Elliptic Curve Based Dynamic Contributory Group Key Agreement Protocol for Secure Group Communication over Ad-hoc Networks

Vankamamidi Srinivasa Naresh[1] and Nistala V.E.S. Murthy[2]
(Corresponding author: Vankamamidi Srinivasa Naresh)

Computer Science Department, S.V.K.P. and Dr. K.S. Raju, Arts and Science College[1]
Penugonda 534320, Andhrapradesh, India.
Computer Science and System Engineering Department, Andhra University[2]
Visakhapatnam 530003, Andhra Pradesh, India
(Email: vsnaresh111@gmail.com)

(Received Feb. 6, 2013; revised and accepted Jan. 3 & Feb. 6, 2014)

Abstract

The aim of this paper is to propose an efficient and simpler Contributory Group Key Agreement protocol (CGKA) based on Elliptic Curve Diffie Hellman (ECDH). In this CGKA protocol, a member acts as a group controller (GC) and forms two-party groups with remaining group members and generates an ECDH-style shared key per each two-party group. It then combines these keys into a single group key and acts as a normal group member. This paper also addresses a Dynamic Contributory Group Key Agreement protocol (DCGKA) by extending CGKA to dynamic groups. The proposed protocol has been compared with other popular DH and ECDH based group key distribution protocols and satisfactory results were obtained.

Keywords: Dynamic group key agreement, elliptic curve Diffie-Hellman, mobile ad-hoc networks (MANETS), secure group communication (SGC)

1 Introduction

Wireless networks are growing rapidly in the last few years and also secure and reliable communication is an increasingly active research area with growing popularity in group oriented and collaborative applications. In the light of advances in Mobile ad-hoc networks the need for mechanism of secure group communications is growing day by day. Providing SGC over ad-hoc mobile networks is a very difficult task because they are mostly without much infrastructure. This problem was overcome by using elliptic curve crypto (ECC) systems. ECC emerged as the cryptographic choice for ad-hoc networks and communication devices because it can provide high security with very smaller key sizes and also at low computational expenses. Recent studies [11] indicate that the execution of ECC operations in mobile ad-hoc networks is feasible with predictable improved performance.

Secure Group Communication (SGC) 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 glean any information even when they are able to intercept the messages. The vast majority of SGC protocols use the (Discrete Logarithm Problem) based or DLP-based Diffie-Hellman as the basic key agreement protocol [16]. Any DLP-based Diffie-Hellman key agreement protocol now-a-days depends on the discrete logarithm problem for its security. The key length for secure DLP-based Diffie-Hellman has increased over recent years, which has simultaneously placed a heavier processing load on applications using DLP-based Diffie-Hellman. However, the processing load is especially critical for ad-hoc networks, which have a relatively limited bandwidth, slower CPU speed, limited battery power and high bit-error rate wireless links.

Elliptic Curve Cryptography (ECC) is a public key cryptosystems based on elliptic curves [10, 12]. The attraction of ECC is that it appears to offer equal security for a far smaller key size, thereby reducing processing overheads. However, the methods for computing general elliptic curve discrete logarithms are much less efficient than those for factoring or computing conventional discrete logarithms, indicating that more computational time is required for ECC. Thus, the overall performance of ECDLP based applications need to be evaluated.

The recent works on performance evaluation of group Diffie-Hellman protocols can be found in [2] and [6]. In [2], the authors evaluated five notable group key agreement protocols: Centralized Group Key Distribution (CKD), Burmeister Desmedt (BD), Steer et al. (STR), etc. The

Group Diffie-Hellman (GDH) key distribution protocols were first presented in [14]. There are three different versions. GDH.2 involves fewer number of rounds and messages than GDH.1 and GDH.3. The GDH protocol consists of two stages: up flow and down flow. The up flow stage collects contributions from all group members. The down flow stage broadcasts the intermediate values to all group members for calculating the shared group key. The Group Elliptic Curve Diffie-Hellman (GECDH) protocol and Tree-based Group Elliptic Curve Diffie-Hellman Protocol based on ECDLP are analyzed in [15]. Also studied Group-DH technique for Multi Party Key in [4, 13]. However, few studies have been conducted in literature on the performance of DLP and ECDLP-based group Diffie-Hellman protocols.

