J. Cryptology (2004) 17: 263–276
DOI: 10.1007/s00145-004-0312-y

Journal of
CRYPTOLOGY

A One Round Protocol for Tripartite Diffie–Hellman

Antoine Joux
DCSSI Crypto Lab, 51 Bd de Latour-Maubourg,
75700 Paris 07 SP, France
Antoine.Joux@m4x.org

Communicated by Arjen K. Lenstra

Received 23 January 2003 and revised 10 June 2003
Online publication 23 June 2004

Abstract. In this paper we propose a three participants variation of the Diffie–Hellman protocol. This variation is based on the Weil and Tate pairings on elliptic curves, which were first used in cryptography as cryptanalytic tools for reducing the discrete logarithm problem on some elliptic curves to the discrete logarithm problem in a finite field.

Key words. Cryptosystem, Key exchange, Discrete logarithm, Elliptic curve, Pairing.

1. Introduction

Since its discovery in 1976, the Diffie–Hellman protocol has become one of the most famous and largely used cryptographic primitives. In its basic version, it is an efficient solution to the problem of creating a common secret between two participants. Since this protocol is also used as a building block in many complex cryptographic protocols, finding a generalization of Diffie–Hellman gives us a new tool which can be used to construct many new and efficient protocols. The goal of this paper is precisely to describe such a generalization of Diffie–Hellman. For this generalization, examples of such new and efficient protocols are numerous and quickly expanding, for example, we can cite identity-based encryption [6], short signatures [7], verifiable random functions [26], . . .

This article is a revised and updated version of [16]. It shows how the Weil and Tate pairings can be used to devise a tripartite generalization of the basic Diffie–Hellman protocol. These pairings were first used in cryptography as cryptanalytic tools to reduce the complexity of the discrete logarithm problem on some "weak" elliptic curves. However, using them for constructive purposes is a novel idea. It was pointed out by Galbraith et al. in [13] that this idea independently appeared in [37].

Of course, the problem of setting a common key between more than two participants can be addressed using the usual Diffie–Hellman (see the protocol for conference keying in [8]). However, all the known protocols, that make use of classical cryptographic assumptions, require at least two rounds of communication. In some cases these two rounds can be somewhat cumbersome, and a single pass would be much preferable. To give a basic example, exchanging an email message key with a two round Diffie–Hellman protocol would require both participants to be connected at the same time, which is a very undesirable property for this application. On the other hand, the basic one round Diffie–Hellman protocol can be fitted to this email application by replacing one of the two ephemeral keys by a static public key. Similarly, the one round tripartite Diffie–Hellman presented here can be transformed into non-interactive public key systems (e.g., identity-based encryption [6]). As a consequence, it presents a real improvement compared with conference keying. Of course, due to its very simplicity, it also lacks some refinements, for example, as the basic Diffie–Hellman key exchange it is unauthenticated and allows man in the middle attacks. Possible ways of achieving authentication are proposed in [3] and [44].

2. The Discrete Logarithm Problem on Weak Elliptic Curves

The discrete logarithm problem on elliptic curves is one of the few standard assumptions that is currently used in public key cryptography. When elliptic curve cryptosystems were first proposed in [34], computing the number of points of a given curve was a challenging task, since the Schoof, Elkies and Atkin algorithm was not yet mature (for a survey of this algorithm see [24]) and the p-adic methods, recently proposed by Satoh [38] and Mestre [32] were unknown. Since their invention, these p-adic methods have been implemented by many people, e.g., see [11], [20], [15] and [25]. Thus, at that time, in order to avoid the hard task of point counting, the idea of using special curves where this problem becomes easy quickly arose. As a side bonus, the addition formulas are somewhat simpler on such curves and the cryptosystems can thus be implemented more efficiently. However, it was shown later that some of these special cases are less secure than general curves.

As of today, two categories of weak special cases have been identified. In one of them, the discrete logarithm problem becomes easy (i.e., polynomial time) as was shown in [42], [40] and [39]. This easiest case happens when the number of points of the elliptic curve over $\mathbb{F}_p$ is exactly p. Luckily, they were never really used[1] in cryptosystems. In the other category of special curves, the discrete logarithm problem on the elliptic curve is transformed into a discrete logarithm problem in a small extension of the field of definition of the elliptic curve. In particular, this category contains the class of supersingular elliptic curves, which were considered in elliptic curve cryptosystems since the early days. In 1993 Menezes, Okamato and Vanstone proposed in [31] a first reduction algorithm called the MOV reduction and based on a mathematical tool, the Weil pairing. A second reduction algorithm was proposed by Frey and Rück in [12], we refer to it as the FR reduction. A survey of these reductions, as cryptanalytic tools, was published at Eurocrypt '99 [14], and gave a comparison of these two reductions. The conclusion was that the FR reduction can be applied to more curves than the MOV reduction and moreover that it can be computed faster than the MOV reduction. Thus for all practical

[1] I thank one of the anonymous referrees for pointing out US patent number 5,272,755 by Miyaji and Tatebayashi (1993) patenting the use of these anomalous curves.

usage, the authors recommended the FR reduction. However, they mistakenly claimed that the computation of the FR and MOV reduction may be a heavy load. In fact, thanks to an idea of Miller [33], this is not the case and these reductions can be computed very efficiently, which is a essential preliminary in order to transform them from cryptanalytic to cryptographic tools. Of course, in that case, speed is of the essence and recent results show that it is possible to improve upon Miller's algorithm by using numerous tricks (see [13] and [4]).

At first, it seemed surprising to use elliptic curves which are known to be weaker than random curves in cryptographic constructions. Indeed, this forces us to rely on the hardness of the discrete logarithm in some finite field $\mathbb{F}_{p^k}$, thus losing one of the most important advantages in elliptic-curve-based systems, i.e., the small size of the keys. However, as recent history has shown, the loss is more than balanced by the extra properties offered by such curves, and their versatility in new constructions.

