Multiparty Key Agreement using Bilinear Map

Nam-Su Jho and Myung-Hwan Kim

ABSTRACT — A key agreement protocol is a cryptographical primitive which allows participants to share a common secret key via insecure channel. In particular, a multiparty key agreement protocol is a key agreement protocol that can manage arbitrary number of participants at once. In the security point of view, authentication and forward secrecy are the most important requirements in such protocols. One interesting problem in key agreement protocols is to construct a multiparty key agreement protocol satisfying the above security requirements with minimal number of communication rounds (i.e. one-round). In literature, there has been no one-round multiparty key agreement protocol that satisfies both of authentication and forward secrecy.

In this paper, we present a new multiparty key agreement protocol using bilinear map and adopting the key generation center. The protocol demands only one–round for arbitrary number of participants to share a group key and satisfies both authentication and (partial) forward secrecy.

Keywords—Multiparty Key Agreement, Authentication, Bilinear Map, Weil Pairing

I. Introduction

A key agreement protocol is a cryptographical primitive which allows participants to share a common secret key via insecure channel. The Diffie–Hellman key agreement protocol [1] in 1976 is the first practical solution to the key agreement problem. After Diffie and Hellman's work, many security requirements and various solutions have been discussed and proposed up to now.

The most important requirement for a key agreement protocol is authentication of participants involved in the protocol. Although the Diffie–Hellman key agreement protocol is very simple and efficient without requiring any preparation for the protocol, it is seriously vulnerable to the well known man–in–the–middle–attack and is not sufficient to accomplish the purpose of key agreement protocols. The reason is that the Diffie–Hellman key agreement allows no authentication of participants. Formal definitions of various security requirements are given in the next section.

In the early days, improving on Diffie-Hellman key agreement protocol adopting authentication is mainly researched. MTI protocol [2] in 1986 and MQV protocol [3] in 1998 are representative results. However, only heuristic security analyses are given. After formal security model is proposed by [4], [5] and [6], attempts of constructing provably secure key agreement protocol are made. Protocol proposed by Jeong, Kats and Lee [7] in 2004 is an example of such a key agreement protocol. Identity–Based key agreement protocol is another topic which attracts one's attention. Smart [8] and Smart [9] in 2002 proposed ID-based key agreement protocols modifying MTI key agreement protocol using the idea of identity based encryption, independently. This works become the basis of other ID-based key agreement protocols proposed later.

On the other hand, multiparty key agreement protocols – key agreement protocols which can manage a group containing arbitrary number of participants – attract attention and various methods to generalize two–party key agreement protocols to multiparty protocols have been proposed.

Ingemarsson, Tang and Wong made a protocol which is natural extension of the classical Diffie–Hellman key agreement protocol, 1982 [10]. Similarly to the classical DH key agreement, this protocol is secure against a passive adversary only. In 1995, Burmester and Desmedt [11] proposed a group key agreement protocol which reduces round complexity (roughly, the number of communication exchanged) remarkably. The modified version and the security proof of this protocol is presented in 2005 by Burmester et al. [12] adopting the formal security model.

. Nam-Su Jho (email: nsjho@etri.re.kr) – Electronics and Telecommunications Research Institute, Daejeon, Korea.

. Myung-Hwan Kim (email: mhkim@math.snu.ac.kr) – Department of Mathematical Sciences, Seoul National University, Seoul, Korea.

Bresson, Chevassut and Pointcheval [6], [13], [14] set up the formal security model about multiparty key agreement protocol and provided protocols which are provably secure authenticated group key agreement protocols. Recently, Abdalla, Bresson, Chevassut and Pointcheval [15] proposed password–based key agreement, i.e. pre–shared (or distributed) password is used to authenticate each participant.

In order to measure the efficiency of key agreement protocols, the computation complexity, the communication complexity and the round complexity are often considered. The computation complexity and the communication complexity denote the amount of computations of each participant to obtain common secret session key and the amount of broadcasted messages of each participant to other ones, respectively. A round means one broadcasting session in which every participant can cast messages to others but all at once. Therefore, in a one round key agreement protocol for example, all the messages necessary should be cast by participants in parallel. The round complexity is simply the number of rounds needed to complete the protocol. Minimizing round complexity is very important in designing key agreement protocols as well as reducing other complexities.

To now, there are few one round multiparty key agreement protocols proposed and the one proposed by Boyd, Manuel and Nieto [16] is the only one which is proved to be secure as an implicitly authenticated protocol. The protocol of Boyd et al. is a simple combination of a public key encryption system and a signature system. However, this protocol does not satisfy forward secrecy and explicit key authenticated. Since the protocol by Boyd et al. does not satisfy some security requirements including forward secrecy, it is not regarded as a good enough solution. So it is still open to construct a one round multiparty key agreement protocol satisfying enough security requirements.

In this paper, we proposed new multiparty key agreement protocol, OAK, which requires just one round without dependency on the number of participants. The OAK satisfies the (partial) forward secrecy and provides the key confirmation secrecy. However, the cost of minimizing the round complexity is to bring in the trusted key generation center, which is used often in ID-based key agreement protocols. A bilinear map defined on an elliptic curve [17], [18], [19] is a main building block used to construct new protocol. Joux's key agreement [20] was the first positive result which used a bilinear map to construct application in cryptography. After this work, a bilinear map comes into the spotlight and a number of pairing-based protocols were proposed

The organization of this paper is as follows: In section 2, we summarize security requirements for key agreement protocols and security model. In section 3, we will describe the construction of our new key agreement protocol OAK. The security proof is presented in section 4 and finally the section 5 concludes this paper.

II. Security Definitions and Model.

1. Authentication

Roughly, authentication of key agreement protocol has two aspects, one is authentication of participants and the other is authentication of the computed session key.

A. Implicit Key Authentication

The implicit key authentication is based on the authentication of participants for preventing impersonation (for example man-in-the-middle) attack. The purpose of the attacker is sharing secret session key without detecting of legitimate participants by replacing legitimate participant. Thus, generally the attacker is not considered to be a malicious participant. The implicit key authentication requires that each legitimate participant is assured that no other one except for other legitimate participants learns the established group key.

B. Key Confirmation

Key confirmation is the other aspect of the authentication. A key agreement protocol is said to provide key confirmation if each participant is assured that the other legitimate participant actually has possession of the same secret key to the key computed by him/her. In key confirmation security, it must be considered the case that the attacker is a malicious participant of the protocol. Because it is usually assumed that the attacker can control communication channel completely, the attacker always can prevent participants from sharing the secret session key. So key confirmation considers the case of that key agreement protocol is finished with no error detected.

C. Explicit Key Authentication

A key agreement protocol satisfying both implicit authentication and key confirmation is called an explicit authenticated key agreement protocol.

2. Forward Secrecy

The basic idea of this security notion is to maintain security of current communication sessions from the event which may occur in the future. Note that compromised long-term keys make the attacker impersonate without detection in the future

protocol runs.

A. (Partial) Forward Secrecy

A key agreement protocol is forward secure if the long-term secret keys of participants or more are compromised, the secrecy of previous session keys is not affected. The word 'partial' means that we assume that the attacker can compromise some, but not all, participants.

