New Key Agreement Protocols Providing Explicit Authentication

Mohamed Nabil, Yasmine Abouelseoud, Galal Elkobrosy, Amr Abdelrazek
Engineering Mathematics and Physics Department
Faculty of Engineering, Alexandria University
Alexandria, Egypt
{eng.m.nabil.m, yasmine.abouelseoud}@gmail.com, {elkobrosy, amr_abdelrazek_62}@yahoo.com

ABSTRACT: *Establishing a common secret key between two or more entities is an essential component of any security system. In this paper, the problem of establishing a shared session key in an authenticated way is addressed. The proposed key agreement protocols support explicit authentication. The protocols are designed in such a way that permits a trusted third party, such as a firewall, to verify the identities of the parties involved in a key agreement session. This reduces the computational burden at the end-user side, which can be a device with limited capabilities. The proposed protocols have been implemented, based on the PBC library provided by Stanford University, to ensure their correctness, the ease of their practical use and their timeliness. A comparative study of the proposed protocols to other protocols in literature revealed the superiority of the proposed protocols from the security viewpoint at the cost of an acceptable increase in the computational burden.*

Keywords: Authentication, Public Key Infrastructure (PKI), Key Agreement, Security, Bilinear Maps

Received: 22 November 2012, Revised 31 December 2012, Accepted 6 January 2013

1. Introduction

The rapid progress in wireless mobile communication technology has prompted new security questions. Since open air is used as the communication channel, the content of the communication may be exposed to an eavesdropper, or system services can be used fraudulently. An authentication mechanism and a key establishment protocol together form the very starting point from which most security services such as data confidentiality and non-repudiation can be provided. Symmetric or asymmetric techniques can be used for key exchange between parties in a wireless network. Protocols based on asymmetric-key encryption algorithms are the main solution for ad-hoc wireless networks because there is no need for a trusted third party in these protocols. On the other hand, symmetric-key encryption algorithms are applicable to the wireless networks with base stations [1], yet asymmetric-key algorithms could still be used to simplify key management of network users and provide more security services.The importance and the need for efficient secure key agreement protocols in communication systems is clear as they are essential components in the Universal Mobile Telecommunications System (UMTS) and the IEEE 802.11 wireless local area network (WLAN).

The Diffie-Hellman key agreement protocol [2] is the earliest example of an asymmetric key establishment technique. It is still found in many secure connectivity protocols over the Internet. It is a cryptographic protocol that enablesthe secure construction

of a shared secret between two parties over an untrusted network. The two parties may have never met previously, but with their new shared secret key they can encrypt their communications over the insecure channel to ensure the confidentiality of their communicated messages. However, this protocol does not involve the verification of the identities of the communicating entities, and thus suffers from man-in-the-middle attack. Different approaches have been developed to address this problem, which rely on the use of long-term keys for authentication purposes [3,4].The authentication phase may be done as a separate phase (explicit authentication), or the authentication is achieved implicitly if the key generated passes some verification check. In the former case, authentication can be done by gateways in networks reducing the computational load on the end-user and providing better security guarantees by forbidding impersonation attacks from both outsiders as well as insiders.

Tripartite key agreement protocols are of particular importance. They are useful in providing essential security in several vital applications such as in e-commerce where the three entities involved in the protocol are the merchant, the customer and the bank. Other interesting applications include a third party being added to chair or referee a conversation for the purpose of ad-hoc auditing, data recovery or escrow purposes [5, 6].

In this paper, new authenticated key agreement protocols are developed based on the existence of a traditional PKI (Public Key Infrastructure) within which the entities involved in the protocols are registered. Thus, every user randomly chooses its private key and computes the corresponding public key using a suitable one-way trapdoor function. Both two-party and three-party protocols are developed. Extensions to more than three participants have been also considered. The proposed protocols provide explicit authentication. The security properties of these protocols are examined and performance enhancements are suggested.

The rest of the paper is organized as follows. In the next section, elliptic curves, bilinear mapsand the computationally hard related problems are reviewed. Section III gives details on the desirable security properties for a sound key agreement protocol. Section IV offersa review of some related key agreement protocols. Section V describes our proposed schemes for two and three parties. The performance and security properties of the proposed protocols are examined in Section VI. A comparative study is provided in the section that follows. In Section VIII, a brief account on our implementation of the proposed protocols. Finally, Section IX concludes the paper.

2. Basic Concepts

In this section, some preliminary concepts necessary to the development of the proposed protocols are introduced.

2.1 Elliptic Curves

Recently, elliptic curves have received much attention in the field of cryptography. They are slowly replacing finite fields in the design of new cryptographic schemes. This is due to the fact that the discrete logarithm problem (defined below) over well-chosen elliptic curves is more difficult than the corresponding problem over finite fields. Consequently, smaller key sizes, in the order of 160 bits instead of 256 bits, can be used while achieving the same level of security [7].

An elliptic curve E [8] over a finite field F_p is defined by the Weirestrass equation

$$y^2 = x^3 + ax^2 + bx + c \tag{1}$$

Where $a^2b^2 - 4a^3c - 4b^3 + 18abc - 27c^2 \neq 0$ and $x \in F_p$ with p a prime greater than 3.

