ID-Based One Round Authenticated Tripartite Key Agreement Protocol with Pairings

Fangguo Zhang[1] Shengli Liu[2] and Kwangjo Kim[1]

[1] International Research center for Information Security (IRIS)
Information and Communications University(ICU),
58-4 Hwaam-dong Yusong-ku, Taejon, 305-732 KOREA
{zhfg, kkj}@icu.ac.kr

[2] Dept. of Computer Science and Engineering, Shanghai Jiao Tong University,
Shanghai 200030, P.R.China
liu-sl@cs.sjtu.edu.cn

Abstract. With various applications of Weil pairing (Tate pairing) to cryptography, ID-based encryption schemes, digital signature schemes, blind signature scheme, two-party authenticated key agreement schemes, and tripartite key agreement scheme were proposed recently, all of them using bilinear pairing (Weil or Tate pairing). In this paper, we propose an ID-based one round authenticated tripartite key agreement protocol. The authenticity of the protocol is assured by a special signature scheme, so that messages carrying the information of two ephemeral keys can be broadcasted authenticly by an entity. Consequently, one instance of our protocol results in eight session keys for three entities. Security attributes of our protocol are presented, and the computational overhead and bandwidth of the broadcast messages are analyzed as well.

Key words: Key agreement, Bilinear pairings, Identity-based cryptography.

1 Introduction

As the first practical solution to key agreement problem, the Diffie-Hellman key agreement protocol [8] enables two entities to establish a *session key*, which can be used to provide security or data integrity for later communications between the two entities. However, the basic Diffie-Hellman protocol does not authenticate the two communication entities, hence suffers from the "man-in-the-middle" attack. Over the years, different approaches have been developed to solve the problem (see [17, 16, 3, 25]).

Another direction of research on key agreement is to generalize the two-party key agreement to multi-party key agreement, amongst which the three-party case receives much interest. The recent work done by Joux [15] showed how to implement an elegant tripartite key agreement protocol using pairings: only one broadcast is required for each entity. Joux's protocol found very good applications to broadcast networks. However, just like the basic Diffie-Hellman protocol, Joux's protocol did not attempt to authenticate the three communicating entities, and is vulnerable to "man-in-the-middle" attack as well. To provide authenticity to tripartite key agreement, Al-Riyami *et.al.* lately presented several protocols in [1], also using pairing. Their proposals belong to the MTI and MQV family of protocols. The main idea is to use certificates of the three entities, which are issued by a Certificate Authority (CA), to bind an entity's identity with his static (long-term) keys. Then the final session key is generated by both ephemeral (short-term) keys and static keys. The authenticity of the static keys (provided by signatures of CA) assures that only the entities who possess the static keys are able to compute the session keys.

However, in a certificate system, before using the public key of a user, the participants must first verify the certificate of the user. As a consequence, this system requires a large amount of computing time and storage. In 1984 Shamir [23] asked for identity-based encryption and signature schemes to simplify key management procedures in certificate-based public key infrastructure. Since then, many ID-based encryption schemes and signature schemes have been proposed [4, 7, 27]. The idea of ID-based cryptosystems is that the identity information of a user functions as his public key. A key generation center which is trusted by all users is responsible for the generation of users' corresponding private keys. The bilinear pairings, namely the Weil pairing and the Tate pairing of algebraic curves, are important tools for research on algebraic geometry. The early applications of the bilinear pairings in cryptography were negative. For example, the MOV attack [18](using Weil pairing) and FR attack [10](using Tate pairing) reduce the discrete logarithm problem on some elliptic curves or hyperelliptic curves to the discrete logarithm problem in a finite field. Recently the bilinear pairings have been found positive application in cryptography [4, 5, 15, 22, 28]. More precisely, they are important tools for construction of ID-based cryptographic schemes. Many ID-based cryptographic schemes have been proposed using the bilinear pairings. Examples are Boneh-Franklin's ID-based encryption scheme [4], Smart's ID-based authentication key agreement protocol [24], several ID-based signatures schemes [6, 14, 20, 22], and an ID-based blind signature scheme and ring signature scheme [29], *etc.* With the construction of ID-based public key cryptosystems, ID-based public key infrastructure can be an alternative for certificate-based public key infrastructures, especially when efficient key management and moderate security are required.