B. Perfect Forward Secrecy

A key agreement protocol is perfect forward secure if the long–term secret keys of all participants are compromised, the secrecy of previous session keys is not affected.

C. Weak Forward Secrecy

A key agreement protocol is weak forward secure if the protocol is forward secure against the attacker who can compromise only long–term secret keys from participants.

D. Strong Forward Secrecy

A key agreement protocol is strong forward secure if the protocol is forward secure against the attacker who can compromise internal states of participants additionally to long–term secret keys. Note that this notion is meaningful where participants has some reusable data, also called {\it ephemeral keys}, in his/her internal memory. It is also assumed that each participant can erase ephemeral keys if it will be no longer used, so an attacker can compromise some, but not all, ephemeral keys used in the past.

3. Other Security Requirements

Followings are often referred security requirements:

A. Known Session Key Secrecy

A protocol is called satisfying known session key security, if the protocol is still secure against the adversary who knows some other session keys. In other words, the knowledge about some previous session keys gives no help to guess other session keys.

B. No Key-Compromise Impersonation Secrecy

This means the compromise of a long-term secret key of a participant doesn't imply that other participants' secret key is compromised. Note that since authentication of participants depends on knowledge of long-term secret keys, the adversary who compromises a long–term secret key may impersonate the compromised participant. However, no key–compromise impersonation security guarantees that the attacker cannot impersonate other participants except compromised one.

C. No Key Control Secrecy

A key agreement protocol satisfying this security notion is sometimes called *contributory group key agreement*. No key control security means that shared secret session key must be determined by all participants who involved in key agreement protocol and no one can enforces some specific value on a shared secret session key.

4. Security Model

In this section, we describe the formal security model for security analysis. This model is following security models of Bresson et al. [6] and Dutta et al. [21], [22]. In these models, each user in the key agreement protocol is considered as an oracle and every interaction of the adversary with users is considered as an oracle query. We assume that the adversary has the ability of controlling the network completely.

Let $\mathcal{S} = \{u_1, u_2, \ldots, u_t\}$ be a set of all users. Without loss of generality, we can give the unique number to each key agreement instance although key agreement instances may occur simultaneously. Let Π_ℓ be the ℓ-th key agreement instance and $\mathcal{S}_\ell$ be the subset of $\mathcal{S}$ which contains every user involved in the ℓ-th key agreement instance with $|\mathcal{S}_\ell| = n_\ell \leq t$.

More precisely, Π_ℓ is the set of instance of users who are in $\mathcal{S}_\ell$, let Π_ℓ^u be the instance of user u in the ℓ-th session. M_ℓ^u means the message which is broadcasted by Π_ℓ^u and M_ℓ means the whole message which is broadcasted in the instance Π_ℓ. In other words, M_ℓ is the all messages in the ℓ-th session.

And K_ℓ is the shared secret session key in the ℓ-th session. For each user u_i, k_i means the user's long-term secret key and $r_{\ell i}$ means the users ephemeral key, i.e. random nonce used to share the session key in the ℓ-th session by the user u_i.

Followings are oracle queries which are executed by the adversary:

Eavesdrop(ℓ) : This query models a passive attack, in which the adversary only eavesdrop messages among the users in $\mathcal{S}_\ell$. The reply of this oracle is M_ℓ.

Initiate($\mathcal{S}_\ell$) : This query models an active attack in which the adversary initiates key agreement protocol among the users in $\mathcal{S}_\ell$. $\mathcal{S}_\ell$ may be chosen by the adversary or randomly given. The adversary obtains all messages among the users, i.e. the reply of this oracle is M_ℓ.

Send(u, $\mathcal{S}_\ell$, M'_ℓ) : This query models an active attack in which the attacker send messages which can be originated or modified by the adversary to other user(s). Here, we simplify

this query as following:

- *Step1 :* The adversary makes a message M'_ℓ.
- *Step2 :* Or the adversary intercepts the message M_ℓ sent to the user $u \in \mathcal{S}_\ell$ and modify to M'_ℓ.
- *Step3 :* The adversary sends M'_ℓ to the user u instead.

The reply of this oracle is M_ℓ^u.

KeyGen : This query models an active attack in which the adversary contacts the KGC to obtain a long–term secret user key. The reply of this oracle is a long-term secret user key k_j. Note that the KGC never issues same long–term key to two or more users, i.e. the adversary can obtains k_j for the index j which is never used.

Reveal(ℓ) : This query models the event of leakage of a particular session key to the adversary. The reply of this oracle is the session key K_ℓ.

Corrupt(u) : This query models the corruption of the long–term secret key of the user u. The reply of this oracle is k_u. Note that in OAK the strong corruption is meaningless. So we make only one corrupt model.

Test(ℓ) : The reply of this oracle is the real session key K_ℓ if b=1 or a random key R_ℓ if b=0, where b is randomly chosen bit by the Test oracle. This oracle is used to compute the adversary's ability to distinguish the real session key from a random key. The adversary can query this oracle at any time but only once.

III. New Protocol

1. Signature Scheme

Let k be the security parameter. A digital signature scheme $\Sigma=(\mathcal{K}_\mathcal{G}, \mathcal{S}_{\mathcal{IG},\mathcal{N}}, \mathcal{V}_{\mathcal{ER}})$ consists of the following three algorithms:

- The key generation algorithm $\mathcal{K}_\mathcal{G}$ is a probabilistic algorithm with an input the security parameter λ and outputs (e, d) the secret key and public key pair.
- The signing algorithm $\mathcal{S}_{\mathcal{IG},\mathcal{N}}$ is a probabilistic algorithm with an input the secret key and the message (d, m) outputs the signature of m, σ_m.
- The verification algorithm $\mathcal{V}_{\mathcal{ER}}$ is a deterministic algorithm with an input the triple verification key, message and signature (e, m, σ), outputs TRUE if the signature is valid or FALSE otherwise.

The signature scheme $\Sigma = (\mathcal{K}_\mathcal{G}, \mathcal{S}_{\mathcal{IG},\mathcal{N}}, \mathcal{V}_{\mathcal{ER}})$ is secure if it is computationally impossible for any adversary to forge a signature on any message (*existential forgery*) even under adaptive chosen–message attack. In other words, there is no polynomial time algorithm with an input the public verification key e outputs (m, σ) which satisfying $\mathcal{V}_{\mathcal{ER}}(e, m, \sigma)$ = TRUE with non–negligible probability, even it can query to signing oracle at any time.

2. Bilinear Map

Assume that $\mathbb{G}_1$ and $\mathbb{G}_2$ are two cyclic groups of prime order p and P is a generator of $\mathbb{G}_1$, for convenience we'll denote $\mathbb{G}_1$ as an additive group. A map e: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is a bilinear map if it satisfies:

- Bilinear: for all $P,Q \in \mathbb{G}_1$ and $a,b \in \mathbb{Z}$, we have $e(aP,bQ) = e(P,Q)^{ab}$.
- Non-degenerate: $e(P,P) \neq 1$.

By a bilinear group, we mean a group in which the group operation can be computed efficiently and there exists an efficiently computable bilinear map as above. Modified Weil pairing [25] and Tate pairing [26] are the typical examples of bilinear maps which are defined on elliptic curves.