For efficiency purposes, usually a point over an elliptic curve is stored in compressed format. In compressed format, the x-coordinate is only stored along with a single bit indicating whether the positive or negative square root of $x^3 + ax^2 + bx + c$ is the designated y-coordinate.

An elliptic curve E over the finite field Z_p^* should be carefully chosen to avoid specialized attacks such as the MOV- attack and the FR- attack [9,10]. Specifications of safe elliptic curves can be found in [11].

The set of points on an elliptic curve E generated by some point P –in addition to the point at infinity O-together with the point addition operation forms an abelian group. The group addition law for elliptic curves over a field of characteristic greater than three is given below.

Let $P = (x_1, y_1) \in E$, then its inverse is defined to be the point $-P = (x_1 - y_1)$. If $Q = (x_2, y_2) \in E$ and $Q \neq -P$, then the addition operation can be defined as follows in terms of the chord and tangent method.

Case (1): If $P \neq Q$, then the chord $\overline{PQ}$ joining the two points intersects the curve in exactly one more point R. By definition, $P + Q = -R$. This is illustrated in Figure 1(a).

Case (2): If $P = Q$, then the tangent line at P intersects the curve at exactly one point R. By definition, $P + P = 2P = -R$, see Figure 1(b).

The coordinates of the point $R = (x_3, y_3)$ can be directly computed according to the following formulae

$$x_3 = \lambda^2 - x_1 - x_2 \tag{2}$$

$$y_3 = \lambda(x_1 - x_3) - y_1 \tag{3}$$

and

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1}, & \text{if } P \neq Q \\ \dfrac{3x_1^2 + a}{2y_1}, & \text{if } P = Q \end{cases}$$

To calculate the multiple of a point repeated doubling, in the manner of the repeated squaring technique, can be employed. For example, to compute $101P$, we double six times to compute $2P$, $4P$, $8P$, $16P$, $32P$, $64P$ and then three point additions are needed namely $((P + 4P) + 32P) + 64P = 101P$.

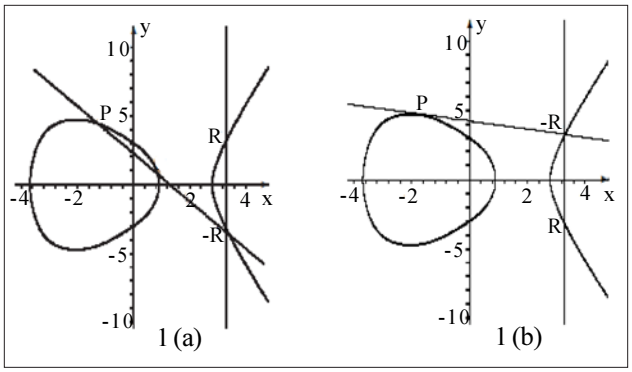

Figure 1. Elliptic curves point addition (a); and point doubling (b) illustrations over the field of real numbers

2.2 Bilinear Maps

Bilinear maps have been extensively used in the development of many cryptographic protocols during the last decade. Bilinear maps were used at first to mount cryptanalysis attacks against cryptographic schemes and then they found positive applications in cryptography [5, 6, 12, 13]. Many traditional certificate-based, as well as identity-based, key agreement protocols for two parties and three parties have been developedin literature based on the use of bilinear pairings. Some examples include Joux's one-round un-authenticated key agreement protocol and the four Tripartite Authenticated Key (TAK) agreement protocols (TAK-1, TAK-2, TAK-3, TAK-4) for sharing a session key among three parties [5, 6].

Bilinear maps and their properties are provided in what follows. More details can be found in Joux [5]. Consider the two groups G_1 (additive) and G_2 (multiplicative) of prime order q, and P a generator for G_1. A symmetric pairing is a computable bilinear map between these two groups.

For our purpose, let $\hat{e}$ be a symmetric bilinear map $\hat{e}: G_1 \times G_1 \rightarrow G_2$ which satisfies the following three properties.

1- **Bilinear:** if $P, Q \in G_1$ and $a, b \in Z_q^*$, then $\hat{e}(aP, bQ) = \hat{e}(abP, Q) = \hat{e}(P, Q)^{ab}$, $\hat{e}(P, Q + R) = \hat{e}(P, Q) \cdot \hat{e}(P, R)$.

2- **Non-degenerative:** there exist non-trivial points $P, Q \in G_1$ both of order q such that $\hat{e}(P, Q) \neq 1$.

3- **Computable:** if $P, Q \in G_1$, $\hat{e}(P, Q) \in G_2$ is efficiently computable in polynomial time.

The modified Weil pairing and the modified Tate pairing are commonly used instantiations of bilinear maps over elliptic curves [14].

2.3 Hard Computational Problems

Many pairing-based cryptographic protocols are based on the hardness of the BDHP (Bilinear Diffie-Hellman Problem) for their security [4]. Some computational problems related to the elliptic curve cryptography are defined below.

• Bilinear Diffie-Hellman Problem(BDHP)

Given $(P, xP, yP, zP) \in G_1$ for some x, y, z chosen at random from Z_q^*, compute $\hat{e}(P, P)^{xyz} \in G_2$.

• Discrete Logarithm Problem(DLP)

Given $P, Q \in G_1$ find an integer n such that $P = nQ$.