All the group key generating techniques can be divided into two classes. In one class, a single member of the group generates the key [5, 17] and distributes it to remaining member. However, it requires a trusted key generator for reliability. In the other class there is a contributory key agreement [1, 5, 7, 9, 14, 17, 18], in which each member of the group contributes a share to generate the group key. This class provides key secrecy. In order to assure the secrecy of communication nodes of the network, the group usually computes the group key dynamically in the sense that the group key will be updated whenever a node joins or leaves the group.

In this paper, in the second part, we propose and evaluate the performance of ECDH-based dynamic contributory group key agreement protocol over ad-hoc networks with the following secure attributes:

- Key Secrecy: The key can computed only by the members of the group.
- Forward Secrecy: As soon as a member leaves the group, it is hard to compute the new key with the previous knowledge of the old key.
- Backward Secrecy: As soon as a new member joins the group, it is hard to compute old key with the knowledge of the new key.

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

2.1 List of Some Common Abbreviations and Notations

Table 1 is the abbreviations used in this paper. Table 2 is the notations used in this paper.

Table 1: Abbreviations

Abbreviations	Full Name
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman
ECDLP	Elliptic Curve Discrete Log Problem
ECDP	Elliptic Curve Domain Parameters
GC	Group Controller
CGKA	Contributory Group Key Agreement
DCGKA	Dynamic Contributory Group Key Agreement
GK	Group Key
NJGK	New Join Group Key
NLGK	New Leave Group Key

2.2 Background of Elliptic Curve Group

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

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

and with the discriminant

$$\Delta = 4a^3 + 27b^2 \neq 0.$$

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

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

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 $E(F_p)$ can be represented as follows. The k th multiple of a point P belongs to $E(F_p)$ computed as follows:

$$[k]P = P + P + ... + (ktimes).$$

Note: For integers j and k, we have

$$[j]([k]P) = [jk]P = [k]([j]P).$$

2.2.1 Elliptic Curve Domain Parameters (ECDP)

$ECDP(p, a, b, P, n, h)$ a set of information for communicating members 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 $h = \#E(F_p)/n$, where $\#E(F_p)$ is the number of points on an elliptic curve group. The base point P generates a cyclic group of order n. In other words, $E(F_p) = \langle P \rangle = \{P, [2]P, ..., [n-1]P, [n]P\}$.

2.2.2 Elliptic Curve Discrete Logarithm Problem (ECDLP)

Given the ECDP as described above and $Q \in < P > = E(F_p)$, ECDLP is to find an integer l, $0 \leq l \leq n-1$ such that $Q = [l]P$.

Table 2: 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,y_P	The x and y coordinates of point P respectively
P	The Base Point is a generator of a sub group of E(F_p)
n	The order of base point P typically, n is a prime of bit length ≥ 224
m	Total number of members in the group
M_i	ith group member, $1 \leq i \leq m$
M_l	The group controller
x_i	The Private Key of member M_i. This is an integer belongs to $\{1, 2, ...n-1\}$
X_i	The Public key of member M_i
$x_{K_{li}}$	ECDH shared key between GC and M_i, for $1 \leq i \leq m$, $i \neq l$

2.2.3 Cryptographically Strong Elliptic Curve Domain Parameters over F_p

The ECDLP is currently considered to be intractable if at least the following condition holds.

- The Order n of the base points 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 $p^r \neq 1$ mod n, for all $1 \leq r \leq 10^4$.
- The class number of the principle order belongs to the endomorphism ring of E should be at least 200.

