MULTI-PARTY AUTHENTICATED KEY AGREEMENT PROTOCOLS FROM MULTILINEAR FORMS

HO-KYU LEE, **HYANG-SOOK LEE**, **YOUNG-RAN LEE**

Department of Mathematics, Ewha Womans University, Seoul, Korea

Abstract. Joux [10] presented a one round protocol for tripartitie key agreement and Al-Riyami et.al. [15] developed a number of tripartitie, one round, authenticated protocols related to MTI and MQV protocols. Recently, Boneh and Silverleg [4] studied multilinear forms, which provides a one round multi-party key agreement protocol. In this paper, we propose $(n+1)$ types of one round authenticated multi-party key agreement protocols from multilinear forms based on the application of MTI and MQV protocols.

Keywords : Multilinear forms, Key Agreement protocol, Authentication

1 Introduction

A number of two party key agreement protocols([13], [16]) have been proposed ever since the famous Diffie-Hellman protocol [9] was first proposed. The situation where three or more parties share a secrete key, which is often called conference keying([8], [13]), is getting more important as group communications grow up on open network. There have been many attempts to extend the well known Diffie-Hellman key exchange protocol to the multi-party setting ([1], [2], [3], [8], [11], [17]). In 2000, Joux [10] presented a one round tripartitie key agreement protocol. However Joux's protocol is unauthenticated and suffers from man-in-the-middle attacks. Al-Riyami et.al. [15] proposed one round authenticated key agrement protocols for three parties which is based on the ideas from Joux's protocol and MTI [14]

H.S.Lee was supported by KOSEF grant No. R06-2002-012-01001. H.K.Lee and Y.R.Lee was supported by Brain Korea 21 Project.
e-mail : hokyu@dreamwiz.com, hsl@ewha.ac.kr, panic@ewha.ac.kr.

and MQV [12] protocols. Recently, Boneh and Silverberg [4] studied the problem of finding efficiently computable non-degenerate multilinear maps and presented several applications to cryptography using multilinear forms. The efficiently computable multilinear forms would enable one round multi-party key exchange, a unique signature scheme and secure broadcast encryption with very short broadcasts. However, the one round multi-party key agreement protocol from multilinear forms is unauthenticated, and hence is subject to a classic man-in-the-middle attacks like Joux's protocol. In this paper, we propose one round multi-party authenticated key agreement protocols using multilinear forms based on the application of MTI and MQV protocols. The security analysis of our protocol is ad hoc and therefore the statements about security can be termed heuristic. This paper is organized as follows. In section 2, we discuss security goals and performance attributes of key agreement protocols. In section 3, we introduce a one round multi-party key agreement protocol from multilinear forms, and give the obvious attacks on the protocol. In section 4, we present one round authenticated key agreement protocols for multi-parties. These protocols are developed from the multi-party key agreement protocol using multilinear forms and the application of MTI and MQV protocols. In section 5, we analyze a number of attacks on our protocols and show how they can be prevented. We also compare the security and efficiency of our protocols. In the final section, we conclude and suggest the future works to develop our protocols based on provable security.

2 Protocol Goals and Attributes

We discuss various security goals and performance attributes that one may wish a key agreement protocol to possess. The following definitions come from the references ([1], [15], [16]).

There are two types of attack : One is *passive attack*, where an adversary attempts to defeat a cryptographic technique by simply recording data and therefore analyzing it. The other is *active attack*, where an adversary additionally subverts the communications themselves in any possible : by injecting messages, intercepting messages, replaying messages, altering messages, and the like.

Now we present concrete security goals for protocols. The fundamental security goals described below are considered to be vital in any application. The other security and performance attributes are important in some environments, but less important in others.

(1) Fundamental security goals

(i) *Key authentication*. In some of the literature, key authentication may imply implicit authentication or explicit authentication. The difference is that *implicit key authentication* to entity A implies that only B may be able to compute a particular key, while *explicit key authentication* to entity A implies that only B has the ability to compute a particular key and has actually done so.
(ii) *Key confirmation*. Key confirmation to entity A is the assurance that entity B has actually computed the shared session key K.

A key agreement protocol which provides implicit key authentication to both participating entities is called an *authenticated key agreement* protocol, while one providing *explicit key agreement* with key confirmation protocol.

(2) Other desirable security attributes

A number of desirable attributes of key agreement protocols have also been identified by the followings.
(i) *Known session key security*. A protocol is called known session key secure if it still achieves its goal in the face of an adversary who has learned some previous session keys.
(ii) *(Perfect) forward secrecy*. A protocol is forward secrecy if, when the long-term secrets of one or more entities are compromised, the secrecy of previous session keys is not affected. Perfect forward secrecy refers to the scenario when the private keys of all the participating entities are compromised.
(iii) *No key-compromise impersonation*. Suppose A's long-term private key is disclosed. Clearly an adversary that knows this value can impersonate A in any protocol. We say that a protocol resists key-compromise impersonation when this loss does not enable an adversary to impersonate other entities as well.
(iv) *No unknown key-share.* In an unknown key share attack, an adversary convinces a group of entities that they share a key with the adversary, whereas in fact the key is shared between the group and another party.
(v) *No key control*. A protocol is no key control if for any participant (or an adversary) can not control or predict the value of the session key.