The Basic Tool—Pairings on an Elliptic Curve

The MOV and FR reductions are both based on a bilinear pairing, in the MOV case it is the Weil pairing and in the FR case it is called the Tate pairing. In this section we describe these pairings for an elliptic curve E defined over $\mathbb{F}_p$. The pairings also exist for elliptic curves defined over $\mathbb{F}_{p^r}$. In order to define these pairings, we first need to introduce the function field and the divisors of the elliptic curve. Very informally, the function field $K(E)$ of E is the set of rational maps in x and y modulo the equation of E (e.g. $y^2 - x^3 - ax - b$). A divisor D is an element of the free group generated by the points on E, i.e., it can be written as a finite formal sum: $D = \sum_i a_i(P_i)$, where the P_i are points on E and the a_i are integers. In what follows, we only consider divisors of degree 0, i.e., such that $\sum_i a_i = 0$.

Given any function f in $K(E)$, we can build a degree 0 divisor $div(f)$ from the zeros and poles of f simply by forming the formal sum of the zeros (with multiplicity) minus the formal sum of the poles (with multiplicity). Any divisor $D = div(f)$ will be called a principal divisor. In the reverse direction, testing whether a degree 0 divisor $D = \sum_i a_i(P_i)$ is principal or not, can be done by evaluating $\sum a_i P_i$ on E. The result is the point at infinity if and only if D is principal.

Given a function f in $K(E)$ and a point P of E not belonging to the support of $div(f)$, f can be evaluated at P by substituting the coordinates of P for x and y in any rational map representing f. Given a divisor $D = \sum_i a_i(P_i)$, whose support is disjoint from the support of $div(f)$, the function f can also be evaluated at D using the following definition:

$$f(D) = \prod_i f(P_i)^{a_i}.$$

Using these notions, we can now define the Weil pairing: it is a bilinear function from the torsion group $E[\ell]$ to the multiplicative group μ_ℓ of ℓth roots of unity in some extension of $\mathbb{F}_p$, say $\mathbb{F}_{p^k}$. Given two ℓ-torsion points P and Q, we compute their pairing $e_\ell(P, Q)$ by finding two functions f_P and f_Q such that $div(f_P) = \ell(P) - \ell(O)$ and $div(f_Q) = \ell(Q) - \ell(O)$, and by evaluating

$$e_\ell(P, Q) = f_P(Q)/f_Q(P).$$

This pairing e_ℓ: $E[\ell] \times E[\ell] \to \mu_\ell$ is bilinear and non-degenerate. This means that $e_\ell(aP, bQ) = e_\ell(P, Q)^{ab}$ and that for some points of P and Q, we have $e_\ell(P, Q) \neq 1$. Using such a pairing for cryptanalytic applications is quite easy, indeed, given a point X "independent" from P and Q, we can shift the discrete logarithm problem $Q = \lambda P$ on the elliptic curve to the discrete logarithm problem $e_\ell(Q, X) = e_\ell(P, X)^\lambda$ in $\mathbb{F}_{p^k}$. Then we can apply a subexponential algorithm to solve the latter problem. This application of the Weil pairing is called the MOV reduction and appeared in [31].

The variant of the Tate pairing described in [12] appears to be more complicated, since it operates on divisors instead of points. However, it is at least twice as fast as the Weil pairing, since it suffices to evaluate a single function from the function field instead of two. In fact, in some cases there is an asymmetry between the divisors involved and it is possible to gain an even larger speed factor by using some fast implementation techniques proposed in [4] which exploit this asymmetry. More precisely, the divisors involved in the Tate pairing are ℓ-fold divisors, i.e., divisors D such that ℓD is principal, it takes values in μ_ℓ and it is bilinear and non-degenerate. Given two ℓ-fold divisors D_1 and D_2 defined over an extension $\mathbb{F}_{p^k}$ that contains the ℓth roots of unity, we find first f_{D_1} such that $div(f_{D_1}) = \ell D_1$. The Tate pairing of D_1 and D_2 is then defined as

$$t_\ell(D_1, D_2) = f_{D_1}(D_2)^{(p^k-1)/\ell}.$$

This pairing is also bilinear and non-degenerate. A very important point to remember with the Tate pairing is the fact that the divisors involved, D_1 and D_2, are in truth representatives of classes of equivalence. Two divisors D_1 and D_1' belong in the same class when their difference $D_1 - D_1'$ is a principal divisor. Moreover, for the purpose of discrete logarithm reduction, the Tate pairing $t_\ell(D_1, D_2)$ can easily be transformed into a pairing that involves points. As in [12], one can simply fix two points R and S, and remark that

$$t_\ell((\lambda P) - (O), (R) - (S)) = t_\ell((P) - (O), (R) - (S))^\lambda.$$

For more information about the Weil and Tate pairings and their cryptanalytic applications, we refer the reader to [31] or [12].

3. A Tripartite Diffie–Hellman Protocol

In this section we want to build an analog of the Diffie–Hellman protocol that involves three participants, A, B and C, requires a single pass of communication and allows the construction of a common secret $K_{\mathrm{A,B,C}}$. By a single pass of communication, we mean that each participant is allowed to talk once and broadcast some data to the other two. The main idea is similar to ordinary Diffie–Hellman, we start from some elliptic curve E and some (ℓ-torsion) point P. Then A, B and C each choose a random number (a, b or c) and they respectively compute $P_{\mathrm{A}} = aP$, $P_{\mathrm{B}} = bP$ and $P_C = cP$ and broadcast these values. Then they respectively compute $F(a, P_B, P_C)$, $F(b, P_A, P_C)$ and $F(c, P_A, P_B)$, where the function F is chosen in a way that ensures that these numbers will be equal and that this common value $K_{\mathrm{A,B,C}}$ will be hard to compute given P_{A}, P_{B} and P_{C}. The problem now is to find such an F.