3. Construction

We now propose the new key agreement protocol which requires only one round to establish shared session key and satisfies explicit authentication. In literature there is no one-round multiparty key agreement protocol which satisfies explicit authentication.

The One–round Authenticated Key agreement protocol (OAK) consists of following three algorithms: Setup, KeyGeneration and KeyAgreement.

A. Setup

Setup is executed by the key generation center KGC which is assumed to be a trusted party.

- With the security parameter λ, the KGC chooses a prime q and generates two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of order q, where $\mathbb{G}_1$ is a bilinear group with a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$.
- The KGC randomly chooses P which is a generator of $\mathbb{G}_1$, α and s from $\mathbb{Z}_q^*$ and an integer t.
- The KGC publishes $\mathbb{G}_1$, $\mathbb{G}_2$, e, t, P, αP, $\alpha^2 P$, …, $\alpha^t P$ as public information of the protocol. Note that α and s is the secret information that only the KGC knows.

B. Key Generation

If a participant wants to join the protocol, he/she sends his/her own information with which the KGC can identify him/her and his/her own verification key. After identifying process, the KGC gives the secret key to him/her as followings:

- Assume that each participant u runs $\mathcal{K}_\mathcal{G}$ to obtain a pair of

signing and verification key (d_u, e_u) before the KeyGeneration process.
- Each participant u is assigned unique identifier $I(u)$ (an integer between 1 and t) and KGC computes $s\alpha^{I(u)}P$.
- The KGC gives the secret key $s\alpha^{I(u)}P$, to the participant u and publishes the participant's identifier together with the verification key e_u.

C. Key Agreement

It is assumed that each session has a unique session identifier, the session number ℓ. Let S_ℓ be a set of participants who want to agree a secret session key in the ℓ-th session.

- Each participant u in S_ℓ randomly chooses $r_u \in Z_p$ and broadcasts $M_u = \{m_{u,v} = r_u\alpha^{t+1-I(v)}P \mid v \in S_\ell, v \neq u\}$ together with the signature $\sigma_u = S_{SK,M}(d_u, \ell \mid M_u)$ to all other users.
- Each participant checks message integrity verifying the signatures. If there is an unverifiable message then KeyAgreement outputs FAIL and terminates.
- Optionally, each participant may check message reliability of $M_u = \{m_{u,v} \mid v \in S_\ell, v \neq u\}$ by verifying the equations $e(\alpha^{t+1-I(v')}P, m_{u,v}) = e(\alpha^{t+1-I(v)}P, m_{u,v'})$. If KeyAgreement finds some unreliable message, it outputs FAIL and terminates.
- Each participant u in S_ℓ computes $\Sigma_{v \in S_\ell} r_v\alpha^{t+1-I(u)}P$ and the secret session key, $K_\ell = e(s\alpha^{I(u)}, \Sigma_{v \in S_\ell} r_v\alpha^{t+1-I(u)}P) = \mathrm{Exp}[e(P,P), \alpha^{t+1}s(\Sigma_{v \in S_\ell} r_v)]$, where $\mathrm{Exp}[a,b]$ means a^b.
- After computing the session key, each user u_i erases his/her own random nonce r_i.

Note that the protocol adopts the KGC for generating of public keys and each user's secret key similarly to ID-based key agreement protocols. By necessity, we assume that the KGC is trusted. Because the leakage of the secret key of the KGC breaks forward secrecy and if the KGC itself is compromised then the protocol cannot be guaranteed to be secure no longer. Even though the assumption is the cost for reducing the round complexity to be optimized, it is somewhat expensive and removing this assumption is the next step of this work.

4. Efficiency

- Rounds: One, regardless to the size of S_ℓ.
- Communication: Each user broadcasts n-1 elements of G_1.
- Computation: For each user, n scalar multiplications in G_1, n-1 additions in G_1, 1 computation of signature, n-1 verifications of signatures and 1 bilinear map computation are required.

[Remark] Checking reliability of messages is required when participants want key confirmation property. Note that message reliability can be checked with computing maximum $2(n\text{-}2)$ pairing computations and $(n\text{-}3)n$ additions in G_1.

[Remark] In the viewpoint of computation, this protocol is not efficient comparing to other group key agreement protocols. The main contribution of this paper is the optimization of round complexity satisfying security requirements.

IV. Security

1. Assumptions

A. Discrete Logarithm Assumption

Let $\mathrm{Gen}(1^\lambda)$ be an algorithm which generates a λ-bit prime q and a multiplicative group $G = \langle g \rangle$ of order q. Then the discrete logarithm assumption says that for all probabilistic polynomial time adversary $\mathcal{A}$,

$$\Pr[(q,g) \leftarrow \mathrm{Gen}(1^\lambda); \mathcal{A}(q,g,y) = x : g^x = y]$$

is negligible.

B. Bilinear Diffie-Hellman Assumption

Let $\mathrm{Gen}(1^\lambda)$ be an algorithm which generates a λ-bit prime q and two groups $G_1 = \langle P \rangle$ and G_2 of order q, where G_1 is a bilinear group with a bilinear map $e : G_1 \times G_1 \rightarrow G_2$. Additionally, we assume that the discrete logarithm problems (DLP) in both G_1 and G_2 are hard. The bilinear Diffie-Hellman assumption says for all probabilistic polynomial time adversary $\mathcal{A}$,

$$\Pr[(q,P) \leftarrow \mathrm{Gen}(1^\lambda); \mathcal{A}(P,aP,bP,cP) = x : x = e(P,P)^{abc}]$$

is negligible.

C. Bilinear Diffie-Hellman Exponent Assumption

In 2005, Boneh et al. [23] proposed bilinear Diffie–Hellman Exponent (BDHE) assumption to generate an efficient HIBE. Later BDHE assumption is used in construction a broadcast encryption system [24] and considered as a general assumption in a bilinear group.

Let $\mathrm{Gen}(1^\lambda)$ be an algorithm which generates a λ-bit prime q and two groups $G_1 = \langle P \rangle$ and G_2 of order q, where G_1 is a bilinear group with a bilinear map $e : G_1 \times G_1 \rightarrow G_2$. Additionally, we assume that the discrete logarithm problems (DLP) in both G_1 and G_2 are hard. The t-BDHE problem in G_1 is stated as follows: given a vector of $2t + 1$ elements

$$(Q, P, \alpha P, \alpha^2 P, \ldots, \alpha^t P, \alpha^{t+2} P, \ldots, \alpha^{2t} P) \in G_1^{2t+1}$$

as input, output $\mathrm{Exp}[e(P,Q), \alpha^{t+1}] \in G_2$. Note that the input vector is missing the term $\alpha^{t+1}P$ so that the bilinear map seems

to be of little help in computing the required Exp[$e(P,Q)$, α^{t+1}].

The t-BDHE assumption says that for all probabilistic polynomial time algorithm $\mathcal{A}$, the probability that

$$\Pr[(q,e,P)\leftarrow \mathrm{Gen}(1^{\lambda}),\ Q \leftarrow_R \mathbb{G}_1,\ \alpha \leftarrow_R \mathbb{Z}_q^*;\ \mathcal{A}(Q, P, \alpha P, \ldots, \alpha^t P, \alpha^{t+2} P, \ldots, \alpha^{2t} P) = x : x = e(P,Q)^{t+1}]$$

is negligible.

