Sādhanā Vol. 40, Part 7, October 2015, pp. 2143–2161. © Indian Academy of Sciences

A new two-round dynamic authenticated contributory group key agreement protocol using elliptic curve Diffie–Hellman with privacy preserving public key infrastructure

VANKAMAMIDI S NARESH[1,*] and NISTALA V E S MURTHY[2]

[1]Department of Computer Science, S V K P and Dr K S R Arts and Science College, Penugonda 534320, Andhra Pradesh, India
[2]Department of Mathematics, Andhra University, Visakhapatnam 530003, Andhra Pradesh, India
e-mail: vsnaresh111@gmail.com

MS received 5 June 2013; revised 20 June 2015; accepted 1 July 2015

Abstract. In this paper a new two-round authenticated contributory group key agreement based on Elliptic Curve Diffie–Hellman protocol with Privacy Preserving Public Key Infrastructure (PP-PKI) is introduced and is extended to a dynamic authenticated contributory group key agreement with join and leave protocols for dynamic groups. The proposed protocol provides such security attributes as forward secrecy, backward secrecy, and defense against man in the middle (MITM) and Unknown key-share security attacks and also authentication along with privacy preserving attributes like anonymity, traceability and unlinkability. In the end, they are compared with other popular Diffie–Hellman and Elliptic Curve Diffie–Hellman based group key agreement protocols and the results are found to be satisfactory.

Keywords. Secure group communication (SGC); mobile ad-hoc networks (MANETS); dynamic authenticated group key agreement (DAGKA); elliptic curve Diffie–Hellman (ECDH); privacy preserving public key infrastructure (PP-PKI).

1. Introduction

With the exponential growth in modern communication, Secure group communication (SGC) is becoming an important research area in various group ware applications, such as tele-conferencing, tele-medicine, real-time information services, distributive interactive simulations,

*For correspondence

grid computing and collaborative work. It refers to a scenario in which a group of participants can send and receive messages to/from other group members in a way that outsiders are unable to make any information even when they are able to intercept the messages. Since key distribution is a corner stone of any SGC, it has naturally received high attention.

Most of the group key agreement protocols are DLP-based. However, the key length for secure DLP-based D–H has increased over recent years, which has also placed a heavier processing load on applications using the same, making them not suitable for the ad-hoc networks. Notice that the processing load is especially critical for ad-hoc networks, which have relatively a limited bandwidth, slower CPU speed, limited battery power and have high bit-error rate wireless links and the ad-hoc networks demanding SGC are growing rapidly in the last few years.

Elliptic curve cryptographic schemes are public-key mechanisms that provide the same functionality as (DLP-based) D–H schemes. However, their security is based on the hardness of the Elliptic Curve Discrete Logarithm Problem (ECDLP). In fact, ECDLP is harder than DLP. Currently the best algorithms known to solve the ECDLP have fully exponential running time in contrast to the sub exponential-time algorithms known for the DLP. This means that the same desired security level can be attained with significantly smaller keys in ECDLP-based schemes when compared to DLP-based ones. For example, it is generally accepted that a 160-bit elliptic curve key of a given length provides the same level of security as a 1,024-bit D–H/RSA key of the same length. Of course, the advantages that can be gained from smaller key sizes include speed, efficient use of power, bandwidth and storage. So Elliptic Curve Cryptography (ECC) (BSI 2009; Koeblitz 1987; Miller 1986) emerged as the cryptographic choice for ad-hoc networks and communication devices. Also, recent results (Batina *et al* 2006; Kazuo *et al* 2007; Malan *et al* 2004) indicate that the execution of ECC operations in mobile ad-hoc networks is feasible with predictable improved performance. In fact, Kazuo *et al* (2007) proposes P-MALU (Parallelized Modular Arithmetic Unit) with which one can achieve over 80K scalar multiplications per second with predictably improved performance. For instance a large group of size $m = 40{,}000$ the proposed protocol requires $2m = 80{,}000$ (Cf. Section 5.1, table 5) sequential scalar multiplication for generating the group key which can be managed within one second time by using P-MALU.

Coming to the group key generating techniques, all of them can be broadly divided into two classes. In the first one, a single member of the group generates the key (Burmester & Desmedt 1994; Wang *et al* 2006) accepting all the computational and communicational loads and distributes it to the remaining members. Further, it requires a trusted key generator for reliability. In the second one, a contributory key is generated using group key agreements (Alves-Foss 2000; Ingemarsson *et al* 1982; Kim *et al* 2004b; Steiner *et al* 1996; Zheng *et al* 2007) in which each member of the group contributes a part to generate the shared group key, distributing both the computational and communicational loads. Schemes in this class provide better key secrecy than the ones in the first class. A Contributory Group Key Agreement (CGKA) (Amir *et al* 2002) is thus computed as a function of contributions from all members. These protocols are appropriate for dynamic peer groups, where new members join or the existing ones leave and; no single member can influence the shared key unilaterally. Consequently, the problems of trust and failure associated with the group members are inherently taken care of.

For dynamic group key agreement protocols with join and leave protocols, one can refer to Alves-Foss (2000); Becker & Wille (1998); Dutta & Barua (2005, 2008); Kim *et al* (2004a, b); Steiner *et al* (1996); Wong *et al* (1998); Wang *et al* (2006); Zheng *et al* (2007). Although the tree based GKA protocols are sometimes considered to be superior to their non-tree based GKA

cousins, note that extending these protocols into dynamic setting is far more difficult, as join and/or leave operations in the tree based GKA protocols can sometimes disturb the entire tree structure, especially when a leaving/joining member is somewhere in the middle of the tree or so. Dutta & Barua (2005) beautifully illustrated how difficult the process of preserving dynamic nature in these tree based GKA protocols. The present paper falls in the category of non-tree based which are more robust to message losses and more appropriate for dynamic groups.

For defense against Man In The Middle (MITM) attack, one can refer to D–H with Public Key Infrastructure (PKI) (Kaufman *et al* 2002). Notice that D–H with PKI is not only secured against active attacks but also has the additional advantage of eliminating the first two messages of those D–H based schemes that do not use PKI. Using ECDH with PKI in our case it amounts to reduce $2(m-1)$ messages over those ECDH-based schemes that do not use PKI, while generating $(m-1)$ ECDH shared keys in the first round.

Using D–H with PKI in any GKA protocol provides group member's authentication. However with the popularity of the internet and the legislation, PKI-based authentication in some ways threatens user privacy. In order to address this issue the concept of group signature (Chaum & van Heyst 1991) was introduced. This adopts group based authentication to achieve privacy of the individual signer against potential verifiers.

A group signature scheme is a method for allowing a member of a group to anonymously sign a message on behalf of the group. For example, a group signature scheme could be used by an employee of a large company where it is sufficient for a verifier to know a message was signed by an employee, but not which particular employee signed it. Another similar example is, key-card access to restricted areas where it is unwanted/inappropriate to track individual employee's movements, but necessary to secure areas to only certain employees in the group. Many schemes of this type have been proposed, however many of them share three basic requirements such as anonymity, traceability, and unlinkability.