Using the Weil pairing, it seems natural to use the following formula:

$$F_{\mathrm{W}}(x, P, Q) = e_\ell(P, Q)^x.$$

With this definition, one can easily check that

$$F_{\mathrm{W}}(a, P_{\mathrm{B}}, P_{\mathrm{C}}) = F_{\mathrm{W}}(b, P_{\mathrm{A}}, P_{\mathrm{C}}) = F_{\mathrm{W}}(c, P_{\mathrm{A}}, P_{\mathrm{B}}) = F_{\mathrm{W}}(1, P, P)^{abc}.$$

However, due to the properties of the Weil pairing, this simple construction fails because $e_\ell(P, P) = 1$ and thus $K_{\mathrm{A,B,C}}$ is the constant 1 and can easily be guessed by any attacker.

Nevertheless, the basic idea is sound and can in fact be implemented. Two different approaches are possible, we can either make use of two independent points or modify the pairing to make it work with a single point. Initially, we proposed to use two independent points P and Q, instead of one in order to derive a working protocol. In this section we focus on this two points approach, the single point variation is addressed in Section 4.

Using the Weil pairing, the two points approach works as follows: we randomly choose two independent[2] points P and Q in the ℓ-torsion of the curve, and we have the three participants compute and broadcast $(P_{\mathrm{A}}, Q_{\mathrm{A}})$, $(P_{\mathrm{B}}, Q_{\mathrm{B}})$ and $(P_{\mathrm{C}}, Q_{\mathrm{C}})$. Then A, B and C can respectively compute $F_{\mathrm{W}}(a, P_{\mathrm{B}}, Q_{\mathrm{C}}) = F_{\mathrm{W}}(a, Q_{\mathrm{B}}, P_{\mathrm{C}})$, $F_{\mathrm{W}}(b, P_{\mathrm{A}}, Q_{\mathrm{C}}) = F_{\mathrm{W}}(b, Q_{\mathrm{A}}, P_{\mathrm{C}})$ and $F_{\mathrm{W}}(c, P_{\mathrm{A}}, Q_{\mathrm{B}}) = F_{\mathrm{W}}(c, Q_{\mathrm{A}}, P_{\mathrm{B}})$. Moreover, all these values are equal and thanks to the independence of P and Q, they are not constant. Moreover, assuming that ℓ is prime and that a, b and c are random integer from $[1; \ell - 1]$, the common value is in fact picked uniformly at random from the set of (non-trivial) ℓth roots of unity.

It is also easy to use the Tate pairing instead of the Weil pairing, and to define another function F as

$$F_{\mathrm{T}}(x, D_1, D_2) = t_\ell(D_1, D_2)^x.$$

Assuming that A, B and C are still broadcasting two points, we can now define two divisors for each user. The simplest technique is for any point X on the curve to define $D_X = (X) - (O)$. Then A, B and C can respectively compute

$$\begin{aligned} F_{\mathrm{T}}(a, D_{P_{\mathrm{B}}}, D_{Q_{\mathrm{C}}}) &= F_{\mathrm{T}}(a, D_{P_{\mathrm{C}}}, D_{Q_{\mathrm{B}}}), \\ F_{\mathrm{T}}(b, D_{P_{\mathrm{A}}}, D_{Q_{\mathrm{C}}}) &= F_{\mathrm{T}}(b, D_{P_{\mathrm{C}}}, D_{Q_{\mathrm{A}}}), \\ F_{\mathrm{T}}(c, D_{P_{\mathrm{B}}}, D_{Q_{\mathrm{A}}}) &= F_{\mathrm{T}}(c, D_{P_{\mathrm{A}}}, D_{Q_{\mathrm{B}}}). \end{aligned}$$

However, with this choice, some care must be taken to evaluate F_{T}. Indeed, we must choose a different representative for one of the divisor classes, otherwise, at some point during the Tate pairing computation, we try to divide zero by zero, the computation fails and returns an undefined result. To avoid this pitfall, we can replace D_Q by the equivalent divisor $(Q + R) - (R)$, where R is some randomly chosen point.

Another approach to avoid this difficulty is to choose different divisors. For a user U, define

$$\begin{aligned} D_1^{(U)} &= (P_U) - (Q_U) \equiv D_{P_U} - D_{Q_U}, \\ D_2^{(U)} &= (P_U + Q_U) - (O) \equiv D_{P_U} + D_{Q_U}. \end{aligned}$$

[2] When the ℓ-torsion points P and Q are independent, no value of a exists such that $P = aQ$ or $Q = aP$. This is easily checked, since in that case $e_\ell(P, Q) \neq 1$.

Then A, B and C can respectively compute

$$\begin{aligned} F_{\mathrm{T}}(a, D_1^{(\mathrm{B})}, D_2^{(\mathrm{C})}) &= F_{\mathrm{T}}(a, D_1^{(\mathrm{C})}, D_2^{(\mathrm{B})}), \\ F_{\mathrm{T}}(b, D_1^{(\mathrm{A})}, D_2^{(\mathrm{C})}) &= F_{\mathrm{T}}(b, D_1^{(\mathrm{C})}, D_2^{(\mathrm{A})}), \\ F_{\mathrm{T}}(c, D_1^{(\mathrm{B})}, D_2^{(\mathrm{A})}) &= F_{\mathrm{T}}(c, D_1^{(\mathrm{A})}, D_2^{(\mathrm{B})}). \end{aligned}$$

Because of the bilinearity of the pairing, all these numbers are equal.

With both choices involving the Tate pairing, it is important to check the non-degeneracy when setting the cryptosystem. This is simply done by checking that $t_\ell(D_P, D_Q)$ (or $t_\ell(D_P - D_Q, D_P + D_Q)$) is not equal to 1. Then, as with the Weil pairing, the common value is uniformly distributed.