The decisional version of the t-BDHE problem in $\mathbb{G}_1$ is defined analogously. Let $Y_{P,\alpha,t} = (\alpha P, \ldots, \alpha^t P, \alpha^{t+2} P, \ldots, \alpha^{2t} P)$. A probabilistic polynomial time algorithm $\mathcal{B}$ that outputs $b \in \{0,1\}$ has advantage ε in solving decision t-BDHE in $\mathbb{G}_1$ if

$$\left|\Pr[\mathcal{B}(Q,P,Y_{P,\alpha,t}, e(\alpha^{t+1}P,Q))=0] - \Pr[\mathcal{B}(Q,P,Y_{P,\alpha,t}, R)=0]\right| \geq \varepsilon$$

where the probability is over the random choice of generators P, Q in $\mathbb{G}_1$, the random choice of a in $\mathbb{Z}_p^*$, the random choice of $R \in \mathbb{G}_2$, and the random bits consumed by $\mathcal{B}$.

The decisional t-BDHE assumption holds in $\mathbb{G}_1$ if no probabilistic polynomial time algorithm has non–negligible advantage in solving the decisional t-BDHE problem in $\mathbb{G}_1$

D. Modified Bilinear Diffie–Hellman Exponent Assumption

Generalizing BDHE assumption, we obtain Modified Bilinear Diffie–Hellman Exponent (MBDHE) assumption. Security of our system is mainly based on MBDHE assumption.

Let $\mathrm{Gen}(1^{\lambda})$ be an algorithm which generates a λ-bit prime q and two groups $\mathbb{G}_1 = \langle P \rangle$ and $\mathbb{G}_2$ of order q, where $\mathbb{G}_1$ is a bilinear group with a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. Additionally, we assume that the discrete logarithm problems (DLP) in both $\mathbb{G}_1$ and $\mathbb{G}_2$ are hard.

Define the t-MBDHE problem in $\mathbb{G}_1$ as follows: given a vector of $2t + 1$ elements

$$\left(P,\ \alpha P,\ \alpha^2 P,\ \ldots,\ \alpha^t P,\ \begin{matrix} - \\ Q \end{matrix},\ \begin{matrix} \{\alpha^{t+i} P & \text{for } i \notin I\} \\ \{\alpha^{1-i} Q & \text{for } i \in I\} \end{matrix} \right) \in \mathbb{G}_1^{2t+1}$$

as input, output Exp[$e(P,Q)$, α^{t+1}] $\in \mathbb{G}_2$, where I is an set of integers such that $\{1\} \subseteq I \subseteq \{1,2,\ldots, t\}$. Note that for any $1 \leq i \leq t$ one of $\alpha^{t+i}P$ or $\alpha^{1-i}Q$ is missing in the input vector so that the bilinear map seems to be of little help in computing the required Exp[$e(P,Q)$, α^{t+1}] directly.

Note that for $I = \{1\}$ we have the t-BDHE problem so we can consider t-MBDHE problem as a generalization of t-BDHE problem. Following is an example of an input for t-MBDHE problem where t=5, I = {1,4}:

$$\left(P, \alpha P, \alpha^2 P, \alpha^3 P, \alpha^4 P, \alpha^5 P,\ \begin{matrix} - \\ Q \end{matrix},\ \alpha^7 P,\ \alpha^8 P,\ \begin{matrix} - \\ - \end{matrix},\ \begin{matrix} - \\ \alpha^{-3} Q \end{matrix},\ \alpha^{10} P,\ \begin{matrix} - \\ - \end{matrix} \right) \in \mathbb{G}_1^{11}$$

Let $Y_{P,Q,\alpha,I} = (\alpha P, \ldots, \alpha^t P, \{\alpha^{t+I} \mid i \text{ which is not in I}\}, \{\alpha^{1-i} Q \mid i \in I\})$. The t-MBDHE assumption says that for all probabilistic polynomial time algorithm $\mathcal{A}$ and all index set I, following probability is negligible.

$$\Pr[(q,e,P)\leftarrow \mathrm{Gen}(1^{\lambda}), Q\leftarrow_R \mathbb{G}_1, \alpha\leftarrow_R \mathbb{Z}_q^*;\ \mathcal{A}(P, Y_{P,Q,\alpha,I}) = x : x = e(P,Q)^{t+1}]$$

The definition of the decisional t-MBDHE assumption is similar to that of the decisional t-BDHE assumption. The decisional t-BMDHE assumption holds in $\mathbb{G}_1$ if for all probabilistic polynomial time algorithm $\mathcal{B}$ and all index set I, the advantage of $\mathcal{B}$

$$\left| \Pr[\mathcal{B}(P, Y_{P,Q,\alpha,I}, e(\alpha^{t+1}P,Q)) = 0] - \Pr[\mathcal{B}(P, Y_{P,Q,\alpha,I}, R) = 0] \right|$$

is negligible.

2. Basic Theorems

To prove the security of proposed OAK protocol, we define following adversaries:

- $\mathcal{A}_i$ is a polynomial time algorithm for a fixed integer i ($1 \leq i \leq t$) with an input

$$\left(\begin{matrix} \alpha P & , \alpha^2 P & , \ldots , & \alpha^{i-1} P & , & \alpha^i P & , & \alpha^{i+1} P & , \ldots , & \alpha^t P \\ r\alpha P & , r\alpha^2 P & , \ldots , & r\alpha^{i-1} P & , & - & , & r\alpha^{i+1} P & , \ldots , & r\alpha^t P \\ & & & & & \alpha^{t+1-i} Q & & & & \end{matrix} \right), R$$

  outputs TRUE if R = Exp[$e(P,Q)$, $r\alpha^{t+1}$] or FALSE otherwise. Adv($\mathcal{A}_i$) means the advantage of $\mathcal{A}_i$ i.e.

  Adv($\mathcal{A}_i$) = | Pr[the answer of $\mathcal{A}_i$ is correct] – 1/2 |

- $\mathcal{A}_I$ is a polynomial time algorithm for a fixed set of index $\emptyset \subsetneq I \subseteq \{1, \ldots, t\}$ with an input

$$\left(\begin{matrix} \alpha P, \alpha^2 P, \ldots , & \alpha^t P \\ \{r\alpha^i P & \text{for } i \notin I\} \\ \{\alpha^{t+1-i} Q & \text{for } i \in I\} \end{matrix} \right), R$$

  outputs TRUE if R = Exp[$e(P,Q)$, $r\alpha^{t+1}$] or FALSE otherwise. Similarly, Adv($\mathcal{A}_I$) means the advantage of $\mathcal{A}_I$.

- $\mathcal{A}'_I$ is a polynomial time algorithm for a fixed set of index $I \subseteq \{1, \ldots, t\}$ with an input

$$\left(\begin{matrix} \alpha P, \alpha^2 P, \ldots , & \alpha^t P \\ \{r\alpha^i P & \text{for } i \notin I\} \\ \{\alpha^{t+1-i} Q & \text{for } i \in I\} \\ \{r'\alpha^i P & \text{for } i \notin I\} \end{matrix} \right), e(P,Q)^{r'\alpha^{t+1}}, R$$

  outputs TRUE if R = Exp[$e(P,Q)$,$r\alpha^{t+1}$] or FALSE otherwise. Similarly, Adv($\mathcal{A}'_I$) means the advantage of $\mathcal{A}'_I$.