(3) Desirable performance attributes

These attributes include :

(i) Number of message exchanges (passes) required between entities ;

(ii) Bandwidth required by messages (total number of bits transmitted) ;

(iii) Complexity of computation by each entity (as it affects execution time) ; and

(iv) Possibility of pre-computation to reduce on-line computational complexity.

Also the protocol is called *role symmetry* if the messages transmitted have the same structure and *non-interactive* if the messages transmitted between the two entities are independent of each other.

3 A one round multi-party key agreement using multilinear forms

In this section, we introduce multilinear forms and a multi-party key agreement protocol based on the multilinear form. We also consider man-in-the-middle attack on the protocol.

3.1 Multi-linear forms

Let G_1, G_2 be two groups of the same prime order. We say that a map $e_n : G_1^n \to G_2$ is an *n-multilinear map* if it satisfies the following properties:

(i) If $a_1, a_2, \cdots, a_n \in Z$ and $x_1, x_2, \cdots, x_n \in G_1$, then

$$e_n(x_1^{a_1}, \cdots, x_n^{a_n}) = e_n(x_1, \cdots, x_n)^{a_1 \cdots a_n}.$$

(ii) The map e_n is non-degenerate in the following sense: if $g \in G_1$ is a generator of G_1 then $e_n(g, \cdots, g)$ is a generator of G_2.

The efficiently computable multilinear forms would enable secure broadcast encryption with very short broadcasts and private keys, a unique signature scheme, and one round multi-party key exchange. Refer to [4] for more detailed applications to the cryptography using multilinear forms.

The multilinear Diffie-Hellman problem (MDHP) *The multilinear Diffie-Hellman problem* says that given $g, g^{a_1}, \cdots, g^{a_n}$ in G_1, compute $e_n(g, \cdots, g)^{a_1 \cdots a_n}$ in G_2.

The multilinear Diffie-Hellman assumption means the multilinear Diffie-Hellman problem is hard.

3.2 Multi-party key agreement protocol

Boneh and Silverberg proposed a simple and elegant one round key agreement protocol using multilinear forms in which the secret session key for n-parties could be created using just one broadcast per entity. Now we introduce the one round multiparty Diffie-Hellman key exchange scheme based on multilinear forms [4].

I. A one-round n-party key exchange protocol $(n > 2)$

Setup : Let G_1, G_2 be finite cyclic groups of the same prime order p and g be a generator of G_1. Let $A_1, \cdots, A_n$ be n-participants who want to share a common secret information. Let $e_{n-1} : G_1^{n-1} \to G_2$ be an $(n-1)$-multilinear map.

Publish : Each participants A_i pick a uniformly random integer $a_i \in [1, p-1]$ and computes $g^{a_i} \in G_1$. Each A_i broadcasts g^{a_i} to all others and keeps a_i secret.

Key generation : Each A_i computes the conference key K_i as follows:

$$\begin{aligned} K_i &= e_{n-1}(g^{a_1}, \cdots, g^{a_{i-1}}, g^{a_{i+1}}, \cdots, g^{a_n})^{a_i} \\ &= e_{n-1}(g, \cdots, g)^{a_1 \cdots a_n} \in G_2. \end{aligned}$$

Therefore all n-participants obtain the same conference key $K = K_i$ for all $i = 1, \cdots, n$. $\square$

The security of this protocol is based on the hardness of the multilinear Diffie-Hellman problem. More precisely, the session key should be derived by applying a suitable key derivation function to the quantity $e(g, \cdots, g)^{a_1 \cdots a_n}$. For otherwise, an attacker might be able to get partial information about session keys even if the MDHP is hard. It is known that the MDHP is no harder than the computational Diffie-Hellman problems in either G_1 or G_2.

3.3 Man-in-the-Middle Attack on the protocol I

Just like Joux's protocol based on pairing maps, the protocol I is subject to a classic man-in-the-middle-attack.

Suppose an adversary D is able to intercept A_1's communications with the other participants $A_2, \cdots, A_n$, impersonating A_1 to the other entities and impersonating the other entities to A_1. We write D_{A_1} to indicate that the adversary D is impersonating A_1 in sending or receiving messages intended for or originating from A_1. Similarly $D_{A_2,\cdots,A_n}$ denotes an adversary impersonating the other entities.

