Extending Joux's Protocol to Multi Party Key Agreement

Rana Barua, Ratna Dutta and Palash Sarkar
Cryptology Research Group
Stat-Math and Applied Statistics Unit
203, B.T. Road, Kolkata
India 700108
e-mail:{rana,ratna_r,palash}@isical.ac.in

Abstract

We present a secure unauthenticated as well as an authenticated multi party key agreement protocol. The unauthenticated version of our protocol uses ternary trees and is based on bilinear maps and Joux's three party protocol. The number of rounds, computation/communication complexity of our protocol compares favourably with previously known protocols. The authenticated version of our protocol also uses ternary trees and is based on public IDs and Key Generation Centres. The authenticated version of our protocol is more efficient than all previously known authenticated key agreement protocols.
Keywords : group key agreement, authenticated key agreement, pairing based cryptography, ID based cryptography.

1 Introduction

Key agreement is one of the fundamental cryptographic primitives. This is required in situations where two or more parties want to communicate securely among themselves. The situation where three or more parties share a secret key is often called *conference keying*. In this situation, the parties can securely send and receive messages from each other. An adversary not having access to the secret key will not be able to decrypt the message.

Key agreement protocols fall naturally into two classes – authenticated and unauthenticated. The first two party key agreement protocol was introduced by Diffie-Hellman in their seminal paper [14]. This is an unauthenticated protocol in the sense that an adversary who has control over the channel can use the man-in-the-middle attack to agree upon two separate keys with the two users without the users being aware of this. This situation is usually tackled by adding some form of authentication mechanism to the protocol.

Unauthenticated key agreement protocols consider the adversary to be passive, i.e., the adversary can listen to the traffic on the network, but cannot alter it. On the other hand, authenticated key agreement protocols consider the adversary to be active, i.e., the adversary can alter/replace the messages flowing through the network. Thus the security requirements for authenticated key agreement is more stringent. Some of the desirable properties in authenticated key agreement are mutual implicit key authentication, known key security, forward secrecy, key compromise impersonation and key control.

Apart from authentication, the other aspects of key agreement protocols are computation and communication efficiency. When the number of parties is more than two, the protocol is usually a multi-round protocol. In each round some or all of the users send messages to the other users and at the end of all the rounds, all the users should agree upon a common key. The total number of bits exchanged in the protocol is a crucial parameter in judging the efficiency of the protocol. Further, in each round, each user has to perform some computation like an exponentiation or a scalar multiplication. The total amount of computation required by all the users is another measure of efficiency of the protocol.

In this paper, we present a secure multi-party key agreement protocol. The protocol can be used for both authenticated and unauthenticated key agreement. For n parties, $n > 2$, the number of rounds required for our protocol is $\lceil \log_3 n \rceil$ for both authenticated and unauthenticated key agreement. For unauthenticated key agreement, the total number of messages exchanged and the total number of scalar multiplications (over a suitable elliptic curve) is less than $\frac{5}{2}(n-1)$. Additionally, $n\lceil \log_3 n \rceil$ pairings have to be computed. For authenticated version of our protocol, the combined message size is two times more than the unauthenticated version, the number of scalar multiplications is at most $9(n-1)$ and pairings is around five times more than the unauthenticated version.

Our protocol is based on bilinear maps. In one of the breakthroughs in key agreement protocols, Joux [17] proposed a three party, single round key agreement protocol. An authenticated version of this protocol has been proposed in [2], [28]. The basic Joux protocol is at the heart of our (unauthenticated) protocol. We group the users into three user sets using top down recursive procedure so that user sets with two users appear only at the first round. The Joux protocol is invoked for each set of three users to agree upon a common key. For the set of two users (if any), one of them generates randomly another short term private key to have two keys. These two keys and the short term private key of the other user is used in another invocation of Joux's protocol. A set of one user is kept unchanged. Our protocol then proceeds over several rounds and each round consists of several invocations of Joux's protocol. Ultimately, we are left with three user sets and Joux's protocol is invoked once more to complete the protocol. The authenticated version of our protocol is structurally same as the unauthenticated version. The only difference is that the invocations of the two and three party protocol now involves the authenticated version of these protocols.

Thus our multi party protocol is essentially a combination of Joux's tripartite Diffie-Hellman protocol and tree based group key agreement using ternary tree structure. Even though the idea of combining Joux's protocol and tree based group key agreement is a natural extension of Joux's original protocol, it is nontrivial to obtain a security proof for the protocol against passive adversary. One of our major contribution is to provide such a proof using techniques from [5], [18], [27]. In fact, any secure two or three party protocol can be used with the ternary tree structure to obtain a secure multi party protocol. Our security analysis against passive adversaries is also a kind of reduction. We argue that if the underlying two and three party protocols are secure, then our multi party protocol is also secure.

The authenticated version of our protocol is an ID-based protocol. ID-based cryptography was proposed by Shamir [25] and there has been a spurt of papers in this area. The authenticated version of our protocol can be seen as another contribution to this field. The unauthenticated version of our protocol uses pairing based cryptography, which was introduced by Boneh and Franklin [6].

Previous Work : As mentioned before, the first two party unauthenticated key agreement protocol was proposed by Diffie-Hellman in their seminal paper [14]. This was modified into an authenticated key agreement protocol by Matsumoto, Takashima and Imai in [22]. Later, Law, Menezes, Qu, Solinas and Vanstone showed in [21] that some of the protocols of [22] are not secure and proposed a new protocol for authenticated key agreement. There have been a number of proposals for authenticated and unauthenticated key agreement [23], [26], [11].

Among the previously known multi party key agreement protocols only two protocols have number of rounds less than our protocol. In [8], Boneh and Silverberg proposed a single round multi party key agreement protocol. This protocol is based on the existence of multi-linear maps. Currently, no such suitable maps are known and the existence of such maps is presently a research problem [8]. The other protocol which requires less number of rounds is due to Burmester and Desmedt [9]. This is an unauthenticated protocol and requires two rounds. However, the computation complexity is higher and the total number of exponentiations required is around n^2. Also the authors indicate that zero-knowledge proof techniques are required to convert this protocol into an authenticated protocol.

The total number of messages exchanged in our protocol is less than all other known protocols. Further, the total number of scalar multiplications/exponentiation required by our protocol is also less than other

known protocols. However, our protocol requires a number of pairings and hence in certain cases can be computationally less efficient than some of the previously known protocols.

The remainder of the paper is organized as follows. Section 2 briefly explains the cryptographic bilinear map and the basic requirements of our protocol. Section 3 describes the protocol. The security analysis is provided in Section 4. Section 5 discusses the efficiency. Section 6 compares with other key agreement protocols. Finally Section 7 concludes the paper. Membership operations insertion and deletion are discussed in the Appendix.

2 Preliminaries

2.1 Cryptographic Bilinear Maps

Let G_1, G_2 be two groups of the same prime order q. We view G_1 as an additive group and G_2 as a multiplicative group. Let P be an arbitrary generator of G_1. Assume that discrete logarithm problem (DLP) is hard in both G_1 and G_2. A mapping $e : G_1^2 \rightarrow G_2$ satisfying the following properties is called a bilinear map from a cryptographic point of view :
Bilinearity : $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in G_1$ and $a, b \in Z_q^*$.
Non-degeneracy : If P is a generator of G_1, then $e(P, P)$ is a generator of G_2. In other words, $e(P, P) \neq 1$.
Computable : There exists an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in G_1$.
Modified Weil Pairing [6] and Tate Pairing [4], [16] are the examples of cryptographic bilinear maps.

2.2 Diffie Hellman Assumptions

In this subsection we specify some versions of Diffie-Hellman problems of which the last one is newly introduced. Consider $\langle G_1, G_2, e\rangle$ where G_1, G_2 are two cyclic subgroups of a large prime order q and $e : G_1^2 \rightarrow G_2$ is a cryptographic bilinear map. We take G_1 as an additive group and G_2 as a multiplicative group. (*By $a \in_R Z_q^*$, we mean a is randomly chosen from Z_q^*.*)

1. **Decisional Diffie-Hellman (DDH) problem in G_1 :**
Instance : (P, aP, bP, cP) for some $a, b, c \in Z_q^*$.
Solution : Output *yes* if $c = ab \bmod q$ and output *no* otherwise.
DDH problem in G_1 is easy : DDH problem in G_1 can be solved in polynomial time by verifying $e(aP, bP) = e(P, cP)$. This is the well known MOV reduction [6] : The DLP in G_1 is no harder than the DLP in G_2.
DDH assumption : There exists no polynomial time algorithm which can solve the DDH problem in G_2 with non-negligible probability of success. See [13] for a detailed discussion.