In order to enhance privacy in PKI, some researchers used pseudonym certificate (Benjumea *et al* 2004; Kwon *et al* 2006). But these approaches seem to have the serious limitation that they cannot satisfy full anonymity, because their authentication transactions are still linkable. Recently, Lee *et al* (2011) proposed a privacy-preserving PKI based on group signature. However, one can extend the proposed protocol using privacy preserving PKI and/or group signature, addressing the issues of providing anonymity, traceability, and unlinkability, without any increase in either time complexity or in computational complexity.

There are several standard security attributes and also privacy preserving attributes which can be considered for incorporation into any cryptic scheme. However, in what follows, we recall some of them that are incorporated in the proposed protocol: (a) Key secrecy – here, the key is computed only by the members of the group; (b) Forward secrecy – as soon as an existing member leaves the group, a new key is computed in such a way that it is hard to find the new key with the knowledge of previous key; (c) Backward secrecy – as soon as a new member joins, a new key is computed in such a way that it is hard to compute old key with the knowledge of new key; (d) Defense against MITM attack – a form of eavesdropping, wherein communication between two users is monitored and modified by an unauthorized party. (Generally, the attacker actively eavesdrops by intercepting a public key message exchange and retransmits the message while replacing the requested key with his own.); (e) Unknown key-share security – it is not possible to coerce A into thinking he is sharing a key with B, when he is actually sharing a key with another (honest) user C (That is, it is possible for A to believe he is sharing a key with $B \neq C$, while C correctly thinks the key is shared with A.) and privacy presurving attributes such as, (f) Anonymity – Given a message and its signature, the identity of the individual signer

cannot be determined without the group identity. (g) Traceability – Given any valid signature, the group manager should be able to trace which user issued the signature. (This and the previous requirement imply that only the group manager can break users' anonymity.) (h) Unlinkability – Given two messages and their signatures, we cannot tell if the signatures were from the same signer or not.

1.1 *Related work*

Ever since two-party Diffie–Hellman key agreement (Diffie & Hellman 1976) was first proposed in 1976, there have been numerous efforts to extend its simplicity and elegance to group key agreements. The vast majority of SGC protocols based on D–H depend on the discrete logarithm problem (DLP) for their security. The recent work on performance evaluation of group D–H protocols can be found in (Amir *et al* 2002; Hagzan & Bischof 2004). In Amir *et al* (2002), notable group key agreement protocols, such as Centralized Group Key Distribution (CKD), Burmester–Desmedt (BD), and Steer *et al* (STR), were evaluated.

On the basis of D–H, three group key algorithms namely GDH.1, GDH.2, and GDH.3 with increased performance are reported in (Steiner *et al* 1996) where it is shown that two of them are proved to be optimal with respect to certain measures of computation and communication complexities. In brief, the basic idea of these methods is to follow two stages known as up-flow and down-flow stages. In the up-flow stage, a participant i ($1 \leq i \leq n$ for n-party group) collects intermediate values from the participant $i-1$ and computes $g^{n_1 n_2 \cdots n_i}$ by raising $g^{n_1 n_2 \cdots n_{i-1}}$ to the power of n_i, where $n_1, n_2, ..., n_i$ are the private keys of the i participants respectively. The up-flow ends when the highest numbered group member receives up-flow messages and computes the intended group key. On the other hand, the highest numbered group member initiates the down-flow stage after generating the group key. In the down-flow stage participant i receives an ordered list of intermediate values from participant $i+1$ and raises all of them to power of n_i the last entry of the vector is the group key. After getting group key participant i forwards all other values to participant $i-1$, continuing this until first member gets the group key.

Recently, DLP-based efficient and simpler GKA protocols (Biswas 2008; Murthy & Naresh 2010) were proposed. More precisely, these protocol may not be a good choice for the ad-hoc networks. In view of the above aspects, ECDH-based GKA protocols came into picture. In this direction we analyzed the Group Elliptic Curve Diffie–Hellman (GECDH) protocol (Zheng *et al* 2007) based on ECDLP which is a natural extension of Group D–H (GDH) and the Tree-based Group Elliptic Curve Diffie–Hellman Protocol (TGECDH) (Zheng *et al* 2007) and proposed an ECDH-based GKA protocol in this paper.

1.2 *Contributions*

In this paper, an ECDLP-based dynamic authenticated contributory group key agreement protocol using Elliptic Curve Diffie–Hellman with *PP-PKI* with optimal communication and computational costs for SGC over data networks, is proposed. It uses only two rounds, dynamically updating the group key *without* a rerun of the total/entire DACGKA protocol anew, as soon as a member joins or leaves the existing group. Further, it is secured under the ECDDH assumption. The protocol requires no special ordering of the participants. For each execution of the protocol, a random participant can be chosen as the group leader. It is robust as loss of messages

from some participants towards the leader, does not prevent other participants from calculating the group key. Further, the proposed protocol being ECDLP-based one, it is less expensive and also more suitable to resource constraint ad-hoc/sensor networks especially in view of its smaller key sizes and the recent advances in the computational prowess. ECDH with PKI is not only secured against active attacks such as MITM, but also has the additional advantage of eliminating the first two messages of those ECDH-based schemes that do not use PKI. In the proposed protocol it amounts to reduce $2(m - 1)$ messages over those ECDH-based schemes that do not use PKI. It provides *authentication* along with such *privacy preserving attributes* as anonimity, traceability and unlinkability, providing security against active adversary. It also provides security attributes such as key secrecy, key indepencdence, forward secrecy, back ward secrecy and secured agnist Man in the middle attacks and Unknown key-share security attacks.

In the end the proposed protocol is compared with popular Diffie–Hellman, Elliptic Curve Diffie–Hellman based group key agreement protocols, and also with recently published GKA protocol (Dutta & Barua 2008) and the results are found to be satisfactory.

The rest of the paper is organized as follows: Section 2 describes the background material necessary to understand the ECDLP-based protocols. Section 3 presents the proposed ECDH-based group schemes. Section 4 discusses security analysis. Section 5 provides comparative analysis. Finally, Section 6 concludes the paper.

2. Preliminaries

In this section, we focus on preliminaries of elliptic curve cryptography (BSI 2009; Koeblitz 1987; Malan *et al* 2004) for constructive terminology to be adopted before entering into the actual proposed protocol.

2.1 *The common abbreviations and notations*

Abbreviations:

- D–H – Diffie–Hellman
- ECDH – Elliptic curve Diffie–Hellman
- ECDLP – Elliptic curve discrete log problem
- GC – Group controller
- GKA – Group key agreement
- ACGKA – Authenticated contributory group key agreement
- DACGKA – Dynamic authenticated contributory group key agreement
- GK – Group key
- PGK – Previous group key
- NJGK – New join group key
- NLGK – New leave group key
- PKI – Public key infrastructure
- MITM – Man in the middle attack
- MANETS – Mobile ad-hoc networks
- WSN – Wireless sensor networks
- PP-PKI – Privacy preserving public key infraustructure

Notations (table 1):

Table 1. Notations.