Theorem 1

Adv($\mathcal{A}_i$) is negligible under t-BDHE assumption.

(Proof) Suppose there exists a $\mathcal{A}_i$ such that Adv($\mathcal{A}_i$) $\geq 1/f(k)$,where $f(\cdot)$ is a polynomial. Then we can make a polynomial time algorithm which attacks t-BDHE problem with the advantage greater than $1/f(k)$.

$\mathcal{B}$ is an adversary who wants to attack t-BDHE problem. So $\mathcal{B}$ is given a problem instance:

$$(Q, \alpha P, \alpha^2 P, \ldots, \alpha^t P, \alpha^{t+2} P, \ldots, \alpha^{2t} P), R$$

$\mathcal{B}$ chooses random r and computes

$$r\alpha^{t-i} P, r\alpha^{t-i+1} P, \ldots, r\alpha^t P, r\alpha^{t+2} P, \ldots, r\alpha^{2t+1-i} P.$$

Note that i is an integer between 1 and t so all of above can be computed from the input for $\mathcal{B}$ and chosen r. Then $\mathcal{B}$ gives $\mathcal{A}$ the following input:

$$\left(\begin{array}{ccccccccc} \alpha P & , \ldots , & \alpha^{i-1} P & , & \alpha^i P & , & \alpha^{i+1} P & , \ldots , & \alpha^t P \\ r\alpha^{t-i} P & , \ldots , & r\alpha^t P & , & - & , & r\alpha^{t+2} P & , \ldots , & r\alpha^{2t+1-i} P \\ & & & & Q & & & & \end{array} \right), R$$

Finally, $\mathcal{B}$ outputs TRUE if $\mathcal{A}$ outputs TRUE or FALSE otherwise.

Now we compute the advantage of $\mathcal{B}$ with above attack. With an input described above $\mathcal{A}$ can distinguish R from $\mathrm{Exp}[e(P,Q), r\alpha^{t-i}\alpha^i] = \mathrm{Exp}[e(P,Q), r\alpha^{t+1}]$ with the advantage greater than $1/f(k)$. Because in this attack the advantage of $\mathcal{B}$ is same to the advantage of $\mathcal{A}$, $\mathrm{Adv}(\mathcal{B})$ is also greater than $1/f(k)$. So, $\mathrm{Adv}(\mathcal{A}_1)$ must be negligible under t-BDHE assumption. □

Theorem 2

$\mathrm{Adv}(\mathcal{A}_1)$ is negligible under t-MBDHE assumption.

(Proof) With similar arguments, we can prove this theorem. Suppose that there exists a polynomial time algorithm $\mathcal{A}$ with an index set $\emptyset \subsetneq I \subseteq \{1, \ldots, t\}$. Then we can construct a polynomial time algorithm $\mathcal{B}$ which attacks t-MBDHE problem with $I' := \{i\text{-}x\text{+}1 \mid i \in I\}$, where x is the smallest element of I.

For a given input

$$\left(\begin{array}{ccccccc} P, & \alpha P, & \alpha^2 P, & \ldots, & \alpha^t P, & - & , \ \{\alpha^{t+i'} P \quad \text{for } i' \notin I'\} \\ & & & & & Q & , \ \{\alpha^{1-i'} Q \quad \text{for } i' \in I'\} \end{array} \right), R$$

$\mathcal{B}$ randomly chooses r and computes following

$$\left(\begin{array}{ll} \alpha P, \alpha^2 P, \ldots, & \alpha^t P \\ \{r\alpha^{t+1-x+i} P & \text{for } i \notin I\} \\ \{\alpha^{x-i} Q & \text{for } i \in I\} \end{array} \right), R$$

Note that $r\alpha^{t+1-x+i} P$ is in fact r times $\alpha^{t+i'} P$ and $\alpha^{x-i} Q$ is exactly same to $\alpha^{1-i'} Q$. $\mathcal{B}$ outputs TRUE if $\mathcal{A}$ outputs TRUE or FALSE otherwise. Since the advantage of $\mathcal{B}$ is same to the advantage of $\mathcal{A}_1$, $\mathrm{Adv}(\mathcal{A}_1)$ cannot be less than $1/f(k)$ for any polynomial $f(\cdot)$ under t-MBDHE assumption. □

Theorem 3

If $\mathrm{Adv}(\mathcal{A}_1)$ is negligible, then $\mathrm{Adv}(\mathcal{A}'_1)$ is negligible under BDH assumption.

(Proof) To prove this theorem we need to define new adversary.

- $\mathcal{A}_0$ is a polynomial time algorithm with an input (P, aP, bP, cP, $e(P, P)^{acs}$, R) which distinguish R from $e(P, P)^{abs}$. Let $\mathrm{Adv}(\mathcal{A}_0)$ be the advantage of $\mathcal{A}_0$.

Step 1: Under the BDH assumption, $\mathrm{Adv}(\mathcal{A}_0)$ is negligible.

Suppose that there exists a polynomial time algorithm $\mathcal{A}_0$ with $\mathrm{Adv}(\mathcal{A}_0)$ is non-negligible. We can make following polynomial time algorithm $\mathcal{B}$ which attacks BDH problem with the non-negligible advantage.

From the input (P, aP, bP, cP, R), $\mathcal{B}$ choose random r and computes rP and $e(aP,cP)^r = e(P, P)^{acr}$. $\mathcal{B}$ gives (P, aP, bP, rP, $e(P, P)^{arc}$, R) to $\mathcal{A}$ as an input. Then $\mathcal{A}$ distinguish R from $e(P, P)^{abc}$ with non–negligible advantage. This is contradiction to the BDH assumption.

Step 2: With the assumption of $\mathrm{Adv}(\mathcal{A}_1)$ is negligible (let $\mathrm{Adv}(\mathcal{A}_1) = \varepsilon_1$), $\mathrm{Adv}(\mathcal{A}_{1'})$ is also negligible.

Suppose that there exists a polynomial time algorithm $\mathcal{A}_{1'}$ with non-negligible advantage. Then we can construct $\mathcal{A}_0$ with non-negligible advantage. With an input (P, aP, bP, cP, $e(P, P)^{acs}$, R), $\mathcal{A}_0$ chooses random x and computes followings:

$$\left(\begin{array}{ll} xP, x^2 P, \ldots, & x^t P \\ \{x^i bP & \text{for } i \notin I\} \\ \{x^{t+1-i} aP & \text{for } i \in I\} \\ \{x^i cP & \text{for } i \notin I\} \end{array} \right), e(P, P)^{acs}, R$$

and gives it as input to $\mathcal{A}'_1$.

Note that $e(P, P)^{acs}$ is not a proper input. But, from the Theorem2, $\mathcal{A}'_1$ cannot distinguish this input from a proper one. Therefore, $\mathcal{A}'_1$ works ordinarily and finally distinguishes R from $e(P, P)^{abs}$ with non–negligible probability. This is contradiction to the result of *Step1* above. Therefore, Theorem3 is proved. □

3. Forward Secrecy

Theorem 4

The proposed key agreement protocol, OAK, satisfies (partial) forward secrecy against the strong corruption.

