Secure Delegation of Elliptic-Curve Pairing

Benoît Chevallier-Mames, Jean-Sébastien Coron, Noel Mccullagh, David Naccache, Michael Scott

Secure Delegation of Elliptic-Curve Pairing

Benoît Chevallier-Mames, Jean-Sébastien Coron[1], Noel McCullagh[2], David Naccache[3], and Michael Scott[2]

[1] Université du Luxembourg
6, rue Richard Coudenhove-Kalergi
L-1359 Luxembourg, Luxembourg
jean-sebastien.coron@uni.lu
[2] School of Computing, Dublin City University
Glasnevin, Dublin 9, Ireland
{noel.mccullagh, mike}@computing.dcu.ie
[3] École normale supérieure
Département d'informatique, Groupe de cryptographie
45, rue d'Ulm, F-75230 Paris CEDEX 05, France
david.naccache@ens.fr

Abstract. In this paper we describe a simple protocol for secure delegation of the elliptic-curve pairing. A computationally limited device (typically a smart-card) will delegate the computation of the pairing $e(A, B)$ to a more powerful device (for example a PC), in such a way that 1) the powerful device learns nothing about the points A and B, and 2) the limited device is able to detect when the powerful device is cheating.

Key-words: Elliptic-curve pairing, secure delegation protocol, Boneh-Franklin IBE.

1 Introduction

Since the discovery of the first practical identity-based cryptosystem based on the elliptic-curve pairing [1], pairing-based cryptography has become a very active research area. Many pairing-based protocols have been proposed with novel and attractive properties, for example for key-exchange [6] and digital signatures [3].

The increasing popularity of pairing-based cryptosystems and their foreseeable deployment in computationally constrained devices such as smart-cards and dongles spurred recent research in the implementation of pairing (*e.g.* [8]). Unfortunately, although pairing is a cubic-time operation, pairing implementation attempts in limited devices such as smart-cards reveal that the embedded code may be slow, resource-consuming and tricky to program.

Work done while authors Chevallier-Mames, Coron and Naccache were with Gemplus (now Gemalto). Authors McCullagh and Scott are also affiliated to NoreTech.

Given that several PC-based pairing libraries exist, it seems natural to find-out whether a smart-card could interact with such packages to privately compute the elliptic-curve pairing. Note that beyond preserving operands from preying eyes, the card must also ascertain that bogus libraries don't mislead it into generating wrong results.

In this paper, we propose a simple protocol for secure delegation of elliptic-curve pairing. A computationally limited device (for example a smart-card) will delegate the computation of the elliptic-curve pairing $e(A, B)$ to a more powerful device (for example a PC), in such a way that 1) the powerful device learns nothing about the points A and B, and 2) the limited device is able to detect when the powerful device is cheating. The limited device will restrict itself to simple curve or field operations. We also describe some efficient variants of our protocol if one of the points A and B or both are already publicly known, or when the point A can be considered as constant, as it is the case for the Boneh-Franklin identity-based encryption scheme [1].

2 Preliminaries

2.1 Bilinear Map

Our protocol for secure pairing delegation is actually more general than just elliptic-curve pairing : as most pairing-based cryptosystems, it works for any bilinear map. Therefore, we briefly review the basic facts about bilinear maps. We follow the notations in [2]. We refer the reader to [7] for an extensive background on elliptic-curve pairing.

1. $\mathcal{G}_1$ and $\mathcal{G}_2$ are two (additive) cyclic groups of prime order p;
2. G_1 is a generator of $\mathcal{G}_1$ and G_2 is a generator of $\mathcal{G}_2$;
3. ψ is a computable isomorphism from $\mathcal{G}_1$ to $\mathcal{G}_2$ with $\psi(G_1) = G_2$;
4. e is a computable bilinear map $e : \mathcal{G}_1 \times \mathcal{G}_2 \rightarrow \mathcal{G}_T$;
5. $\mathcal{G}_T$ is a multiplicative cyclic group of order p.