2.2.4 Elliptic Curve Diffie-Hellman

Elliptic Curve Diffie-Hellman protocol (ECDH) is one of the key exchange protocols used to establish a shared key between two members. ECDH protocol is based on the additive elliptic curve group. First A and B agree on elliptic curve domain parameters and proceed as Table 3.

The secret key K is a point on 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 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 and thus the session key may have a different length with the private key.

3 Proposed Protocols

3.1 Contributory Group Key Agreement Protocol (CGKA)

We propose a contributory group key agreement protocol 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) 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 $1 \leq l \leq m$.

Step 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:

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

$$X_l = [x_l]P.$$

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

$$X_i = [x_i]P, for, 1 \leq i \leq m, i \neq l.$$

3) The GC M_l, broadcast, X_l to the remaining group members and each M_i transmits X_i to the group controller, M_l

4) After exchanging the public keys, each member generates a ECDH-style shared key with GC as

$$K_{li} = [x_i]X_l = [x_i]([x_l]P) = [x_i x_l]P = (x_{K_{li}}, y_{K_{li}}),$$

for $1 \leq i \leq m, i \neq l$. Where $x_{K_{li}}, y_{K_{li}} \in F_p$ are x and y coordinates of K_{li}, respectively.

Similarly GC, M_l generates the same shared keys as

$$K_{li} = [x_l]X_i = [x_l]([x_i]P) = [x_l x_i]P = (x_{K_{li}}, y_{K_{li}}),$$

for $1 \leq i \leq m, i \neq l$. Where $x_{K_{li}}$, $y_{K_{li}} \in F_p$ are x and y coordinates of K_{li}, respectively.

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

Table 3: ECDH

Party-A	Communication	Party-B
Choose a random number $x \in \{1, 2, ...n-1\}$		Choose a random number $y \in \{1, 2, ...n-1\}$
Compute $[x]\,P$		Compute $[y]\,P$
Retrieve $[y]\,P$	$\overset{[x]P}{\rightarrow}$ $\overset{[y]P}{\leftarrow}$	Retrieve $[x]\,P$
Compute [x][y]P=[xy]P		Compute [y][x]P=[yx]P=[xy]P

Step 2. Now the group controller computes the $(m-1)$ public keys L_i as follows and send to M_i respectively.

$$L_i = [\prod_{j=1, j\neq i}^{m} x_{K_{li}}]P, for 1 \leq i \leq m, i \neq l, and\ j \neq l.$$

After retrieving L_i each member M_i of the group generates group key K as follows:

$$\begin{aligned} K &= [x_{K_{li}}]L_i \\ &= [x_{K_{li}}][\prod_{j=1, j\neq i}^{m} x_{K_{lj}}]P \\ &= [\prod_{i=1}^{m} x_{K_{li}}]P \\ &= (x_K, y_K)\,. \end{aligned}$$

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

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

Hence take x_K as group key among the group members.

3.2 Dynamic Contributory Group Key Agreement Protocol (DCGKA)

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

3.2.1 Join Protocol

The main security requirement of the member addition is the secrecy of the previous group key with respect to outsider and new group members.

1) When 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.

2) GC generates a random number $R^{'}_{m+1}$ and broadcasts $\left[x_{K_{lm+1}}.R_{m+1'}\right]P$ to all the previous members of the group, M_i on receiving they compute the new key:

$$NJGK = (x_K)x_{K_{lm+1}}R^{'}_{m+1}P = (\prod_{i=1}^{m+1} x_{K_{li}}.R^{'}_{m+1})P,$$

where x_K is the previous group key.

3) GC transmits $[(x_K)R^{'}_{m+1}]P$ to M_{m+1} and then M_{m+1} computes the new key as follows:

$$NJGK = (x_K)x_{K_{lm+1}}R^{'}_{m+1}P = (\prod_{i=1}^{m+1} x_{K_{li}}R^{'}_{m+1})P,$$

where x_K is the previous group key.