In this paper, we propose a one round tripartite authenticated key agreement protocol to further consummate ID-based public key infrastructure.

The rest of the paper is organized as follows: The next section briefly explains the bilinear pairing and ID-based public key infrastructure. Section 3 gives a detailed description of our key agreement protocol. In Section 4, a heuristic security analysis is presented. Section 5 concludes this paper.

2 ID-Based Public Key Infrastructure with Pairing

In this section, we briefly describe the basic definition and properties of the bilinear pairing and the BDH Assumption. Then we present ID-based public key infrastructure based on pairing.

2.1 Bilinear Pairings and the BDH Assumption

Let G_1 be a cyclic additive group generated by P, whose order is a prime q, and G_2 be a cyclic multiplicative group of the same order q. We assume that the discrete logarithm problems (DLP) in both G_1 and G_2 are hard. Let $e : G_1 \times G_1 \rightarrow G_2$ be a pairing which satisfies the following conditions:

1. Bilinear: $e(P_1+P_2, Q) = e(P_1, Q)e(P_2, Q)$ and $e(P, Q_1+Q_2) = e(P, Q_1)e(P, Q_2)$;
2. Non-degenerate: There exists $P \in G_1$ and $Q \in G_1$ such that $e(P, Q) \neq 1$;
3. Computability: There is an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in G_1$.

We note that the Weil and Tate pairings associated with supersingular elliptic curves or abelian varieties can be modified to create such bilinear maps. We refer to [4, 5, 12, 13] for more details.

Definition 1. *The* **Bilinear Diffie-Hellman (BDH) Problem** *for a bilinear pairing* $e : G_1 \times G_1 \rightarrow G_2$ *is defined as follows: given* $P, aP, bP, cP \in G_1$, *compute* $e(P, P)^{abc}$, *where* a, b, c *are randomly chosen from* Z_q^*. *An algorithm is said to solve the BDH problem with an advantage of* ε *if*

$$Pr[\mathcal{A}(P, aP, bP, cP) = e(P, P)^{abc}] \geq \varepsilon.$$

BDH Assumption: We assume that the BDH problem is hard, which means there is no polynomial time algorithm to solve BDH problem with non-negligible probability.

2.2 ID-based Public Key Infrastructure

In ID-based public key infrastructure (IDPKC), everyone's public keys are predetermined by information that uniquely identifies them, such as name, social security number, email address, etc. rather than an arbitrary string. This concept was first proposed by Shamir [23]. Since then, many researchers devote their effort on ID-based cryptographic schemes [9, 26, 27, 19]. How to construct ID-based schemes using Weil or Tate pairings on supersingular elliptic curves or abelian varieties recently receives much research interest [4–6, 22, 24, 14, 20, 22, 29].

ID-based public key infrastructure involves a Key Generation Center (KGC) and users. The basic operations consists of **set up** and **private key extraction**. When we use bilinear pairings to construct IDPKC, **set up** and **private key extraction** can be implemented as follows:

Let P be a generator of G_1. Remember that G_1 is an additive group of prime order q and the bilinear pairing is given by $e : G_1 \times G_1 \rightarrow G_2$. Define two cryptographic hash functions $H : \{0, 1\}^* \rightarrow Z_q$ and $H_1 : \{0, 1\}^* \rightarrow G$.

– **Setup:** KGC chooses a random number $s \in Z_q^*$ and set $P_{pub} = sP$. The center publishes system parameters $params = \{G, q, P, P_{pub}, H, H_1\}$, and keep s as the *master-key*, which is known only by itself.

– **Private key extraction:** A user submits his identity information ID to KGC. KGC computes the user's public key as $Q_{ID} = H_1(ID)$, and returns $S_{ID} = sQ_{ID}$ to the user as his private key.

Having set up the above IDPKC, different ID-based cryptographic schemes have been proposed, including Boneh-Franklin's ID-based encryption scheme [4], and several ID-based signature schemes (see [14, 22]), etc. Smart recently presented an identity-based two-party authenticated key agreement protocol in [24]. This paper will focus on tripartite authenticated key agreement.