A bilinear map is a map $e : \mathcal{G}_1 \times \mathcal{G}_2 \rightarrow \mathcal{G}_T$ with the following properties :

1. Bilinear: for all $U \in \mathcal{G}_1, V \in \mathcal{G}_2$ and $a, b \in \mathbb{Z}$, $e(a \cdot U, b \cdot V) = e(U, V)^{a \cdot b}$
2. Non-degenerate: $e(G_1, G_2) \neq 1$

Note that the previous conditions imply that $e(G_1, G_2)$ is a generator of $\mathcal{G}_T$.

2.2 Computational Indistinguishability

We recall the notion of computational indistinguishability [5], which will be used in the definition of secure pairing delegation. Two distribution ensemble $X = \{X_n\}_{n\in\mathbb{N}}$ and $Y = \{Y_n\}_{n\in\mathbb{N}}$ are said to be computationally indistinguishable and denoted $X \stackrel{c}{\equiv} Y$ if for every (probabilistic) polynomial-time algorithm A, and every $c > 0$, there exists an integer N such that for all $n > N$

$$|\Pr[A(X_n) = 1] - \Pr[A(Y_n) = 1]| < \frac{1}{n^c}$$

3 Secure Pairing Delegation

In this section, we formalize the security notions for secure pairing delegation. Our setting is the following : a computationally limited device, called the card and denoted $\mathcal{C}$, will delegate the computation of the pairing $e(A, B)$ to a more powerful device, called the terminal and denoted $\mathcal{T}$. Both devices $\mathcal{C}$ and $\mathcal{T}$ are actually probabilistic polynomial-time Turing machines. We denote by $\text{View}_T(A, B)$ the terminal's view when interacting with $\mathcal{C}$ with points A, B. The terminal's view includes the randomness used by the terminal, and the data received from the card.

The security notions could be formalized in the general framework of secure multiparty computation (for standard definitions, see for example [4]). However, we observe that our setting is much simpler than for general secure multiparty computation : the terminal has no secret and outputs nothing; moreover only the terminal can be malicious. Therefore, we adapt the general notions for secure multiparty computation to our restricted setting. We obtain that a protocol for pairing delegation is secure if it satisfies the three following security notions :

Completeness : after completion of the protocol with an honest terminal, the card obtains $e(A, B)$, except with negligible probability.

Secrecy : a (possibly cheating) terminal should not learn any information about the points A and B. More formally, for any malicious terminal $\mathcal{T}$, there exists a simulator $\mathcal{S}$ such that for any A, B, the output of S is computationally indistinguishable from the terminal's view :

$$S \stackrel{c}{\equiv} \text{View}_T(A, B)$$

Note that the simulator S is not given A, B as input.

Correctness : The card should be able to detect a cheating terminal, except with negligible probability. More formally, for any cheating terminal $\mathcal{T}$ and for any A, B, the card outputs either $\perp$ or $e(A, B)$, except with negligible probability.

4 Our Protocol

In order to delegate the pairing computation, one could think of the following protocol. On input A, B, the card could generate random x, y and ask the terminal to compute the pairing :

$$\alpha = e(x \cdot A, y \cdot B)$$

The card would then recover $e(A, B)$ by simply computing :

$$e(A, B) = \alpha^{(x \cdot y)^{-1}}$$

However, it is easy to see that this is not a secure pairing delegation protocol. Namely, although the terminal learns nothing about A, B, the card cannot detect a cheating terminal. Namely, if the terminal outputs α^r for some r instead of α, the card will obtain $e(A, B)^r$ instead of $e(A, B)$, and will not be able to detect the cheating terminal. In the following, we describe a secure pairing delegation protocol, such that if the terminal is cheating, then the card outputs either the correct $e(A, B)$ or nothing with overwhelming probability.