In OAK, each user erases the random exponent after computing the session key. It is can be done because the random exponent is never reused in the future. And the adversary cannot obtain any more information from the strong corruption. So we can assume that the reply of Corrupt query is just the long-term key of participant.

To prove the forward secrecy, we consider following attack scenario: The adversary $\mathcal{A}_f$ chooses a user u and do Corrupt(u) query. After that, $\mathcal{A}_f$ chooses the session index ℓ which satisfies the condition that u is contained in $\mathcal{S}_\ell$. $\mathcal{A}_f$ do Test(ℓ) query to obtain the challenge. The challenge is K_ℓ if the random bit b chosen by Test oracle is 1 or R_ℓ otherwise. In this attack, the purpose of the adversary is to distinguish the shared session

key K_ℓ from a randomly chosen R_ℓ with the same length which are the reply of the oracle Test$_\ell$. The $\mathcal{A}_f$ is allowed to query to Eavesdrop, Initiate, KeyGen, Reveal and Send oracles freely at any time. Note that Reveal(ℓ) query is trivially forbidden.

In this attack model, we may ignore Initiate and Send queries. Since the reply of these oracles are information theoretically independent to K_ℓ and gives no help to distinguish the K_ℓ. Except Eavesdrop(ℓ), Eavesdrop queries are also helpless. Here, to ignore Eavesdrop queries, we assume that the adversary knows M_ℓ already. Since the reply of the Reveal can be duplicated by the $\mathcal{A}_f$ with randomly chosen exponent, we include one reply of Reveal as an input of $\mathcal{A}_f$.

Lemma 5

Let r_i be the random exponent of $u = u_i$ used in the session ℓ to share the session key K_ℓ. Then the probability of $\mathcal{A}_f$ to distinguish K_ℓ from R_ℓ is same to the probability of $\mathcal{A}_f$ to distinguish Exp[e(P, P), $s\alpha^{t+1}r_i$] from R'_ℓ, where R_ℓ and R'_ℓ are randomly chosen elements from G_2.

(Proof) It is trivial since that if $\mathcal{A}_f$ knows $s\alpha^i P$ then $\mathcal{A}_f$ can compute Exp[$e(P, P)$, $s\alpha^{t+1}r_j$] for $j \neq i$ and $j \in \mathcal{S}_\ell$ from M_ℓ and $s\alpha^i P$. So distinguishing K_ℓ is equivalent to distinguishing $\mathrm{Exp}[e(P, P), s\alpha^{t+1}r_i] = K_\ell / \Pi_{j \neq I}\mathrm{Exp}[e(P, P), s\alpha^{t+1}r_j]$ □

From this lemma, we can summarize $\mathcal{A}_f$ as a polynomial time algorithm with the inputs: the public information

$$\alpha P, \alpha^2 P, \ldots, \alpha^t P,$$

a reply of the Eavesdrop(ℓ) query

$$\{ r_i \alpha^j P \mid j \in \mathcal{S}_\ell, i \neq j \},$$

a long-term key

$$s\alpha^i P$$

from the Corrupt(i) query, replies of KeyGen queries

$$\{ s\alpha^j P \mid \text{for } j \text{ which is not in } \mathcal{S}_\ell \}$$

a reply from the Reveal query

$$\{ r'\alpha^j P \mid j \in \mathcal{S}_\ell, i \neq j \}, \mathrm{Exp}[e(P, P), r's\alpha^{t+1}],$$

and a challenge R_ℓ from the Test oracle. Therefore, $\mathcal{A}_f$ is exactly same to the $\mathcal{A}'_1$ defined in the section IV–2.

Theorems in the section IV–2 say that $\mathcal{A}_f$ cannot distinguish K_ℓ from R_ℓ under t-MBDHE and BDH assumptions. Note that if we restrict $\mathcal{A}$ not to query KeyGen query then OAK is forward secure under t-BDHE and BDE assumptions.

[Remark] The OAK does not satisfy perfect forward secrecy. Since the adversary can compute previous session keys with two long-term secret keys assigned to the participants who are involved to share the session key.

4. Authenticity

The authenticity of OAK is guaranteed by following two theorems :

Theorem 6

The OAK is an implicit authenticated key agreement protocol under the t-MBDHE and BDH assumptions.

Theorem 7

The OAK satisfies key confirmation security under the unforgeability of the signature scheme.

To prove the Theorem6, we consider the adversary $\mathcal{A}_a$ who wants to learn a session key K_ℓ without participating in $\mathcal{S}_\ell$.

The $\mathcal{A}_a$ chooses the user set $\mathcal{S}_\ell$ and do Initiate($\mathcal{S}_\ell$) query to obtain the reply M_ℓ. After that $\mathcal{A}_a$ do Test(ℓ) query, the purpose of the $\mathcal{A}_a$ is to distinguish the session key K_ℓ form the random challenge R_ℓ which is the reply of Test(ℓ). $\mathcal{A}_a$ can do any other queries freely in any time except Corrupt and Reveal(ℓ) queries. Since we assumed that the signature scheme is unforgeable and the $\mathcal{A}_a$ is not a member of $\mathcal{S}_\ell$, the $\mathcal{A}_a$ cannot success Send queries with non-negligible probability. So we can ignore Send queries. Since the reply of the Reveal can be duplicated by the $\mathcal{A}_a$ with randomly chosen exponent, we include one reply of Reveal as an input of $\mathcal{A}_a$.

We can summarize the $\mathcal{A}_a$ as a polynomial time algorithm with the inputs: the public information

$$\alpha P, \alpha^2 P, \ldots, \alpha^t P,$$

a reply of the query Initiate(ℓ)

$$\{ r_i \alpha^j P \mid i,j \in \mathcal{S}_\ell, i \neq j \},$$

replies of KeyGen queries

$$\{ s\alpha^j P \mid \text{for } j \text{ which is not in } \mathcal{S}_\ell \},$$

replies from the Reveal queries and a challenge R_ℓ from Test oracle.

Therefore, $\mathcal{A}_a$ is a polynomial time algorithm with input

$$\begin{pmatrix} \alpha P, \alpha^2 P, \ldots, \alpha^t P \\ \{r_1 \alpha^{t+1-j} P | j \in S, j \neq i_1\} \\ \{r_2 \alpha^{t+1-j} P | j \in S, j \neq i_2\} \\ \vdots \\ \{r_n \alpha^{t+1-j} P | j \in S, j \neq i_n\} \end{pmatrix}, \{s\alpha^j P | j \notin S\}, R_\ell$$

and replies of Reveal queries to distinguish Exp[$e(P, P)$, $s\alpha^{t+1}$ $(r_1 + r_2 + \ldots + r_n)$] from R_ℓ. And let $\mathcal{A}_1$ be a polynomial time algorithm with an input

$$\begin{pmatrix} \alpha P, \alpha^2 P, \ldots, \alpha^t P \\ \{r_1 \alpha^{t+1-j} P | j \in S, j \neq i_1\} \end{pmatrix}, \{s\alpha^j P | j \notin S\}, R'_\ell$$

and replies of Reveal queries to distinguish Exp[$e(P, P)$, $s\alpha^{t+1}r_1$] from R'_ℓ.

