to the base P. Let r be the order of P and define $M = \lfloor \sqrt{r} \rfloor$. The standard description of the algorithm is to compute and store a list of 'baby steps' $P, [2]P, [3]P, \ldots, [M]P$ and find a collision with the list of 'giant steps' $Q, Q - P', Q - [2]P', \ldots$ where $P' = [M]P$. The obstacles are that, a priori, one cannot compute $-P'$ and one cannot compute $Q + (-P')$.

Instead, one can formulate the baby-step-giant-step algorithm as follows. If $Q = [\lambda]P$ then we can write $\lambda = jM - i$ for some $1 \leq j \leq M + 1$ and some $0 \leq i \leq M$. Hence, we compute and store the baby steps $Q, Q + P, Q + [2]P, \ldots, Q + [M]P$ by successively applying the operation g. Then we compute the giant steps $[jM]P$ using the double-and-add algorithm for each value of j and check if there is a match using the predicate. Note that each giant step is now a full point multiplication, rather than a single addition. Nevertheless, the final complexity is still $\tilde{O}(\sqrt{r})$.

To implement a random walk method, such as Pollard rho, we require canonical representatives of group elements. Hence it seems that such methods cannot be implemented on disguised projective elliptic curves.

With the partial group description coming from a disguised elliptic curve we do not have an equality predicate or canonical representatives of group elements. Hence the best algorithm for solving the discrete logarithm problem seems to be brute-force search! This suggests that there could be applications where we can safely reduce the group size to 80 bits.

Our second observation is that one can recover a full group description from a partial group description in the case of finite fields. More precisely, let $G = \mathbb{F}_{q^m}^*$ where $q = 2^s$ and suppose we compute a linear blinding of the Weil restriction of G with respect to $\mathbb{F}_{q^m}/\mathbb{F}_q$. We therefore compute with m-tuples $(x_0, \ldots, x_{m-1})$ of elements of $\mathbb{F}_q$. Suppose that we publish a group element g as an m-tuple $(g_0, \ldots, g_{m-1})$ and descriptions of the unary operations of squaring and multiplication by g. Since multiplication by g is linear we assume that this operation is represented by a matrix M_1. Similarly, we assume squaring is represented by a matrix M_2 applied to the vector $(x_0^2, x_1^2, \ldots, x_{m-1}^2)$. One might hope that it is not possible to deduce the full group law from the partial group description.

We now show how to recover the full group description. One simply computes the characteristic polynomial of the matrix M_1. This gives the character-

istic polynomial $P(T)$ of the element g. One may therefore represent $\mathbb{F}_{q^m}/\mathbb{F}_q$ as $\mathbb{F}_q[T]/(P(T))$. By simple linear algebra one can translate between the given m-tuple representation $(x_0, \ldots, x_{m-1})$ for an element of G and the polynomial representation. The general multiplication rule can then be recovered.

While we do not think that this method can be used to recover the full group description in the elliptic curve case, it does cast some doubt over our claims for security of the disguised elliptic curve idea. More research is needed to clarify this.

6 Simple Applications of Trapdoor Groups

In this secion we present a few simple cryptographic applications for trapdoor DDH and trapdoor DL groups. We will assume that the order r of g in G is known (or that it is possible to compute it efficiently). If this is not the case, then we may still use all of the following applications by taking r to be much larger than the order of g.

6.1 A Simple Identification Scheme Based on Trapdoor DDH Groups

The simplest application of trapdoor DDH groups is to create an identification scheme. Here a central authority Charlie wishes to identify a user Alice.

- At the time of registration, Alice generates a trapdoor DDH group $(G, g, \tau) = Gen(1^k)$ and gives Charlie (G, g, r), where r is the order of the element g.
- When Charlie wishes to identify Alice, he randomly selects a bit $\sigma \in \{0, 1\}$ and integers a and b from $\{1, 2, \ldots, r\}$, and computes $A = g^a$ and $B = g^b$. If $\sigma = 0$ then he computes $C = g^{ab}$, otherwise he randomly chooses a value $c \in \{1, 2, \ldots, r\}$ such that $c \neq ab \bmod r$ and computes $C = g^c$. Charlie then sends the triple (A, B, C) to Alice.
- Alice receives the challenge triple (A, B, C), and checks whether it is a valid DDH triple using the trapdoor information τ. If (A, B, C) is a DDH triple, then Alice sends $\sigma' = 0$ to Charlie; otherwise Alice sends $\sigma' = 1$.
- Charlie receives a bit σ' from Alice, and accepts Alice's identity if $\sigma = \sigma'$.

It is obvious that any attacker that fools the identification scheme with probability $1/2 + \epsilon$ has advantage at least $\epsilon + 1/r$ in breaking the DDH problem in the trapdoor group, therefore ϵ is negligible. Clearly, if this scheme is meant to be practical, then the identification scheme would have to be run multiple times before Alice's identity is actually accepted.

6.2 An Encryption Scheme Based on the Discrete Logarithm Problem

We present an encryption scheme whose security depends upon the difficulty of solving the discrete logarithm problem in an arbitrary trapdoor DL group. For simplicity we present this encryption scheme as a KEM [4].