4.1 Description

The card and the terminal are given as input a description of the groups $\mathcal{G}_1$, $\mathcal{G}_2$ and $\mathcal{G}_T$, and a description of the bilinear map $e : \mathcal{G}_1 \times \mathcal{G}_2 \to \mathcal{G}_T$. The card and the terminal receive the generators G_1 and G_2; we also assume that the card receives $e(G_1, G_2)$. The card is given as input the points A and B and must eventually output $e(A, B)$. Recall that $\mathcal{G}_1$, $\mathcal{G}_2$ and $\mathcal{G}_T$ are additive groups of order p.

1. The card generates a random $g_1 \in \mathbb{Z}_p$ and a random $g_2 \in \mathbb{Z}_p$, and queries the three following pairings to the terminal :

$$\alpha_1 = e(A + g_1.G_1, G_2), \quad \alpha_2 = e(G_1, B + g_2.G_2)$$
$$\alpha_3 = e(A + g_1.G_1, B + g_2.G_2)$$

2. The card checks that $\alpha_1, \alpha_2, \alpha_3 \in \mathcal{G}_T$, by checking that $(\alpha_i)^p = 1$ for $i = 1, 2, 3$. Otherwise, the card outputs $\perp$ and halts.
3. The card computes a purported value for $e(A, B)$:

$$e_{AB} = \alpha_1^{-g_2} \cdot \alpha_2^{-g_1} \cdot \alpha_3 \cdot e(G_1, G_2)^{g_1 g_2} \tag{1}$$

4. The card generates four random values $a_1, r_1, a_2, r_2 \in \mathbb{Z}_p$ and queries the pairing :

$$\alpha_4 = e(a_1.A + r_1.G_1, a_2.B + r_2.G_2)$$

5. The card computes :

$$\alpha_4' = (e_{AB})^{a_1 a_2} \cdot (\alpha_1)^{a_1 r_2} \cdot (\alpha_2)^{a_2 r_1} \cdot e(G_1, G_2)^{r_1 r_2 - a_1 g_1 r_2 - a_2 g_2 r_1} \tag{2}$$

and checks that $\alpha_4' = \alpha_4$. In this case, the card outputs e_{AB}; otherwise it outputs $\perp$.

4.2 Security Proof

The following theorem shows that our protocol is secure :

Theorem 1. *The previous protocol is a secure pairing delegation protocol.*

Proof. The completeness property is easily established. We obtain from the bilinear property :

$$e(A + g_1.G_1, B + g_2.G_2) = e(A, B) \cdot e(A, G_2)^{g_2} \cdot e(G_1, B)^{g_1} \cdot e(G_1, G_2)^{g_1 g_2}$$

Then, for an honest terminal, we have :

$$\alpha_1 = e(A + g_1.G_1, G_2) = e(A, G_2) \cdot e(G_1, G_2)^{g_1} \quad (3)$$
$$\alpha_2 = e(G_1, B + g_2.G_2) = e(G_1, B) \cdot e(G_1, G_2)^{g_2} \quad (4)$$
$$\alpha_3 = e(A + g_1.G_1, B + g_2.G_2) \quad (5)$$

Combining the four previous equations, we obtain :

$$\alpha_3 = e(A, B) \cdot (\alpha_1)^{g_2} \cdot (\alpha_2)^{g_1} \cdot e(G_1, G_2)^{-g_1 g_2}$$

which, using (1), shows that the card computes the correct $e_{AB} = e(A, B)$. Moreover, using :

$$\begin{aligned} \alpha_4 &= e(a_1.A + r_1.G_1, a_2.B + r_2.G_2) \\ &= e(A, B)^{a_1 a_2} \cdot e(A, G_2)^{a_1 r_2} \cdot e(G_1, B)^{r_1 a_2} \cdot e(G_1, G_2)^{r_1 r_2} \end{aligned}$$

we obtain from equations (3) and (4) :

$$\alpha_4 = (e_{AB})^{a_1 a_2} \cdot (\alpha_1)^{a_1 r_2} \cdot (\alpha_2)^{r_1 a_2} e(G_1, G_2)^{r_1 r_2 - a_1 g_1 r_2 - a_2 g_2 r_1}$$

which, using (2), gives $\alpha_4 = \alpha'_4$ and shows that the card eventually outputs the correct $e_{AB} = e(A, B)$.