3 One Round ID-Based Tripartite Authenticated Key Agreement Protocol

In this section, we present a one round protocol for ID-based tripartite authenticated key agreement. One instance of the protocol results in multiple session keys.

Let A, B and C be the three entities who are going to agree to some session keys. The identities of A B and C are ID_A, ID_B and ID_C respectively. With the ID-based public key infrastructure, their public keys and private keys are as follows:

A's public key is $Q_A = H_1(ID_A)$, and the private key is $S_A = sQ_A$.

B's public key is $Q_B = H_1(ID_B)$, and the private key is $S_B = sQ_B$.

C's public key is $Q_C = H_1(ID_C)$, and the private key is $S_C = sQ_C$.

The pairs (Q_{ID}, S_{ID}) for A B and C serve as their static (or long term) public/private key pairs.

The protocol is as follows:

$$A \to B, C: \ P_A = aP, \ {P_A}' = a'P, \ T_A = H(P_A, {P_A}')S_A + a{P_A}'.$$

$$B \to A, C: \ P_B = bP, \ {P_B}' = b'P, \ T_B = H(P_B, {P_B}')S_B + b{P_B}'.$$

$$C \to A, B: \ P_C = cP, \ {P_C}' = c'P, \ T_C = H(P_C, {P_C}')S_C + c{P_C}'.$$

A verifies:

$$e(T_B + T_C, P) = e(H(P_B, {P_B}')Q_B + H(P_C, {P_C}')Q_C, P_{pub})e(P_B, {P_B}')e(P_C, {P_C}').$$

If the above equation holds, then A computes:

$$K_A^{(1)} = e(P_B, P_C)^a, \ K_A^{(2)} = e(P_B, P'_C)^a, \ K_A^{(3)} = e(P'_B, P_C)^a, \ K_A^{(4)} = e(P'_B, P'_C)^a,$$
$$K_A^{(5)} = e(P_B, P_C)^{a'}, \ K_A^{(6)} = e(P_B, P'_C)^{a'}, \ K_A^{(7)} = e(P'_B, P_C)^{a'}, \ K_A^{(8)} = e(P'_B, P'_C)^{a'},$$

B verifies:

$$e(T_A + T_C, P) = e(H(P_A, {P_A}')Q_A + H(P_C, {P_C}')Q_C, P_{pub})e(P_A, {P_A}')e(P_C, {P_C}').$$

If the above equation holds, then B computes

$$K_B^{(1)} = e(P_A, P_C)^b,\ K_B^{(2)} = e(P_A, P'_C)^b,\ K_B^{(3)} = e(P_A, P_C)^{b'},\ K_B^{(4)} = e(P_A, P'_C)^{b'},$$
$$K_B^{(5)} = e(P'_A, P_C)^b,\ K_B^{(6)} = e(P'_A, P'_C)^b,\ K_B^{(7)} = e(P'_A, P_C)^{b'},\ K_B^{(8)} = e(P'_A, P'_C)^{b'}.$$

C verifies:

$$e(T_B + T_A, P) = e(H(P_B, P_B{}')Q_B + H(P_A, P_A{}')Q_A, P_{pub})e(P_B, P_B{}')e(P_A, P_A{}').$$

If the above equation holds, then C computes

$$K_C^{(1)} = e(P_A, P_B)^c,\ K_C^{(2)} = e(P'_A, P_B)^c,\ K_C^{(3)} = e(P_A, P'_B)^c,\ K_C^{(4)} = e(P'_A, P'_B)^c,$$
$$K_C^{(5)} = e(P_A, P_B)^{c'},\ K_C^{(6)} = e(P'_A, P_B)^{c'},\ K_C^{(7)} = e(P_A, P'_B)^{c'},\ K_C^{(8)} = e(P'_A, P'_B)^{c'}.$$

Each entity takes the eight values $K_{ID}^{(i)},\ i = 1, 2, \ldots, 8$, as the final session keys. The correctness of the protocol can be easily checked by the bilinear property of the pairing:

$$\begin{aligned} K_A^{(1)} = e(P_B, P_C)^a = e(abP, cP) = e(aP, cP)^b = e(P_A, P_C)^b = K_B^{(1)} \\ = e(bP, aP)^c = e(P_B, P_A)^c = K_C^{(1)}. \end{aligned}$$