As we explained above in Section 2, since F_{T} is based on the Tate pairing, it will be faster to evaluate than F_{W}. Finally, our initial tripartite Diffie–Hellman protocol can be summarized as follows:

Alice	Bob	Charlie
Choose a Compute $(P_{\mathrm{A}}, Q_{\mathrm{A}})$	Choose b Compute $(P_{\mathrm{B}}, Q_{\mathrm{B}})$	Choose c Compute $(P_{\mathrm{C}}, Q_{\mathrm{C}})$
Broadcast $P_{\mathrm{A}}, P_{\mathrm{B}}, P_{\mathrm{C}}$ and $Q_{\mathrm{A}}, Q_{\mathrm{B}}, Q_{\mathrm{C}}$		
Compute the common key as		
$F_{\mathrm{T}}(a, D_{P_{\mathrm{B}}}, D_{Q_{\mathrm{C}}})$	$F_{\mathrm{T}}(b, D_{P_{\mathrm{A}}}, D_{Q_{\mathrm{C}}})$	$F_{\mathrm{T}}(c, D_{P_{\mathrm{B}}}, D_{Q_{\mathrm{A}}})$
Alternatively, compute the common key as:		
$F_{\mathrm{T}}(a, (P_{\mathrm{B}}) - (Q_{\mathrm{B}}), (P_{\mathrm{C}} + Q_{\mathrm{C}}) - (O))$	$F_{\mathrm{T}}(b, (P_{\mathrm{A}}) - (Q_{\mathrm{A}}), (P_{\mathrm{C}} + Q_{\mathrm{C}}) - (O))$	$F_{\mathrm{T}}(c, (P_{\mathrm{B}}) - (Q_{\mathrm{B}}), (P_{\mathrm{A}} + Q_{\mathrm{A}}) - (O))$

Choice of Parameters and Construction of the Elliptic Curve

For this tripartite Diffie–Hellman protocol to be efficient, we need to choose a finite field $\mathbb{F}_q$ (with $q = p^r$) and an elliptic curve such that the chosen pairing can be efficiently computed. We recall that the pairing takes values in a multiplicative subgroup of some extension field $\mathbb{F}_{q^k}$. In this section we focus on the two points protocol, however, most of the arguments remain unchanged with the single point variant discussed in Section 4. Together with the curve, we need to choose two points P and Q such that the chosen pairing will be nondegenerate. As explained above this property can easily be checked by testing this pairing on the base points. Note that when $k \neq 1$ at least one of the points P and Q must be defined over the extension field $\mathbb{F}_{q^k}$ rather than over $\mathbb{F}_q$, for the pairing not to be degenerate.

The most important parameter when choosing an elliptic curve together with a pairing is the value of k. This value should be small enough, otherwise computations in $\mathbb{F}_{q^k}$ become infeasible and the pairing cannot be computed. On the other hand, for some applications such as short signatures [7], it might be useful to have a moderately large value of k for the discrete logarithm problem in $\mathbb{F}_{q^k}$ to be as hard as possible. At present, several values of k can be efficiently constructed, namely $k = 1$, $k = 2$, $k = 3$, $k = 4$ and $k = 6$. These values can be reached by using either supersingular curves or complex multiplication techniques.

With supersingular curves, the possible values of k depend on the characteristic p of the field of definition, and also on the parity of the exponent r in the cardinality $q = p^r$ of the field. The possible values of k for supersingular curves are described in [30]. With care, they can also be derived from the complete classification of supersingular curves given on p. 140 of [41]. With $p = 2$, there is (up to isomorphism in the algebraic closure of $\mathbb{F}_2$) a single supersingular curve, with the equation $y^2 + y = x^3$. However, when restricting ourselves to isomorphisms in $\mathbb{F}_q$, more curves are available. A complete classification is given on pp. 46–48 of [30]. We take the example of the curve $y^2+y = x^3$. When r is odd, this curve has $q + 1$ points and $k = 2$. When r is even, this curve has $(p^{r/2} + 1)^2$ points and $k = 1$. The value $k = 4$ can be reached when r is odd with a different curve, such as $y^2 + y = x^3 + x$ or $y^2 + y = x^3 + x + 1$. With $p = 3$, there is (up to isomorphism in the algebraic closure of $\mathbb{F}_3$) a single supersingular curve, with the equation $y^2 = x^3 + 2x + 1$. As in the characteric 2 case, matters are more complicated when restricting ourselves to isomorphisms in $\mathbb{F}_q$. A classification for this case is given in [36]. For example, with the curve $y^2 = x^3 + 2x + 1$, we find that when $r \equiv 0 \pmod 6$, then $k = 1$. When $r \equiv 3 \pmod 6$, we get $k = 2$. When $r \equiv 2 \pmod 6$ or $r \equiv 4 \pmod 6$, we get $k = 3$. When $r \equiv 1 \pmod 6$ or $r \equiv 5 \pmod 6$, we get $k = 6$. Finally, with p a prime greater than 4, when r is odd we have $k = 2$ and when r is even, $k = 1$ or $k = 3$.

With complex multiplication techniques, it is also possible to construct efficiently curves with k up to 6. These constructions are described in [22] and [35]. In particular, complex multiplication techniques give a construction for curves of trace 2. However, they only work when the number of points of the curve, i.e., $q - 1$, is a square or a small multiple of a square. Allowing for slower key generation, larger values of k are possible (see [5] and [10]). The current record, set in [10], is $k = 50$.

4. Single Point Approach

In Section 3 our first try to devise a tripartite Diffie-Hellman protocol used a single point P. However, this simple approach did not work and we had to switch to a two points variation. In this section we revisit the first proposal and see that with some small changes a single point variant of the protocol can work. We describe three different techniques to reach this goal and examine their respective advantages and drawbacks.

The first technique was hinted at in our initial proposal, and it involves curves of trace 2, i.e., curves with $q - 1$ points. Let E be a curve of trace 2 and let ℓ be a prime dividing $q-1$. Moreover, assume that ℓ^2 does not divide $q-1$, then the curve E contains exactly ℓ points of ℓ-torsion. Indeed, any elliptic curve contains either ℓ or ℓ^2 points of ℓ-torsion, moreover, in the latter case, ℓ^2 must divide the cardinality of the curve. In this context, any non-zero ℓ-torsion point P generates the full subgroup of ℓ-torsion. As a consequence, any ℓ-torsion divisor can be written in term of P and its multiples. Since the Tate pairing is non-degenerate on the group of ℓ-torsion divisors, the basic proposal of Section 3 must work using for parameters the curve E, the point P and the function F_{T} derived from the Tate pairing. As a consequence, this approach seems to meet all the requirements for a single point pairing protocol. However, there is a major drawback, namely, we known of no algorithm to construct such a curve E with trace 2 efficiently.