The secrecy property follows from the fact that the terminal receives only random, independently distributed points in the groups $\mathcal{G}_1$ and $\mathcal{G}_2$. Therefore, the simulator $\mathcal{S}$ simply consists in running the terminal $\mathcal{T}$ with randomly generated points. The simulator's output and the terminal's view when interacting with $\mathcal{C}$ are then identically distributed.

The correctness property is established as follows : we show that if the value e_{AB} computed by the card at step 3 is not equal to $e(A, B)$, then the element α'_4 computed by the card at step 5 has a nearly uniform distribution in $\mathcal{G}_T$, independent from the terminal's view. Then, the probability that $\alpha_4 = \alpha'_4$ at step 5 will be roughly $1/p$. Therefore, the card will output $\perp$, except with negligible probability.

We let $U = a_1.A + r_1.G_1$ and $V = a_2.B + r_2.G_2$. Moreover, we let $a, b, u, v \in \mathbb{Z}_p$ be such that $A = a.G_1$, $B = b.G_2$, $U = u.G_1$, $V = v.G_2$, which gives :

$$u = a_1 \cdot a + r_1 \tag{6}$$
$$v = a_2 \cdot b + r_2 \tag{7}$$

The card checks that $\alpha_1, \alpha_2, \alpha_3 \in \mathcal{G}_T$. Therefore, we must have $e_{AB} \in \mathcal{G}_T$, and since $e(G_1, G_2)$ is a generator of $\mathcal{G}_T$, we can let $\beta_1, \beta_2, \beta_3 \in \mathbb{Z}_p$ be such that :

$$\alpha_1 = e(A, G_2) \cdot e(G_1, G_2)^{g_1+\beta_1} \tag{8}$$
$$\alpha_2 = e(G_1, B) \cdot e(G_1, G_2)^{g_2+\beta_2} \tag{9}$$
$$e_{AB} = e(A, B) \cdot e(G_1, G_2)^{\beta_3} \tag{10}$$

Therefore, the value e_{AB} is correct iff $\beta_3 = 0$.

From the previous observation, we also have $\alpha'_4 \in \mathcal{G}_T$. Therefore, we can assume that $\alpha_4 \in \mathcal{G}_T$, since otherwise $\alpha'_4 \neq \alpha_4$ and the card outputs $\perp$. Then we can let $\beta_4, \beta'_4 \in \mathbb{Z}_p$ be such that :

$$\alpha_4 = e(U, V) \cdot e(G_1, G_2)^{\beta_4} \tag{11}$$
$$\alpha'_4 = e(U, V) \cdot e(G_1, G_2)^{\beta'_4} \tag{12}$$

Therefore, the card outputs e_{AB} iff $\beta_4 = \beta'_4$.

In the following, we assume that $u \neq 0$ and $v \neq 0$. Since (u, v) is uniformly distributed in $\mathbb{Z}_p$, this happens with probability $(1 - 1/p)^2 \geq 1 - 2/p$.

We show that if $\beta_3 \neq 0$, then β'_4 has a nearly uniform distribution in $\mathbb{Z}_p$, independent from the terminal's view, and therefore $\beta_4 = \beta'_4$ happens with negligible probability.

From equations (2), (8), (9), (10) and (12), we obtain :

$$\beta'_4 = a_1 a_2 \beta_3 + a_1 r_2 \beta_1 + a_2 r_1 \beta_2 \tag{13}$$

The terminal's view includes the points $A + g_1.G_1$, $B + g_2.G_2$, U and V and the group elements $\alpha_1, \alpha_2, \alpha_3$ and α_4. Therefore, the terminal's view is entirely determined by $(\beta_1, \beta_2, \beta_3, \beta_4, u, v, r)$, where r is the randomness used by the terminal. Moreover, given $(\beta_1, \beta_2, \beta_3, \beta_4, u, v, r)$, the element (a_1, a_2) is uniformly distributed over $\mathbb{Z}_p^2$.