• Computational Diffie-Hellman Problem(CDHP)

Given a tuple $(P, aP, bP) \in G_1$ for $a, b \in Z_q^*$, find the element abP.

3. Desirable Security Properties of a Key Agreement Protocol

In order to develop a sound key agreement protocol, we need to understand the desirable security properties it must satisfy, which are described in detail in [15]. Here, assume A and B are two honest entities. It is desired for an authenticated key agreement protocol to possess the following security attributes as suggested in [16, 17].

3.1 Known-Key Security

Each key generated in one protocol round is independent and should not be exposed if other secret keys are compromised.

3.2 Forward Secrecy

If the long-term private keys of one or more of the entities are compromised, the secrecy of previously established session keys should not be affected. We say that a system has ***partial forward secrecy*** if some but not all of the entities' long-termkeys can be corrupted without compromising previously established session keys, and we say that a system has perfect ***forward secrecy*** if the long-term keys of all the entities involved may be corrupted without compromising any session key previously established by these entities.

3.3 Key-Compromise Impersonation

Assume that A and B are two entities. Suppose A's secret key is disclosed. Obviously, an adversary who knows this secret key can impersonate A to B. However, it is desired that this disclosure does not allow the adversary to impersonate B to the real A. In the two-party case, only an outsider would impersonate the communicating parties. However, in the n-party case for $n \geq 3$, one party of the communicating group might impersonate another party to the rest of the parties of the group. This kind of impersonation attack is called the insider impersonation attack.

3.4 Key Control

The key should be determined jointly by both A and B. Neither A nor B can control the key alone.

4. Related Work

The Diffie-Hellman key agreement protocol was developed by Diffie and Hellman in 1976 and published in the ground-breaking paper "*New Directions in Cryptography*" [2]. The protocol allows two users to exchange a secret key over an insecure medium without any prior secrets.

4.1 The Original Protocol

The original Diffie-Hellman protocol between two entities A and B is described below.

A and B begin by selecting an appropriate (large) prime p and a generator g of Z_q^*. The following steps must be taken each time a session key is required:

1. A selects an ephemeral random integer a; $1 \le a \le p-1$.

2. B selects an ephemeral random integer b; $1 \le b \le p-1$.

3. A and B then exchange the following messages, in either order:

$$A \to B: g^a \bmod P.$$

$$A \to B: g^b \bmod P.$$

4. On receipt of the message $g^b \bmod P$, A compute $K_A = (g^b \bmod P)^a \bmod P$, and on receipt of the message, B computes $K_B = (g^a \bmod P)^b \bmod P$. We find that $K_A = K_B = K = g^{ab} \bmod P$ which can be used as a secret session key shared between A and B. The ephemeral values a and b should be erased on completion of the protocol.

Intuitively, the security of this protocol appears to be related to the computational Diffie-Hellman (CDH) problem (defined in Section II) since a passive adversary would have to solve the CDH problem in order to determine the session key.

The original Diffie-Hellman protocol is in fact insecure against active adversaries since neither A nor B have any assurance of the source of the messages they receive or the identity of the party with whom they share the resulting key. This is demonstrated by a well-known attack on the original Diffie-Hellman protocol, known as a man-in-the-middle attack.

4.2 Authenticated Diffie-Hellman protocols

The use long-term keys is a solution to the authentication problem. One of the authenticated key agreement protocols presented by Blake-Wilson et al. is described in what follows[18].

As in the original DH-protocol, this protocol is defined between two entities A and B who can communicate over an open channel and who wish to generate a shared secret session key. Since the protocol uses public key techniques for authentication, A and B require public and private key pairs (x_A, x_A) and (x_B, x_B) respectively. We assume that p and q are large primes where q divides $(p-1)$, and g is an element of of Z_q^* order q. We also assume that H: $\{0, 1\}^* \to \{0, 1\}^k$ is a cryptographic hash function for a fixed value k (usually the security parameter). We assume that x_A and x_B are chosen randomly from Z_q^* and that $x_A = g_{x_A} \bmod p$ and $x_A = g_{x_B} \bmod p$.

The following steps must be taken each time a session key is required:

1. A selects an ephemeral random integer $a \in Z_q$.

2. B selects an ephemeral random integer $b \in Z_q$.

A and B then exchange the following messages, in either order:

$$A \to B: T_A = g^a \bmod p.$$

$$A \to B: T_B = g^b \bmod p.$$

On receipt of the message $g^b \bmod p$, A computes

$$K_A = H(T_B^a \bmod p, X_B^{x_A} \bmod p),$$

and on receipt of the message $g^b \bmod p$, B computes

$$K_B = H(T_A^b \bmod p, X_A^{x_B} \bmod p),$$

It is clear $K_A = K_B = K = H(g^{ab} \bmod p, g^{x_A x_B} \bmod p)$ that which can be used as a secret session key shared between A and B. The ephemeral values a and b are erased on completion of the protocol.

5. The Proposed Protocols

In this section, new schemes for authenticated key agreement are developed, which are extensions of the schemes summarized in [19] to the traditional PKI-based cryptosystems. These schemes consist of two phases, which are the setup phase and the session key generation phase. The setup phase is common to all schemes and it is described here.