Lemma 8

There exists a polynomial adversary $\mathcal{A}_1$ with non-negligible

advantage if there exists a polynomial time adversary $\mathcal{A}_a$ with non-negligible advantage.

(Proof) We suppose that there exits a polynomial time algorithm $\mathcal{A}_a$. Then for given input

$$\begin{pmatrix} \alpha P, \alpha^2 P, \ldots, \alpha^t P \\ \{r_1 \alpha^{t+1-j} P | j \in S, j \neq i_1\} \end{pmatrix}, \{s\alpha^j P | j \notin S\}, R'_\ell,$$

$\mathcal{A}_1$ chooses $r_2, r_3, \ldots, r_n$ randomly and computes other vectors $\{r_2\alpha^{t+1-j} | j \in \mathcal{S}, j \neq i_2\}, \ldots, \{r_n\alpha^{t+1-j} | j \in \mathcal{S}, j \neq i_n\}$. $\mathcal{A}_1$ can compute Exp[$e(P, P)$, $s\alpha^{t+1}r_2$], …, Exp[$e(P, P)$, $s\alpha^{t+1}r_n$] trivially. Therefore $\mathcal{A}_1$ can set $R_\ell = R'_\ell \cdot$Exp[$e(P, P)$, $s\alpha^{t+1}r_2$] ⋯ Exp[$e(P, P)$, $s\alpha^{t+1}r_n$]. And input

$$\begin{pmatrix} \alpha P, \alpha^2 P, \ldots, \alpha^t P \\ \{r_1 \alpha^{t+1-j} P | j \in S, j \neq i_1\} \\ \{r_2 \alpha^{t+1-j} P | j \in S, j \neq i_2\} \\ \vdots \\ \{r_n \alpha^{t+1-j} P | j \in S, j \neq i_n\} \end{pmatrix}, \{s\alpha^j P | j \notin S\}, R_\ell$$

to $\mathcal{A}_a$. Because $\mathcal{A}_a$ can distinguish Exp[$e(P, P)$, $s\alpha^{t+1}(r_1 + r_2 + \ldots + r_n)$] from R_ℓ with non-negligible advantage $\mathcal{A}_1$ also can distinguish Exp[$e(P, P)$, $s\alpha^{t+1}r_1$] from R'_ℓ with same advantage. It is the end of proof of the above lemma. □

Now, consider the adversary $\mathcal{A}_2$ which is a polynomial time algorithm with input

$$\begin{pmatrix} \alpha P, \alpha^2 P, \ldots, \alpha^t P \\ \{r\alpha^{t+1-j} P | j \in S, j \neq i\} \\ \{r'\alpha^{t+1-j} P | j \in S, j \neq i\} \end{pmatrix}, \{s\alpha^j P | j \notin S\}, e(P, P)^{s\alpha^{t+1}r'}, \mathcal{R}_\ell,$$

to distinguish Exp[$e(P, P)$, $s\alpha^{t+1}r$] from R_ℓ. Note that for $\mathcal{A}_2$ the Reveal queries are not allowed.

Lemma 9

There exists a polynomial adversary $\mathcal{A}_2$ with non-negligible advantage if there exists a polynomial time adversary $\mathcal{A}_1$ with non-negligible advantage.

(Proof) To prove this lemma, it is sufficient to show the $\mathcal{A}_2$ can simulate Reveal queries from $\mathcal{A}_1$. Note that the reply of Reveal query can be constructed using public keys, a secret key which can be obtained from KeyGen query and randomly chosen exponent. □

From the result of section IV-2 and above lemmas the Theorem6 is proved under the t-MBDHE and BDH assumptions. Note that the $\mathcal{A}_2$ is exactly same to $\mathcal{A}'_1$ in section IV-2.

To prove the Theorem7, we define a proper form of the message for the index set $\mathcal{S}$ as $\{ r\alpha^i P \mid i \in \mathcal{S} \}$ first. It means that the message vector is obtained by multiplying a random exponent to the vector consists public keys with the index $i \in \mathcal{S}$, i.e. $\{\alpha^i P \mid i \in \mathcal{S}\}$. Note that if the message sent from each participant has proper form, each participant obtains same secret session key. And messages are forced to satisfy the proper form, because that if one of messages has non-proper form the key agreement protocol halts. Therefore, to disturb participants from sharing the session key, although the adversary is a malicious participant, is impossible.

Trivially, if the adversary intercepts a message and discards it then no key agreement protocol achieves key confirmation security. So we do not consider such an adversary here.

The Theorem6 and the Theorem7 imply that the OAK is a explicit key agreement protocol.

5. Other Security Requirements

Security analyses about Known Session Key Secrecy, No Key-Compromise Impersonation Secrecy and No Key Control Secrecy are omitted here. Those are presented in the appendix.

V. Conclusion

A key agreement protocol is a cryptographical primitive which allows participants to share a common secret key via insecure channel. One interesting problem which is still unsolved in key agreement is to construct an one-round multiparty key agreement protocol satisfying various security requirements. In this paper, we present a new multiparty key agreement protocol using bilinear map and adopting the key generation center. The protocol demands only one–round for arbitrary number of participants to share a group key and satisfies both authentication and (partial) forward secrecy.

References

[1] W. Diffie and M. Hellman, "New Directions in Cryptography", *IEEE Transaction on Information Theory, IT-22*(6), 1976, pp. 644-654.

[2] T. Matsumoto, Y. Takashima and H. Imai, "On Seeking Smart Public-key Distribution Systems", *Transactions of the IECE of Japan, E69*, 1986, pp. 99-106

[3] 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.

[4] M. Bellare and P. Rogaway, "Entity Authentication and Key Distribution}, *LNCS 773, Proc. CRYPTO* '93, 1993, pp. 232-249.

[5] M. Bellare and P. Rogaway, "Random Oracles are

Practical : A Paradigm for Designing Efficient Protocols", *Proc. ACM CCS* '93, 1993, pp. 62-73.

[6] E. Bresson, O. Chevassut and D. Pointcheval, "Dynamic Group Diffie-Hellman Key Exchange under Standard Assumptions", *LNCS 2332, Proc. Eurocrypt* '02, 2002, pp. 321-336.

[7] I. R. Jeong, J. Kats and D. H. Lee, "One-round protocols for Two-Party Authenticated Key Exchange", LNCS 3089, *Proc. ACNS* '04, 2004, pp. 220-232.

[8] N. P. Smart, "An Identity-based Authenticated Key Agreement Protocol Based on the Weil Pairing", *Electronic Letters*, 38, 2002, pp. 630-632.

[9] M.. Scott, "Authenticated ID-based Key Exchange and Remote Log-in with Insecure Token and PIN Number", Cryptology ePrint Archive, http://eprint.iacr.org/2002/164, 2002.

[10] I. Ingemarsson, D. T. Tang and C. K. Wong, "A Conference Key Distribution System", *IEEE Transaction on Information Theory* 28(5), 1982, pp. 714-720.