Let $\delta_1, \cdots, \delta_n \in [1, p-1]$ be random values of D's choice. We assume A_1 initiates a run of the protocol I. The following is the man-in-the-middle attack :

1. $D_{A_2,\cdots,A_n}$ intercepts g^{a_1} from A_1, and D_{A_1} forwards g^{δ_1} to $A_2, \cdots, A_n$.
2. D_{A_1} intercepts g^{a_j} from A_j, and D_{A_j} forwards g^{δ_j} to A_1.

At the end of this attack, D impersonating A_1 has agreed a key $K_{D_{A_1}A_2\cdots A_n} = e(g, \cdots, g)^{\delta_1 a_2 \cdots a_n}$ with other A_j's, $j \neq 1$, while D impersonating the other entities A_j's, $j \neq 1$ has agreed a second key $K_{A_1 D_{A_2 \cdots A_n}} = e(g, \cdots, g)^{a_1 \delta_1 \cdots \delta_n}$ with A_1. If these keys are used to encrypt subsequent communications, then D, by appropriately decrypting and re-encrypting messages, can continue his masquerade as A_1 to A_j's, $j \neq 1$ and A_j's to A_1. Now D can share a separate session key with each user and can masquerades any entity to any other entity.

4 One round multi-party authenticated key agreement protocols

The one round multi-party key agreement protocol I is established via just one round broadcast per entity. However this protocol is not authenticated. In this section we present authenticated multi-party key agreement protocols. Our protocols are generalizations of the MTI family of protocols and the MQV protocol to the setting of multilinear forms. We present a single protocol with $n+1$ different methods for deriving a session key. These different derivations result in protocols with slightly different security attributes, and we examine these in detail. A summary is given in Table 1.

As with the MTI protocols, a certification authority (CA) is used in the initial set-up stage to provide certificates which bind user's identities to long-term keys. The certificate for entity A_i will be of the form :

$$\mathrm{Cet}_{A_i} = (\mathcal{I}_{A_i} \| \mu_{A_i} \| g \| \mathcal{S}_{CA}(\mathcal{I}_{A_i} \| \mu_{A_i} \| g)).$$

Here $\mathcal{I}_{A_i}$ denotes the identity string of A_i, $\|$ denotes the concatenation of data items, and $\mathcal{S}_{CA}$ denotes the CA's signature. Entity A_i's long-term public key is $\mu_{A_i} = g^{x_i}$, where $x_i \in Z_p^*$ is the long-term secret key of A_i. Element g is the public value and is induced in order to specify which element is used to construct μ_{A_i} and the short term public values.

II. Multi-party authenticated key agreement protocols (MAK) $(n > 2)$

Setup : Let G_1, G_2 be finite cyclic groups of the same prime order p and g be a generator of G_1. Let $A_1, \cdots, A_n$ be n-participants who want to share a common secret information. Let $e_{n-1} : G_1^{n-1} \to G_2$ be an $(n-1)$-multilinear map.

Publish : Each participant A_i pick an uniformly random integer $a_i \in [1, p-1]$ and computes $g^{a_i} \in G_1$. Each A_i broadcasts to all other entities the short-term public value g^{a_i} along with a certificate Cert_{A_i} containing his long-term public key and each A_i keeps a_i secret. The ordering of protocol messages is unimportant and any of the other entities can initiate the protocol.

Key generation : Each A_i verifies the authenticates he receives. If any check fails, the protocol should be aborted. When no check fails, one of the following possible session keys described below should be computed.

MAK key generation :

1. Type A (MAK-A)

The keys computed by the entities are :

$$\begin{aligned} K_{A_i} &= e_{n-1}(g^{a_1}, \cdots, g^{a_{i-1}}, g^{a_{i+1}} \cdots, g^{a_n})^{a_i} \cdot e_{n-1}(g^{x_1}, \cdots, g^{x_{i-1}}, g^{x_{i+1}}, \cdots, g^{x_n})^{x_i} \\ &= e_{n-1}(g, \cdots, g)^{a_1 \cdots a_n + x_1 \cdots x_n}. \end{aligned}$$

2. Type B-j (MAK B-j), $(j = 1, \cdots, n-1)$

The keys computed by the entities are :

$$\begin{aligned} K_{A_i} = & \prod_{\substack{\binom{n-1}{j} \\ i \neq i_1, \cdots, i_j}} e_{n-1}(g^{a_1}, \cdots, g^{x_{i_1}}, \cdots, \widehat{g^{a_i}}, \cdots, g^{x_{i_j}}, \cdots, g^{a_n})^{a_i} \\ & \cdot \prod_{\substack{\binom{n-1}{j-1} \\ i \neq i_1, \cdots, i_{j-1}}} e_{n-1}(g^{a_1}, \cdots, g^{x_{i_1}}, \cdots, \widehat{g^{x_i}}, \cdots, g^{x_{i_{j-1}}}, \cdots, g^{a_n})^{x_i} \\ & = e_{n-1}(g, \cdots, g)^{\sum_{i_k \neq i_l}^{\binom{n}{j}} a_1 \cdots x_{i_1} \cdots x_{i_j} \cdots a_n} \end{aligned}$$

where $\widehat{g^{a_i}}$, $\widehat{g^{x_i}}$ are the terms which do not appear.