2. **Hash Decisional Diffie-Hellman (HDH) problem in G_1 :**
Instance : (P, aP, bP, r) for some $a, b, c, r \in Z_q^*$ and a one way hash function $H : G_1 \rightarrow Z_q^*$.
Solution : Output *yes* if $r = H(abP) \bmod q$ and output *no* otherwise.
This problem was introduced by Abdalla, Bellare and Rogaway in [1]. The only difference is that the co-domain set for the hash function used by them is a set of finite (fixed) length strings whereas in our version, we take it to be the set Z_q^*.
The advantage of any probabilistic , polynomial time, 0/1-valued algorithm $\mathcal{A}$ in solving HDH problem in G_1 is defined to be :
$Adv_{\mathcal{A}}^{HDH} = |Prob[\mathcal{A}(P, aP, bP, r) = 1 : a, b, c, r \in_R Z_q^*] - Prob[\mathcal{A}(P, aP, bP, H(abP)) = 1 : a, b \in_R Z_q^*]|$
HDH assumption : There exists no polynomial time algorithm which can solve the HDH problem in G_1 with non-negligible probability of success. In otherwords, for every probabilistic, polynomial time, 0/1-valued algorithm $\mathcal{A}$, $Adv_{\mathcal{A}}^{HDH} < \frac{1}{m^l}$ for every fixed $l > 0$ and sufficiently large m. For more details,

see [1].

3. **Bilinear Diffie-Hellman (BDH) problem in $\langle G_1, G_2, e\rangle$:**
Instance : (P, aP, bP, cP) for some $a, b, c \in Z_q^*$
Solution : Output $e(P,P)^{abc}$.
BDH assumption : There exists no polynomial time algorithm which can solve the BDH problem in $\langle G_1, G_2, e\rangle$ with non-negligible probability of success. See [6] for more details.

4. **Decisional Hash Bilinear Diffie-Hellman (DHBDH) problem in $\langle G_1, G_2, e\rangle$:**
Instance : $(P, aP, bP, cP, r))$ for some $a, b, c, r \in Z_q^*$ and a one way hash function $H : G_2 \rightarrow Z_q^*$.
Solution : Output *yes* if $r = H(e(P,P)^{abc})$ mod q and output *no* otherwise.
Similar to the HDH problem already introduced in [1] which is a hash version of DDH problem, the DHBDH problem in $\langle G_1, G_2, e\rangle$ is also a hash version of the decisional BDH problem in $\langle G_1, G_2, e\rangle$.
The advantage of any probabilistic, polynomial time, 0/1-valued algorithm $\mathcal{A}$ in solving DHBDH problem in $\langle G_1, G_2, e\rangle$ is defined to be :
$Adv_{\mathcal{A}}^{DHBDH} = |Prob[\mathcal{A}(P, aP, bP, cP, r) = 1 : a, b, c, r \in_R Z_q^*] - Prob[\mathcal{A}(P, aP, bP, cP, H(e(P,P)^{abc})) = 1 : a, b, c \in_R Z_q^*]|$
DHBDH assumption : There exists no polynomial time algorithm which can solve the DHBDH problem with non-negligible probability of success. In otherwords, for every probabilistic, polynomial time, 0/1-valued algorithm $\mathcal{A}$, $Adv_{\mathcal{A}}^{DHBDH} < \frac{1}{m^l}$ for every fixed $l > 0$ and sufficiently large m.

The DHBDH assumption is a "natural" combination of the HDH and BDH assumption and hence appears to be a reasonable assumption to make.

2.3 Protocol Requirements

Consider the n users who wish to agree upon a conference key to be the set $\{1, 2, \ldots, n\}$. Let $s_1, s_2, \ldots, s_n \in Z_q^*$, be their respective private keys. Let U be a subset of $\{1, 2, \ldots, n\}$ consisting of consecutive integers. We call U a user set. Let $Rep(U)$ stand for the representative of the set U. To be specific, let $Rep(U) = \min(U)$. We use the notation $A[1, \ldots, n]$ for an array of n elements $A_1, \ldots, A_n$ and write $A[i]$ and A_i interchangeably. We take G_1 to be a cyclic subgroup of an elliptic curve group of some large prime order q and the bilinear map $e : G_1^2 \rightarrow G_2$ to be either a modified Weil pairing or a Tate pairing [4], [16]. Let P be an arbitrary generator of G_1. Choose a hash function $H : G_2 \rightarrow Z_q^*$. The system parameters for the unauthenticated protocol are $params = \langle G_1, G_2, e, q, P, H\rangle$.

For ID-based authenticated key agreement, we will additionally require the followings:

1. Hash functions $\widehat{H} : G_1 \rightarrow Z_q^*$, $H_1 : \{0,1\}^* \rightarrow G_1$.
2. A key generation centre (KGC) which chooses a random $s \in Z_q^*$ and sets $P_{pub} = sP$. It publishes P_{pub} as a system parameter and keeps s as secret which it treates as the master key. Each user i has an identity $ID_i \in \{0,1\}^*$ and long term public key $Q_i = H_1(ID_i)$. User i sends Q_i to KGC and KGC sends back the long term private key $S_i = sQ_i$ to user i.
3. The keys $s_1, s_2, \ldots, s_n$ are short term private keys.

The system parameters for the authenticated protocol are $params = \langle G_1, G_2, e, q, P, P_{pub}, H, \widehat{H}, H_1\rangle$.

3 Protocol

In this section, we present a three-group and a two-group Diffie-Hellman key agreement protocol CombineThree and CombineTwo respectively together with an n-party recursive algorithm KeyAgreement which makes use of CombineThree and CombineTwo. *The boxed portions are executed for the authenticated version.*

The three-group Diffie-Hellman key agreement for unauthenticated as well as ID-based authenticated versions are jointly given by the subroutine CombineThree as described below. In this subroutine, when cardinality of each of these three groups is one, then the ID-based authenticated version is simply the three-party protocol proposed by Zhang, Liu, Kim in [28] while the unauthenticated version is the Joux [17] three-party key agreement protocol using bilinear map.

procedure CombineThree($U[1,2,3], s[1,2,3]$)
$i = 1$ to 3 **do**
$Rep(U_i)$ computes $P_i = s_i P$
$\boxed{\text{and } T_{Rep(U_i)} = \widehat{H}(P_i) S_{Rep(U_i)} + s_i P_i;}$
Let $\{j,k\} = \{1,2,3\} \backslash \{i\}$;
$Rep(U_i)$ sends P_i, $\boxed{T_{Rep(U_i)}}$ to all members of both U_j, U_k;
end do

$i = 1$ to 3 **do**
Let $\{j,k\} = \{1,2,3\} \backslash \{i\}$;
each member of U_i
$\boxed{\text{verifies : } e(T_{Rep(U_j)} + T_{Rep(U_k)}, P) = e(\widehat{H}(P_j) Q_{Rep(U_j)} + \widehat{H}(P_k) Q_{Rep(U_k)}, P_{pub}) e(P_j, P_j) e(P_k, P_k) \text{ and}}$
computes $H(e(P_j, P_k)^{s_i})$;
end do

end CombineThree

This subroutine does a key agreement among three user sets U_1, U_2, U_3 with s_1, s_2, s_3 respectively as their private keys (short term for ID-based) with common key $H(e(P,P)^{s_1 s_2 s_3})$.

Similarly, the two-group Diffie-Hellman key agreement for unauthenticated as well as ID-based authenticated versions are jointly given by the subroutine CombineTwo as described below. This subroutine reduces to a two-party Diffie-Hellman key agreement protocol when each of the two groups has cardinality one.

procedure CombineTwo($U[1,2], s[1,2]$)
$i = 1$ to 2 **do**
$Rep(U_i)$ computes $P_i = s_i P$
$\boxed{\text{and } T_{Rep(U_i)} = \widehat{H}(P_i) S_{Rep(U_i)} + s_i P_i;}$
end do
$Rep(U_1)$ generates $\overline{s} \in_R Z_q^*$ and sends $\overline{s}P$
$\boxed{\text{and } \overline{T}_{Rep(U_1)} = \widehat{H}(\overline{s}P) S_{Rep(U_1)} + \overline{s}^2 P}$ to the rest of the users;
$\boxed{\begin{array}{l}\text{each member of } U_1, U_2 \text{ except } Rep(U_1) \text{ verifies :} \\ e(\overline{T}_{Rep(U_1)}, P) = e(\widehat{H}(\overline{s}P) Q_{Rep(U_1)}, P_{pub}) e(\overline{s}P, \overline{s}P);\end{array}}$
$Rep(U_1)$ sends P_1, $\boxed{T_{Rep(U_1)}}$ to all members of U_2;

$Rep(U_2)$ sends P_2, $\boxed{T_{Rep(U_2)}}$ to all members of U_1;
each member of U_1
$\boxed{\text{verifies : } e(T_{Rep(U_2)}, P) = e(\widehat{H}(P_2)Q_{Rep(U_2)}, P_{pub})e(P_2, P_2) \text{ and}}$
computes $H(e(P_2, \overline{s}P)^{s_1})$;
each member of U_2
$\boxed{\text{verifies : } e(T_{Rep(U_1)}, P) = e(\widehat{H}(P_1)Q_{Rep(U_1)}, P_{pub})e(P_1, P_1) \text{ and}}$
computes $H(e(P_1, \overline{s}P)^{s_2})$;

end CombineTwo

This subroutine does a key agreement among two user sets U_1, U_2 with s_1, s_2 respectively as their private keys (short term for ID-based) with common key $H(e(P, P)^{s_1 s_2 \overline{s}})$ where $\overline{s}$ is generated randomly by the representative of the user set U_1. Thus, this subroutine is essentially the Joux's protocol invoked for two user sets.