[11] M. Burmester and Y. Desmedt, "A Secure and Efficient Conference Key Distribution System", *LNCS 950, Proc. Eurocrypt* '94, 1995, pp.275-286.

[12] M. Burmester and Y. Desmedt, " A Secure and Scalable Group Key Exchange System", *In Information Processing Letters,* 94(3), 2005, pp. 137-143.

[13] E. Bresson, O. Chevassut and D. Pointcheval, "Provably Authenticated Group Diffie-Hellman Key Exchange – The Dynamic Case", *LNCS 2248, Proc. Asiacrypt* '01, 2001, pp. 290-309.

[14] E. Bresson and D. Catalano, "Constant Round Authenticated Group Key Agreement via Distributed Computing", *LNCS 2947, Proc. PKC* '04, 2004, pp. 115-129.

[15] M. Abdalla, E. Bresson, O. Chevassut and D. Pointcheval, "Password-based Group Key Exchange in a Constant Number of Rounds", *LNCS 3958, Proc. PKC* '06, 2006, pp. 427-442

[16] C. Boyd, J. Manuel and G. Nieto, "Round-optimal Contributory Conference Key Agreement", *LNCS 2567, Proc. PKC* '03, 2003, pp.161-174.

[17] G. Frey, M. Müller and H. Rück, " The Tate Pairing and the Discrete Logarithm Applied to Elliptic Curve Cryptosystems", *IEEE Transaction on Information Theory Vol.* 45, 1999, pp. 1717-1718.

[18] S. Lang, *Elliptic Functions*, Addison-Wesley, Reading, 1973..

[19] J. Silverman, *The arithmetic of elliptic curve*, Springer-Verlag, 1986.

[20] A. Joux, "One Round Protocol for Tripartite Diffie-Hellman", *LNCS 1838, Proc. ANTS* 4, 2000, pp. 385-394.

[21] R. Dutta and R. Barua, "Constant Round Dynamic Group Key Agreement", *LNCS 3650, Proc. ISC* '05, 2005, pp. 74-88.

[22] R. Dutta and R. Barua. "Overview of Key Agreement Protocols", Cryptology ePrint Archive, http://eprint.iacr.org/2005 /289, 2005.

[23] D. Boneh, X. Boyen and E. Goh, " Hierarchical Identity Based Encryption with Constant Size Ciphertext", *LNCS 3494, Proc. Eurocrypt* '05, 2005, pp. 440-456.

[24] D. Boneh, C. Gentry and B. Waters, "Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys, *LNCS 3621, Proc. CRYPTO* '05, 2005, pp. 258-275.

[25] D. Boneh and M. Franklin, "Identity-Based Encryption from the Weil Pairing", *LNCS 2139, Proc. CRYPTO* '01, 2001, pp. 213-229.

[26] P. S. L. M. Barreto, H. Y. Kim and M. Scott, "Efficient Algorithms for Pairing-Based Cryptosystems", *LNCS 2442, Proc. CRYPTO* '02, 2002, pp. 354-368.

Appendix

1. Known Session Key Secrecy

This security requirement is guaranteed by proof of the Theorem6. Note that in proving the Theorem6, we assume that the adversary $\mathcal{A}_a$ can query Reveal freely. This means that there is no adversary who can distinguish the given session key from a random key although the adversary knows many other session keys.

2. No Key-Compromise Impersonation Secrecy

Remind that no key-compromise impersonation security means the adversary who knows a long-term private key of a participant u cannot impersonate u', for $u \neq u'$. Basically, the security against impersonation attack is based on the security of the signature scheme used. Here we claim that even in the case of the adversary can forge the signature of other users, the adversary cannot success the impersonation attack.

Because that if an adversary cannot success impersonation attack to one user then trivially the adversary cannot success impersonation attack to a set of users, we will consider an impersonation attack to one user. Consider the adversary who acts as followings: $\mathcal{A}_{\text{im}}$ chooses a target user u and executes Initiate($\mathcal{S}_\ell$) query on the index set $\mathcal{S}_\ell$ which contains u. The purpose of $\mathcal{A}_{\text{im}}$ is sharing the same key with u by sending modified messages to u.

Let $u = u_1$ and $\mathcal{S}_\ell = \{ u_1, u_2, \ldots, u_n \}$. And suppose that the message to u from $\mathcal{A}_{\text{im}}$ is

$$\begin{pmatrix} \{m_{2,j} | j \in S, j \neq i_2\} \\ \{m_{3,j} | j \in S, j \neq i_3\} \\ \vdots \\ \{m_{n,j} | j \in S, j \neq i_n\} \end{pmatrix}.$$

Assume that $\mathcal{A}_{\text{im}}$ successfully forges all signatures of these messages. After u receives these messages u chooses r and sends $\{ r \cdot \text{Exp}[\alpha, t+1-i_2] \cdot P, \ldots, r \cdot \text{Exp}[\alpha, t+1-i_n] \cdot P \}$ to $\mathcal{A}_{\text{im}}$. And u computes the session key $\text{Exp}[e(P, P), rs\alpha^{t+1}] \cdot \Pi_{j=2, \ldots, n} \text{Exp}[e(P, m_{j,1}), s\alpha^i]$ Because $m_{j,1}$ are all chosen by $\mathcal{A}_{\text{im}}$, we can assume that $\mathcal{A}_{\text{im}}$ can compute $\text{Exp}[e(P, m_{j,1}), s\alpha^i]$ easily. At last the problem is reduced to whether $\mathcal{A}_{\text{im}}$ can distinguish $\text{Exp}[e(P, P), rs\alpha^{t+1}]$ from a random with the inputs $\{ r \cdot \text{Exp}[\alpha, t+1-i_2] \cdot P, \ldots, r \cdot \text{Exp}[\alpha, t+1-i_n] \cdot P \}$ and the replies of some oracle queries.

Note that $\mathcal{A}_{\text{im}}$ is same to $\mathcal{A}_2$ in the section IV-4. From the result of the section \ref{groundwork} we can deduce the $\mathcal{A}_{\text{im}}$ cannot distinguish this with non-negligible advantage.

3. No Key Control Secrecy

From the security of signature scheme, if the adversary is not in the $\mathcal{S}_\ell$, he cannot make a change of the session key. So we assume that the adversary is a malicious user in the set $\mathcal{S}_\ell$ who want to make the session key a certain value R.

Because the adversary $\mathcal{A}_c$ cannot change the random exponent which is chosen by other users, only possible scenario is following: $\mathcal{A}_c$ initiates key agreement protocol among the set $\mathcal{S}_\ell$ which is selected by $\mathcal{A}_c$, using Initiate($\mathcal{S}_\ell$)$ query. After receiving all messages from other users, let $u_2, \ldots, u_n$, $\mathcal{A}_c$ chooses random r satisfying

$\text{Exp}[e(P, P), rs\alpha^{t+1}] = R / \Pi_{j=2, \ldots, n} \text{Exp}[e(P, P), r_j s\alpha^{t+1}]$.

So, we can summarize this as a problem that for given R, R' whether the $\mathcal{A}_c$ can choose r satisfying $R^r = R'$ or not. As you know this is an instance of the DLP problem. So it is not possible that $\mathcal{A}_c$ can control the session key for a certain fixed value R with non-negligible advantage.