3. Type C (MAK-C)

The keys computed by the entities are :

$$\begin{aligned} K_{A_i} &= e_{n-1}(g^{a_1+H(g^{a_1}\|g^{x_1})x_1}, g^{a_2+H(g^{a_2}\|g^{x_2})x_2}, \cdots, g^{a_{i-1}+H(g^{a_{i-1}}\|g^{x_{i-1}})x_{i-1}}, \\ &\quad g^{a_{i+1}+H(g^{a_{i+1}}\|g^{x_{i+1}})x_{i+1}}, \cdots, g^{a_n+H(g^{a_n}\|g^{x_n})x_n})^{(a_i+H(g^{a_i}\|g^{x_i})x_i)} \\ &= e_{n-1}(g,\cdots,g)^{(a_1+H(g^{a_1}\|g^{x_1})x_1)\cdots(a_n+H(g^{a_n}\|g^{x_n})x_n)}. \end{aligned}$$

Protocols MAK-A and MAK B-j have originated from MTI protocols. Protocol MAK-C has a root in the MQV protocol but avoids protocol's unknown key share weakness by using cryptographic hash function H. In each case, key generation is role symmetric and each entity uses both short term and long term keys to produce a unique shared secret key. No party has control over the resulting session key. The communication of each protocol is identical using a single broadcast per entity. However, the computation of MAK B-j requires more computation compared to MAK-A. In MAK B-j, each entity takes $\binom{n}{j}$ pairing calculations, but MAK-A takes two pairing computations and MAK-C require only a single pairing computation per entity. MAK-A and MAK B-$(n-1)$ can exploit pre-computation if entities know in advance with whom they will be sharing a key. In MAK-A, all entities can pre-compute the term $e_{n-1}(g,\cdots,g)^{x_1x_2\cdots x_n}$ and use this term until the long term keys are expired. In case of MAK B-$(n-1)$, A_i can pre-compute $e_{n-1}(g,\cdots,g)^{x_1\cdots a_i\cdots x_n}$ as long as the the short term key a_i is available. Our MAK protocols prevent man-in-the-middle attacks of the type introduced in the section 3.3. However, other forms of active attack can still occur. We consider such attacks and also suggest how to prevent them in the following section.

5 Attacks on MAK Protocols

We present various attacks on our MAK protocols. These are mostly inspired by earlier attacks on the two-party MTI protocols. Nevertheless some of the attacks are preventable, and others require rather unrealistic scenarios, all of the attacks are important since they determine the security attributes of our various protocols. The summary of our security attributes is provided in Table 1.

5.1 Two Key-Compromise Attacks on MAK-A

We consider a very serious attack on MAK-A. It requires the adversary D to obtain just a session key and one of the short-term secret keys used in a protocol run, and after which the adversary D is able to impersonate any of the other entities in subsequent protocol runs. Since this attack does not require the adversary to learn a long-term secret key, it is more severe than a key-compromise impersonation attack.

The pre-requisites for the attack are the followings;

1. The adversary D, by eavesdropping on a protocol run, has obtained the short-term public values $g^{a_2}, \cdots, g^{a_n}$

2. The adversary D has also obtained the session key

$$K_{A_1,\cdots,A_n} = e_{n-1}(g, g, \cdots, g)^{a_1 \cdots a_n + x_1 \cdots x_n}$$

agreed in that protocol run.

3. The adversary D has also somehow acquired the short-term key "a_1" used in that run.

The adversary D can evaluate $K_{A_1 A_2 \cdots A_n} \cdot e_{n-1}(g^{a_2}, \cdots, g^{a_n})^{-a_1}$. D can impersonate any of $A_1, A_2, \cdots A_{n-1}$ or A_n in subsequent protocol runs. She does this simply by sending $\mathrm{Cert}_{A_1}, \mathrm{Cert}_{A_2}, \cdots, \mathrm{Cert}_{A_n}$ along with her chosen short-term public value g^δ. She can compute session keys agreed in subsequent protocol runs since she knows $e_{n-1}(g, g, \cdots, g)^{x_1 x_2 \cdots x_n}$ and δ was chosen by her. By symmetry, this attack can be mounted once D is in possession of any short-term secret key. This attack is prevented by using a hash function to perform key derivation.

Our MAK-A protocol fails to achieve the key-compromise impersonation attribute, that is, if entity A_i discloses its long-term secret key x_i then the adversary D is not only able to impersonate A_i to any entity, but also can impersonate any entity A_j to A_i, since in this event the adversary D is able to compute the value $e_{n-1}(g, \cdots, g)^{x_1 x_2 \cdots x_n}$ using x_i and public data in A_j's $(j \neq i)$ certificates. Session key derivation is not helpful to resist this attack. However, the attacks do not appear to MAK B-j and MAK-C since the long-term key components are combined with short-term key components in $K_{A_1 \cdots A_n}$.