Setup:
The system set up algorithm is responsible for generatingthe following parameters for the users. The public domain parameter ($p, q, E, P, \hat{e}, H$), where E is an elliptic curve defined over Z_p, P is a generator for a group of points on E (thegroupG_1) with order q. The hash function H is a one- way hash function that maps from G_1 into G_2 and is a bilinear map.

Each entity obtains a certificate for its static public key. Let $Cert_A$ denote A's public-key certificate, which includes her static public key $Q_A = aP$ and a certification authority (CA) signature over this information, where a is the long-term private key ofthe entity A.

5.1 Protocol 1
Suppose there are two entities A and B who want to agree on a session key. They exchange their public key certificates and the CA signature is verified.

5.1.1 Key generation
A and Bselect x, y randomly and independently, then theycompute and broadcast the following:

1. $A \rightarrow B : P_A = xP, T_A = aH(P_A) + xQ_A$
2. $B \rightarrow A : P_B = yP, T_B = bH(P_B) + xQ_B$

A verifies $\hat{e}(T_B, P) = \hat{e}(H(P_B) + P_B, Q_B)$

B also verifies $\hat{e}(T_A, P) = \hat{e}(H(P_A) + P_A, Q_A)$

If the above equations hold, then A and B compute:

$$K_A = \hat{e}(P_B, Q_B)^{ax}, K_B = \hat{e}(P_A, Q_A)^{by}$$

Then, the session key is $K_A = K_B = \hat{e}(P, P)^{axby}$

The correctness of the protocol can be easily verified as follows based on the properties of the bilinear map. The verification equation that A uses is only investigated and clearly similar arguments hold for B.

$$\begin{aligned}\hat{e}(H(P_B) + P_B, Q_B) &= \hat{e}(H(P_B) + yP, bP) \\ &= \hat{e}(bH(P_B) + byP, P) \\ &= \hat{e}(T_B, P)\end{aligned}$$

5.2 Protocol 2
This protocol extends the above protocol to the case where two entities A and B need to agree on a set of ***four*** session keys. The public key certificates are exchanged and the associated CA signatures are verified.

5.2.1 Key generation
A and B select the pairs (x, x') and (y, y') randomly and independently, and then compute and broadcast the following:

1. $A \rightarrow B : P_A = xP, P'_A = x'P, T_A = aH(P_A, P'_A) + xQ_A$
2. $B \rightarrow A : P_B = yP, P'_B = y'P, T_B = bH(P_B, P'_B) + yQ_B$

Upon receiving the broadcasted points, each entity proceeds to verify the authenticity of the received data.

A verifies $\hat{e}(T_B, P) = ?\ \hat{e}(H(P_B) + P_B, Q_B)$

B also verifies $\hat{e}(T_A, P) = ?\ \hat{e}(H(P'_A) + P_A, Q_A)$

If the above equations hold, then A and B compute the first key as:

$$K_{A(1)} = \hat{e}(P_B, Q_B)^{ax}, K_{B(1)} = \hat{e}(P_A, Q_A)^{by}$$

Then, the first session key is

$$K_{A(1)} = K_{B(1)} = \hat{e}(P, P)^{axby}$$

The remaining three session keys as will be computed by A are given below.

$$K_{A(2)} = \hat{e}(P_B, Q_B)^{ax'}, K_{A(3)} = \hat{e}(P_B, Q_B)^{ax}, K_{A(4)} = \hat{e}(P'_B, Q_B)^{ax'}$$

Again, the consistency check of the verification equation for one of the entities (A) is provided below based on the properties of the bilinear map.

$$\begin{aligned}\hat{e}(H(P_B, P'_B) + P_B, Q_B) &= \hat{e}(H(P_B, P'_B) + yP, bP) \\ &= \hat{e}(H(P_B, P'_B) + yP, bP) \\ &= \hat{e}(bH(P_B, P'_B) + yP, bP) \\ &= \hat{e}(T_B, P)\end{aligned}$$

5.3 Protocol 3

Suppose there are three entities A, B and C who want to agree on a session key. They exchange their public key certificates and the CA signature is verified.

5.3.1 Key generation

A, B and C select x, y, z randomly and independently, then they compute and broadcast the following:

1. $A \rightarrow B, C: P_A = xP, T_A = aH(P_A) + xQ_A$
2. $B \rightarrow A, C: P_B = yP, T_B = bH(P_B) + yQ_B$
3. $C \rightarrow A, B: P_C = zP, T_C = cH(P_C) + zQ_c$

A verifies $\hat{e}(T_B + T_C, P) = ?$

$$\hat{e}(H(P_B) + P_B, Q_B) \,.\, \hat{e}(H(P_C) + P_C, Q_C)$$

B also verifies $\hat{e}(T_A + T_C, P) = ?$

$$\hat{e}(H(P_A) + P_A, Q_A) \,.\, \hat{e}(H(P_C) + P_C, Q_C)$$

C also verifies $\hat{e}(T_A + T_B, P) = ?$

$$\hat{e}(H(P_A) + P_A, Q_A) \,.\, \hat{e}(H(P_B) + P_B, Q_B)$$

If the above equations hold, then A and B compute the first key as:

$$K_A = \hat{e}(P_B, P_C)^x, K_B = \hat{e}(P_A, P_C)^y, K_C = \hat{e}(P_A, P_B)^z$$