Indeed, the only efficient known approach to building trace 2 curves is by using complex multiplication techniques. However, in that case the number of points on the curve E is a small multiple of a square. Thus, we cannot find a sufficiently large prime ℓ such that ℓ divides $q-1$ and ℓ^2 does not.

The second technique stems from a proposal of Verheul in [43] and uses an additional property of supersingular curves. More precisely, on a supersingular curve, the group of points has more automorphisms than on an ordinary curve. These extra automorphisms were called distortions by Verheul. They nicely map points defined over the base field to points defined over an extension field. This implies that the image $\varphi(P)$ of a point P defined over the base field by a distortion φ must be linearly independent from P. As a consequence, the Weil pairing of P and $\varphi(P)$ is non-trivial, i.e., $e_\ell(P, \varphi(P)) \neq 1$. Then, on the group generated by P, i.e., if ℓ is prime on the set of all ℓ-torsion points defined over the base field, we define a modified pairing $\hat{e}_\ell$ as follows. Let R and S be two multiples of P, then $\hat{e}_\ell(R, S)$ is defined as

$$\hat{e}_\ell(R, S) = e_\ell(R, \varphi(S)).$$

This modified pairing is symmetric and non-degenerate. It satisfies all the required properties for realizing a single point tripartite Diffie–Hellman protocol. For better efficiency, it is also possible to define similarly a modified Tate pairing $\hat{t}_\ell$ as

$$\hat{t}_\ell(R, S) = t_\ell(R, \varphi(S)).$$

The only potential drawback is that this construction applies only to supersingular curves. For a list of possible distortions on some useful supersingular curves, we refer the reader to [17] and [19].

The third technique works as follows. Given any curve E over $\mathbb{F}_p$, and ℓ a large prime divisor of its cardinality, such that ℓ divides $q^k - 1$, we know that the Tate pairing is non degenerate on the group of ℓ-torsion points which contains ℓ^2 elements. Then, according to [19], two possibilities arise, either the Tate pairing is anti-symmetric on the curve and (up to a constant power) is equal to the Weil pairing, or it is not anti-symmetric. In the latter case, among $\ell + 1$ subgroups of the ℓ-torsion, at most two are self-degenerate. So by choosing such a curve together with a random point P, it is easy to verify whether or not the subgroup generated by P can be used for one point pairing systems. Indeed, it suffices to check that $t_\ell(D_P, D_P) \neq 1$. Of course, as explained in Section 3, one of the two instances of D_P must be replaced by an equivalent divisor $(P+R) - (R)$. The main drawback with this construction is the lack of a "distinguished" subgroup as in the case of supersingular curves. With supersingular curves, the distinguished subgroup is the subgroup of points defined over the base field. It is useful in some applications such as identity-based encryption since it provides a way to construct random points in the subgroup with an unknown logarithm (in [6] this is done by using a function called `MapToPoint`).

5. Security Issues

Clearly in order to be secure, the tripartite Diffie–Hellman described here requires the discrete logarithm on the chosen elliptic curve to be hard, and the discrete logarithm in the finite field $\mathbb{F}_{q^k}$ to be hard. Since we placed ourselves in the cases where either the MOV or the FR reduction applies, we know that the hardness of the elliptic curve discrete logarithm problem implies the hardness of the finite field discrete logarithm problem. However, it is not known whether the elliptic curve discrete logarithm on a weak curve is as hard as the discrete logarithm in the corresponding finite field (in the sense of the MOV or FR reduction). In fact, this is a very interesting open problem. Moreover, as in the Diffie–Hellman case this is not the whole story, some Diffie–Hellman-like problem and Diffie–Hellman-like decision problem should be hard in order to get security.

Another important remark is that on curves where either the MOV or FR reduction applies, the usual Diffie–Hellman decision problem is mostly easy. Remember that the usual Diffie–Hellman problem is given a quadruple (g, g^a, g^b, g^c) to decide whether $c = ab$. Combined with the work of Maurer and Wolf (see [27]–[29]), this leads to a construction of groups where the DDH problem is easy, while the CDH and DL problems are equivalent and presumably hard. Such a construction is detailed in [19]. To deal with the two points version of pairing, it is useful to express the problem as follows. Given a quadruple (g, g^a, h, h^b), decide whether $a = b$. When h is in the group generated by g, the two formulations are equivalent. Now on an elliptic curve where the MOV (or alternatively the FR) reduction applies, we can easily test for a quadruple (P, aP, Q, bQ) whether $a = b$; it suffices to compute $e_\ell(aP, Q)$ and $e_\ell(P, bQ)$ and to compare them. This test works as soon as P and Q are independent (i.e., when $e_\ell(P, Q) \neq 1$). Note than when Q is a multiple of P, the test does not work, except when a single point pairing, as discussed in Section 4, is available.

With the current knowledge of elliptic curves, we believe that cryptosystems of this kind are secure in practice as soon as the discrete logarithm in $\mathbb{F}_{q^k}$ is hard. In the large characteristic, q^k should be at least a 1024-bit number. In the small characteristic, the best currently known algorithm for computing a discrete logarithm, i.e., the function field sieve (see [2], [1], and [18]), outperforms the large characteristic algorithms. As a consequence, in this case, q^k should be a larger number with 1536 or even 2048 bits. In both cases, we should work in large enough subgroups of the elliptic curve and $\mathbb{F}_{q^k}$, i.e., choose some large prime divisor ℓ of the order of the elliptic curve.