Next we describe the tree structure KeyAgreement as a top down recursive procedure which uses the above two subroutines CombineTwo and CombineThree.

procedure KeyAgreement$(m, U[i+1, \ldots, i+m])$
  if $(m = 1)$ **then**
    $KEY = s[i+1]$;
  end if
  if $(m = 2)$ **then**
    call CombineTwo$(U[i+1, i+2], s[i+1, i+2])$;
    Let KEY be the agreed key between user sets U_{i+1}, U_{i+2};
  end if
  $n_0 = 0$; $n_1 = \lfloor \frac{m}{3} \rfloor$; $n_3 = \lceil \frac{m}{3} \rceil$; $n_2 = m - n_1 - n_3$;
  $j = 1$ to 3 **do**
    call KeyAgreement$(n_j, U[i + n_{j-1} + 1, \ldots, i + n_{j-1} + n_j])$;
    $\widehat{U}_j = U[i + n_{j-1} + 1, \ldots, i + n_{j-1} + n_j]$; $\widehat{s}_j = KEY$; $n_j = n_{j-1} + n_j$;
  end do;
  call CombineThree$(\widehat{U}[1, 2, 3], \widehat{s}[1, 2, 3])$;
  Let KEY be the agreed key among user sets $\widehat{U}_1, \widehat{U}_2, \widehat{U}_3$;
end KeyAgreement

The start of the recursive protocol KeyAgreement is made by the following two statements:

1. $U_j = j$ for $1 \leq j \leq n$;
2. **call** KeyAgreement$(n, U[1, \ldots, n])$;

The algorithm is recursive and goes through several levels starting with level 0. In each level, there are sets of users who have (or agree upon) a common secret key. In level 0, each user is in a set by himself/ herself and the secret key of the singleton set is the secret key of the concerned user. For n users, let the levels be numbered $0, \ldots, R(n)$. In level i, let the number of user sets be n_i. Thus $n_0 = n$, and $n_k = 1$, where $k = R(n)$. We identify the rounds of the algorithm as follows. There are $R(n)$ rounds of computation. The i^{th} round of computation takes the state of the algorithm from level $i-1$ to level i for $i = 1, \ldots, R(n)$. We introduce some notations for convenience of analysing the algorithm.
–$U_j^{(i)}$: the j-th user set at level $i, 0 \leq i \leq R(n), 1 \leq j \leq n_i$,

$-s_j^{(i)}$: common secret key agreed upon by users in the user set $U_j^{(i)}$,
$-P_j^{(i)}$: i-th level j-th public key, *i.e.* $P_j^{(i)} = s_j^{(i)} P$.
Let $p = \lfloor \frac{n}{3} \rfloor$ and $r = n \bmod 3$. We partition the set of users $U_1^{(k)} = \{1, \ldots, n\}$ into three user sets $U_1^{(k-1)}, U_2^{(k-1)}, U_3^{(k-1)}$ with cardinality p, p, p respectively if $r = 0$, with cardinality $p, p, p+1$ respectively if $r = 1$ or with cardinality $p, p+1, p+1$ respectively if $r = 2$. We use this top down recursive procedure for each user set $U_j^{(i)}$ to split it into three user sets for $1 < i < k, 1 \leq j \leq n_i$ and for $i = 1$, each such user set is partioned into either one user set, or two user sets or three user sets depending on n. Note that with this tree structure, an user set with two users appears only at level 1 and so CombineTwo is never invoked for round ≥ 2. For $n = 10$, the working of the algorithm (unauthenticated version) is shown in figure 1.

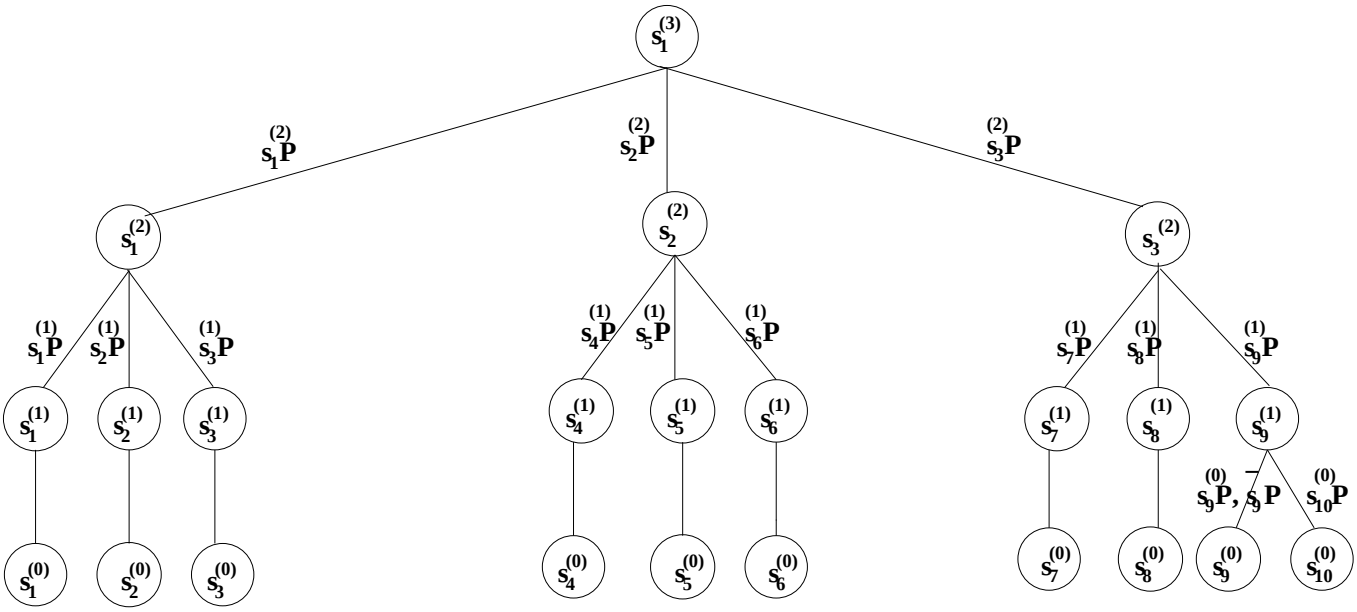

Figure 1: Key agreement among $n = 10$ parties, user 9 generates $\overline{s}_9$ randomly and sends $\overline{s}_9 P$ to user 10 at the first level.

Lemma 3.1 *The final agreed key KEY among n users in the subroutine* KeyAgreement *is*

$$KEY = s_1^{(k)} = H(e(P,P)^{s_1^{(k-1)} s_2^{(k-1)} s_3^{(k-1)}})$$

where $k = R(n)$, $n > 2$.

Lemma 3.2 *Each member of $U_j^{(i)}$ can compute $s_j^{(i)}$ for $1 \leq j \leq n_i$, $i \leq k$ where $k = R(n)$. Consequently, all users are able to compute the common key $KEY = s_1^{(k)}$.*

Proof : We prove this lemma by induction on i.
Base Step : Initially $U_j^{(0)} = j$ and private key $s_j^{(0)}$ are assigned to each user for $1 \leq j \leq n_0$. So each member of $U_j^{(1)}, 1 \leq j \leq n_1$ can compute $s_j^{(1)}$.
Induction Step: Let for $2 \leq i \leq k, 1 \leq j \leq n_{i-1}$, each member of user set $U_j^{(i-1)}$ has computed $s_j^{(i-1)}$ at the $(i-1)^{th}$ round. Note that KeyAgreement never calls CombineTwo for round $i \geq 2$. Now at the i-th round, for $j = 1, \ldots, n_i$, KeyAgreement calls CombineThree in which user set $U_j^{(i)}$ computes $s_j^{(i)}$ using the corresponding s of the $(i-1)^{th}$ round and user set $U_j^{(i)}$ consists of all members of $U_{3j-2}^{(i-1)}, U_{3j-1}^{(i-1)}$ and $U_{3j}^{(i-1)}$

all of who know the corresponding s by induction hypothesis. So each member of $U_j^{(i)}$ can compute the required $s_j^{(i)}$. Hence the proof follows. ■

One question arises here: How does user u ($1 \leq u \leq n$), in a certain round $i \geq 1$ determine to which user set he belongs and whether he is the representative of that user set? Using the above recursions, user u can easily compute all the user sets $U_j^{(i)}, 1 \leq j \leq n_i$, of this round and can efficiently check its position. Note that each $U_j^{(i)}$ is a subset of $\{1, 2, \ldots, n\}$ consisting of consecutive integers. So if $u \in U_j^{(i)}$ for some j, $1 \leq j \leq n_i$, user u verifies whether u equals $\min(U_j^{(i)})$ and if so, u is the representative of the user set $U_j^{(i)}$.