Similarly, we get

$$K^{(i)} = K_A^{(i)} = K_B^{(i)} = K_C^{(i)},\ i = 2, 3, \ldots, 8.$$

Remark: One entity uses 4 pairings for verification of the broadcast messages from the other two entities, and 4 pairings to computes the 8 session keys. Entity A knows that

$$K_A^{(5)} = \left(K_A^{(1)}\right)^{a^{-1}a'}, K_A^{(6)} = \left(K_A^{(2)}\right)^{a^{-1}a'}, K_A^{(7)} = \left(K_A^{(3)}\right)^{a^{-1}a'}, K_A^{(8)} = \left(K_A^{(4)}\right)^{a^{-1}a'}.$$

Entity B and C also knows some similar relationship between the 8 session keys. However, since an entity keeps his ephemeral keys secret, other entities cannot learn the relationship that is known the specific entity. When ephemeral keys are randomly chosen, the 8 session keys are also random.

Take $a' = a$ in the above protocol, we can easily obtain a simplified version of ID-based tripartite authenticated key agreement, i.e., the three entities agree to one session key instead of eight keys. The simplified version works in the following way.

$$A \to B, C:\ P_A = aP,\ T_A = H(P_A)S_A + aP_A.$$

$$B \to A, C:\ P_B = bP,\ T_B = H(P_B)S_B + bP_B.$$

$$C \to A, B:\ P_C = cP,\ T_C = H(P_C)S_C + cP_C.$$

User A verifies:

$$e(T_B + T_C, P) = e(H(P_B)Q_B + H(P_C)Q_C, P_{pub})e(P_B, P_B)e(P_C, P_C)$$

and computes:

$$K_A = e(P_B, P_C)^a$$

User B verifies:

$$e(T_A + T_C, P) = e(H(P_A)Q_A + H(P_C)Q_C, P_{pub})e(P_A, P_A)e(P_C, P_C)$$

and computes:

$$K_B = e(P_A, P_C)^b$$

User C verifies:

$$e(T_B + T_A, P) = e(H(P_B)Q_B + H(P_A)Q_A, P_{pub})e(P_B, P_B)e(P_A, P_A)$$

and computes:

$$K_C = e(P_B, P_A)^c.$$

Then the shared secret key is:

$$K_A = K_B = K_C = e(P, P)^{abc}.$$

Remark: With the simplified version of our protocol, an entity needs to use 5 pairings (4 for verification and one for the generation of the session key) to get one session key. The computational overhead is heavy compared with the original protocol, which uses 8 pairings to generate 8 keys.

4 Analysis of the proposed protocol

4.1 Security attributes

The principle of the proposed protocol is that the ephemeral keys a, a', b, b' and c, c' of the three entities determines the final session keys $K^{(1)} = e(P, P)^{abc}, \ldots,$ $K^{(8)} = e(P, P)^{a'b'c'}$. Due to the bilinearity of the pairing, the three entities don't have to exchange ephemeral keys over secure channels. They exchange $(aP, a'P)$, $(bP, b'P)$, and $(cP, c'P)$ publicly to determine the session keys instead. The secrecy of the session keys relies on the assumption of hardness of BDH problem, *i.e.*, given P, aP, bP, cP, it is hard to determine $e(P, P)^{abc}$.

The resulting session keys are **good keys,** as long as one of the three entities chooses his ephemeral keys uniformly.

The authenticity of $(aP, a'P)$, $(bP, b'P)$, and $(cP, c'P)$ are achieved by attaching authenticators T_A, T_B, and T_C. The authenticators are computed by the entities using their static private key. We can also consider T_A to be A's signature for the message $(aP, a'P)$, T_B B's signature for the message $(bP, b'P)$, and T_C C's signature for the message $(cP, c'P)$. As a consequence, the authenticity of the tripartite key agreement protocol is assured by the security of the following signature scheme, which relies on the ID-based public key infrastructure introduced in Section 2. Without loss of generality, we take entity A as the signing entity. Signer A has a private key S_A, while the public key is Q_A.