5.2 Forward Security Weakness in MAK B-j $(j = 1, 2, \cdots, n-1)$

We say a protocol is not forward secure if the compromise of long-term secret keys of one or more entities also allows an adversary to obtain session keys previously established between honest entities. Both the protocols MAK-A and MAK-C achieve perfect forward secrecy. Indeed if all n long-term secret keys are available to the adversary in MAK-A protocol, then extracting the session key $K_{A_1\cdots A_n}$ from an old session key can be shown to be equivalent to solving the MDHP. The same is true of MAK-C, because the key $K_{A_1\cdots A_n}$ agreed in that case also includes the component $e_{n-1}(g, g, \cdots, g)^{a_1 a_2 \cdots a_n}$. However, MAK B-$j$ are not forward secure. It is not hard to see that if the adversary obtains $(n-j+1)$ long-term secret keys in MAK B-j $(j = 1, \cdots, n-1)$, then she can obtain old session key(assuming she keeps a record of the public values $g^{a_1}, g^{a_2}, \cdots, g^{a_n}$). The protocols can be made perfectly forward secure by using the key $K_{A_1\cdots A_n} \cdot e_n(g, g, \cdots, g)^{a_1 a_2 \cdots a_n}$ instead of the key $K_{A_1\cdots A_n}$. Of course, it needs some additional computational cost.

5.3 Unknown Key-Share Attacks

(1) *Basic Source Substitution Attacks on MAK-A to MAK-C*
This is a practical attack, which utilizes a potential registration weakness for public keys to create fraudulent certificates. The attack scenario is the following: An adversary D registers A's public key μ_{A_1} as her own; i.e. $\mu_{A_1} = \mu_D$, $\mathrm{Cert}_D = (I_D \parallel \mu_{A_1} \parallel g \parallel S_{\mathcal{CA}}(I_D \parallel \mu_{A_1} \| g))$. When A_1 broadcasts a message $g^{a_1} \parallel \mathrm{Cert}_{A_1}$ to $A_2, \cdots, A_n$, D intercepts the message and replaces $g^{a_1} \parallel \mathrm{Cert}_D$. Note that D registered the A_1's long-term public key g^{x_1} as her own without knowing the value of x_1. Therefore she cannot learn the key $K_{A_1\cdots A_n}$. However $A_2, \cdots, A_n$ accept the key $K_{A_1\cdots A_n}$ and believe that they have agreed a key with D, when in fact they have shared a key with A_1. They will interpret any subsequent encrypted messages emanating from A_1 as coming from D. This basic source substitution attack is usually prevented if CA does not allow two entities to register the same long-term public key. However, this solution may not scale well to large or distributed systems. The better solution follows.

(2) *Second Source Substitution on MAK B-j*
The adversary can attack protocols MAK B-j even if the **CA** does the previous check. She obtains a Cert_D from the **CA** which contains a component μ_D which is some power of μ_{A_i}, and alters short-term keys in subsequent protocol messages by appropriate multiples. As with the last attack, the adversary does not create the shared key. She is able to fool other participants into believing messages came from her rather than from honest participants A_i's. We present in detail the attack on MAK B-j.

1. A_1 sends $g^{a_1}||\mathrm{Cert}_{A_1}$ to $D_{A_2,\cdots,A_n}$.
2. D_{A_1} computes $\mu_{D_{A_1}} = g^{\delta^{n-j}x_1}$ and registers $\mu_{D_{A_1}}$ as part of her $\mathrm{Cert}_{D_{A_1}}$.
3. D_{A_1} initiates a run of protocol MAK B-j by sending $g^{\delta^{n-j-1}a_1}||\mathrm{Cert}_D$ to A_2, $\cdots$, A_n.
4. A_i sends $g^{a_i}||\mathrm{Cert}_{A_i}$ to D and A_k, $\;k \neq i, k \neq 1, i = 2, 3, \cdots, n$.
5. $A_i, i = 2, \cdots, n$ computes

$$K_{D_{A_1}A_2\cdots A_n} = \prod_{\binom{n-1}{j}} e_{n-1}(g,\cdots,g)^{\delta^{n-j}x_1a_2\cdots x_{i_1}\cdots x_{i_{j-1}}a_n} \cdot \prod_{\binom{n-1}{j-1}} e_{n-1}(g,\cdots,g)^{\delta^{n-j-1}a_1\cdots x_{i_1}\cdots x_{i_j}\cdots a_n}.$$

6. D_{A_i} sends $g^{\delta a_i}||\mathrm{Cert}_{A_i}$ to A_1, $\;i = 2, \cdots, n$.
7. A_1 computes a key

$$K_{A_1D_{A_2\cdots A_n}} = \prod_{\binom{n-1}{j}} e_{n-1}(g,\cdots,g)^{\delta^{n-j}x_1a_2\cdots x_{i_1}\cdots x_{i_{j-1}}\cdots a_n} \cdot \prod_{\binom{n-1}{j-1}} e_{n-1}(g,\cdots,g)^{\delta^{n-j-1}a_1\cdots x_{i_1}\cdots x_{i_j}\cdots a_n}.$$

8. Now D, forwarding A_1's messages encrypted under key $K_{D_{A_1}A_2\cdots A_n} = K_{A_1D_{A_2\cdots A_n}}$ to $A_2, \cdots, A_n$, and fools them into believing that A_1's message come from her.