Symbol	Comment
p	Large prime number
F_p	The finite field of p elements
E	An elliptic curve defined by Weierstrass equation
$E(F_p)$	An elliptic curve group over the finite field F_p
P, Q	Points on the elliptic curve $E(F_p)$
$P + Q$	The sum of two points P and Q in $E(F_p)$
$[k]P$	The k-th multiple of a point P i.e., $[k]P = P + P + ...+$ (k times)
x_P (or) $(P)_x$, y_P (or) $(P)_y$	The x and y coordinates of point P respectively
m	Total number of members in the group
P	The base point of $E(Fp)$ i.e., a generator of a sub group of $E(Fp)$
n	The order of base point P typically, n is a prime of bit length ≥ 224
M_i	i-th group member, $1 \leq i \leq m$
M_l	The group controller (GC)
x_i	The private key of member M_i, This is an integer belongs to $[1, n-1]$
$x \leftarrow [1, n-1]$	Choose an integer x from $[1, n-1]$
X_i	The public key of member M_i
$x_{K_{l,i}}$	ECDH shared key between GC and M_i, for $1 \leq i \leq m, i \neq l$

2.2 *Background of elliptic curve group*

Let E be an elliptic Curve over F_p described in terms of Weierstrass equation,

$$E(x, y) : y^2 = x^3 + ax + b, a, b \in F_p \tag{1}$$

and with the discriminant

$$\Delta = 4a^3 + 27b^2 \neq 0 \tag{2}$$

The set of rational points in E over F_p denoted by $E(F_p)$

$$E(F_p) = \left\{(x, y) \in F_p^2 : E(x, y) = 0\right\} \cup O \tag{3}$$

where O is the point at infinite.

$E(F_p)$ carries a group structure under point addition with the point at infinity acting as identity element. Scalar multiplication over $E(F_p)$ i.e, the kth multiple of a point P belongs to $E(F_p)$ can be represented as follows:

$$[k]P = P + P + ... + (ktimes) \tag{4}$$

Note: For integers j and k, we have

$$[j]([k]P) = [jk]P = [k]([j]P). \tag{5}$$

2.2a *Elliptic curve domain parameters*: Elliptic Curve Domain Parameters (BSI 2009) (p, a, b, P, n, h) a set of information for communicating parties to identify a certain elliptic curve group used in cryptography. Here p is a large prime number, a and b are the coefficients of the Weierstrass equation, P is the base point of $E(F_p)$, having order n, and finally the co-factor $h = \#E(F_p)/n$. The base point P generates a cyclic group of order n. i.e, $E(F_p) =< P >= \{P, [2]P, ..., [n-1]P, [n]P\}$.

2.2b *Elliptic curve discrete logarithm problem*: The Elliptic curve discrete logarithm problem (ECDLP) (BSI 2009) is defined as follows:

Given the elliptic curve domain parameters as described above and a point $Q \in < P >= E(F_p)$, find the unique integer k, $0 \leq k \leq n-1$, such that $Q = [k]P$.

2.2c *Elliptic curve group computational Diffie–Hellman assumption*: ECGCDHA (Boneh 1998) states that given the values $[\prod x_i]P$, for some choice of proper subsets of $\{1, 2, \ldots, n\}$ a computationally bounded adversary cannot recover the group Diffie–Hellman secret $[\prod_{i=1}^{n} x_i]P$.

An elliptic curve group is called cryptographically strong if the underlined ECDLP is considered to be computationally intractable for the application in use.

2.2d *Cryptographically strong elliptic curve domain parameters over F_p*: The ECDLP is currently considered to be intractable if at least the following conditions (BSI 2009) hold.

- The order n of the base point P must be prime of at least 224 bits.
- To avoid the elliptic curve to be anomalous, the order n must be different from p.
- The ECDLP must not be reducible to DLP in a multiplicative group (F_{p^r}, for a small integer r) Thus it is required that $p^r \neq 1 \bmod n$, for all $1 \leq r \leq 10^4$.
- The class number of the principle order belonging to the endomorphism ring of E should be at least 200.

2.2e *Elliptic curve Diffie–Hellman with public key infrastructure*: In ECDH with PKI, the participants publish their public keys in PKI, instead of exchanging their public keys, as in the third step of below ECDH. To generate the shared key the participants knowing their own private keys and looking for public keys of persons they are intended to communicate in PKI.

Elliptic curve Diffie–Hellman: ECDH (BSI 2009) is one of the key exchange protocols used to establish a shared key between two members. It is based on the additive elliptic curve group. First A and B agree on elliptic curve domain parameters and proceeds as in table 2.

The secret key K is a point in the elliptic curve. If this secret key is to be used as a session key, a single integer must be derived. There are two categories of derivation: reversible and irreversible. If the session key is also required to be decoded as a point on elliptic curve, it is called reversible. Otherwise, it is irreversible. The reversible derivation will result in a session key which doubles the length of the private key. In the irreversible derivation, we can simply use the x-coordinate or simple hash function of the x-coordinate as the session key. In this paper we use irreversible key derivation by simply taking the x-coordinate of the elliptic curve point.

Defense against man in the middle (MITM) attack using D–H with PKI: Diffie–Hellman is secured against active attacks by published D–H numbers (Kaufman *et al* 2002). Here participants A and B compute their public keys from their own private keys and publish them through

Table 2. ECDH.

Party-A	Communication	Party-B
Choose a random number x∈{1,2, . . . $n-1$}		Choose a random number y∈{1,2, . . . $n-1$}
Compute $[x]P$		Compute $[y]P$
Retrieve $[y]P$	[x]P → [y]P ←	Retrieve $[x]P$
Compute $K = [x][y]P = [xy]P$		Compute $K = [y][x]P = [yx]P = [xy]P$

PKI. As a result an intruder cannot get in and modify the published public keys, this makes D–H immune to active attacks. It has the additional advantage of eliminating the first two messages of D–H protocol. A and B knowing their own private keys, look for B and A's public keys respectively in PKI and compute their shared key.

Anonymous authentication using D–H with privacy preserving public key infraustructure (PP-PKI)

In order to enhance privacy in PKI, Lee *et al* (2011) proposed a *PP-PKI* based on group signature. Using D–H with *PP-PKI* based on group signature one can provide anonymous authentication so that such privacy preserving secure attributes such as anonymity, traceability and unlinkability can be attained.

3. Proposed protocols

3.1 *Authenticated contributory group key agreement (ACGKA) protocol*

In this subsection, an ACGKA protocol is proposed to generate a group key among the group members. In this technique, an arbitrary group member acts as a group controller that publicly publishes cryptographically strong elliptic curve domain parameters (p, a, b, P, n, h), using ECDH with *PP-PKI* (Lee *et al* 2011) arrangement that binds public keys with respective user identities by means of a Certificate Authority (CA) and proceeds as follows:

Let $M_1, M_2, ..., M_l, ..., M_m$ be the group members and let the group controller be M_l, where $1 \leq l \leq m$.