– Bob, who wishes to be able to receive encrypted messages, generates a trapdoor DL group $(G, g, \tau) = Gen(1^k)$ and publishes (G, g, r) as his public key, where r is the order of the element g. Bob also publishes a key derivation function KDF which maps elements of the set $\{1, 2, \ldots, r\}$ onto bit-strings of the appropriate key length.
– Alice, who wishes to compute a symmetric key for use in sending an encrypted message to Bob, randomly generates an integer $x \in \{1, 2, \ldots, r\}$. She computes $C = g^x$ and $K = KDF(x)$, and sends C to Bob (along with the encryption of a message computed using the DEM and the key K).
– Bob recovers first x from C using the trapdoor information τ. Bob then computes key $K = KDF(x)$.

It is not difficult to see that, in the random oracle model, an attacker who has an advantage ϵ in breaking the IND-CCA2 security of this scheme, can be used to construct an algorithm that solves the discrete logarithm problem in G with probability ϵ.

7 Conclusions

We have suggested the concept of a hidden pairing, which gives rise to a trapdoor DDH group, and potentially a trapdoor DL group. We have suggested two possible ways to implement such an idea. Our work suggests several problems for further study, which we list below. We hope that the ANTS community will be motivated to study some of these problems further.

– Can the storage requirement be reduced for the group law description of a disguised elliptic curve?
– For which values of m is the proposed version of disguising an elliptic curve secure against Gröbner basis or linearisation attacks?
– Is there a way to perform Miller's algorithm to compute pairings on a disguised elliptic curve?
– Is there a trapdoor DDH group onto which one can hash in a non-trivial way?
– Are there cryptosystems which can be securely implemented using an 80-bit partial group law?
– Do there exist partial group descriptions for other groups which may allow interesting cryptographic functionalities?
– Are there further cryptographic applications of hidden pairings?

Acknowledgements

We thank Kenny Paterson for his comments.

References

1. I. Blake, G. Seroussi and N. P. Smart, Advances in elliptic curve cryptography, Cambridge (2005).
2. D. Boneh, B. Lynn, and H Shacham. Short signatures from the Weil pairing. In C. Boyd, editor, *Advance in Cryptology – Asiacrypt 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer-Verlag, 2001.
3. The Magma computer algebra system. University of Sydney.
4. R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2004.
5. N. Demytko, A new elliptic curve based analogue of RSA, in T. Helleseth (ed.), EUROCRYPT 1993, Springer LNCS 765 (1994) 40–49.
6. G. Frey, How to disguise an elliptic curve (Weil descent), Talk at ECC 1998. Slides available from:
   `http://www.cacr.math.uwaterloo.ca/conferences/1998/ecc98/frey.ps`
7. G. Frey, H.-G. Rück, A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves, *Math. Comp.*, **62**, No.206 (1994) 865–874.
8. S. D. Galbraith and J. F. McKee, Pairings on elliptic curves over finite commutative rings, in N. P. Smart (ed.), Cryptography and Coding: 10th IMA International Conference, Cirencester, UK, Springer LNCS 3796 (2005) 392–409.
9. D. M. Gordon, Designing and detecting trapdoors for discrete log cryptosystems, in E. F. Brickell (ed) CRYPTO 92, Springer LNCS 740 (1993) 66–75.
10. D. Hühnlein, M. J. Jacobson, D. Weber, Towards practical non-interactive public key cryptosystems using non-maximal imaginary quadratic orders, in D. R. Stinson and S. Tavares (eds.), SAC 2000, Springer LNCS 2012 (2001) 275–297.
11. A. Joux and R. Lercier, Discrete logarithms in $GF(2^{607})$ and $GF(2^{613})$, posting to the Number Theory Mailing List, 23 Sep 2005.
12. K. Koyama, U.M. Maurer, T. Okamoto and S.A. Vanstone, New public-key schemes based on elliptic curves over the ring $\mathbb{Z}_n$, in J. Feigenbaum (ed.), CRYPTO 1991, Springer LNCS 576 (1992) 252–266.
13. H. W. Lenstra Jr., Factoring integers with elliptic curves, *Annals of Mathematics*, **126** (1987) 649–673.
14. H. W. Lenstra Jr., Elliptic curves and number theoretic algorithms, *Proc. International Congr. Math.*, Berkeley 1986, AMS (1988) 99–120.
15. A. J. Menezes, T. Okamoto and S. A. Vanstone, Reducing elliptic curve logarithms to logarithms in a finite field, *IEEE Trans. Inf. Theory*, **39**, No. 5 (1993) 1639–1646.
16. D. Naccache and J. Stern, A new public-key cryptosystem based on higher residues, ACM Conference on Computer and Communications Security (1998) 59–66.
17. T. Okamoto and S. Uchiyama, A new public key cryptosystem as secure as factoring, in K. Nyberg (ed.), EUROCRYPT '98, Springer LNCS 1403 (1998) 308–318.
18. P. Paillier, Public-key cryptosystems based on composite degree residuosity classes, in J. Stern (ed.), Eurocrypt 1999, Springer LNCS 1592, 1999, pp. 223-238.
19. E. Teske, An elliptic curve trapdoor scheme, *J. Crypt.*, 19 (2006) 115–133.
20. E. Thomé, Computation of discrete logarithms in $GF(2^{607})$, in C. Boyd (ed.), ASIACRYPT 2001, Springer LNCS 2248 (2001) 107–124.
21. E. Thomé, Personal communication, January 9, 2006.
22. S. Vanstone and R. J. Zuccherato, Elliptic curve cryptosystem using curves of smooth order over the ring $\mathbb{Z}_n$, *IEEE Trans. Inf. Theory*, Vol. 43, No. 4 (1997) 1231–1237.