4 Security Analysis

4.1 Security Against Passive Adversary

A secure key agreement protocol should withstand both passive and active attacks. In this subsection we define the **Decisional ternary tree group key agreement (DTGKA) problem** for our unauthenticated protocol and show that this problem is hard for the passive adversary by reducing the hardness of this problem to the hardness of DHBDH problem following the technique used in [5], [18], [27]. The authentication is introduced in our unauthenticated protocol using a special signature scheme to get security against an active adversary which has been discussed in the next subsection.

Given a ternary tree T of height at most k with n leaf nodes ($n > 2$) and $X = (s_1, s_2, \ldots, s_n)$ for $s_i \in Z_q^*$, the public and secret values are collectively defined as follows :
$-vw(k, X, T) := \{P_j^{(i)}$ where j and i are defined according to $T\}$
$-K(k, X, T) := KEY = H(e(P, P)^{s_1^{(k-1)} s_2^{(k-1)} s_3^{(k-1)}})$

$vw(k, X, T)$ is exactly the view of the passive adversary in the ternary tree T where final key is $K(k, X, T)$. We call the key $K(k, X, T)$ a DHBDH key. Our goal is to show that this DHBDH key generated by the unauthenticated protocol can not be distinguished by a polynomial time algorithm from a random number if all the transmitted values during a protocol run are known.
Suppose $\mathcal{T}_k$ is the set of all ternary trees of height k having structure of KeyAgreement. A tree T of height k is chosen randomly from $\mathcal{T}_k$ and let $X \in_R (Z_q^*)^n$ ($n \leq 3^k$) be the labels of (short term private keys associated with) the leaf nodes of T. Then k is the number of rounds with n users. Let us define two random variables $A_k, \widehat{A}_k$ as follows :
$-A_k := (vw(k, X, T), y), y \in_R Z_q^*$
$-\widehat{A}_k := (vw(k, X, T), K(k, X, T))$

Let $\mathcal{S}_k = \{(T, X) : T \in_R \mathcal{T}_k$ and $X \in_R (Z_q^*)^n$, where n is the number of leaf level nodes in $T\}$. Let B_T be the number of edges in the ternary tree T. For $(T, X) \in \mathcal{S}_k$, define $\Gamma(T, X)$ to be the ordered tuple of all public information along the arcs of T. Clearly, $\Gamma(T, X) \subseteq (G_1)^{B_T}$. Then the random variable A_k takes values from the sample space $(G_1)^{B_T} \times Z_q^*$ according to the uniform probability distribution and $\widehat{A}_k$ takes values from the sample space $\Gamma(T, X) \times Z_q^* \subseteq (G_1)^{B_T} \times Z_q^*$ with the uniform probability distribution.

Definition 4.1 *Consider $\langle G_1, G_2, e\rangle$. Let $n > 2$ be a positive integer, $X = (s_1, s_2, \ldots, s_n)$ for $s_i \in Z_q^*$ and T be a ternary tree of height k with n leaf nodes labeled by X, and $A_k, \widehat{A}_k$ are defined as above. A* **Decisional ternary tree group key agreement (DTGKA) algorithm** *$\mathcal{F}$ for $\langle G_1, G_2, e\rangle$ is a*

probabilistic polynomial time algorithm that outputs either 0 *or* 1*, satisfying, for some fixed* $l > 0$ *and sufficiently large* m *:*

$$|Prob[\mathcal{F}(A_k) = 1] - Prob[\mathcal{F}(\widehat{A}_k) = 1]| > \frac{1}{m^l}.$$

We call $\mathcal{F}$ *a polynomial time distinguisher that distinguishes* A_k *and* $\widehat{A}_k$*.*

The **DTGKA problem** is to find a polynomial time distinguisher $\mathcal{F}$ for A_k and $\widehat{A}_k$ defined above.

Theorem 4.2 *If the DHBDH problem in* $\langle G_1, G_2, e\rangle$ *is hard, then* A_k *and* $\widehat{A}_k$ *are polynomially indistinguishable.*

Proof : First let us provide a plan of the proof. The proof is by induction on k.

Base Step : $k = 1$: Distinguishing A_1 and $\widehat{A}_1$ implies "solving" DHBDH problem in $\langle G_1, G_2, e\rangle$. Thus it is not possible to distinguish A_1 and $\widehat{A}_1$ assuming DHBDH problem is hard in $\langle G_1, G_2, e\rangle$.
Induction Hypothesis : Assume that for some $k \geq 2$, it is not possible to distinguish A_{k-1} and $\widehat{A}_{k-1}$.
Induction Step : We show that the ability to distinguish A_k and $\widehat{A}_k$ implies either (a) "solving" DHBDH problem in $\langle G_1, G_2, e\rangle$ or (b) ability to distinguish A_{k-1} and $\widehat{A}_{k-1}$. Since (a) is given to be hard and (b) is hard by induction hypothesis, it follows that it is not possible to distinguish between A_k and $\widehat{A}_k$.

Now we turn to a proof of the induction step. Let $T \in_R \mathcal{T}_k$ be a ternary tree of height k with n leaf nodes. Let $X \in_R (Z_q^*)^n$ be the labels of the leaf nodes of T. Let T_1, T_2, T_3 respectively be the left, middle and right subtree of height at most $k-1$ of the tree T. This implies that at least one of these subtrees has height exactly $k-1$. If $X_1 = (s_1, \ldots, s_l), X_2 = (s_{l+1}, \ldots, s_m)$ and $X_3 = (s_{m+1}, \ldots, s_n)$, where s_1 through s_l are associated with T_1, s_{l+1} through s_m with T_2 and s_{m+1} through s_n with T_3, then A_k and $\widehat{A}_k$ can be rewritten as :

$A_k := (vw(k, X, T), y)$ for random $y \in Z_q^*$
$= (vw(k-1, X_1, T_1), vw(k-1, X_2, T_2), vw(k-1, X_3, T_3), P_1^{(k-1)}, P_2^{(k-1)}, P_3^{(k-1)}, y)$
$= (vw(k-1, X_1, T_1), vw(k-1, X_2, T_2), vw(k-1, X_3, T_3), s_1^{(k-1)}P, s_2^{(k-1)}P, s_3^{(k-1)}P, y)$

$\widehat{A}_k := (vw(k, X, T), K(k, X, T))$
$= (vw(k-1, X_1, T_1), vw(k-1, X_2, T_2), vw(k-1, X_3, T_3), P_1^{(k-1)}, P_2^{(k-1)}, P_3^{(k-1)}, KEY)$
$= (vw(k-1, X_1, T_1), vw(k-1, X_2, T_2), vw(k-1, X_3, T_3), s_1^{(k-1)}P, s_2^{(k-1)}P, s_3^{(k-1)}P, KEY)$

Let us consider the following random variables:
$A_k := (vw(k-1, X_1, T_1), vw(k-1, X_2, T_2), vw(k-1, X_3, T_3), s_1^{(k-1)}P, s_2^{(k-1)}P, s_3^{(k-1)}P, y)$
$B_k := (vw(k-1, X_1, T_1), vw(k-1, X_2, T_2), vw(k-1, X_3, T_3), r_1P, s_2^{(k-1)}P, s_3^{(k-1)}P, y)$
$C_k := (vw(k-1, X_1, T_1), vw(k-1, X_2, T_2), vw(k-1, X_3, T_3), r_1P, r_2P, s_3^{(k-1)}P, y)$
$D_k := (vw(k-1, X_1, T_1), vw(k-1, X_2, T_2), vw(k-1, X_3, T_3), r_1P, r_2P, r_3P, y)$
$\widehat{D}_k := (vw(k-1, X_1, T_1), vw(k-1, X_2, T_2), vw(k-1, X_3, T_3), r_1P, r_2P, r_3P, K_1)$
$\widehat{C}_k := (vw(k-1, X_1, T_1), vw(k-1, X_2, T_2), vw(k-1, X_3, T_3), r_1P, r_2P, s_3^{(k-1)}P, K_2)$
$\widehat{B}_k := (vw(k-1, X_1, T_1), vw(k-1, X_2, T_2), vw(k-1, X_3, T_3), r_1P, s_2^{(k-1)}P, s_3^{(k-1)}P, K_3)$
$\widehat{A}_k := (vw(k-1, X_1, T_1), vw(k-1, X_2, T_2), vw(k-1, X_3, T_3), s_1^{(k-1)}P, s_2^{(k-1)}P, s_3^{(k-1)}P, KEY)$
where $r_1, r_2, r_3 \in_R Z_q^*$ and $K_1 = H(e(P,P)^{r_1 r_2 r_3})$, $K_2 = H(e(P,P)^{r_1 r_2 s_3^{(k-1)}})$ and $K_3 = H(e(P,P)^{r_1 s_2^{(k-1)} s_3^{(k-1)}})$.