Then, the session key is

$$K_A = K_B = K_C = \hat{e}(P, P)^{xyz}$$

The correctness of the protocol can be easily verified as follows based on the properties of the bilinear map. The verification equation that A uses is only investigated and clearly similar arguments hold for $B$and C.

$$\hat{e}(H(P_B) + P_B, Q_B) \,.\, \hat{e}(H(P_C) + P_C, Q_C)$$

$$= \hat{e}(H(P_B) + P_B, Q_B) \,.\, \hat{e}(H(P_C) + P_C, Q_C)$$
$$= \hat{e}(H(P_B) + yP, bP) \,.\, \hat{e}(H(P_C) + zP, cP)$$
$$= \hat{e}(bH(P_B) + yQ_B, P) \,.\, \hat{e}(H(P_C) + zQ_C, P)$$
$$= \hat{e}(T_B + T_C, P)$$

5.4 Protocol 4

Again, the above protocol is extended to the case where there are three entities *A*, *B* and *C* who want to agree on a set of ***eight*** session keys. The public key certificates as usual are exchanged and the associated CA signatures are verified.

5.4.1 Key generation

A, *B* and *C* select the pairs (x, x'), (y, y') and (z, z') randomly and independently, and then compute and broadcast the following:

1. $A \rightarrow B, C : P_A = xP, P'_A = x'P$, and
   $T_A = aH(P_A, P'_A) + x\,Q_A$
2. $B \rightarrow A, C : P_B = yP, P'_B = y'P$, and
   $T_B = bH(P_B, P'_B) + y\,Q_B$
3. $C \rightarrow A, B : P_C = zP, P'_C = z'P$, and
   $T_C = cH(P_C, P'_C) + z\,Q_C$

Upon receiving the broadcasted points, each entity proceeds to verify the authenticity of the received data.

A verifies

$\hat{e}(T_B + T_C, P) = ?$
$$\hat{e}(H(P_B, P'_B) + P_B, Q_B) \,.\, \hat{e}(H(P_C, P'_C) + P_C, Q_C)$$

B also verifies

$\hat{e}(T_A + T_C, P) = ?$
$$\hat{e}(H(P_A, P'_A) + P_A, Q_A) \,.\, \hat{e}(H(P_C, P'_C) + P_C, Q_C)$$

C also verifies

$\hat{e}(T_A + T_B, P) = ?$
$$\hat{e}(H(P_A, P'_A) + P_A, Q_A) \,.\, \hat{e}(H(P_B, P'_B) + P_B, Q_B)$$

If the above equations hold, then *A* and *B* compute the first key as:

$$K_{A(1)} = \hat{e}(P_B, P_C)^x, K_{B(1)} = \hat{e}(P_A, P_C)^y, K_{C(1)} = \hat{e}(P_A, P_B)^z$$

Then, the session key is

$$K_{A(1)} = K_{B(1)} = K_{C(1)} = \hat{e}(P, P)^{xyz}$$

The remaining seven session keys as will be computed by *A* are given below.

$$K_{A(2)} = \hat{e}(P_B, P'_C)^x, K_{A(3)} = \hat{e}(P'_B, P_C)^x$$
$$K_{A(4)} = \hat{e}(P'_B, P'_C)^x, K_{A(5)} = \hat{e}(P_B, P_C)^{x'}$$
$$K_{A(6)} = \hat{e}(P_B, P'_C)^{x'}, K_{A(7)} = \hat{e}(P'_B, P_C)^{x'}$$
$$K_{A(8)} = \hat{e}(P'_B, P'_C)^{x'}$$

The consistency check of the verification equation is quite similar to that of protocol 3 and is thus omitted.

5.5 Protocol 5

The third protocol can be extended to the case of more than three users. Assume that a broadcast channel is available for the users. Each user i (from 1 to n) computes and broadcasts, $P_i = x_i P$, $T_i = a_i H(P_i) + x_i Q_i$ where x_i is the ephemeral key of user i and a_i is the long-term private key of this user with a corresponding public key. Each user i can authenticate his colleagues by applying the following verification equation.

$$\hat{e}(\Sigma_{j\in\{1,n\}/\{i\}} T_i, P) = \Pi_{j\in\{1,n\}/\{i\}} \hat{e}(H(P_j) + P_j, Q_j)$$

After the authentication phase succeeds, they will communicate over an unsecured channel to choose a coordinator to the conservation, suppose this coordinator is user number j. Each user will compute and broadcast the following information

$$X_i = \hat{e}(P_j, x_i(P_{i+1} - P_{i-1}))$$

Now, each user will compute the shared session key according to the following equation.

$$K_i = \hat{e}(P_j, nx_i P_{i-1}) \cdot X_i^{n-1} \cdot X_{i+1}^{n-2} \ldots X_{i-2}$$

The shared session key will be as follows.

$$K = \hat{e}(P, P)^{(x_1x_2 + x_2x_3 + \ldots + x_nx_1)x_j}$$

It is noteworthy that all user indexes are taken modulo n, as if the users were arranged on a hypothetical ring, as shown in Figure 2. This protocol is inspired by the work in [20], and extends it to the PKI-model.