3.2.2 Leave Protocol

The main security requirement of member leaving is the secrecy of the subsequent (future) group key with respect to both outsiders and former group members.

1) When M_j wants to leave the group, intimates the GC and then GC, M_l generates a random number R'_j.

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

$$M_l \xrightarrow{E_{K_{li}}\left[R'_j x^{-1}_{K_{lj}}\right]P} M_i, for 1 \leq i \leq m, i \neq j.$$

After receiving each member computes the new key as follows:

$$NLGK = (x_K)R'_j x^{-1}_{K_{lj}}P = \left[\prod_{i=1, i\neq j}^{m} x_{K_{li}}R'_j\right]P,$$

where x_K is the previous group key.

3) Also M_l computes the new key as follows.

$$NLGK = (x_K)R'_j x^{-1}_{K_{lj}}P = \left[\prod_{i=1, i\neq j}^{m} x_{K_{li}}R'_j\right]P,$$

where x_K is the previous group key.

4 Security Analysis

We Prove that our protocols meet the desirable attributes under the assumption that the Elliptic Curve Discrete Logarithm Problem is secure.

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

Proof. If the m-group members execute CGKA protocol then they clearly share a group key K. During the computation of group key K, in Step 1 we have generated $(m-1)$, two-party ECDH style keys.

An adverser tries to extract the private keys x_i from unknown public keys $X_i = [x_i]P$, but this is an Elliptic Curve Discrete Problem and hence two-party ECDH-style keys generated in Step 1 are indistinguishable in polynomial time.

In Step 2 of CGKA, GC generates $(m-1)$ public keys L_i and sends to M_i, respectively. That is

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

An adversary tries to extract all the products

$$\prod_{j=1, j\neq i}^{m} x_{K_{li}}, for 1 \leq i \leq m.$$

From publicly known,

$$L_i = [\prod_{j=1, j\neq i}^{m} x_{K_{li}}]P.$$

But this is again an elliptic curve discrete logarithm problem. Therefore all the products

$$\prod_{j=1, j\neq i}^{m} x_{K_{li}}, for 1 \leq i \leq m,$$

are indistinguishable from random numbers in polynomial time and hence it is difficult to find $x_{K_{li}}$. □

Theorem 2. *DCGKA with join protocol satisfies the properties of backward security.*

Proof. The GC generates a random number $R^{'}_{m+1}$ as soon as a new member joins the network group and broadcasts GC generates a random number $R^{'}_{m+1}$ and broadcasts $\left[x_{K_{lm+1}}.R_{m+1'}\right]P$ to all the previous members of the group, M_i on receiving they compute the new key

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

where x_K is the previous group key.

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

Theorem 3. *DCGKA with leave protocol satisfies the properties of the forward security.*

Proof.

1) When M_j wants to leave the group, intimates the GC and then GC, M_l generates a random number R'_j.

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

$$M_l \overset{E_{K_{li}}\left[R'_j x^{-1}_{K_{lj}}\right]P}{\Longrightarrow} M_i, \quad \text{for} \quad 1 \leq i \leq m, i \neq j.$$

$$NLGK = (x_K)R'_j x^{-1}_{K_{lj}} P = \left[\prod_{i=1, i\neq j}^{m} x_{K_{li}} R'_j\right] P,$$

where x_K is the previous group key.

3) Also M_l computes

$$NLGK = (x_K)R'_j x^{-1}_{K_{lj}} P = \left[\prod_{i=1, i\neq j}^{m} x_{K_{li}} R'_j\right] P,$$

where x_K is the previous group key.

As $\left[R'_j x^{-1}_{K_{lj}}\right]P$ is in encrypted form it is secured from outsiders and also GC keeps it secure from leaving member, we have the main security requirement of member leaving are satisfied with respective both outsiders and former group members.

□

5 Comparative Analysis

In this section, the proposed ECDLP-based DCGKA protocol has been firstly compared with DLP based group key distribution protocols, and then with ECDLP based protocols in terms of number of rounds, messages, operations and so on.