Claim : If $A_k, \widehat{A}_k$ are distinguishable in polynomial time, then at least one of the followings can be distinguished : $(A_k, B_k), (B_k, C_k), (C_k, D_k), (D_k, \widehat{D}_k), (\widehat{D}_k, \widehat{C}_k), (\widehat{C}_k, \widehat{B}_k)$ or $(\widehat{B}_k, \widehat{A}_k)$.
Proof of the Claim : Let $a_1 = Prob[\mathcal{F}(A_k) = 1]$, $a_2 = Prob[\mathcal{F}(B_k) = 1]$, $a_3 = Prob[\mathcal{F}(C_k) = 1]$,

$a_4 = Prob[\mathcal{F}(D_k) = 1]$, $a_5 = Prob[\mathcal{F}(\widehat{D}_k) = 1]$, $a_6 = Prob[\mathcal{F}(\widehat{C}_k) = 1]$, $a_7 = Prob[\mathcal{F}(\widehat{B}_k) = 1]$, $a_8 = Prob[\mathcal{F}(\widehat{A}_k) = 1]$. Since A_k and $\widehat{A}_k$ are distinguishable in polynomial time, $|a_1 - a_8| > \frac{1}{m^l}$ for sufficiently large m and for a fixed $l > 0$. Now we will show that at least one of the followings must hold : $|a_i - a_{i+1}| > \frac{1}{m^{l+1}}$ for $i = 1, \ldots, 7$. If not, let $|a_i - a_{i+1}| \leq \frac{1}{m^{l+1}}$ for all $i = 1, \ldots, 7$.
Then $|a_1 - a_8| \leq |a_1 - a_2| + \cdots + |a_7 - a_8| \leq \frac{7}{m^{l+1}} \leq \frac{1}{m^l}$, a contradiction, if $m \geq 7$.

We shall show that the ability to distinguish any one of (A_k, B_k), (B_k, C_k), (C_k, D_k), $(\widehat{D}_k, \widehat{C}_k)$, $(\widehat{C}_k, \widehat{B}_k)$ or $(\widehat{B}_k, \widehat{A}_k)$ reduces to solving DTGKA problem with height $k-1$ and ability of distinguishing $(D_k, \widehat{D}_k)$ reduces to solving the DHBDH problem in $\langle G_1, G_2, e\rangle$. The proof of the two cases : (A_k, B_k) and $(D_k, \widehat{D}_k)$ are discussed here in details. A proof similar to the case (A_k, B_k) follows for others.

Case : Distinguish (A_k, B_k) : Suppose $\mathcal{F}_{AB_k}$ is a polynomial time distinguisher that can distinguish A_k and B_k in polynomial time. We will show that $\mathcal{F}_{AB_k}$ can be used to solve DTGKA problem with height $k-1$. We construct a polynomial time distinguisher $\mathcal{F}_{A\widehat{A}_{k-1}}$ that distinguishes A_{k-1} and $\widehat{A}_{k-1}$ in polynomial time as follows :
Let $V^*_{k-1} = (vw(k-1, X^*, T^*), r^*)$ where T^* is a ternary tree of height $k-1$ with $|X^*| = n$ having structure of KeyAgreement. The distinguisher $\mathcal{F}_{A\widehat{A}_{k-1}}$ first constructs two ternary trees $\widetilde{T}$ and $\overline{T}$ with leaf level secret key distribution $\widetilde{X}$ and $\overline{X}$ respectively in the following manner : if $n = 3^k$, then take $|\widetilde{X}| = |\overline{X}| = n$, else take either $|\widetilde{X}| = |\overline{X}| = n$ or $|\widetilde{X}| = n, |\overline{X}| = n+1$ or $|\widetilde{X}| = |\overline{X}| = n+1$. Then $\mathcal{F}_{A\widehat{A}_{k-1}}$ constructs a tree of height k with $|X^*| + |\widetilde{X}| + |\overline{X}|$ users and $T^*, \widetilde{T}, \overline{T}$ as the left, middle and right subtree respectively. The resulting tree is clearly a random member of $\mathcal{T}_k$. Next $\mathcal{F}_{A\widehat{A}_{k-1}}$ sets :
$V^*_k = (vw(k-1, X^*, T^*), vw(k-1, \widetilde{X}, \widetilde{T}), vw(k-1, \overline{X}, \overline{T}), r^*P, \widetilde{K}P, \overline{K}P, y)$, where $\widetilde{K} = K(k-1, \widetilde{X}, \widetilde{T})$, $\overline{K} = K(k-1, \overline{X}, \overline{T})$, $y \in_R Z^*_q$ and runs $\mathcal{F}_{AB_k}$ on input V^*_k. Now $Prob[\mathcal{F}_{AB_k}(A_k = V^*_k) = 1] = Prob[\mathcal{F}_{A\widehat{A}_{k-1}}(\widehat{A}_{k-1} = V^*_{k-1}) = 1]$ and $Prob[\mathcal{F}_{AB_k}(B_k = V^*_k) = 1] = Prob[\mathcal{F}_{A\widehat{A}_{k-1}}(A_{k-1} = V^*_{k-1}) = 1]$.
Consequently, $|Prob[\mathcal{F}_{A\widehat{A}_{k-1}}(\widehat{A}_{k-1} = V^*_{k-1}) = 1] - Prob[\mathcal{F}_{A\widehat{A}_{k-1}}(A_{k-1} = V^*_{k-1}) = 1]|$
$= |Prob[\mathcal{F}_{AB_k}(A_k = V^*_k) = 1] - Prob[\mathcal{F}_{AB_k}(B_k = V^*_k) = 1]|$. Hence if $\mathcal{F}_{AB_k}$ can distinguish between A_k and B_k, then $\mathcal{F}_{A\widehat{A}_{k-1}}$ can distinguish between A_{k-1} and $\widehat{A}_{k-1}$.

Case : Distinguish $(D_k, \widehat{D}_k)$: Suppose $\mathcal{F}_{D\widehat{D}_k}$ is a polynomial time distinguisher that can distinguish D_k and $\widehat{D}_k$ in polynomial time. We shall show that $\mathcal{F}_{D\widehat{D}_k}$ can be used to construct a polynomial time algorithm $\mathcal{A}$ that solves the DHBDH problem in $\langle G_1, G_2, e\rangle$. Note that r_1P and r_2P are independent variables from $vw(k-1, X_1, T_1)$ and $vw(k-1, X_2, T_2)$.
Given $V^*_{k-1} = (P, r^*P, \tilde{r}P, \overline{r}P, H(e(P,P)^r))$, the algorithm $\mathcal{A}$ has to decide whether $r = r^*\tilde{r}\overline{r} \bmod q$ ($\mathcal{A}$ outputs *yes* in this case) or r is random ($\mathcal{A}$ outputs *no* in this case) where $r^*, \tilde{r}, \overline{r} \in_R Z^*_q$. For this, $\mathcal{A}$ first generates a tree of height k having structure of KeyAgreement with three subtrees T^*, $\widetilde{T}$ and $\overline{T}$. The leaf level secret key distribution of these subtrees are X^*, $\widetilde{X}$ and $\overline{X}$ respectively. Then $\mathcal{A}$ sets :
$V^*_k = (vw(k-1, X^*, T^*), vw(k-1, \widetilde{X}, \widetilde{T}), vw(k-1, \overline{X}, \overline{T}), r^*P, \tilde{r}P, \overline{r}P, H(e(P,P)^r))$ and runs $\mathcal{F}_{D\widehat{D}_k}$ on input V^*_k. Now $Prob[\mathcal{A}$ outputs *no* on input $V^*_{k-1}] = Prob[\mathcal{F}_{D\widehat{D}_k}(D_k = V^*_k) = 1]$
and $Prob[\mathcal{A}$ outputs *yes* on input $V^*_{k-1}] = Prob[\mathcal{F}_{D\widehat{D}_k}(\widehat{D}_k = V^*_k) = 1]$.
Consequently, $|Prob[\mathcal{A}$ outputs *no* on input $V^*_{k-1}] - Prob[\mathcal{A}$ outputs *yes* on input $V^*_{k-1}]|$
$= |Prob[\mathcal{F}_{D\widehat{D}_k}(D_k = V^*_k) = 1] - Prob[\mathcal{F}_{D\widehat{D}_k}(\widehat{D}_k = V^*_k) = 1]|$. Hence if $\mathcal{F}_{D\widehat{D}_k}$ can distinguish between D_k and $\widehat{D}_k$, then $\mathcal{A}$ solves DHBDH problem in $\langle G_1, G_2, e\rangle$. ■

4.2 Security Against Active Adversary

This is discussed in the Appendix.