Public parameters: G_1 is a group of prime order q. P is a generator of G_1. The bilinear pairing is given as $e : G_1 \times G_1 \rightarrow G_2$. A cryptographic hash function is defined as $H : \{0,1\}^* \rightarrow Z_q$.

Signing: Suppose that the message to be signed is $m = a'P$. Signer A randomly chooses an integer $a \in Z_q^*$. He computes $P_A = aP$ and $T_A = H(P_A, m)S_A + am$. Then the signature of m is (P_A, T_A).

Verification: After getting a message m and its corresponding signature (P_A, T_A), the verifier accepts the signature if and only the following equation holds

$$e(T_A, P) = e(H(P_A, m)Q_A, P_{pub})e(m, P_A).$$

We note that this signature is not quite functioning as the normal signature scheme in the sense that the message m takes a special form of a multiple of P instead of any value. However, the above signature scheme can be easily converted into a normal version using a function $f : \{0,1\}^* \rightarrow G_1$, for example $f(m) = H(m)P$.

Our signature scheme is secure against existential forgery under an adaptively chosen message attack in the random oracle model. The proof is similar to that of Scheme 3 in [14].

Now we give a brief security analysis to show that the above signature scheme is secure against existential forgery. Suppose that there is a polynomial time probabilistic Turing machine E which takes m and Q_A as input, and output an existential forgery of a signature from A with a non-negligible probability. Here we assume that H is a random oracle. Then we show that there is another polynomial time algorithm E', who take advantage of the Turing machine E, solves the weak version of Diffie-Hellman problem. E' inputs the same random tape to E. When E inquires the hash values of (P_A, M), E' returns a random value to E. According to the Forking Lemma of Pointcheval and Stern [21], E may get two forgeries of signature from A for the same message m within a polynomial time. Let the two signature forgeries for m are (P_A, T_A) and $(\tilde{P}_A, \tilde{T}_A)$. We have

$$T_A = H(P_A, m)S_A + am$$

and

$$\tilde{T}_A = H(\tilde{P}_A, m)S_A + am.$$

It follows that

$$T_A - \tilde{T}_A = (H(P_A, m) - H(\tilde{P}_A, m))S_A,$$

hence

$$e(T_A - \tilde{T}_A, P) = e(Q_A, P_{pub})^{H(P_A, m) - H(\tilde{P}_A, m)}.$$

The above equation means that for a random element $v \in G_2$, an element $R \in G_1$ can be found such that $e(R, P) = v$ in a polynomial time, i.e. there is a polynomial time algorithm $f : G_2 \rightarrow G_1$ inverting the pairing. Suppose that

g is a generator of G_2. Let $g' = e(f(g), f(g))$, then given g^u and g^v, $g'^{ut} = e(f(g^u), f(g^t))$ can be easily determined, i.e., we have solved the weak Diffie-Hellman problem.

The above proof can be easily extended into a detailed proof of the security of the signature scheme against an adaptively chosen message attack, allowing the adversary to query a limited number of a signature oracle, a key extraction oracle, and a random oracle (see [14]).

Upon employment of the above signature scheme in our tripartite key agreement protocol, we can consider aP as the message, and $(a'P, T_A)$ as the corresponding signature, or $a'P$ as the message, and (aP, T_A) as the corresponding signature. It is easy to see that aP and $a'P$ are role symmetric: either one can serve as the message and the other as the part of the signature. As a result, both aP and $a'P$ are authenticly broadcasted.

Without knowledge of the static private key of an entity, an adversary can hardly impersonate that entity for a key agreement since he can hardly forge a signature of the broadcast message within a polynomial time with a non-trivial probability. Therefore, only those entities who have the corresponding static private keys can generate the resulting session keys. In other words, our proposed protocol is an **implicit key authentication** protocol.

The ephemeral keys of the participating entities should be kept by the entities with high secrecy. When an adversary knows an ephemeral key, say a, he can extract the corresponding static private key by

$$S_A = (T_A - aP'_A)H(P_A, P'_A)^{-1}. \tag{1}$$

Having obtained the static key of some entity, the adversary can impersonate the entity in every kind of cryptographic primitives, key agreement, encryption scheme, signature systems, etc.

Below is security attributes of our protocol.