This attack does not seem to apply to MAK-A and MAK-C because of the way in which long-term private key components are separated from the short-term components in $K_{A_1,\cdots,A_n}$ in MAK-A and due to the use of a hash function in MAK-C.

Unlike the unknown key share attack on the MQV protocol, the adversary in our attack does not know his long term private key. Therefore all these source substitution attacks are easily prevented if the CA insists that each registering party provides a proof of possession of his private key when registering a public key.

5.4 Known Session Key Attack on MAK-A

We now present a known session key attack on MAK-A that makes use of session interleaving and message reflection. In the attack, D interleaves n sessions and reflect message originating from A_1 back to A_1 in the different protocol runs. The result is that the session keys agreed in the n runs are identical. So D, upon obtaining one of them, gets keys for $(n-1)$ subsequent sessions as well.

A_1 is convinced to initiate n sessions with D:
Session $S_1 : A_1 \rightarrow D_{A_2 \cdots A_n} : \quad g^{a_{11}} \parallel \mathrm{Cert}_{A_1} \ (S_{11})$
Session $S_2 : A_1 \rightarrow D_{A_2 \cdots A_n} : \quad g^{a_{12}} \parallel \mathrm{Cert}_{A_1} \ (S_{21})$
$\vdots$
Session $S_n : A_1 \rightarrow D_{A_2 \cdots A_n} : \quad g^{a_{1n}} \parallel \mathrm{Cert}_{A_1} \ (S_{n1})$

D reflects and replays pretending to be $A_2, \cdots, A_n$, to complete session S_1.
$D_{A_k} \rightarrow A_1 : \ g^{a_{1k}} \parallel \mathrm{Cert}_{A_k} \ (S_{1k}), \ \ k = 2, 3, \cdots, n.$
Similarly the next (n-1) sessions are completed by $D_{A_2, \cdots, A_n}$ as follows.
$D_{A_k} \rightarrow A_1 : \ g^{a_{1k+1}} \parallel \mathrm{Cert}_{A_k} \ (S_{2k}), \ k = 2, 3, \cdots, n, \ \ a_{1n+1} = a_{11}.$
$\vdots$
$D_{A_k} \rightarrow A_1 : \ g^{a_{1k+(n-2)}} \parallel \mathrm{Cert}_{A_k} \ (S_{nk}), \ k = 2, 3, \cdots, n, \ \ a_{1n+i} = a_{1i}.$

D now obtains the first session key $e_{n-1}(g, \cdots, g)^{a_{11}a_{12}\cdots a_{1n} + x_1 x_2 \cdots x_n}$. She knows the keys for the next $(n-1)$ sessions, as these are identical to the first session key. This attack only works on MAK-A because of the symmetry of the short-term components, and attacks of this type do not appear to apply to MAK B-j or MAK-C.

5.5 Multilateral Attack on MAK B-$(n-1)$

Our multilateral attack on MAK B-$(n-1)$ allows an adversary D(who has a certificate Cert_D containing $\mu_D = g^{\triangle}$) to compute a session key $K_{A_1 \cdots A_n}$ previously shared by the honest entities $A_i (1 \leq i \leq n)$.

The attack is summarized as follows.

1. D eavesdrops to obtain $g^{a_{11}}, g^{a_{21}}, \cdots, g^{a_{n1}}$ from the session in which $K_{A_1 \cdots A_n} = e_{n-1}(g, \cdots, g)^{(x_1 x_2 \cdots x_{n-1}) a_{n1} + \cdots + (x_2 \cdots x_n) a_{11}}$ is agreed among entities $A_i (1 \leq i \leq n)$.