5 Efficiency

This involves the communication and computation efficiency. In each round, a user may have to transmit publicly an element of G_1 to some or all the other users. Also it has to perform some operations like scalar multiplications, pairing computations. The number of rounds, total group elements sent, total messages exchanged provides the communication overhead in the protocol whereas total pairing computation, total scalar multiplications used incurs the computation costs.

First consider the unauthenticated version. Let $R(n)$ denote the total number of rounds, $P(n)$ the total pairings computed, $B(n)$ the combined message size and $E(n)$ the total number of scalar multiplications. If an user sends publicly an element of G_1 to some or all remaining users, then this counts one to $B(n)$. Thus $B(n)$ is the total number of such group elements. Proofs of the following results will be provided in the full version of the paper.

Lemma 5.1 *For $n > 2$, the following recursions and bounds hold for $R(n), B(n), E(n)$ and $P(n)$:*

1. *$R(3n) = 1 + R(n)$; $R(3n+1) = 1 + R(n+1)$; $R(3n+2) = 1 + R(n+1)$; with initial conditions $R(1) = 0, R(2) = 1$. Consequently, $R(n) = \lceil \log_3 n \rceil$ for all n.*

2. *$B(3n) = 3 + 3B(n)$; $B(3n+1) = 3 + 2B(n) + B(n+1)$; $B(3n+2) = 3 + B(n) + 2B(n+1)$; with initial conditions $B(1) = 0, B(2) = 3$. Consequently, $B(n) < \frac{5}{2}(n-1)$ for $n > 2$.*

3. *$E(3n) = 3 + 3E(n)$; $E(3n+1) = 3 + 2E(n) + E(n+1)$; $E(3n+2) = 3 + E(n) + 2E(n+1)$; with initial conditions $E(1) = 0, E(2) = 3$. Consequently, $E(n) < \frac{5}{2}(n-1)$ for $n > 2$.*

4. *$P(3n) = 3n + 3P(n)$; $P(3n+1) = 3n + 1 + 2P(n) + P(n+1)$; $P(3n+2) = 3n + 2 + P(n) + 2P(n+1)$; with initial conditions $P(1) = 0, P(2) = 2$. Consequently, $P(n) \leq n\lceil \log_3 n \rceil$.*

For authenticated version, let $R_a(n)$, $B_a(n)$, $E_a(n)$ and $P_a(n)$ be the corresponding terms of the unauthenticated version.

Lemma 5.2 *For $n > 2$, the following relations hold for $R_a(n)$, $B_a(n)$, $E_a(n)$ and $P_a(n)$:*

1. *$R_a(n) = R(n)$. Consequently, $R_a(n) \leq n\lceil \log_3 n \rceil$.*

2. *$B_a(n) = 2B(n)$. Consequently, $B_a(n) < 5(n-1)$ for $n > 2$.*

3. *$E_a(3n) = 9 + E_a(n)$; $E_a(3n+1) = 9 + 2E_a(n) + E_a(n+1)$; $E_a(3n+2) = 9 + E_a(n) + 2E_a(n+1)$; with $E_a(1) = 0$, $E_a(2) = 12$. Consequently, $E_a(n) \leq 9(n-1)$ for $n > 2$.*

4. *$P_a(3n) = 5(3n) + 3P_a(n)$ for $n > 2$; $P_a(3n+1) = 5(3n+1) + 2P_a(n) + P_a(n+1)$; $P_a(3n+2) = 5(3n+2) + P_a(n) + 2P_a(n+1)$; with $P_a(1) = 0, P_a(2) = 11$. Also $P_a(n) \leq 5P(n) + 3$. Consequently, $P_a(n) \leq 5n\lceil \log_3 n \rceil + 3$.*

6 Comparison

In this section, we compare our protocol to some of the previously known protocols. Burmester and Desmedt present in [9] an efficient multi-party protocol that can be executed only in two rounds. A class of generic n-party Diffie-Hellman protocols ($n > 2$) is defined in [27]. The entire protocol class is shown

to be secure against passive adversaries based on the intractability of the DDH problem. One group key distribution protocols introduced in [27] is GDH-3. A tree based Diffie-Hellman group key agreement protocol TGDH has been proposed by Kim, Perrig and Tsudik in [18] which is shown to be secure against passive adversaries. We note that the security assumptions behind the various protocols are different and hence in a strict sense an efficiency comparison might not be meaningful. However, we believe that the discussion presented below does provide some idea of the relative efficiency of the various protocols. Table 1 compares the unauthenticated version of our protocol with these protocols. (Inv stands for total number of modular inversions and Mul stands for Total number of multiplications used).

	$R(n)$	$B(n)$	$E(n)$	$P(n)$	Inv
BD [9]	2	$2n$	$n(n+1)$	–	n
GDH-3 [27]	$n+1$	$3(n-1)$	$5n-6$	–	–
TGDH [18]	$\lceil \log_2 n \rceil$	$n\lceil \log_2 n \rceil$	$n\lceil \log_2 n \rceil$	–	–
Our Protocol	$\lceil \log_3 n \rceil$	$< \frac{5}{2}(n-1)$	$< \frac{5}{2}(n-1)$	$\leq n\lceil \log_3 n \rceil$	–

Table 1 : Protocol Comparison (unauthenticated versions).

Points to note for unauthenticated protocols :

1. The underlying group of GDH-3 and BD protocol is a multiplicative subgroup of Z_p^* of order q where p and q both are prime.
2. The communication complexity is measured by $R(n)$ and $B(n)$ and our protocol achieves the minimum for both among all known protocols, except BD protocol. The computation complexity of our protocol consists of two parts – exponentiation and pairing. The number of exponentiation is less than all other protocols, but additionally $n\lceil \log_3 n \rceil$ pairings are required. Assuming that each pairing computation is approximately equal to three exponentiations [4], [16], the TGDH and GDH-3 algorithms are more efficient than ours. This is based on the current state of the art in the algorithm for computing pairings. Any improvement in pairing computation algorithms will improve the efficiency of our protocol with respect to both TGDH and GDH-3
3. Moreover, all the above protocols give DH-key except BD protocol.

It is not a trivial task to provide authentication in BD protocol. The authors indicate that zero-knowledge proofs are required to convert this protocol into an authenticated protocol. We compare the authenticated version of our protocol with the existing authenticated Diffie-Hellman based group key agreement protocols SA-GDH-2 [3] and ID-AGKA [15] in Table 2.

	$R_a(n)$	$B_a(n)$	$E_a(n)$	$P_a(n)$	Inv	Mul
SA-GDH-2 [3]	n	n^2	n^2	–	n	$2n^2-2n$
ID-AGKA [15]	$\lceil \log_2 n \rceil$	$3n\lceil \log_2 n \rceil$	$\leq 2n\lceil \log_2 n \rceil + 4n - 2$	$2n\lceil \log_2 n \rceil$	–	–
Our ID-based auth. prot.	$\lceil \log_3 n \rceil$	$< 5(n-1)$	$\leq 9(n-1)$	$\leq 5n\lceil \log_3 n \rceil + 3$	–	–

Table 2 : Protocol Comparison (authenticated version).

Points to note for authenticated protocols :

1. The underlying group of SA-GDH-2 protocol is multiplicative subgroup of Z_p^* of order q where both p and q are prime.
2. Number of rounds and combined message size in our protocol is less as compared to other protocols. Number of exponentiations or EC scalar multiplication in our protocol is also less than that required for other protocols. Pairing computations of our protocol is slightly more than that for ID-AGKA.

7 Conclusion

We have described an unauthenticated as well as a ID-based authenticated multi-party key agreement protocol using pairing. In fact, our protocol can use any secure two and three party protocol and provides all the desirable security attributes possessed by both of them. Thus the security analysis of our protocol against an active adversary relies on the security of the underlying two and three party protocols. Furthermore, the computation and communication complexity of our protocol compares favourably with other known protocols.

References

[1] M. Abdalla, M. Bellare and P. Rogaway. *DHIES : An encryption scheme based on the Diffie-Hellman problem*, CT-RSA 2001 : 143-158.