From equations (6), (7) and (13), we obtain :

$$\beta'_4 = a_1 a_2 (\beta_3 - b\beta_1 - a\beta_2) + a_1(v\beta_1) + a_2(u\beta_2)$$

Lemma 1. *Let p be a prime integer and let $a, b, c, d \in \mathbb{Z}$ such that $(a, b, c) \neq (0, 0, 0)$. Then the number of solutions $(x, y) \in \mathbb{Z}_p^2$ to the polynomial equation $a \cdot xy + b \cdot x + c \cdot y + d = 0 \mod p$ is at most $2p - 1$.*

Proof. The proof is straightforward and is therefore omitted.

Since $u, v \neq 0$, then $\beta_3 \neq 0$ implies $(\beta_3 - b\beta_1 - a\beta_2, v\beta_1, u\beta_2) \neq (0,0,0)$. Then using the previous lemma, for any $\gamma \in \mathbb{Z}_p$, the probability over $(a_1, a_2) \in \mathbb{Z}_p^2$ that $\beta_4' = \gamma$ is such that :

$$\Pr[\beta_4' = \gamma] \leq \frac{2p-1}{p^2} \leq \frac{2}{p}$$

Therefore, if $\beta_3 \neq 0$, the probability that $\beta_4' = \beta_4$ is at most $2/p$.

Since we have that $u = 0$ or $v = 0$ with probability at most $2/p$, we conclude that if $e_{AB} \neq e(A, B)$, then the card outputs $\perp$, except with probability at most $4/p$. □

Note that the security of the protocol is not based on any computational assumption; namely the protocol achieves unconditional security.

4.3 Efficiency

Our protocol requires a total of 4 scalar multiplications in $\mathcal{G}_1$ and $\mathcal{G}_2$, and a total of 10 exponentiations in $\mathcal{G}_T$. Our protocol is actually a one-round protocol since the four pairing queries can be performed in the same round.

5 Efficient Variants with Public A or B

In this section, we describe more efficient variants of our protocol, when one of the points A and B or both are already publicly known.

For example, when decrypting with Boneh and Franklin's identity-based encryption scheme [1], the point A is the user's private key, and the point B is some part of the ciphertext. Therefore, the point B is already publicly known and does not need to be protected. Similarly, when encrypting with Boneh and Franklin's scheme, the point A is the recipient's identity, and the point B is the trusted party's public-key. Therefore, both points A and B are already publicly known and don't need to be protected.

When the point B is publicly known, the definition of the secrecy property is modified by simply giving B to the simulator. When both points A and B are publicly known, the secrecy property is not necessary anymore.

5.1 Secure Pairing Delegation with Public B

The protocol is the same as the protocol described in the previous section, except that we can take $g_2 = 0$ since the point B does not need to be protected.

1. The card generates a random $g_1 \in \mathbb{Z}_p$ and queries the three following pairings to the terminal :

$$\alpha_1 = e(A + g_1.G_1, G_2), \quad \alpha_2 = e(G_1, B), \quad \alpha_3 = e(A + g_1.G_1, B)$$

2. The card checks that $\alpha_1, \alpha_2, \alpha_3 \in \mathcal{G}_T$, by checking that $(\alpha_i)^p = 1$ for $i = 1, 2, 3$. Otherwise, the card outputs $\perp$ and halts.
3. The card computes a purported value for $e(A, B)$:

$$e_{AB} = (\alpha_2)^{-g_1} \cdot \alpha_3 \tag{14}$$

4. The card generates four random values $a_1, r_1, a_2, r_2 \in \mathbb{Z}_p$ and queries the pairing :

$$\alpha_4 = e(a_1.A + r_1.G_1, a_2.B + r_2.G_2)$$

5. The card computes :

$$\alpha'_4 = (e_{AB})^{a_1 a_2} \cdot (\alpha_1)^{a_1 r_2} \cdot (\alpha_2)^{a_2 r_1} \cdot e(G_1, G_2)^{r_1 r_2 - a_1 g_1 r_2} \tag{15}$$

and checks that $\alpha'_4 = \alpha_4$. In this case, the card outputs e_{AB}; otherwise it outputs $\perp$.