Round 1: Initially GC, M_l forms $(m-1)$ two party groups with each of the remaining group members M_i and produces $(m-1)$ shared keys for $(m-1)$ two party groups, as follows:

(i) The group controller M_l, selects a private key $x_l \in \{1, 2, \ldots, n-1\}$ and generates a public key as

$$X_l = [x_l]P \tag{6}$$

(ii) Each group member M_i, where $i \neq l$, also selects a private key $x_i \in \{1, 2, \ldots, n-1\}$ and generates a public key as

$$X_i = [x_i]P, 1 \leq i \leq m, i \neq l. \tag{7}$$

(iii) All the group members including the group controller (GC) publish their public keys in PKI.

(iv) Each member by knowing their own private key x_i, look up GC's public key X_l in PKI and generates a ECDH-style shared key with GC as follows:

$$K_{li} = [x_i]X_l = [x_i]([x_l]P) = [x_i x_l]P = (x_{K_{li}}, y_{K_{li}}), 1 \leq i \leq m, i \neq l, \tag{8}$$

where $x_{K_{li}}, y_{K_{li}} \in F_p$ are x and y coordinates of K_{li} respectively.

Similarly GC, M_l by knowing its own private key and by picking up each member's public key from PKI and generates a ECDH-style shared key with each of the remaining group member as follows:

$$K_{li} = [x_l]X_i = [x_l]([x_i]P) = [x_l x_i]P = (x_{K_{li}}, y_{K_{li}}), 1 \leq i \leq m, i \neq l, \tag{9}$$

where $x_{K_{li}}, y_{K_{li}} \in F_p$ are x and y coordinates of K_{li} respectively.

Hence $x_{K_{li}}$ are the $(m-1)$ shared keys between the GC, M_l and each of the group member M_i, where $1 \leq i \leq m, i \neq l$ respectively.

Round 2: Now the GC computes the $(m-1)$ public numbers L_i, using the two party shared keys $x_{K_{li}}$ generated in round 1 as follows and sends L_i to M_i respectively.

$$L_i = \left[\prod_{j=1, j\neq i}^{m} x_{K_{lj}} \right] P, 1 \leq i \leq m, i \neq l, j \neq l. \tag{10}$$

$$M_l \overset{L_i}{\rightarrow} M_i, 1 \leq i \leq m, i \neq l. \tag{11}$$

After receiving L_i each member M_i in the group then generates group key as follows:

$$K = [x_{K_{li}}]L_i = [x_{K_{li}}] \left[\prod_{j=1, j\neq i}^{m} x_{K_{lj}} \right] P = \left[\prod_{i=1}^{m} x_{K_{li}} \right] P = (x_K, y_K). \tag{12}$$

Since the GC knows all the shared keys, it also generates the group key as follows:

$$K = \left[\prod_{i=1}^{m} x_{K_{li}} \right] P = (x_K, y_K) \tag{13}$$

Hence x_K is now a group key among the group members.

3.2 *Dynamic authenticated contributory group key agreement protocol (DACGKA)*

ACGKA addresses GKA for static groups. However, it is often times necessary either to add a new member (or) delete an existing group member from the initial group. Naturally, it is desirable to do so without executing entire protocol anew. To address this issue we extend ACGKA to DACGKA by proposing a join protocol and a leave protocol.

3.2a *Join protocol*: The main security requirement in the member addition is maintenance of the secrecy of the previous group key from the new group members as well as outsiders.

(i) As soon as a new member M_{m+1} wants to join the group, it intimates the group controller and generates a ECDH-style key $x_{K_{lm+1}}$ with GC using ECDH with PKI.

(ii) The GC generates a random number R'_{m+1} and broadcasts $[x_{K_{lm+1}}.R'_{m+1}]P$ to all the previous members M_i of the group. On receiving, they compute the new key as follows:

$$NJGK = (PGK)x_{K_{lm+1}}R'_{m+1}P = \left(\prod_{i=1}^{m+1} x_{K_{li}}.R'_{m+1}\right)P \tag{14}$$

(iii) GC transmits $[(PGK)R'_{m+1}]P$ to M_{m+1}. Then M_{m+1} computes the new key as follows:

$$NJGK = (PGK)x_{K_{lm+1}}R'_{m+1}P = \left(\prod_{i=1}^{m+1} x_{K_{li}}R'_{m+1}\right)P \tag{15}$$

3.2b *Leave protocol*: The main security requirement in member leaving is the maintenance of the secrecy of the subsequent (future) group key from leaving member as well as the out-sider.

(i) As soon as M_j wants to leave the group, it intimates the GC. Then GC, M_l generates a random number R'_m.

(ii) M_l sends. $\left[R'_m.x^{-1}_{K_{lj}}\right]P$ by encrypting with $x_{K_{li}}$ to the corresponding group member M_i, $i \neq j$, (i.e.) except leaving member.

$$M_l \overset{E_{K_{li}}\left[R'_m x^{-1}_{K_{lj}}\right]P}{\rightarrow} M_i,\ 1 \leq i \leq m, i \neq j. \tag{16}$$

After receiving each member M_i decrypts with $x_{K_{li}}$ and computes the new key as follows:

$$NLGK = (PGK)R'_m x^{-1}_{K_{lj}}P = \left(\prod_{i=1, i\neq j}^{m} x_{K_{li}}R'_m\right)P. \tag{17}$$

(iii) Also M_l computes the new key as follows:

$$NLGK = (PGK)R'_m x^{-1}_{K_{lj}}P = \left(\prod_{i=1, i\neq j}^{m} x_{K_{li}}R'_m\right)P. \tag{18}$$

4. Security analysis

In this section we show that the the proposed protocols, being ECDLP based using *PP-PKI* and contributory in nature, have all the security features which are inherent with them such as Key secrecy, Forward secrecy, Backward secrecy, and Defense against MITM attack. Further using *PP-PKI* one can easly provide such privacy preserving attributes as anonymity, traceability, and unlinkability.

Theorem 1. *The group key derived using ACGKA PROTOCOL is authenticated and indistinguishable in polynomial time from random numbers.*

Proof. Each of the ECDH-two-party shared key generated in the round 1 of ACGKA is secure, because it uses an ECDH protocol with PKI that supports ECDLP assumption. That is, all the two-party shared keys exchanged in round 1 are authenticated and indistinguishable from random numbers in polynomial time.

Also the generation of the public keys made in round 2 follows similar procedure as used in the basic ECDH protocol, because instead of multiplying P by a single secret value, a product of multiple values is used. Note that this product is also secured as it is obtained by multiplying the secured shared keys generated in round 1. Also, each group member again follows the same ECDH protocol to multiply public key by its own shared key. Since the ACGKA imitates the basic ECDH that supports generalized ECDLP (Boneh 1998) in round 2, the group key generated by ACGKA is authenticated and indistinguishable from random numbers in polynomial time, and thus secured. □

Theorem 2. *The join protocol of DACGKA satisfies the properties of backward security.*

Proof. (i) As soon as a new member M_{m+1} wants to join the group, intimates the group controller and generates a ECDH-style key $x_{K_{lm+1}}$ with GC.