[2] S. Al-Riyami and K. G. Paterson. *Tripartite Authenticated Key Agreement Protocols from Pairings.* Cryptology ePrint Archive, Report 2002/035, avalable at http://eprint.iacr.org/2002/035.

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

[4] P. S. L. M. Barreto, H. Y. Kim and M. Scott. *Efficient algorithms for pairing-based cryptosystems.* Advances in Cryptology - Crypto '2002, LNCS 2442, Springer-Verlag (2002), pp. 354-368.

[5] K. Becker and U. Wille. *Communication Complexity of Group Key Distribution.* ACMCCS '98.

[6] D. Boneh and M. Franklin. *Identity-Based Encryption from the Weil Pairing.* In Advances in Cryptology - CRYPTO '01, LNCS 2139, pages 213-229, Springer-Verlag, 2001.

[7] D. Boneh, B. Lynn, and H. Shacham. *Short Signature from Weil Pairing*, Proc. of Asiacrypt 2001, LNCS, Springer, pp. 213-229, 2001.

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

[9] M. Burmester and Y. Desmedt. *A Secure and Efficient Conference Key Distribution System.* In A. De Santis, editor, Advances in Cryptology EUROCRYPT '94, Workshop on the theory and Application of Cryptographic Techniques, LNCS 950, pages 275-286, Springer-Verlag, 1995.

[10] J. Cha and J. Cheon. *An Identity-Based Signature from Gap Diffie-Hellman Groups.* Available from http://eprint.iacr.org, 2002.

[11] L. Chen and C. Kudla. *Identity Based Authenticated Key Agreement Protocols from Pairings.* Cryptology ePrint Archive. Avalable at http://eprint.iacr.org, 2002

[12] C. Cocks. *An Identity Based Encryption Scheme based on Quadratic Residues.* In Cryptography and Coding, LNCS 2260, pp. 360-363, Springer-Verlag, 2001.

[13] R. Cramer and V. Shoup. *A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack.* In H. Krawczyk, editor, Advances in Cryptology - CRYPTO '98, Volume 1462 of LNCS. Springer-Verlag, Berlin , Germany, Aug. 1998.

[14] W. Diffie and M. Hellman. *New Directions In Cryptography.* IEEE Transactions on Information Theory, IT-22(6) : 644-654, November 1976.

[15] D. Nalla and K. C. Reddy. *Identity Based Authenticated Group Key Agreement Protocol.* In Proceedings of INDOCRYPT-2002, LNCS , Springer-Verlag, 2002.

[16] S. Galbraith, K. Harrison and D. Soldera. *Implementing the Tate Pairing.* Algorithm Number Theory Symposium - ANTS V, LNCS 2369, Springer- Verlag (2002), pp. 324-337.

[17] A. Joux. *A One Round Protocol for Tripartite Diffie-Hellman.* ANTS IV, LNCS 1838, pp. 385-394, Springer-Verlag, 2000.

[18] Y. Kim, A. Perrig, and G. Tsudik. Simple and Fault-tolerant Key Agreement for Dynamic Collaborative Groups. In S. Jajodia, editor, *7th ACM Conference on Computation and Communication Security*, pages 235-244, Athens, Greece, Nov. 2000, ACM press.

[19] Y. Kim, A. Perrig, and G. Tsudik. *Tree based Group Key Agreement*, Report 2002/009, http://eprint.iacr.org, 2002.

[20] Y. Kim, A. Perrig, and G. Tsudik. *Communication-efficient Group Key Agreement.* In information systems security, Proceedings of the 17th International Information Security Conference, IFIP SEC '01, 2001.

[21] 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. Avalable at citeseer.nj.nec.com/law98efficient.

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

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

[24] K. G. Paterson. *ID based signature from pairings on elliptic curves*, available from http://eprint.iacr.org, 2002.

[25] A. Shamir. *Identity-based Cryptosystems and Signature Schemes.* In Advances in Cryptology - CRYPTO '84, LNCS 196, pages 47-53, Springer-Verlag, 1984.

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

[27] M. Steiner, G. Tsudik, M. Waidner. *Diffie-Hellman Key Distribution Extended to Group Communication*, ACM Conference on Computation and Communication Security, 1996.

[28] F. Zhang, S. Liu and K. Kim. *ID-based One Round Authenticated Tripartite Key Agreement Protocol with Pairings.* Avalable at http://eprint.iacr.org, 2002.

A Security Against Active Adversary

The two and three party protocols are the subroutines invoked by the main algorithm KeyAgreement. The first thing to note is that the security of KeyAgreement is based on the security of the underlying two and three party protocols. If these two protocols are replaced by some other two and three party protocols, then we also obtain a secure multiparty protocol.
In active attack, an adversary not only just records the data, but also can alter, inject, intercept and replay

messages. The goal of the authentication mechanism is to instill confidence in a user that the persons (s)he is communicating with are indeed the persons they claim to be. In the authenticated version of our protocol, this is done by sending a special signature, called authenticator, on each public key. For a particular round, if user j is a representative and s is the common secret key shared by each user in the user set to which j belongs in this round (or random secret key generated by user j), then this authentication is provided by sending sP together with a special signature T_j on sP given by $T_j = \widehat{H}(sP)S_j + s^2P$, where S_j is the long term private key of user j as generated by the KGC.
The concrete security goals against an active adversary are as follows:

Implicit Key Authentication : Implicit key authentication to a user A implies that only the users with whom A wants to agree upon a common key may be able to compute a particular key. This is a fundamental security goal for any authenticated protocol and is independent of the protocol details.

Known Session Key Security : A protocol is called known session key secure, if an adversary, having obtained some previous session keys, still can't get the session keys of the current run of the protocol.

(Perfect) Forward Secrecy : A protocol is said to satisfy forward secrecy if compromise of the long term private keys of one or more users does not affect the security of the previous session keys. If this property holds even when the private keys of all the participating users are compromised, we say the protocol satisfies perfect forward secrecy.

No Key-compromise Impersonation : A protocol resists key-compromise impersonation when the compromise of one user's long term private key does not imply that the private keys of other users will also be compromised. The adversary may impersonate the compromised user in the subsequent protocols, but cannot impersonate other users.

No Unknown Key-share : We say that a protocol is subjected to unknown key-share attack if an adversary convinces a group of users that they share a key with the adversary, whereas in fact the key is shared between the group and another party.

No Key Control : A protocol is said to have no key control if it is not possible to control or predict the value of the session key by any participant (or an adversary).

Next we argue that the authenticated version of the protocol KeyAgreement satisfies the above properties. Our argument proceeds as follows for implicit key authentication. A similar argument can be given for other properties.

From level 0 to level 1 : The user sets are triplets or sets with a pair of users or singleton sets. For a triplet $U_j^{(0)}, U_{j+1}^{(0)}, U_{j+2}^{(0)}$ – authenticated key agreement takes place among these three users according to the tripartite algorithm of Zhang et.al. in [28]. They have used a special signature scheme on each public key to provide this authentication. The signature scheme used by them is secure against existential forgery under an adaptively chosen message attack in the random oracle model. They have argued heuristically that this signature scheme provides in their protocol all the security attributes stated above against an active adversary. For a pair $U_j^{(0)}, U_{j+1}^{(0)} - Rep(U_j^{(0)})$ which in this case is user j, generates randomly another short term secret key $\overline{s}_j$ so that user j has now two keys $s_j^{(0)}, \overline{s}_j$. These two keys and the key $s_{j+1}^{(0)}$ of user set $U_{j+1}^{(0)}$ is used in another invocation of Joux's protocol to agree a common secret key. The same special signature scheme is used on the transmitted public values to provide authentication and using heuristic

arguments, this two party protocol can be shown to have all the security attributes stated above against an active adversary following [28].

From level $i-1$ to level i : Suppose user $u \in U_j^{(i-1)}$ wants to agree upon a common secret key with users in the user sets $U_{j+1}^{(i-1)}, U_{j+2}^{(i-1)}$. User u receives the authenticators T_{j_2}, T_{j_3}, where $j_2 = Rep(U_{j+1}^{(i-1)})$ and $j_3 = Rep(U_{j+2}^{(i-1)})$. Let $j_1 = Rep(U_j^{(i-1)})$. User u verifies T_{j_2} and T_{j_3} using Q_{j_2} and Q_{j_3}. Thus u gains confidence that the communication is taking place among users j_1, j_2 and j_3. Also users j_2 and j_3 will respectively be using the common secret key for user sets $U_{j+1}^{(i-1)}$ and $U_{j+2}^{(i-1)}$ to provide this confidence. User u uses the common secret key of user set $U_j^{(i-1)}$ to compute the new common secret key for the user set $U_j^{(i-1)} \bigcup U_{j+1}^{(i-1)} \bigcup U_{j+2}^{(i-1)}$ in the next level. Thus user u, agreeing upon a secret key with users j_2 and j_3, gains confidence that he has agreed upon a common key with all users in the user set $U_j^{(i-1)} \bigcup U_{j+1}^{(i-1)} \bigcup U_{j+2}^{(i-1)}$ and with none outside this set. This confidence is attained without verifying the IDs of all users in $U_{j+1}^{(i-1)}$ and $U_{j+2}^{(i-1)}$. Thus an active adversary, without knowing the long term secret keys of the users j_1, j_2, j_3, is not able to compute this common secret key. This provides implicit key authentication property at this round.