- **Known session key security:** A protocol is called *Known session key security*, if an adversary, having obtained some previous session keys, cannot get the session keys of the current run of the key agreement protocol. As with our protocol, suppose that the adversary learned all the eight keys of a previous key agreement protocol. To extract the ephemeral key from a session key, for example, to determine a from $K^{(1)} = e(P_B, P_C)^a$, is equivalent to solving the Discret-Log problem in G_2, i.e., given $e(P_B, P_C)$ and $e(P_B, P_C)^a$, compute a. However, without a, the adversary cannot extract A's private key S_A from Equation (1). The security of the signature scheme also prevents the adversary from forging a valid signature from A and impersonating A to get session keys.
- **Perfect forward secrecy:** A protocol is called *Perfect forward secrecy* if compromise of the three private keys of the three participating entities does not affect the security of the previous session keys. As with our protocol, to learn the previous session keys, the adversary has to get the corresponding ephemeral keys. Suppose the adversary has got S_A, A's static private key.

From A's broadcast messages (P_A, P'_A, T_A), he can computes $aP'_A = a'P_A = T_A - H(P_A, P'_A)S_A$. However, given $(P_A, a'P_A)$ (or (P'_A, aP'_A)), extract a' (or a) is equivalent to solve Discret-Log problem in G_1, which is assumed to be a hard problem. Therefore, our protocol is *Perfect forward secrecy.*
- **No key-compromise impersonation:** The compromise of one entity's static private key doesn't imply that the private keys of other entities will also be compromised in our protocol. The adversary may impersonate the compromised entity in subsequent protocols, but he cannot impersonate other entities. This property is called *no key-compromise impersonation.*
- **No unknown key-share:** If the adversary convinces a group of entities that they share some session key with the adversary, while in fact they share the key with another entity, we call the protocol suffering from *unknown key-share* attack. To implement such an attack on our protocol, the adversary is required to learn the private key of some entity. Otherwise, the attack hardly works. Hence, we claim that our protocol has the attribute of no unknown key-share.
- **No key control:** Just like the session key obtained from Joux's tripartite key agreement protocol, the session keys in our protocol are determined by all the three entities, and no one can influence the outcome of the session keys, or enforce them to fall into a pre-determined interval. In other words, there is *no key control* in our protocol.

The above security attributes are also listed in [1], where three MTI type protocols and one MQV type protocol was presented and analyzed. The MTI type protocols more or less suffer from the above listed attacks, while the MQV type protocol, which is named TAK-4 in [1], have all the above list security attributes.

4.2 Computational Overhead and Bandwidth

In the previous subsection, we see that our proposed protocol is comparable to the so-called TAK-4 protocol, the MQV type protocol proposed in [1], with respect to security attributes. The main advantage of our protocol over TAK-4 is that certificates are not involved in our protocol, since our protocol is a kind of ID-based cryptographic primitives.

A main criticism of authenticated key agreement using digital signature scheme is that the message length is much greater than the non-authenticated version of key agreement. Our protocol also has a great data expansion. However, We stress that one instance of our protocol results in eight session keys. This corresponds to eight instances of a normal key agreement protocol.

As we mentioned, our protocol is comparable to TAK-4 protocol with respect to security attributes. We now compare the bandwidth and computational overhead of our protocol with eight instances of TAK-4.

First, we present TAK-4 protocol.

$$A \to B, C : aP || Cert_A$$

$$B \rightarrow A, C : bP||Cert_B$$

$$C \rightarrow A, B : cP||Cert_C$$

A B and C computes the following three keys (the three keys are the same) respectively

$$K_A = e(bP + H(bP, yP)yP, cP + H(cP, zP)zP)^{a+H(aP,xP)x},$$

$$K_B = e(aP + H(aP, xP)xP, cP + H(cP, zP)zP)^{b+H(bP,yP)y},$$

$$K_C = e(aP + H(aP, xP)xP, bP + H(bP, yP)yP)^{a+H(cP,zP)z}.$$

In the protocol, $Cert_{ID}$ is an entity's certificate consisting of

$$(ID, wP, P, Sig_{CA}(ID, wP, P)),$$

where ID is the identity information of the entity, P is the generator of the group G_1, and wP is the entity's public key, $Sig_{CA}(ID, wP, P)$ is CA's signature for the public information (ID, wP, P). The entity's private key is w.