(ii) GC generates a random number R'_{m+1} and broadcasts $[x_{K_{lm+1}}.R'_{m+1}]P$ to all the previous members of the group, M_i. On receiving they compute the new key as follows:

$$NJGK = (PGK)x_{K_{lm+1}}R'_{m+1}P = \left(\prod_{i=1}^{m+1} x_{K_{li}}.R'_{m+1}\right)P \tag{19}$$

(iii) GC transmits $[(P.G.K)R'_{m+1}]P$ to M_{m+1}. After receiving M_{m+1} computes the new key as follows:

$$NJGK = (PGK)x_{K_{lm+1}}R'_{m+1}P = \left(\prod_{i=1}^{m+1} x_{K_{li}}R'_{m+1}\right)P. \tag{20}$$

On basis of ECDLP, it is hard for an out-sider and new group members to compute previous group key. □

Theorem 3. *The leave protocol of DACGKA satisfies the properties of the forward security.*

Proof. (i) As soon as M_j wants to leave the group, it intimates the GC. Then GC, M_l generates a random number R'_m.

(ii) M_l sends $[R'_m.x_{lj}^{-1}]P$ by encrypting with $x_{K_{li}}$ to the corresponding group member M_i, $i \neq j$, (i.e.) except leaving member.

$$M_l \overset{E_{K_{li}}\left[R'_m x_{K_{lj}}^{-1}\right]P}{\rightarrow} M_i,\ 1 \leq i \leq m,\ i \neq j. \tag{21}$$

After receiving each member by M_i decrypts with $x_{K_{li}}$ and computes the new key as follows:

$$NLGK = (PGK)R'_m x_{K_{lj}}^{-1}P = \left(\prod_{i=1,i\neq j}^{m} x_{K_{li}}R'_m\right)P. \tag{22}$$

(iii) Also M_l computes the new key as follows:

$$NLGK = (PGK)R'_m x_{K_{lj}}^{-1} P = \left(\prod_{i=1, i\neq j}^{m} x_{K_{li}} R'_m \right) P. \tag{23}$$

As $[R'_m.x_{lj}^{-1}]P$ is in encrypted form, it is secured from outsiders and also GC keeps it secure from leaving member. So we have the main security requirement of member leaving is satisfied with respect to both outsiders and former group members. □

Theorem 4. *The proposed protocol DACGKA is as much secured as ECDH with PKI against (a) Man in the middle attacks and (b) Unknown key-share security attacks.*

Proof. (a) *Defense against MITM attack:* First the group members M_i, $1 \leq i \leq m$ compute their public keys from their own private keys and publish them in PKI. Next the GC and each of the remaining group member knowing their own private keys, look for the public key of each other respectively in PKI and establish $m - 1$ ECDH two-party keys together with authenticated communication links between GC, M_l and each of the remaining group members M_i, $i \neq l$, $1 \leq i \leq m$. Since the proposed protocol DACGKA uses ECDH with PKI and also published public keys in PKI cannot be forged or replaced, this protocol is as much immune to MITM attacks as ECDH with PKI itself.

(b) *Unknown key-share security:* For any three members of the group M_i, M_j, and M_k, $i \neq j \neq k$. Since Diffie–Hellman with PKI (Kaufman *et al* 2002) binds public keys with respective user identities by means of a Certificate Authority (CA), it should be impossible to coerce M_i into thinking he is sharing a key with M_j, when he is actually sharing a key with another (honest) user M_k. That is, it should not be possible for M_i to believe he is sharing a key with $M_j M_k$, while M_k correctly thinks the key is shared with M_i. □

5. Comparative analysis

In this section, we first make a comparative analysis of the proposed protocol with popular DLP-based group key agreement protocols and then with ECDLP-based group key agreement protocols, becausethe focus of this paper is on a search for an efficient group key agreement protocol suitable to wireless sensor networks and the proposed protocol being a ECDH-based dynamic authenticated contributory group key agreement protocol, a comparison is made with such protocols with respect to the number of rounds, number of messages exchanged, operations etc..

Tables 3 and table 4 show the comparable key sizes of an ECDLP-based scheme and DLP-based scheme (Amir *et al* 2002). This shows that the same desired security level can be attained with significantly smaller keys in ECDLP-based schemes when compared to DLP-based ones. In view of the advantages and adaptability of ECDLP for ad-hoc networks over DLP, in this paper an ECDLP-based group key agreement protocol DACGKA is proposed.

5.1 *Computation and communication complexities*

5.1a *Description for values in table 5 of DACGKA protocol*:

- *Total number of messages:* For "m" group members to initialize the group key in first round $(m - 1)$ ECDH shared keys are generated by picking up intended members public key from

Table 3. Key sizes.

ECDLP-based scheme (size of n in bits)	*DLP-based scheme* (modular size in bits)
112	512
160	1024
224	2048
256	3072
384	7680
512	15360

Table 4. Computation cost.

Security level (bits)	Computation cost ratio of DH:ECDH scheme
112	6:1
128	10:1
192	32:1
256	64:1

PKI. So no message isrequired for exchanging public keys. In the second round, GC sends $(m-1)$ generated public numbers to corresponding $(m-1)$ group members using $(m-1)$ unicast messages. So the total number of messages required is $(m-1)$ only.

- *Sequential Scalar multiplications:* In the first round, group members use one sequential scalar multiplication for finding their public keys and GC uses $(m-1)$ sequential scalar multiplications to generate $(m-1)$ shared keys. In the second round, GC uses m scalar multiplications to generate the group key and hence a total of $2m$ sequential scalar multiplications are required to generate the group key.
- It is clear from *join protocol* that it uses one unicast and one broadcast message and 4 scalar multiplications for updating the key.
- It is clear from *leave protocol* that it uses $m-2$ unicast messages and 3 scalar multiplications for updating the key.
- Provides authentication along with such privacy preserving attributes as anonimity, traceability and unlinkability.
- *Number of rounds:* The proposed protocol DACGKA consists of only two rounds for initialization of group key, one round for join group key and one round for leave group key. Further, it uses the least number of rounds when compared to other protocols.
- *Number of messages:* For initialization of group key in DACGKA requires $m-1$ messages which is less than both TGECDH and GECDH. The join protocol of DACGKA uses two (one unicast and one broad cast) messages on a member join. The leave protocol of DACGKA uses $m-2$ unicast messages on a member leave, making it optimal among the dynamic protocols discussed in this paper.
- *Storage cost:* Storage cost is proportional to the memory required to store the keys at group member nodes. In tree-based approaches each node has to keep the keys at its leaf nodes and so on which requires much storage cost. Also DH-based protocols require much storage

Table 5. Comparative analysis of popular group key agreement protocols.