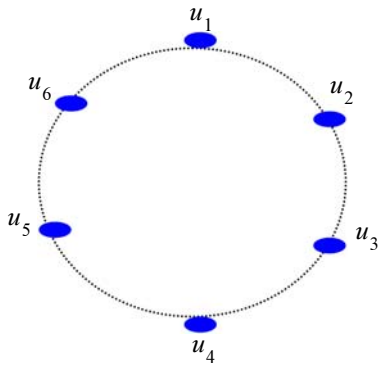

Figure 2. Arrangement of users in the multi-party variant of protocol 3

6. Performance Analysis and Security Analysis for the Proposed Protocols

In this section, the performance of the proposed schemes is investigated. In addition, the increase in computations involved in the schemes is justified due to the high security guarantees offered by these schemes and the possibility of off-loading some of the computational burden to a trusted third party such as a firewall.

6.1 Computational Burden

First, the two-party schemes are studied. Three (Four) scalar point multiplications and one (four) pairing evaluation are needed for the generation of the session key for protocol 1 (2). In addition, two pairing evaluations are required in the authentication phase for protocols 1 and 2; that is, verifying the identities of the parties involved in the protocol. However, it is clear from the verification equation that neither long-term nor short-term keys are required in this phase and thus the verification step can be done by a firewall reducing the computational load significantly.

As for the proposed three-party schemes, three (four) scalar point multiplications and one (eight) pairing evaluations are required for the generation of session keys for protocol 3 (protocol 4). In the authentication step, three pairing evaluations are needed for protocols 3 and 4. However, in protocol 4, since eight session keys are generated in one step, it can be envisioned that the computational load per key is just about one pairing evaluation and one-half of a scalar point multiplication. Again, the

verification equations in this phase involve no private keys and hence the computational load can be easily moved to a more powerful server such as a firewall.

6.2 Security Properties

The two and three party schemes security properties are examined in what follows.

6.2.1 Security Properties of Protocols 1 and 2

Known key security: In each run of these protocols, a new session key is computed which depends on short-term private keys x and y ((x, x') and (y, y')) selected randomly in each session. Thus, the knowledge of a past session key will not allow an adversary to deduce the future keys.

Partial forward secrecy: If the adversary knows the long-term private key of one entity, he will not be able to compute a previous session key. Assume, for example, that A's private key is compromised. It is clear that computing $\hat{e}(P_B, Q_B)^{ax}$ is infeasible without the knowledge of the short-term private key that is chosen randomly every session.However, if he knows long-term private keys of all entities, he will able to compute a previous session key by the relation $\hat{e}(P_A, P_B)^{ab}$. In practical scenarios, there is usually a highly secure end involved in the communication (a remote server), whose key compromise is rather difficult and thus the proposed protocol can still provide a desirable level of security.

Key control: All the entities contribute an equal share to the computation of the key. No one can force the session key to take on a specific pre-computed value.

Key-compromise impersonation: Suppose an adversary E knows the private key of A. He will not able to impersonate B to A unless he knows the private key of B, because of the fact that A authenticates B before computing the session key. No one can impersonate B unless he knows his private key; this is clear from the calculation of T_B.

6.2.2 Security Properties of Protocols 3 and 4

Known key security: In each run of Protocol 3 (4), keys are computed depending on short-term private key pairs x, y and z ((x, x'), (y, y') and (z, z')) which are selected randomly in each session.

Prefect forward secrecy: Even if the adversary knows the long-term private keys of all entities, he will not able to compute a previous session key. Assume, for example, that A's private key is compromised. It is clear that computing $\hat{e}(P_B, P_C)^x$ is infeasible without the knowledge of the short-term private key that is chosen randomly every session.

Key control: All the entities contribute an equal share to the computation of the key. No one can force the session key to take on a specific pre-computed value.

Key-compromise impersonation: Suppose an adversary E knows the private key of A. He will not able to impersonate B to A unless he knows private key of B, because A-before computing the session key- authenticates both B and C. No one can impersonate B or C unless he knows his private key; as is apparent from the calculations T_B of and T_C. Moreover, this protocol provides explicit authentication and not just implicit authentication, which makes this protocol resistant to the insider impersonation attack (suppose A, B and C are the communicating entities, insider impersonation means that one of them, suppose C, impersonates other entities like B to A. Thus, C will talk with A once as he is C and another time as if he is B). Explicit authentication avoids this attack. Furthermore, it similarly avoids outsider impersonation, as no one among communicating entities can impersonate one to the other, then no one outside them can mount this impersonation attack.

7. Comparative Study

In this section, we compare our protocols with other protocols with regard to security and performance. From the security point of view, the criterion to compare the security of the protocols is given by the extent to which a specific protocol fulfills the security properties as discussed in Section III. Fromthe performance point of view, the criterion for comparingthe efficiency is expressed in terms of the number of arithmeticoperations required per run of the protocol and the number of session keys generated in this run.