The protocol is more efficient than the protocol of Section 4 since only 3 scalar multiplications in $\mathcal{G}_1$ and $\mathcal{G}_2$, and 8 exponentiations in $\mathcal{G}_T$ are required.

Theorem 2. *The previous protocol with public B is a secure pairing delegation protocol.*

Proof. The proof is similar to the proof of theorem 1 and is therefore omitted.

5.2 Secure Pairing Delegation with Public A and B

The protocol is similar to the previous protocol except that we can also take $g_1 = 0$ since A does not need to be protected.

1. The card queries the three following pairings to the terminal :

$$\alpha_1 = e(A, G_2), \quad \alpha_2 = e(G_1, B), \quad \alpha_3 = e(A, B)$$

2. The card checks that $\alpha_1, \alpha_2, \alpha_3 \in \mathcal{G}_T$, by checking that $(\alpha_i)^p = 1$ for $i = 1, 2, 3$. Otherwise, the card outputs $\perp$ and halts.
3. The card computes a purported value for $e(A, B)$:

$$e_{AB} = \alpha_3$$

4. The card generates four random values $a_1, r_1, a_2, r_2 \in \mathbb{Z}_p$ and queries the pairing :

$$\alpha_4 = e(a_1.A + r_1.G_1, a_2.B + r_2.G_2)$$

5. The card computes :

$$\alpha'_4 = (e_{AB})^{a_1 a_2} \cdot (\alpha_1)^{a_1 r_2} \cdot (\alpha_2)^{a_2 r_1} \cdot e(G_1, G_2)^{r_1 r_2}$$

and checks that $\alpha'_4 = \alpha_4$. In this case, the card outputs e_{AB}; otherwise it outputs $\perp$.

The protocol is more efficient than the protocol of Section 4 since only 2 scalar multiplications in $\mathcal{G}_1$ and $\mathcal{G}_2$, and 7 exponentiations in $\mathcal{G}_T$ are required.

Theorem 3. *The previous protocol with public A and B is a secure pairing delegation protocol.*

Proof. The proof is similar to the proof of theorem 1 and is therefore omitted.

6 Efficient Variant for Constant Point

In this section, we provide two efficient variants of the previous protocol, when the point A can be considered as constant. In the first protocol, both points A and B are public, whereas in the second protocol, A is private whereas B is public.

Those two variants are particularly useful for Boneh and Franklin's identity-based encryption scheme [1]. Namely, when encrypting with Boneh and Franklin's IBE, the point B is the trusted server public-key, and the point A is the receiver's identity-based public-key. Therefore, B can be considered as constant, and both A and B are public. This corresponds to the first protocol (with constant B instead of constant A, but the protocol modification is straightforward).

Moreover, when decrypting with Boneh and Franklin's IBE, the point A is the user's private key, and the point B is some part of the ciphertext. Therefore, A can be considered as constant and private, whereas B can be considered as public. This corresponds to the second protocol.

6.1 Efficient Variant for constant A and public A, B

As in the previous protocol, the card and the terminal are given as input a description of the groups $\mathcal{G}_1$, $\mathcal{G}_2$ and $\mathcal{G}_T$, and a description of the bilinear map $e : \mathcal{G}_1 \times \mathcal{G}_2 \to \mathcal{G}_T$. Moreover, the card receives $e(A, Q)$ for some random $Q \in \mathcal{G}_2$. The point Q and $e(A, Q)$ are kept private by the card. The card is given as input the point B and must eventually output $e(A, B)$.