B Dynamic Membership Operations : Member Insert, Member Delete :

Suppose T is a keytree for n users $\{1, 2, \ldots, n\}$ having structure of KeyAgreement with $t = R(n)$ rounds and let a new user $n+1$ with private key $s_{n+1}^{(0)}$ requests for join.

procedure Insert(T, n)

//finding keypath
$p = \lfloor \frac{n}{3} \rfloor$; $k = R(n)$; $T' = T$;
while ($k > 1$) **do**
  Let T_L, T_M, T_R be respectively the left, middle and right subtrees of T';
  if ($|T_L| = |T_M| = |T_R| = p$) **then**
    $T' = T_R$; output (k, R);
  end if
  if ($|T_L| = |T_M| = p; |T_R| = p+1$) **then**
    $T' = T_M$; output (k, M);
  end if
  if ($|T_L| = p; |T_M| = |T_R| = p+1$) **then**
    $T' = T_L$; output (k, L);
  end if
  $p = \lfloor \frac{p}{3} \rfloor$; $k = k - 1$;
end do
Let the keypath be $(t, B_1), (t-1, B_2), \ldots, (2, B_{t-1})$ where $B_i = L$ or M or R for $1 \leq i \leq t-1$);
Let this path reaches a node at level 1 having $m(\leq 3)$ children (leaf nodes).
//updating keypath
if ($m = 1$) **then**
  Let $U_{i-1}^{(0)}$ be the user set at level 0;
  $\widehat{U}_1 = U_{i-1}^{(0)}; \widehat{U}_2 = n+1$; $\hat{s}_1 = s_{i-1}^{(0)}; \hat{s}_2 = s_{n+1}^{(0)}$;
  call CombineTwo $(\widehat{U}[1,2], \hat{s}[1,2])$;
  Let $s_j^{(1)}$ be the agreed key between user sets $\widehat{U}_1, \widehat{U}_2$; $U_j^{(1)} = \widehat{U}[1,2]$;
end if

if $(m = 2)$ **then**
Let $U_{i-2}^{(0)}, U_{i-1}^{(0)}$ be the user sets at level 0;
$\widehat{U}_1 = U_{i-2}^{(0)}; \widehat{U}_2 = U_{i-1}^{(0)}; \widehat{U}_3 = n+1$; $\hat{s}_1 = s_{i-2}^{(0)}; \hat{s}_2 = s_{i-1}^{(0)}; \hat{s}_3 = s_{n+1}^{(0)}$;
call CombineThree $(\widehat{U}[1,2,3], \hat{s}[1,2,3])$;
Let $s_j^{(1)}$ be the agreed key among user sets $\widehat{U}_1, \widehat{U}_2, \widehat{U}_3$; $U_j^{(1)} = \widehat{U}[1,2,3]$;
enf if
if $(m = 3)$ **then**
Let $U_{i-2}^{(0)}, U_{i-1}^{(0)}$ and $U_i^{(0)}$ be the user sets at level 0;
$\widehat{U}_1 = U_i^{(0)}; \widehat{U}_2 = n+1$; $\hat{s}_1 = s_i^{(0)}; \hat{s}_2 = s_{n+1}^{(0)}$;
call CombineTwo $(\widehat{U}[1,2], \hat{s}[1,2])$;
Let KEY be the agreed key between user sets $\widehat{U}_1, \widehat{U}_2$; $s_i^{(0)} = KEY; U_i^{(0)} = \widehat{U}[1,2]$;
call CombineThree $(U^{(0)}[i-2, i-1, i], s^{(0)}[i-2, i-1, i])$;
Let $s_j^{(1)}$ be the agreed key among user sets $U_{i-2}^{(0)}, U_{i-1}^{(0)}, U_i^{(0)}$; $U_j^{(1)} = U^{(0)}[i-2, i-1, i]$;
end if
$l = 1$ to $t-1$ **do**
if $(B_{t-l} = L)$ **then**
call CombineThree $(U^{(l)}[j, j+1, j+2], s^{(l)}[j, j+1, j+2])$;
Let $s_m^{(l+1)}$ be the agreed key among user sets $U_j^{(l)}, U_{j+1}^{(l)}, U_{j+2}^{(l)}$; $U_m^{(l+1)} = U^{(l)}[j, j+1, j+2]$;
end if
if $(B_{t-l} = M)$ **then**
call CombineThree $(U^{(l)}[j-1, j, j+1], s^{(l)}[j-1, j, j+1])$;
Let $s_m^{(l+1)}$ be the agreed key among user sets $U_{j-1}^{(l)}, U_j^{(l)}, U_{j+1}^{(l)}$; $U_m^{(l+1)} = U^{(l)}[j-1, j, j+1]$;
end if
if $(B_{t-l} = R)$ **then**
call CombineThree $(U^{(l)}[j-2, j-1, j], s^{(l)}[j-2, j-1, j])$;
Let $s_m^{(l+1)}$ be the agreed key among user sets $U_{j-2}^{(l)}, U_{j-1}^{(l)}$ and $U_j^{(l)}$; $U_m^{(l+1)} = U^{(l)}[j-2, j-1, j]$;
end if
$j = m$;
end do

end Insert

Suppose T_L, T_M, T_R are respectively the left, middle and right subtree of key tree T with n leaf nodes $\{1, 2, \ldots, n\}$. The tree T has the structure of KeyAgreement and so has each of it's subtrees. Now suppose a member i, $1 \leq i \leq n$, wants to leave the group.

procedure Delete (T, n, i)

$p = \lfloor \frac{n}{3} \rfloor$;
Case 1 : $|T_L| = |T_M| = |T_R| = p$
if (i is the leaf node of T_L) **then**
remove it from T_L; adjust the resulting subtree T_L' such that the structure of KeyAgreement is preserved in T_L' and update the key paths;
end if
if (i is the leaf node of T_M) **then**

remove it from T_M; adjust the resulting subtree T'_M to preserve it's structure of KeyAgreement and update the key paths;
$T_{temp} = T'_M$; $T'_M = T_L$; $T_L = T_{temp}$;
end if
if (i is the leaf node of T_R) **then**
remove it from T_R; adjust the resulting subtree T'_R so that the structure of KeyAgreement is preserved in T'_R and update the key paths;
$T_{temp} = T'_R$; $T'_R = T_L$; $T_L = T_{temp}$;
end if

Case 2 : $|T_L| = |T_M| = p; |T_R| = p + 1$
if (i is the leaf node of T_L) **then**
remove it from T_L; extract one leaf node of T_R in such a way that the resulting right subtree T'_R has the structure of KeyAgreement and minimum number of key path updates are required; insert this extracted node to T_L as the i^{th} leaf node resulting the left subtree T'_L having structure of KeyAgreement and finally update the key paths.
end if
if (i is the leaf node of T_M) **then**
remove it from T_M; extract one leaf node of T_R in such a way that the resulting right subtree T'_R has the structure of KeyAgreement and minimum number of key path updates are required; insert this extracted node to T_M as the i^{th} leaf node resulting the middle subtree T'_M having structure of KeyAgreement and finally update the key paths.
end if
if (i is the leaf node of T_R) **then**
remove it from T_R; adjust the resulting subtree T'_R such that the structure of KeyAgreement is preserved in T'_R and update the key paths.
end if

Case 3 : $|T_L| = p$; $|T_M| = |T_R| = p + 1$;
if (i is the leaf node of T_L) **then**
remove it from T_L; extract one leaf node of T_M so that the resulting middle subtree T'_M has the structure of KeyAgreement and minimum number of key path uptates; insert this extracted node to T_L as the i^{th} leaf node resulting in the left subtree T'_L having structure of KeyAgreement and finally update the key paths.
end if
if (i is the leaf node of T_M) **then**
remove it from T_M; adjust the resulting subtree T'_M in such a way that the structure of KeyAgreement is preserved in T'_M and update the key paths.
end if
if (i is the leaf node of T_R) **then**
remove it from T_R; adjust the resulting subtree T'_R so that it has the structure of KeyAgreement and update the key paths.
$T_{temp} = T'_R$; $T'_R = T_M$; $T_M = T_{temp}$;
end if
end Delete