7.1 Security Comparison

The security comparison of the protocols is conducted with regard to three criteria: the fulfillment of security properties as

defined in Section III, and the existence of insider impersonation attack, and type of authentication (implicit, explicit).
Table Icompares the fulfillment of security properties of some 2-party protocolsin literature and our protocols. The following abbreviations and notations are used in Table 1 and Table 2:

KKS: Known-Key Secrecy, FS: Forward Secrecy,

KCI: Key-Compromise Impersonation, KC: Key Control,

IKA: Implicit Key Authentication,

EKA: Explicit Key Authentication, II: Insider Impersonation,

+: means protocol satisfies the property,

- : means protocol does not satisfy the property, *: perfect forward secrecy

Protocol	KKS	FS	KCI	KC	IKA	EKA
$ADHP_1$[18]	+	+*	-	+	+	-
$ADHP_2$[18]	+	+	+	+	+	-
MTI/A0[21]	+	+	+	+	+	-
Two-Pass Unified Model [21]	+	+*	-	+	+	-
Protocols 1 and 2	+	+	+	+	+	+

Table 1. Security Properties for 2-Party Protocols

Table 2 provides a comparison for the fulfillment of security propertiesfor some3-party protocols in literature and our protocols.

Protocol	KKS	FS	KCI	KC	IKA	EKA	II
TAK-1[6]	-	+*	-	+	+	-	-
TAK-2[6]	-	+	-	+	-	-	-
TAK-3[6]	+	-	-	+	+	-	-
TAK-4[6]	+	+*	-	+	+	-	-
Shim's [22]	+	+*	-	+	+	-	-
Protocols 3 and 4	+	+*	+	+	+	+	+

Table 2. Security Properties for 3-Party Protocols

It is clear from the above tables that the proposed protocols satisfy various security requirements of a key agreement protocol.

7.2 Efficiency Comparison
The computational load per user per key (number of computationsperformed) for the reviewed protocols as well as the proposed ones is given in Table 3 and Table 4.

We consider operations which are expensive from the computational point of view - pairing operations, scalar multiplications and exponentiations. The following abbreviations are used.

PairOpA: pairing operations in Authentication,

PairOpG: pairing operations in Generation,

ScMul: scalar point multiplications in G_1,

MULG2: scalar multiplications in G_2,

EXPMP: exponentiation modulo p,

MULMP: multiplication modulo p

Protocol	PairOpA	PairOpG	ScMul	EXPMP	MULMP
ADHP$_1$				3	
ADHP$_2$				3	
MTI/A0				3	1
Two-Pass Unified Model				3	
Protocol 1	2	1	3	1	
Protocol 2	2/4	4/4	4/4	4/4	

Table 3. Computational Load Per User in 2-Party Protocols

It is clear that, for frequently communicating parties with sufficient secure storage media, it is more efficient to use Protocol 2 rather than Protocol 1. Similar arguments hold for protocols 3 and 4, as apparent from the following table.

Protocol	PairOpA	PairOpG	ScMul	EXPMP	MULG2
TAK-1		2	1	2	1
TAK-2		3	1	3	2
TAK-3		3	1	3	2
TAK-4		1	1	1	2
Shim's Protocol		2	1	2	
Protocol 3	3	1	3	1	
Protocol 4	3/8	8/8	4/8	8/8	

Table 4. Computational Effort Per User of 3-Party Protocols

8. Implementation

The proposed four protocols have been implemented using the C++ PBC Library under Ubuntu operating system. Type A elliptic curves have been used in our sample runs for testing the validity and ensuring the timeliness of the proposed protocols.

Type A pairings are symmetric pairings constructed on the elliptic curve $y^2 = x^3 + x$ over the field F_q for some prime $q = 3 \bmod 4$. G_1 is the group of points $E(F_q)$. It turns out that $\#E(F_q) = q + 1$ and $\#E(F_q^2) = (q+1)^2$. Thus, the embedding degree k is 2, and hence G_2 is a subgroup of F_q^2. The order r is some prime factor of $q + 1$. Write $q + 1 = r * h$. For efficiency, r is picked to be a Solinas prime, that is, r has the form of $2^a \pm 2^b \pm 1$ for some integers $0 < b < a$. Moreover, choose $q = -1 \bmod 12$, so F_q^2 can be implemented as $F_q[i]$ (where $i = \sqrt{-1}$) [23]. The values used in one of the sample runs were:

q
67482757493960849107808804251905877426576536547233963656131460282213044792781368793846434454833639711994367788502369447668028429043299746806849694863238009858842252639881869011902897775185925452144670326607992336223363965380169867103259095832603178683592924084368913647031289576778910078145339638253871000123
h
92347147273700015298741233963742404178098136128074622330616838108561169824077048597452686611323248517470974394503822805818802865253410823165123164580695317179330104741886556392613373770104557671851186264142413721849719456188302996411048914825284267887978179132
r
730750862221594424981965739670091261094297337857
a 159
b 135

9. Conclusion

In this paper, new authenticated key agreement protocols offering high level security guarantees have been proposed. The authenticity of the identities of the communicating parties can be done by means of a firewall relieving the users involved from much of the computational burden associated with the authentication step. Moreover, employing dedicated hardware for pairing evaluations could greatly enhance the system performance.

Two of the proposed schemes addressed the problem of sharing a secret key between two entities. In addition, two tripartite key agreement protocols have been proposed. Furthermore, it has been demonstrated how one of the tripartite protocols can be extended to the case of more than three users based on arranging the communicating parties on a hypothetical ring structure.