Security Assumptions and Their Relations

When using standard cryptographic groups in a discrete logarithm-based cryptosystem, it is well known that the security relies on one of the three following assumptions: the hardness of the discrete logarithm problem (DL), of the computational Diffie–Hellman problem (CDH) or of the decision Diffie–Hellman problem (DDH). When dealing with cryptographic groups that admit pairings, one cannot use the same set of basic problems. Indeed, the existence of a pairing implies that DDH becomes easy. In this section we describe some (hopefully) natural problems which are involved when using the single point approach. We assume that we are using the (modified) Tate pairing and we denote by $\mathbb{G}_1$ the group generated by the base point P. Following[3] Boneh and Franklin in [6],

[3] They were working with the modified Weil pairing and thus defined a one point Weil Diffie–Hellman problem. However, our definition is based on theirs.

we define the one point (modified) Tate Diffie–Hellman (TDH) problem as follows:

— Given (P, aP, bP, cP) for random a, b, c compute $\hat{t}(P, P)^{abc}$.

As noted in [6], the TDH assumption implies that CDH is hard in the group of points $\mathbb{G}_1$, it also implies that CDH is hard in the group $\mathbb{G}_2$ of roots of unity, where the pairing takes its values. The security of the IBE scheme from [6] is based on TDH in the random oracle model, thanks to the use of a hash function H. Without such a hash function, as in the tripartite Diffie–Hellman protocol described in Section 3, we need to assume the hardness of the decision problem associated with TDH, that we call DTDH. DTDH is defined as follows:

— Given (P, aP, bP, cP) a quadruplet of elements from $\mathbb{G}_1$ and $\hat{t}(P, P)^d$ an element of $\mathbb{G}_2$ for random a, b, c and d, decide whether $d = abc$.

The DTDH assumption implies DDH in $\mathbb{G}_2$ and CDH in $\mathbb{G}_1$ (remember that DDH in $\mathbb{G}_1$ is easy). The first implication can be shown by remarking that when DDH is easy in $\mathbb{G}_2$ then DTDH is also easy. Indeed, $d = abc$ if and only if $(\langle P, P\rangle, \langle aP, bP\rangle, \langle P, cP\rangle, \langle P, P\rangle^d)$ is a valid decision Diffie–Hellman instance.

Further, other related problems can be introduced to get a deeper understanding of the security of pairing-based systems. Before introducing these problems, we digress and ask the following question which arises quite naturally when looking at pairings: Can they be used as cryptanalytic tools to solve DDH in more general groups? Indeed, it suffices to find a group morphism from any group $\mathbb{G}_3$ to (one of the many possible) $\mathbb{G}_1$, to obtain an efficient algorithm for deciding DDH in $\mathbb{G}_3$. This would become even more interesting if we could choose for $\mathbb{G}_3$ the multiplicative subgroup of order ℓ of $\mathbb{F}_{q^r}$, i.e., $\mathbb{G}_2$ itself. Indeed, this would give a partial solution to solve DDH in some finite fields and would have a wide impact on many cryptographic schemes. Such an "attack" was recently proposed in [9]. It requires the construction of a special auxiliary curve, whose existence is conjectured by the authors of [9]. A recent preprint by Koblitz and Menezes [21] shows that the approach of [9] is completely flawed, since the existence of the required auxiliary curve is extremely unlikely. However, one might wonder about variants of this attack.

In fact, we can get strong evidence against the existence of this kind of attack by generalizing a result of Verheul from [43] and showing that any such attack would also lead to an efficient algorithm against the computational (and not only decision) Diffie–Hellman in the group $\mathbb{G}_3 = \mathbb{G}_2$. The result of Verheul was proved in the special case of the multiplicative subgroup of order $p^2 - p + 1$ in $\mathbb{F}_{p^6}$, which is sometimes called the XTR subgroup due to its relation to the XTR public key cryptosystem [23].

First, we describe more precisely how the DDH attack could work. As explained in [9] and [21], we need a group morphism φ from $\mathbb{G}_2$ (the multiplicative subgroup of order ℓ in $\mathbb{F}_{q^r}$) to $\mathbb{G}_1$ (an additive subgroup of order ℓ of an elliptic curve defined over $\mathbb{F}_{q^r}$). We also need to consider the modified Tate pairing $\hat{t}(\cdot, \cdot)$ that solves the DDH in $\mathbb{G}_1$ by mapping pairs of points to ℓth roots of unity (i.e., back to $\mathbb{G}_2$). Given g, g^a, g^b and g^c in $\mathbb{G}_3$, testing whether $c = ab$ can be done as in [9] by computing $\hat{t}(\varphi(g), \varphi(g^c))$ and $\hat{t}(\varphi(g^a), \varphi(g^b))$ and testing equality. As long as φ is non-constant and $\hat{t}$ non-degenerate, we get an efficient way of testing DDH. However, given φ and $\hat{t}$ we can in fact do much more. Indeed, if g is a generator of the ℓth root of unity, then $\hat{t}(\varphi(g), \varphi(g))$ can be written as g^{λ}. Moreover, because of the non-degeneracy properties, $\hat{t}(\varphi(g), \varphi(g)) \neq 1$ and thus $\lambda \neq 0$. Thanks to the bilinearity of $\hat{t}$, we can now check that $\hat{t}(\varphi(g^a), \varphi(g^b)) = g^{\lambda ab}$. If we could remove the constant λ, then we would clearly be computing CDH. Assume that ℓ is prime, then to remove λ we proceed as follows. First note that

$$\lambda^{q-3} \equiv \lambda^{-2} \pmod{\ell}.$$

Moreover, thanks to the relation

$$\hat{t}(\varphi(g^{\lambda^i}), \varphi(g^{\lambda^j})) = g^{\lambda^{i+j+1}},$$

it is easy by using addition chains to compute $\Lambda = g^{\lambda^{q-3}} = g^{\lambda^{-2}}$. Now verify that $\hat{t}(\varphi(g^{\lambda ab}), \varphi(\Lambda)) = g^{ab}$, which gives the expected solution for the CDH problem in $\mathbb{G}_3 = \mathbb{G}_2$ (and also in $\mathbb{G}_1$) with two applications of the pairing $\hat{t}$.