1. The card generates a random $r \in \mathbb{Z}_p$ and queries the following pairings to the terminal :
$$\alpha_1 = e(A, B), \quad \alpha_2 = e(A, r \cdot B + Q)$$
2. The card checks that
$$(\alpha_1)^r \cdot e(A, Q) = \alpha_2 \tag{16}$$
and that $(\alpha_1)^p = 1$. In this case, it outputs α_1, otherwise it outputs $\perp$.

The protocol is more efficient than the protocol of section 5.2 since it requires only one scalar multiplication and 2 exponentiations in $\mathcal{G}_T$.

Theorem 4. *The previous protocol with constant public A and public B is a secure pairing delegation protocol.*

Proof. The completeness property is straightforward to establish. The protocol's correctness is showed as follows :

Let b be such $B = b \cdot G_2$. Let q be such that $Q = q \cdot G_2$. Let
$$u = r \cdot b + q \mod p$$
which gives $r \cdot B + Q = u \cdot G_2$. We have that the terminal's view is entirely determined by (b, u) and by the randomness used by $\mathcal{T}$. Since r and q are randomly generated in $\mathbb{Z}_p$, we obtain that the distribution of r is independent from the terminal's view. Let β_1, β_2 be such that :
$$\alpha_1 = e(A, B) \cdot e(A, G_2)^{\beta_1}$$
$$\alpha_2 = e(A, r \cdot B + Q) \cdot e(A, G_2)^{\beta_2}$$
We have that β_1, β_2 are a function of the terminal's view, and that $\alpha_1 = e(A, B)$ if $\beta_1 = 0$. Moreover, we obtain from (16) that the card outputs α_1 iff :
$$r \cdot \beta_1 = \beta_2 \mod p \tag{17}$$
Assume now that $\beta_1 \neq 0$. Then since β_1 and β_2 are a function of the terminal's view, and the distribution of r is independent from the terminal's view, equality (17) holds with probability at most $1/p$. Therefore, for any cheating terminal, the card outputs either $\perp$ or the correct $e(A, B)$, except with probability at most $1/p$. □

6.2 Efficient variant for constant private A and for public B

As in the previous protocol, the card and the terminal are given as input a description of the groups $\mathcal{G}_1$, $\mathcal{G}_2$ and $\mathcal{G}_T$, and a description of the bilinear map $e : \mathcal{G}_1 \times \mathcal{G}_2 \to \mathcal{G}_T$. Moreover, the card receives $e(A, Q)$ for some random $Q \in \mathcal{G}_2$. The points A, Q and the value $e(A, Q)$ are kept private by the card. The card is given as input the point B and must eventually output $e(A, B)$.

1. The card generates random $x, y, z \in \mathbb{Z}_p$ and queries the following pairings to the terminal :
$$\alpha_1 = e(x \cdot A, B), \quad \alpha_2 = e(y \cdot A, z \cdot (B + Q))$$
2. The card computes :
$$e_{AB} = (\alpha_1)^{x^{-1}}, \quad \alpha_3 = (\alpha_2)^{(yz)^{-1}}$$
3. The card checks that
$$e_{AB} \cdot e(A, Q) = \alpha_3 \tag{18}$$
and that $(e_{AB})^p = 1$. In this case, it outputs e_{AB}; otherwise it outputs $\perp$.

The protocol is more efficient than the protocol of section 5.1 as it requires only 3 scalar multiplications and 3 exponentiations in $\mathcal{G}_T$.