2. D initiates n-protocol runs. The first one is :

• 1st run(S_1)

$D \rightarrow A_2, \cdots, A_n \ : \ g^{a_{11}} \| \ \mathrm{Cert}_D \qquad (S_{11})$
$A_2 \rightarrow D, A_3, \cdots, A_n \ : \ g^{a_{12}} \| \ \mathrm{Cert}_{A_2} \qquad (S_{12})$
$A_3 \rightarrow D, A_2, A_4, \cdots, A_n \ : \ g^{a_{13}} \| \ \mathrm{Cert}_{A_3} \qquad (S_{13})$
$\vdots$

$A_n \to\ D, A_2, A_3, \cdots, A_{n-1}\ :\ g^{a_{1n}} \|\ \mathrm{Cert}_{A_n} \qquad (S_{1n})$

The session key agreed is

$K_{DA_2A_3\cdots A_n} = e_{n-1}(g, \cdots, g)^{(\triangle x_2\cdots x_{n-1})a_{1n}+\cdots+(\triangle x_3\cdots x_{n-1}x_n)a_{12}+(x_2\cdots x_n)a_{11}}$

• 2nd run(S_2) :

$D \to\ A_1, A_3\cdots, A_n\ :\ g^{a_{21}} \|\ \mathrm{Cert}_D \qquad (S_{21})$

$A_2 \to\ D, A_3, \cdots, A_n\ :\ g^{a_{22}} \|\ \mathrm{Cert}_{A_1} \qquad (S_{22})$

$A_3 \to\ D, A_1, A_4, \cdots, A_n\ :\ g^{a_{23}} \|\ \mathrm{Cert}_{A_3} \qquad (S_{23})$

$\vdots$

$A_n \to\ D, A_1, A_3, \cdots, A_{n-1}\ :\ g^{a_{2n}} \|\ \mathrm{Cert}_{A_n} \qquad (S_{2n})$

The session key agreed is

$K_{A_1DA_3\cdots A_n} = e_{n-1}(g, \cdots, g)^{\alpha_2}$

where $\alpha_2 =(\Delta x_1x_3\cdots x_{n-1})a_{2n}+(\Delta x_1x_3\cdots x_{n-2}x_n)a_{2n-1}+\cdots+(\Delta x_1x_4\cdots x_n)a_{23}$ $+(x_1x_3\cdots x_n)a_{22} + (\Delta x_3\cdots x_n)a_{21}$.

In the final run,

• nth run(S_n) :

$D \to\ A_1, A_2\cdots, A_{n-1}\ :\ g^{a_{n1}} \|\ \mathrm{Cert}_D \qquad (S_{n1})$

$A_1 \to\ D, A_1, \cdots, A_{n-1}\ :\ g^{a_{n2}} \|\ \mathrm{Cert}_{A_1} \qquad (S_{n2})$

$\vdots$

$A_{n-1} \to\ D, A_1, A_2, \cdots, A_{n-2}\ :\ g^{a_{nn-1}} \|\ \mathrm{Cert}_{A_{n-1}} \qquad (S_{nn})$

The session key agreed is

$K_{A_1A_2\cdots A_{n-1}D} = e_{n-1}(g, \cdots, g)^{\alpha_n}$

where $\alpha_n =(x_1x_2\cdots x_{n-1})a_{nn}+(\Delta x_1x_2\cdots x_{n-2})a_{nn-1}+\cdots+(\Delta x_1x_3\cdots x_{n-1})a_{n2}$ $+(\Delta x_2x_3\cdots x_{n-1})a_{n1}$.

3. Therefore D can obtain the session key by computing

$$K_{A_1A_2\cdots A_n} = K_{DA_2\cdots A_n}\cdot e_{n-1}(g, \cdots, g)^{-I_1}\cdot K_{A_1DA_3\cdots A_n}\cdot e_{n-1}(g, \cdots, g)^{-I_2}\cdot\ \cdots$$
$$\cdot K_{A_1\cdots A_{n-1}D}\cdot e_{n-1}(g, \cdots, g)^{-I_n},$$

where $I_1 = (\Delta x_2\cdots x_{n-1})a_{1n}+(\Delta x_2\cdots x_{n-2}x_n)a_{1n-1}+\cdots+(\Delta x_3\cdots x_{n-1}x_n)a_{12}$,

$I_2 = (\Delta x_1x_3\cdots x_{n-1})a_{2n}+(\Delta x_1x_3\cdots x_{n-2}x_n)a_{2n-1}+\cdots+(\Delta x_1x_4\cdots x_n)a_{23}$ $+(\Delta x_3\cdots x_n)a_{21}$, $\cdots\cdots\cdots$ and

$I_n = (\Delta x_1x_2\cdots x_{n-2})a_{nn-1}+\cdots+(\Delta x_1x_3\cdots x_{n-1})a_{n2}+(\Delta x_2\cdots x_{n-1})a_{n1}$.

This multilateral attack is possible because of the algebraic relationship between the long and short term key components in $K_{A_1A_2\cdots A_n}$. It can be prevented using appropriate key derivation. This attack does not work on MAK-A and MAK $B-j(j=1,2,\cdots,n-2)$ because we can not isolate individual short term key components. This type of attack is eliminated in MAK-C because of the binding of each entity's short and long-term key using a hash function.