As a consequence of this digression, we can now remark that the hardness of the CDH problem in either $\mathbb{G}_1$ or $\mathbb{G}_2$ implies that the Tate pairing is hard to invert when one side of the pairing is fixed. More precisely, it is hard to find a point R in $\mathbb{G}_1$ and a morphism φ from $\mathbb{G}_2$ to $\mathbb{G}_1$ such that, for all g in $\mathbb{G}_2$,

$$\hat{t}(R, \varphi(g)) = g.$$

We call this problem the fixed Tate inversion (FTI). A related, possibly easier, problem is: given g, find any pair of points (S, T) in $\mathbb{G}_1$ such that $\langle S, T\rangle = g$. We call it the generalized Tate inversion (GTI).This last problem also appears in the following relation: the hardness of the discrete logarithm in $\mathbb{G}_2$ implies either the hardness of either GTI or the discrete logarithm in $\mathbb{G}_1$. Indeed, when both the GTI and the discrete logarithm in $\mathbb{G}_1$ are easy, it is possible to compute the discrete logarithm in $\mathbb{G}_2$. Assume that $g = \langle P, P\rangle$ and h are two elements of $\mathbb{G}_2$. In order to find α such that $h = g^{\alpha}$, we first use GTI and find two points Q and R such that $\langle Q, R\rangle = h$. Using discrete logarithm computations in $\mathbb{G}_1$, we find a and b such that $Q = aP$ and $R = bP$. Then $h = \langle aP, bP\rangle = g^{ab}$ and $\alpha = ab$.

We summarize the relations between all the complexity assumptions involved with a single point pairing[4] in Fig. 1. Each arrow in the figure goes from a complexity assumption to a weaker one. The figure does not include the conditional and non-uniform equivalences between DL and CDH in a group that come from [29]. These equivalences hold when an auxiliary curve defined over $\mathbb{F}_{\ell}$ and of sufficiently smooth order is known.

$$\begin{tikzcd}
 & & \mathrm{CDH}_{\mathbb{G}_1} \arrow[rr] \arrow[dr] & & \mathrm{DL}_{\mathbb{G}_1} \arrow[d] & \\
\mathrm{DTDH} \arrow[r] \arrow[dr] & \mathrm{TDH} \arrow[ur] \arrow[dr] & \mathrm{GTI} \arrow[r] & \mathrm{FTI} \arrow[r] & \mathrm{DL}_{\mathbb{G}_2} \arrow[r, leftrightarrow] & \mathrm{DL}_{\mathbb{G}_1} \text{ or GTI} \\
 & \mathrm{DDH}_{\mathbb{G}_2} \arrow[r] & \mathrm{CDH}_{\mathbb{G}_2} \arrow[ur] & & &
\end{tikzcd}$$

Fig. 1. Relations between complexity assumptions in pairing cryptography.

[4] A similar, but much more complicated diagram could be drawn for the two points pairing case.

Note that in our case, $\mathbb{G}_1$ and $\mathbb{G}_2$ have the same cardinality ℓ and that the same auxiliary curve can serve this purpose for both groups.

6. Conclusion

In this article we described a generalization of the Diffie–Hellman protocol to three parties using the Weil or Tate pairing on elliptic curves. In recent years, this tripartite Diffie–Hellman protocol has been an essential tool for building new cryptographic protocols and applications.

References

[1] L. M. Adleman. The function field sieve. In *Algorithmic Number Theory*, volume 877 of Lecture Notes in Computer Science, pages 108–121. Springer-Verlag, Berlin, 1994.

[2] L. M. Adleman and M. A. Huang. Function field sieve method for discrete logarithms over finite fields. In *Information and Computation*, volume 151, pages 5–16. Academic Press, New York, 1999.

[3] S. Al-Riyami and K. Paterson. Tripartite authenticated key agreement protocols from pairings. http://eprint.iacr.org, 2002.

[4] P. Barreto, H. Kim, B. Lynn, and M. Scott. Efficient algorithms for pairing-based cryptosystems. In M. Yung, editor, *Proceedings of CRYPTO* '2002, volume 2442 of Lecture Notes in Computer Science, pages 354–368. Springer-Verlag, Berlin, 2002.

[5] P. Barreto, B. Lynn, and M. Scott. Constructing elliptic curves with prescribed embedding degrees. http://eprint.iacr.org, 2002.

[6] D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, editor, *Proceedings of CRYPTO* '2001, volume 2139 of Lecture Notes in Computer Science, pages 213–229. Springer-Verlag, Berlin, 2001.

[7] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In C. Boyd, editor, *Proceedings of ASIACRYPT* '2001, volume 2248 of Lecture Notes in Computer Science, pages 514–532. Springer-Verlag, Berlin, 2001. Updated version available from the authors.

[8] M. Burmester and Y. Desmedt. A secure and efficient conference key distribution system. In A. De Santis, editor, *Advances in Cryptology — EUROCRYPT* '94, volume 950 of Lecture Notes in Computer Science, pages 275–286. Springer-Verlag, Berlin, 1995.

[9] Q. Cheng and S. Uchiyama. Nonuniform polynomial time algorithm to solve decisional Diffie–Hellman problem in finite fields under conjecture. In *CR–RSA* 2002, number 2271 in Lecture Notes in Computer Science, pages 290–299. Springer-Verlag, Berlin, 2002.

[10] R. Dupont, A. Enge, and F. Morain. Building curves with arbitrary small MOV degree over finite prime fields. http://eprint.iacr.org, 2002.

[11] M. Fouquet, P. Gaudry, and R. Harley. An extension of Satoh's algorithm and its implementation. *J. Ramanujan Math. Soc.*, 15:281–318, 2000.

[12] G. Frey and H. Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Math. Comput.*, 62:865–874, 1994.

[13] S. Galbraith, K. Harrison, and D. Soldera. Implementing the tate pairing. In C. Fieker and D. Kohel, editors, *Algorithmic Number Theory*, volume 2369 of Lecture Notes in Computer Science, pages 324–337. Springer-Verlag, Berlin, 2002.

[14] R. Harasawa, J. Shikata, J. Suzuki, and H. Imai. Comparing the MOV and FR reductions in elliptic curve cryptography. In J. Stern, editor, *Advances in Cryptology — EUROCRYPT* '99, volume 1592 of Lecture Notes in Computer Science, pages 190–205. Springer-Verlag, Berlin, 1999.