Theorem 5. *The previous protocol with constant private A and public B is a secure pairing delegation protocol.*

Proof. The protocol's completeness is easily established. The protocol's secrecy follows from the fact that the terminal receives only randomly distributed points. The protocol's correctness is established as follows :

Let b be such $B = b \cdot G_2$. Let q be such that $Q = q \cdot G_2$. Let
$$u = z \cdot (b + q) \mod p$$
which gives $z \cdot (B + Q) = u \cdot G_2$. The terminal's view is then entirely determined by $(b, u, x \cdot A, y \cdot A)$ and by the randomness used by $\mathcal{T}$. Since z and q are randomly generated in $\mathbb{Z}_p$, we obtain that the distribution of z is independent from the terminal's view.

Let β_1, β_2 be such that :
$$\begin{aligned} \alpha_1 &= e(x \cdot A, B)^{1+\beta_1} \\ \alpha_2 &= e(y \cdot A, z \cdot (B + Q))^{1+\beta_2} \end{aligned}$$
We have that β_1 and β_2 are a function of the terminal's view. Moreover, we obtain :
$$\begin{aligned} e_{AB} &= e(A, B)^{1+\beta_1} \\ \alpha_3 &= e(A, B + Q)^{1+\beta_2} \end{aligned}$$
Therefore, $e_{AB} = e(A, B)$ iff $\beta_1 = 0$. Moreover, we obtain from (18) that the card outputs e_{AB} if :
$$e(A, B + Q)^{\beta_1} = e(A, B)^{\beta_2}$$

which gives :

$$b \cdot \beta_1 = (b + q) \cdot \beta_2 \mod p \tag{19}$$

Then since b, β_1, β_2 are a function of the terminal's view, and the distribution of q is uniform in $\mathbb{Z}_p$, independent of the terminal's view, we obtain that if $\beta_1 \neq 0$, the equality (19) holds with probability at most $1/p$. Therefore, for any cheating terminal, the card outputs either $\perp$ or the correct $e(A, B)$, except with probability $1/p$. □

7 Conclusion

In this paper we have described a simple protocol for secure delegation of elliptic-curve pairing. Our protocol allows a computationally limited device (for example a smart-card) to delegate the computation of the pairing $e(A, B)$ to a more powerful device (for example a PC), in such a way that 1) the powerful device learns nothing about the points A and B, and 2) the limited device is able to detect when the powerful device is cheating. We have also described more efficient variants of our protocol when one of the points or both are already publicly known, and when one of the points can be considered as constant.

We observe that our protocols achieve unconditional security. An interesting research direction would be to further optimize the protocols by trading-off unconditional security against computational security. A second interesting question consists in bounding the number of protocol rounds (passes) necessary to delegate pairing in diverse contexts.

References

1. D. Boneh and M. Franklin, *Identity based encryption from the Weil pairing*, SIAM J. of Computing, Vol. 32, No. 3, pp. 586-615, 2003. Extended abstract in proc. of Crypto '2001, LNCS Vol. 2139, Springer-Verlag, pp. 213-229, 2001.
2. D. Boneh, H. Shacham and B. Lynn, *Short signatures from the Weil pairing*. In proceedings of Asiacrypt '01, LNCS Vol. 2248, Springer-Verlag, pp. 514-532, 2001.
3. D. Boneh and X. Boyen, *Short Signatures Without Random Oracles*. In proceedings of Eurocrypt 2004, LNCS 3027, pp. 56-73, 2004.
4. R. Canetti, *Security and Composition of Multiparty Cryptographic Protocols*, Journal of Cryptology, (2000) 13: pp. 143–202.
5. S. Goldwasser and S. Micali, *Probabilistic Encryption*, JCSS, vol. 28, No. 2, 1984, pp. 270-299. Previous version in STOC 2002.
6. A. Joux, *A one round protocol for tripartite Diffie-Hellman*. In proceedings of ANTS IV, LNCS vol 1838, pp. 385–394. Springer-Verlag, 2000.
7. A. Menezes, *Elliptic Curve Public Key Cryptosystems*. Kluwer Academic Publishers, 1993.
8. M. Scott and P. Barreto, *Compressed Pairings*, Proceedings of Crypto 2004, LNCS vol. 3152, 2004.