5.6 Security summary

We examined attacks on our protocols heuristically through the section 5 and suggested how to prevent them. Now we summarize the security attributes of our protocols for given attacks. From the following table 1, we agree the protocol MAK-C which requires hash function is the most preferable.

Table 1

	MK(B & S)	MAK-A	MAK B-$j^{(i)}$	MAK-C
implicit key authentication	No	Yes	Yes	Yes
Known session key secure	No	No	Yes	Yes
Perfect forward secure	n/a	Yes	No$^{(ii)}$	Yes
KC impersonation secure	n/a	No	Yes	Yes
Unknown key-share secure	No	Yes$^{(iii)}$	Yes$^{(iii)}$	Yes$^{(iv)}$

<Comparison of security goals and attributes for one round MAK protocols>

(i) $j=1,\cdots,n-1$
(ii) Not forward secure when $(n-j+1)$ long-term secret keys are compromised, but still forward secure for a compromise of $n-j$ or less such keys.
(iii) If the **CA** checks that public keys are only registered once, and if inconvenient use (iv).
(iv) If the **CA** verifies that each user is in possession of the long-term secret key corresponding to his public key.

6 Conclusions

We have constructed multi-party authenticated key agreement protocols from multilinear forms. We developed the protocols on theoretical basis since it is still open problem to build efficiently computable multilinear forms. Our analysis tells that MAK-C is the most secure, followed by MAK B-1, MAK B-2, $\cdots$, MAK B-$(n-1)$. It is also desirable to develop appropriate models for security of conference key agreement protocols and find multilinear-based protocols that are provably secure in that setting. The work of ([6], [7]) provides an excellent start in this direction.

References

[1] G. Atenies, M. Steiner, G. Tsudik. Authenticated group key agreement and friends, ACM Conference on Computer and Communications Security, 1998.

[2] G. Ateniese, M. Steiner, and G. Tsudik. New Multiparty Authentication Services and Key Agreement Protocols, *Journal of Selected Areas in Communications,* 18(4):1-13, IEEE, 2000.

[3] C. Becker and U. Willie, Communication complexity of group key distribution, ACM conference on Computer and Communication Society, 1998.

[4] D. Boneh and A. Silverberg, Applications of Multilinear forms to Cryptography, Report 2002/080, http://eprint.iacr.org, 2002.

[5] D.Boneh and R.Venkatesan, Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes, Advances in Cryptology-Crypto' 96 , vol 1109 LNCS. pages 129-142, Springer-Verlag, 1996.

[6] E. Bresson, O. Chevassut, D. Pointcheval, and J. J. Quisquater. Provably Authenticated Group Diffie-Hellman Key Exchange, In Proceedings of the 8th ACM Conference on Computer and Communications Security, Philadelphia, USA. pp. 255-264. November 2001.

[7] E. Bresson, O.Chevassut and D. Pointcheval. Provably Authenticated Group Diffie-Hellman Key Exchange - The Dynamic Case, In Advances in Cryptology - ASIACRYPT 2001, Gold Coast, Australia. LNCS 2248, pp. 290-309.

[8] M. Burmester and Y. Desmedt. A Secure and Efficient Conference key Distribution System, Advances in Cryptology-Eurocrypto'94, LNCS, Springer Verlag, 275-286, 1995.

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

[10] A. Joux. A one round protocol for tripartitie Diffie-Hellman, In W. Bosma, editor, *Proceedings of Algorithmic Number Theory Symposium -ANTS IV*, volume 1838 of *Lecture Notes in Computer Science*, pages 385-394. Springer Verlag, 2000.

[11] Y, Kim, A. Perrig, and G. Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative groups, *In ACM CCS'00*, pages 235-244, ACM press, November 2000.

[12] L. Law, A. Menezes, M. Qu, J. Solinas, and S. Vansone. An eficient protocol for authenticated key agreement, Technical Report CORR 98-05, Department of C & O, University of Waterloo, 1998, To appear in Designs, Codes and Cryptography.

[13] A. Menezes, P.C. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*, CRC Press, Boca Raton, 1997.

[14] T. Matsumoto, Y. Takashima, and H. Imai. On seeking smart public-key-distribution systems, *Trans. IECE of Japan,* E69:99-106, 1986.

[15] Sattam S. Al-Riyami and Kenneth G. Paterson, Authenticated Three Party Key Agreement Protocols from Pairings, Report 2002/035, http://eprint.iacr.org, 2002

[16] S. Blake-Wilson and A. Menezes. Security profs for entity authentication and authenticated key transport protocols employing asymmetric techniques, In B. Christinason, B. Crispo, T. Lomas, and M. Roe, editors, *Proceedings of the 5th International Workshop on Security Protocols,* volume 1361 of *Lecture Notes in Computer Science,* pages 137-158. Springer Verlag, 1997.

[17] M. Stein, G. Tsudik, M. Waidner. Diffie Hellman Key Distribution Extended to Group Communication, ACM conference on computer and communication security, 1996.