DLP-protocols		Communication				Computation	Authentication	
Protocol		Rounds	Messages	Unicast	Broadcast	Sequential exponentiation	Sequential signatures	Verifications
EGK	Initialize	h	$2m-2$	0	$2m-2$	$2h-2$	h	$2m-2$
(Alves-Foss	Join	1	2	0	2	1	1	2
2000)	Leave	h	$2(m-1)$	0	$2(m-1)$	$2h$	1	$2(m-1)$
TGDH	Initialize	h	$2m-2$	0	$2m-2$	$2h-2$	h	$2(m-1)$
Kim *et al*	Join	2	3	0	3	$3h-3$	2	3
(2004b)	Leave	1	1	0	1	$3h-3$	1	1
STR	Initialize	$m-1$	$2m-2$	0	$2m-2$	$2(m-1)$	$m-1$	$2(m-1)$
(Kim *et al*	Join	2	3	0	3	4	2	3
2004a)	Leave	1	1	0	1	$m-1$	1	1
GDH.3	Initialize	$m+1$	$2m-1$	$2m-3$	2	$5m-6$	Not provided	
(Steiner *et al*	Join	4	$m+3$	0	$m+3$	$m+3$		
1996)	Leave	1	1	0	1	$m-1$		
Dutt's	Initialize	$2m$	$m+1$	0	$m+1$	$3m$	2	$m+1$
(Dutta &	Join	2	$2m-1$	0	$2m-1$	3	2	5
Barua 2008)	Leave	2	m	0	m	3	2	$m-1$

ECDLP-protocols		Communication				Computation	Authentication	
Protocols		Rounds	Messages	Unicast	Broadcast	Sequential Scalar multiplications	Sequential signatures	Verifications
GECDH	Initialize	m	m	$m-2$	2	$5m-6$	Not Provided	
(Wang *et al*	Join	m	m	0	m	$m+3$		
2006)	Leave	$m-1$	$m-1$	0	$m-1$	$m-1$		
TGECDH	Initialize	h	$2m-2$	0	$2m-2$	$2h-2$	Not Provided	
(Wang *et al*	Join	2	3	0	3	$3h-3$		
2006)	Leave	1	1	0	1	$3h-3$		
DACGKA	Initialize	2	$m-1$	$m-1$	0	2m	Provided using PP-PKI	
(proposed	Join	1	2	1	1	4		
protocol)	Leave	1	$m-2$	$m-2$	0	3		

cost with their long key sizes and hence the storage cost for them is high. Being a non-tree and ECDH-based one, the proposed protocol is optimal with respect to storage cost.

In view of the above comparative analysis in table 5, the proposed protocol DACGKA is optimal among DLP- and ECDLP-based schemes discussed in this paper with respect to communication and computation costs and also it provides the same security level with smaller key sizes, making it more suitable to resource-constrained networks such as MANETS and WSN.

5.2 *Computational and communication complexities using graphs*

In this section the main focus was to compare the proposed protocol with the other ECDLP-based ones using graphs. However, in table 3 a comparative analysis between ECDLP- and

DLP-based ones with respect to key sizes is made. Since a desired security level can be attained with significantly smaller keys in ECDLP-based schemes than the same sized ones (keys) in DLP-based schemes, our focus was on ECDLP-based ones because of their advantages in ad-hoc networks.

5.2a *Initialization of group key*: *Computational complexity* – As shown in figure 1, the number of sequential scalar multiplications required for initialization of group key of DACGKA is lesser than GECDH. Although it uses more number of sequential scalar multiplications than TGECDH

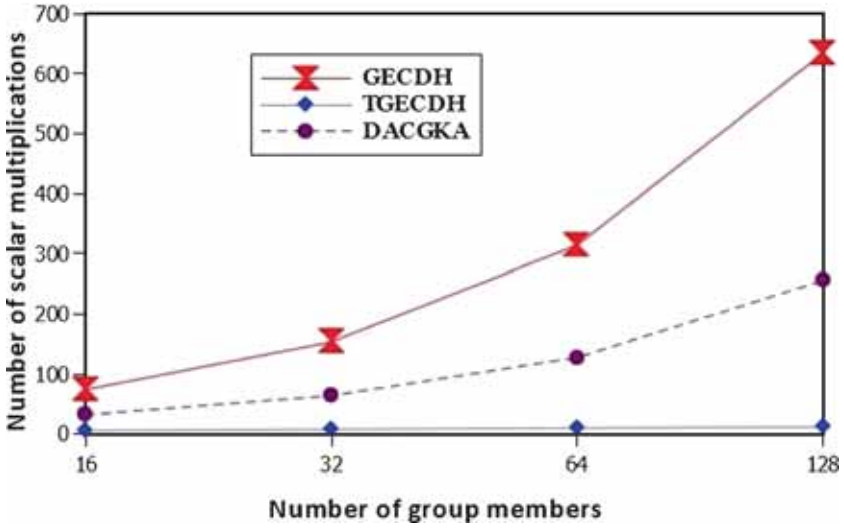

Figure 1. Comparative analysis on computation of ECDLP-based protocols for initial group key generation.

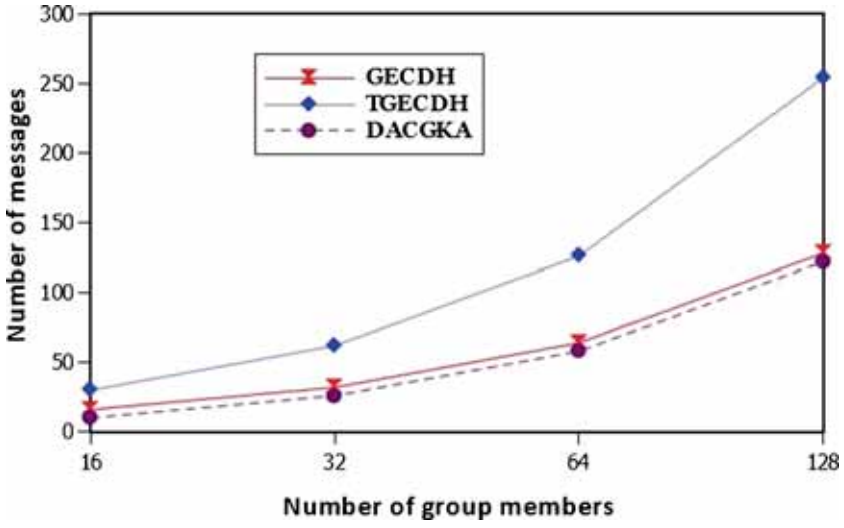

Figure 2. Comparative analysis on communication of ECDLP-based protocols for initial group key generation.

as in table 5. The proposed protocol is non-tree based, much simpler comprising only two rounds with simple operations.

Communication complexity – As shown in figure 2, for initialization of group key of DACGKA requires $m-1$ unicast messages which is lesser than the messages required for TGECDH and GECDH.