First we analyze the bandwidth of one broadcast of TAK-4. One broadcast consists of an element of G_1 and a certificate. In a certificate, the signature of CA generally will bring data expansion, but we conservatively take $Sig_{CA}(\cdot)$ as an element from G_1. Then there are three elements of G_1 in a certificate. Totally there are four elements of G_1 per broadcast.

The computational overhead for each entity in TAK-4 is dominated by two verifications of CA's signatures for the other two parties, one pairing and three scalar multiplications over G_1. Other (less dominant) computation involves one exponentiation and three hashings.

Now we analyze the bandwidth and computational overhead of our protocol and give a comparison with TAK-4. In our protocol, one broadcast consists of three elements of G_1. One entity needs two scalar multiplications to hide two ephemeral keys, two scalar multiplications over G_1 to prepare T_{ID} for the broadcast, two scalar multiplications and four pairings for verification, and four pairings for the generation of the session keys. Eight exponentiations and three hashings are also involved in the computation.

Let VS denote verification of a signature, PA denote pairing, SM denote scalar multiplication, EX denote exponentiation, and HA hashing. The comparison of our protocol and 8 instances of a TAK-4 protocol (8×TAK-4) is illustrated in Table 1. We can see that our protocol is more efficient than TAK-4 in terms of bandwidth and computational overhead. The essential operation in both our schemes and TAK-4 is to compute the bilinear pairing. Due to [2] and [11], the computation of the bilinear pairing is efficient enough nowadays.

5 Conclusion

ID-based public key cryptosystem can be an alternative for certificate-based public key infrastructures, especially when efficient key management and moderate

Protocol	Bandwidth per broadcast	Computational overhead per entity
Ours	3 elements in G_1	8PA + 6SM + 8EX + 3HA
8 × TAK-4	8×4 elements in G_1	8×(2VS + 1PA + 3SM + 1EX + 3HA)

Table 1. Bandwidth and computational overhead of our protocol and TAK-4

security are required. In this paper, we proposed an authenticated tripartite key agreement protocol. The resulting key is determined by the ephemeral keys of the three entities, as Joux's protocol [15] does. The authenticity of the protocol is assured by a digital signature scheme. The signature scheme is role symmetric so that messages containing information of two ephemeral keys can be broadcasted authenticly by an entity. As a consequence, eight sessions keys results after one instance of our protocol. Like TAK-4 proposed in [1], our protocol has the following security attributes: implicit key authentication, known session key security, perfect forward secrecy, no key-compromise impersonation, no unknown key-share, and no key control. Since eight session keys can be generated with our protocol, our protocol is more efficient than TAK-4 in terms of bandwidth and computational overhead.

References

1. S. Al-Riyami and K. Paterson, *Authenticated three party key agreement protocols from pairings*, Cryptology ePrint Archive, Report 2002/035, available at http://eprint.iacr.org/2002/035/.
2. P.S.L.M. Barreto, H.Y. Kim, B.Lynn, and M.Scott, *Efficient algorithms for pairing-based cryptosystems*, To appear in Cryptology-Crypto'2002, available at http://eprint.iacr.org/2002/008/.
3. S. Blake-Wilson and A. Menezes, *Authenticated Diffie-Hellman Key Agreement Protocols*, Proceedings of the 5th Annual Workshop on Selected Areas in Cryptography (SAC '98), LNCS 1556, Springer-Verlag, pp.339-361, 1999.
4. D. Boneh and M. Franklin, *Identity-based encryption from the Weil pairing*, Advances in Cryptology-Crypto 2001, LNCS 2139, pp.213-229, Springer-Verlag, 2001.
5. D. Boneh, B. Lynn, and H. Shacham, *Short signatures from the Weil pairing*, In C. Boyd, editor, Advances in Cryptology-Asiacrypt 2001, LNCS 2248, pp.514-532, Springer-Verlag, 2001.
6. J.C. Cha and J.H. Cheon, *An identity-based signature from gap Diffie-Hellman groups*, Cryptology ePrint Archive, Report 2002/018, available at http://eprint.iacr.org/2002/018/.
7. C. Cocks, *An identity based encryption scheme based on quadratic residues*, In Cryptography and Coding, LNCS 2260, pp.360-363, Springer-Verlag, 2001.
8. W. Diffie and M. Hellman, *New directions in cryptography*, In IEEE Transations on Information Theory, 1976, No.22, pp.644-654, 1976.
9. Y. Desmedt and J. Quisquater, *Public-key systems based on the difficulty of tampering*, Proc. Crypto 86, LNCS 263, pp.111-117, Springer-Verlag, 1986.
10. G. Frey and H.Rück, *A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves*, Mathematics of Computation, 62, pp.865-874, 1994.