All proposed schemes resist various known attacks promoting their use for highly confidential communications. Moreover, implementing the schemes on a Pentium(R) Dual Core PC revealed that the proposed protocols can be used in real-time applications. For devices with limited computational capabilities, the verification of user identities can be moved to a trusted third party as stated above.

References

[1] Mustafa Ergen. (2002). IEEE 802.11 Tutorial, University of California Berkeley. Available online at http://www.eecs.berkeley.edu/~ergen/docs/ieee.pdf.

[2] Diffie, W., Hellman, M. (1976). New directions in cryptography, *IEEE Transactions on Information Theory*, IT-22 (6) 644-654.

[3] Menezes, A., Van Oorschot, P. C., Vanstone, S. (1997). Handbook of Applied Cryptography, CRC Press, USA.

[4] Dutta, R., Barua, R. (2005). Overview of Key Agreement Protocols, Cryptology ePrint Archive, Report 2005/289.

[5] Joux, A. (2000). A one round protocol for tripartiteDiffie-Hellman, 4th International Symposium on AlgorithmicNumber Theory, LNCS Vol. 1838, Springer,UK, p. 385-393.

[6] Al-Riyami, Sattam S.(2004). Cryptographic Schemes based on Elliptic Curve Pairings,Information Security Group, Department of Mathematics, Royal Holloway, University of London.

[7] Giuliani, K. (1999). Attacks on the Elliptic Curve Discrete Logarithm Problem, Master of Mathematics, University of Waterloo, Ontario, Canada.

[8] Silverman, J. H. (1986). The Arithmetic of Elliptic Curves, GTM 106, Springer-Verlag.

[9] Menezes, A., Okamoto, T., Vanstone, S. (1993). Reducing Elliptic CurveLogarithm to Logarithms in a Finite Field, *IEEE Transactions on Information Theory*, 39, 1639-1646.

[10] Frey, G., Ruck, H. (1994). A Remark Concerning *m*-divisibility and the Discrete Logarithm Problem in the Divisor Class Group of Curves, *Mathematics of Computation*, 62, 865-874.

[11] Standards for Efficient Cryptography. (2000). SEC 2: Recommended EllipticCurves Domain Parameters, Certicom Research, Version 1.0.

[12] Sakai, R., Ohgishi, K., Kasahara, M. (2000). Cryptosystemsbased on pairing, Symposium on Cryptographyand Information Security (SCIS'2000), Japan.

[13] Chen, L., Kudla, C. (2003). Identity Based Authenticated Key Agreement Protocols from Pairings, 16th IEEE Computer Security Foundations Workshop, IEEE Press, USA, p. 219-233.

[14] Miller, V. (2004). The Weil Pairing and Its Efficient Calculation, *Journal of Cryptology*, 17 (4) 235-262.

[15] Kudla, C. (2006). Special Signature Schemes and Key Agreement Protocols, PhD Thesis, Royal Holloway University of London.

[16] Bellare, M., Rogaway. P. (1993). Entity Authenticationand Key Distribution, Advances in Cryptology -*CRYPTO* '93, Springer, UK, p. 232-249.

[17] Nalla, D., Reddy, K. C. (2003). ID-based tripartitekey agreement with signatures, *Cryptology ePrint*Archive, Report 2003/004.

[18] Blake-Wilson, S., Johnson, D., Menezes, A. (1992). Key agreement protocols and their security analysis(Extended abstract), 6th IMA International Conferenceon Cryptography and Coding, LNCS V. 1355, Springer, UK, p. 30-45.

[19] Marko Hölbl, Tatjana Welzer, Boštjan Brumen. (2009). Comparative Study of Tripartite Identity-Based Authenticated Key Agreement Protocols, *Informatica*, 33, 347–355.

[20] Xinjun Du, Ying Wang, Jianhua Ge, Yumin Wang. (2003). ID-based Authenticated Two Round Multi-Party Key Agreement, IACR Cryptology ePrint Archive 01/2003, 2003/247.

[21] Song, B., Kim, K. (2000), Two-Pass Authenticated Key AgreementProtocol with Key Confirmation, Progress in Cryptology – Indocrypt 2000, LNCS 1977, Springer-Verlag, p.237-24.

[22] Sun, S., Hsieh, B. (2003). Security analysis of Shim's authenticated key agreement protocols from Pairings. Cryptology ePrint Archive, Report 2003/113.

[23] Benn Lynn. (2006). PBC Library Manual 0.5.11.

Authors Biographies

Eng. Mohamed Nabil is a teaching and research assistant at the Engineering Mathematics department. He graduated from computer science and automatic control with honours in 2008. He is interested in cryptography and its applications with distinguished programming skills.

Dr. Yasmine Abouelseoud is an assistant professor at the Engineering Mathematics department. She received her B.Sc. in computer science in 2001 with honours. She is interested in computer algebra, cryptography and optimization techniques. She published research papers in these fields in international conferences.

Prof. Dr. Galal Elkobrosy is a full professor of Engineering Mathematics. His research interests include simulation and modeling. In addition, he is interested in statistics and time series analysis. He also has interests in number theory and cryptography.

Dr. Amr Abdelrazek is an assistant professor at Engineering Mathematics department. He is interested in numerical methods and their engineering applications. Especially, he is interested in solution of partial differential equations numerically and gained interest in cryptography recently.