5.2b *Join protocol*: *Computational complexity* – As shown in figure 3, the number of sequential scalar multiplications required for new member join group key of DACGKA is fewer than GECDH and TGECDH protocol, in fact it requires only four scalar multiplications independent of group size.

Communication complexity – As shown in figure 4, the join protocol of DACGKA uses two (one unicast and one broad cast) messages on new member join.

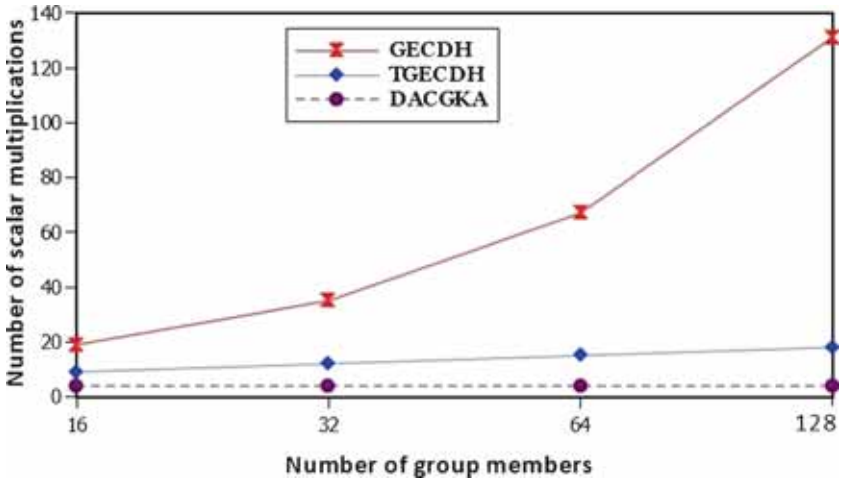

Figure 3. Comparative analysis on computation of ECDLP-based protocols for member join group key.

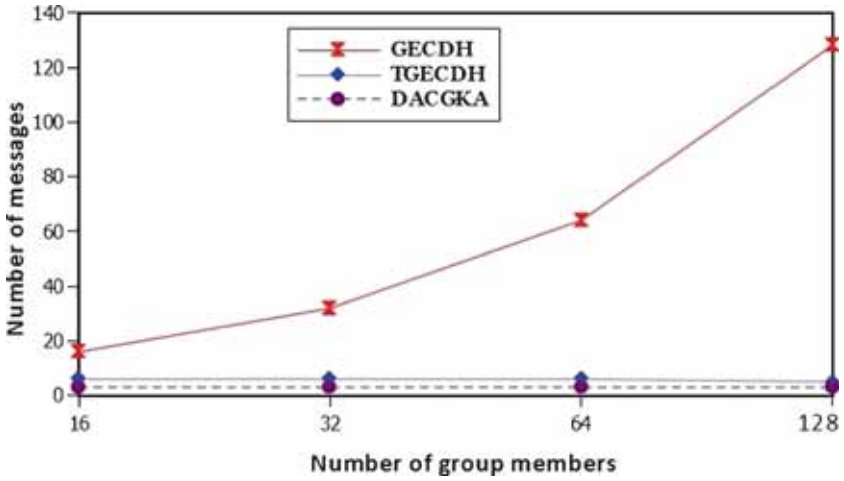

Figure 4. Comparative analysis on communication of ECDLP-based protocols for member join group key.

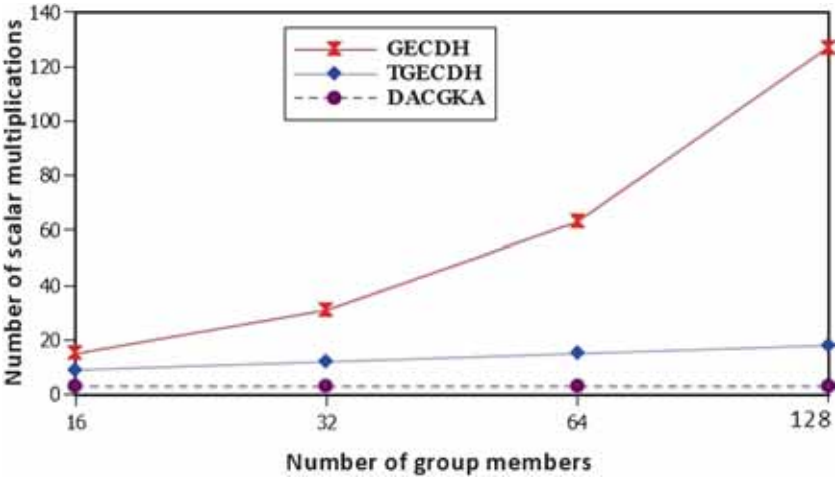

Figure 5. Comparative analysis on computation of ECDLP-based protocols for member leave group key.

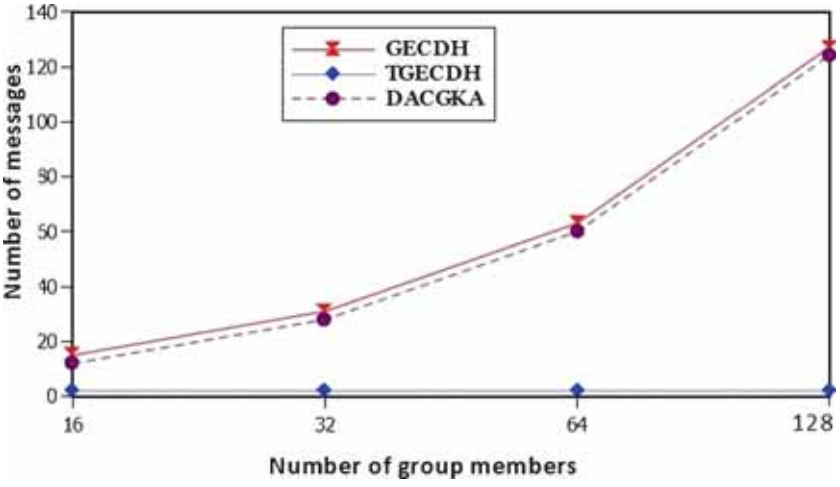

Figure 6. Comparative analysis on communication of ECDLP-based protocols for member leave group key.

5.2c *Leave protocol*: *Computational complexity* – As shown in figure 5, the number of sequential scalar multiplications required for member leave group key using leave protocol of DACGKA is three scalar multiplications independent of group size.

Communication complexity – As shown in figure 6, the number of messages required for member leave group key using leave protocol of DACGKA is $m - 2$ unicast. Although higher than TGECDH, the proposed protocol does not require broadcast messages.