11. S. D. Galbraith, K. Harrison, and D. Soldera, *Implementing the Tate pairing*, ANTS 2002, LNCS 2369, pp.324-337, Springer-Verlag, 2002.
12. C. Gentry and A. Silverberg, *Hierarchical ID-based cryptography*, To appear in Cryptology-Asiacrypt 2002.
13. J. Horwitz and B. Lynn, *Toward hierarchical identity-based encryption*, Proc. Eurocrypt 2002, LNCS 2332, pp.466-481, Springer-Verlag, 2002.
14. F. Hess, *Exponent group signature schemes and efficient identity based signatureschemes based on pairings*, Cryptology ePrint Archive, Report 2002/012, available at http://eprint.iacr.org/2002/012/.
15. A. Joux, *A one round protocol for tripartite Diffie-Hellman*, ANTS IV, LNCS 1838, pp.385-394, Springer-Verlag, 2000.
16. L. Law, A. Menezes, M. Qu, J. Solinas, and S. Vanstone, *An efficient protocol for authenticated key agreement*, Technical Report CORR 98-05, Department of C & O, University of Waterloo, 1998.
17. T. Matsumoto, Y. Takashima, and H. Imai, *On seeking smart public-key distribution systems*, Trans. IECE of Japan, E69, pp.99-106, 1986.
18. A. Menezes, T. Okamoto, and S. Vanstone, *Reducing elliptic curve logarithms to logarithms in a finite field*, IEEE Transaction on Information Theory, Vol.39, pp.1639-1646, 1993.
19. U. Maurer and Y. Yacobi, *Non-interactive public-key cryptography*, proc. Eurocrypt 91, LNCS 547, pp.498-507, Springer-Verlag, 1991.
20. K.G. Paterson, *ID-based signatures from pairings on elliptic curves*, Cryptology ePrint Archive, Report 2002/004, available at http://eprint.iacr.org/2002/004/.
21. D. Pointcheval and J. Stern, *Security arguments for digital signatures and blind signatures*, Journal of cryptology, No. 13, pp.361-396, 2000.
22. R. Sakai, K. Ohgishi, M. Kasahara, *Cryptosystems based on pairing*, SCIS 2000-C20, Jan. 2000. Okinawa, Japan.
23. A. Shamir, *Identity-based cryptosystems and signature schemes*, Advances in Cryptology-Crypto 84, LNCS 196, pp.47-53, Springer-Verlag, 1984.
24. N.P. Smart, *An identity based authenticated key agreement protocol based on the Weil pairing*, Electron. Lett., Vol.38, No.13, pp.630-632, 2002.
25. B. Song and K. Kim, *Two-Pass Authenticated Key Agreement Protocol with Key Confirmation*, Proc. of Indocrypt 2000, LNCS 1977, pp.237-249, Springer-Verlag, 2000.
26. H. Tanaka, *A realization scheme for the identity-based cryptosystem*, proc. Crypto 87, LNCS 293 pp.341-349, Springer-Verlag, 1987.
27. S. Tsuji and T.Itoh, *An ID-based cryptosystem based on the discrete logarithm problem*, IEEE Journal of Selected Areas in Communications, Vol.7, No.4, pp.467-473, 1989.
28. E. R. Verheul, *Self-blindable credential certificates from the Weil pairing*, Advances in Cryptology-Asiacrypt 2001, LNCS 2248, pp.533-551, Springer-Verlag, 2001.
29. F. Zhang and K. Kim, *ID-based blind signature and ring signature from pairings*, To appear in Cryptology-Asiacrypt 2002.