[15] R. Harley. Elliptic curve point counting: 32 003 bits. Available at http://listserv.nodak.edu/archives/-nmbrthry.html, August 2002.

[16] A. Joux. A one round protocol for tripartite Diffie–Hellman. In W. Bosma, editor, *Algorithmic Number Theory*, volume 1838 of Lecture Notes in Computer Science, pages 385–394. Springer-Verlag, Berlin, 2000.

[17] A. Joux. The Weil and Tate pairings as building blocks for public key cryptosystems. In C. Fieker and D. Kohel, editors, *Algorithmic Number Theory*, volume 2369 of Lecture Notes in Computer Science, pages 20–32. Springer-Verlag, Berlin, 2002.

[18] A. Joux and R. Lercier. The function field sieve is quite special. In C. Fieker and D. Kohel, editors, *Algorithmic Number Theory*, volume 2369 of Lecture Notes in Comput Science, pages 431–445. Springer-Verlag, Berlin, 2002.

[19] A. Joux and K. Nguyen. Separating decision Diffie–Hellman from computational Diffie–Hellman in cryptographic groups. *J. Cryptology*, 16(4):239–247, 2003.

[20] H. Kim, J. Park, J. Cheon, J. Park, J. Kim, and S. Hahn. Fast elliptic curve point counting using gaussian normal basis. In C. Fieker and D. Kohel, editors, *Algorithmic Number Theory*, volume 2369 of Lecture Notes in Computer Science, pages 292–307. Springer-Verlag, Berlin, 2002.

[21] N. Koblitz and A. Menezes. Obstacles to the torsion-subgroup attack on the decision Diffie–Hellman problem. Technical Report CORR 2002-05, CACR, 2002. Available at http://www.cacr.math.uwaterloo-.ca/tech_reports.html.

[22] G.-J. Lay and H. Zimmer. Constructing elliptic curves with given group order over large finite fields. In L. Adleman, editor, *Algorithmic Number Theory*, volume 877 of Lecture Notes in Computer Science, pages 250–263. Springer-Verlag, Berlin, 1994.

[23] A. Lentra and E. Verheul. The XTR public key system. In Mihir Bellare, editor, *Proceedings of CRYPTO* '2000, volume 1880 of Lecture Notes in Computer Science, pages 1–19. Springer-Verlag, Berlin, 2000.

[24] R. Lercier. Algorithmique des courbes elliptiques dans les corps finis. Thèse, École polytechnique, June 1997.

[25] R. Lercier and D. Lubicz. Elliptic curve point counting, 100 002 bits. Available at http://-listserv.nodak.edu/archives/nmbrthry.html, December 2002.

[26] A. Lysyanskaya. Unique signatures and verifiable random functions from the DH–DDH separation. In M. Yung, editor, *Proceedings of CRYPTO* '2002, volume 2442 of Lecture Notes in Computer Science, pages 597–612. Springer-Verlag, Berlin, 2002.

[27] U. Maurer. Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms. In *Advances in Cryptology - CRYPTO* '94, volume 839 of Lecture Notes in Computer Science, pages 271–281. Springer-Verlag, Berlin, 1994.

[28] U. Maurer and S. Wolf. Diffie–Hellman oracles. In N. Koblitz, editor, *Advances in Cryptology - Crypto* '96, Volume 1109 of Lecture Notes in Computer Science, pages 268–282. Springer-Verlag, Berlin, 1996.

[29] U. Maurer and S. Wolf. The relationship between breaking the Diffie–Hellman protocol and computing discrete logarithms. *SIAM J. Comput.*, 28(5):1689–1721, 1999.

[30] A. Menezes. *Elliptic Curve Public Key Cryptosystems*. Kluwer, Dordrecht, 1994.

[31] A. Menezes, T. Okamoto, and S. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Trans. Inform. Theory*, 39:1639–1646, 1993.

[32] J.-F. Mestre. Lettre à Gaudry et Harley. Available from http://www.math.jussieu.fr/˜mestre/, December 2000.

[33] V. Miller. Short programs for functions on curves. Unpublished manuscript, 1986.

[34] V. Miller. Use of elliptic curves in cryptography. In H. Williams, editor, *Advances in Cryptology — CRYPTO* '85, volume 218 of Lecture Notes in Computer Science, pages 417–428. Springer-Verlag, Berlin, 1986.

[35] A. Miyaji, M. Nakabayashi, and S. Takano. New explicit conditions of elliptic curve traces for FR-reduction. *IEICE Trans. Fundamentals*, E84A(5):1234–1243, 2001.

[36] F. Morain. Classes d'isomorphismes des courbes elliptiques supersingulières en caractéristiques ≥ 3. *Util. Math.*, 52:241–253, December 1997.

[37] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing. In *SCIS*, Okinawa, Japan, 2000.

[38] T. Satoh. The canonical lift of an ordinary elliptic curve over a finite field and its point counting. *J. Ramanujan Math. Soc.*, 15:247–270, 2000.

[39] T. Satoh and K. Araki. Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. *Comment. Math. Univ. St. Paul.*, 47:81–92, 1998.

[40] I. Semaev. Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p. *Math. Comp.*, 67:353–356, 1998.

[41] J. Silverman. *The Arithmetic of Elliptic Curves*. Number 106 in Graduate Texts in Mathematics. Springer-Verlag, New York, 1992.

[42] N. Smart. The discrete logarithm problem on elliptic curves of trace one. *J. Cryptology*, 12:193–196, 1999.

[43] E. Verheul. Evidence that XTR is more secure than supersingular elliptic curve cryptosystems. In B. Pfizmann, editor, *Proceedings of EUROCRYPT* '2001, volume 2045 of Lecture Notes in Computer Science, pages 195–210. Springer-Verlag, Berlin, 2001.

[44] F. Zhang, S. Liu, and K. Kim. ID-based one round authenticated tripartite key agreement protocol with pairings. http://eprint.iacr.org, 2002.