6. Conclusions and future work

In this paper, we proposed an ECDLP-based dynamic authenticated contributory group key agreement protocol using Elliptic Curve Diffie–Hellman with *PP-PKI* for SGC over data

networks. Since it is ECDLP based using *PP-PKI* and contributory in nature, it has all the advantages which are inherent with them such as (a) offering relatively low communication overheads and low computational loads, (b) consumption of relatively lower memory storages, (c) offering authentication along with such privacy preserving attributes as anonimity, traceability and unlinkability, and providing security against active adversary, (d) distribution of computational and communicational loads among all the members (Although, the ad-hoc group controller has more scalar multiplications to do, the proposed protocol being ECDLP-based one, additions and scalar multiplications are used instead of multiplications and exponentiation as in DLP-based protocols, making it less expensive and also more suitable to resource constraint ad-hoc/sensor networks especially in view of its smaller key sizes and the recent advances in the computational prowess. In fact, (Kazuo *et al* 2007) proposes P-MALU, Parallelized Modular Arithmetic Unit, with which one can achieve over 80,000 scalar multiplications per second with predictably improved performance.). Further, it has the added advantage of dynamically updating the group key *without* a rerun of the total/entire DACGKA protocol anew, as soon as a member joins or leaves the existing group.

The proposed protocol provides authentication of the participants using ECDH with Public Key Infrastructure, which may be difficult in certain environments. It may be possible to provide authentication using ECDH integrated signature scheme for group key agreement, with overall reduced computational and communicational loads. Also such security issues as, perfect forward secrecy, replay attack, forgery attack, key compromise impersonation, key control, etc. are yet to be studied for the proposed protocol.

Acknowledgments

The authors are extremely grateful to the anonymous referee(s) for their many helpful comments. The authors are thankful to the University Grants Commission (UGC), Government of India, for financial assistance [UGC Reference No. F MRP-4519/14 (SERO/UGC)].

References

Alves-Foss J 2000 An efficient secure group key exchange algorithm for large and dynamic groups. In: *Proceedings of 23rd National Information Systems Security Conference*, pp. 254–266

Amir Y, Kim Y, Nita-Rotaru C and Tsudik G 2002 On the performance of group key agreement protocols In: *Proceedings of the 22nd IEEE International Conference on Distributed Computing Systems*, Vienna, Austria

Batina L, Mentens N, Sakiyama K, Preneel B and Verbauwhede I 2006 Low-cost elliptic curve cryptography for wireless sensor networks. In: *Proceedings of Third European Workshop on Security and Privacy in Ad Hoc and Sensor Networks*, Lecture notes in computer science. Springer, vol. 4357, pp. 415–429

Becker K and Wille U 1998 Communication complexity of group key distribution. In: *5th Conference on Computer and Communication Security* pp. 1–6

Benjumea V, López J, Montenegro J A and Troya J M 2004 A first approach to provide anonymity in attribute certificates. In: Bao F, Deng R H and Zhou J (eds.), PKC 2004, 2004 *International Workshop on Practice and Theory in Public Key Cryptography.* Lecture notes in computer science, Springer-Verlag, vol. 2947, pp. 402–415

Biswas G P 2008 Diffie-Hellman technique extended to multiple two party keys and one multi party key. *IET Information Security* 2(1): 12–18

Boneh D 1998 The decision Diffie-Hellman problem. In: *Proceedings of the Third Algorithmic Number Theory Symposium*. Lecture notes in computer science, Springer, vol. 1423, pp. 48–63

Bundesamt fuer Sicherheit in der Informationstechnik (BSI) 2009 Technical Guideline TR-03111 – Elliptic Curve Cryptography, Version 1.11

Burmester M and Desmedt Y 1994 A secure and efficient conference key distribution system. *Advances in cryptography – EUROCRYPT'94*, pp. 275–286

Chaum D and van Heyst E 1991 Gropu signatures. In: *Advances in Cryptology-EUROCRYPT.* Lecture notes in computer science, vol. 547, pp. 257–265

Diffie W and Hellman M E 1976 New directions in cryptography. *IEEE Trans. Inf. Theory* 22: 644–654

Dutta R and Barua R 2005 Dynamic group key agreement in tree-based setting. In: *Proceedings of ACISP 2005*. Lecture notes in computer science, vol. 3574, pp. 101–112

Dutta R and Barua R 2008 Provably secure constant round contributory group key agreement in dynamic setting. *IEEE Trans. Inf. Theory (TIT)* 54(5): 2007–2025

Hagzan K S and Bischof H P 2004 The performance of group Diffie-Hellman paradigms. In: *International Conference on Wireless Networks*, Las Vegas, Nevada, USA

Ingemarsson I, Tang D and Wong C 1982 A conference key distribution system. *IEEE Trans. Inf. Theory* 28(5): 714–720

Kaufman C, Perlman R and Speciner M 2002 Network security: private communication in public. Second edition, *Prentice Hall*, ISBN 0130460192

Kazuo S, Lejla B, Bart P and Ingrid V 2007 High-performance public-key crypto processor for wireless mobile applications. *Mobile Netw. Appl.* 12(4): 245–258

Kim Y, Perrig A and Tsudik G 2004a Group key agreement efficient in communication. *IEEE Trans. Comput.* 53(7): 905–921

Kim Y, Perrig A and Tsudik G 2004b Tree-based group key agreement. *ACM Trans. Inf. Syst. Security* 7(1): 60–96

Koeblitz N 1987 Elliptic curve cryptosystems. *Math. Comput.* 48: 203–209

Kwon T, Cheon J H, Kim Y and Lee J 2006 Privacy protection in PKIs: A separation-of- authority approach. *WISA 2006.* Lecture notes in computer science, vol. 4298, pp. 297–311

Lee S, Kwon H C and Seo D (eds) 2011 Privacy-preserving PKI design based on group signature. Ninth Australian Information Security Management Conference, Perth, Australia

Malan D, Welsh M and Smith M D 2004 A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography. In: *Proceedings of 1st IEEE International Conference on Sensor and Ad-Hoc Communications and Networks* (*SECON*), Santa Clara, CA

Miller V S 1986 Use of elliptic curves in cryptography. *Crypto'85*. Lecture notes in computer science, Springer Verlag, vol. 218, pp. 417–426

Nistala V E S Murthy and Vankamamidi S Naresh 2010 Diffie-Hellman technique extended to efficient and simpler group key distribution protocol. *Int. J. Comput. Appl.* 4(11): 1–5

Steiner M, Tsudik G and Waidner M 1996 Diffie-Hellman key distribution extended to groups. In: *3rd ACM Conference on Computer and Communication Security,* ACM Press, 1996, pp. 31–37

Wang Y, Ramamurthy B and Zou X 2006 The performance of elliptic curve based group Diffie-Hellman protocols for secure group communication over ad-hoc networks. In: *IEEE International Conference on Communications ICC '06*, pp. 2243–2248

Wong C, Gouda M and Lam S 1998 Secure group communication using key graphs. In: *Proceedings of ACM SIGCOMM'98 Conference* on *Applications, Techniques, Architectures and Protocols for Computer Communication,* pp. 68–99

Zheng S, Manz D and Alves-foss J 2007 A communication-computation efficient group key algorithm for large and dynamic groups. *Comput. Netw.* 51(1): 69–93