Table 4 shows the comparable key sizes (Table 5) of the same security level for an ECDLP-based group scheme and DLP-based scheme. It shows that ECDLP-based schemes can use a much smaller key size than DLP-based group schemes.

The key length for secure DLP-based Diffie-Hellman has increased over recent years, which has also placed a heavier processing load on applications using DLP-based Diffie-Hellman. However, the processing load is especially critical for ad-hoc networks, which have a relatively limited bandwidth, slower CPU speed, limited battery power and high bit-error rate wireless links and ECDLP-based group schemes are having lower communication overheads and less computation load than DLP-based group scheme.

As per the advantages and adaptability for ad-hoc networks of ECDLP over DLP In this paper, we proposed ECDLP-based group key distribution protocol DCGK at

Table 4: Comparative analysis of popular group key agreement protocols

DLP-Protocols		Rounds	Messages	Unicast	Broadcast	Seq exponent ions	Seq scalar multiplications
CEGK [3]	Initialize	h	$2m-2$	m	$m-2$	$2h-2$	0
	Join	1	2	1	1	1	0
	Leave	1	1	0	1	$h-1$	0
EGK [1]	Initialize	h	$2m-2$	0	$2m-2$	$2h-2$	0
	Join	1	2	0	2	1	0
	Leave	h	$2(m-1)$	0	$2(m-1)$	$2h$	0
TGDH [9]	Initialize	h	$2m-2$	0	$2m-2$	$2h-2$	0
	Join	2	3	0	3	$3h-3$	0
	Leave	1	1	0	1	$3h-3$	0
STR [8]	Initialize	$m-1$	$2m-2$	0	$2m-2$	$2(m-1)$	0
	Join	2	3	0	3	4	0
	Leave	1	1	0	1	$m-1$	0
GDH.3 [14]	Initialize	$m+1$	$2m-1$	$2m-3$	2	$5m-6$	0
	Join	4	$m+3$	0	$m+3$	$m+3$	0
	Leave	1	1	0	1	m-1	0
ECDLP-based Protocol		Rounds	Messages	Unicast	Broadcast	Seq exponent ions	Seq scalar multiplications
GECDH [15]	Initialize	m	m	$m-2$	2	0	$5m-6$
	Join	m	n	0	m	0	$m+3$
	Leave	$m-1$	$m-1$	0	$m-1$	0	$m-1$
TGECDH [15]	Initialize	h	$2m-2$	0	$2m-2$	0	$2h-2$
	Join	2	3	0	3	0	$3h-3$
	Leave	1	1	0	1	0	$3h-3$
DCGKA [our protocol]	Initialize	$m+1$	$2m-1$	$2m-2$	1	0	$2m$
	Join	1	2	1	1	0	6
	Leave	1	1	0	1	0	3

Table 5: 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

the same security level as the DLP-based Diffie-Hellman schemes.

Our protocol uses only two steps which involve very simple operations. Being ECDLP-based protocol additions and scalar multiplications are used instead of multiplications and exponentiations (as in DLP-based protocols) respectively and also it uses smaller key sizes. Hence our protocol works with lesser computational expense. However, the group controller needs to execute comparatively more key exchange operations than the other group members, but these operations are very simple with lesser computational expense. The overall delay of key generation depends on the performance of group controller. Since most of today's machines have high computation power, the proposed technique may not be a problem for practical applications.

In view of above comparative analysis in Table 5, our protocol [DCGK] is optimal in terms of comparatively less communication and computation cost and also it provides same security level with smaller key sizes. Our protocol is relatively best protocol for secure group key distribution over ad-hoc networks among the DLP and ECDLP based schemes discussed in this paper.

Computational Complexities.

Initialization of group key. The number of sequential scalar multiplications for initialization of group key in our protocol [CGKA] is lesser than GECDH. Although our protocol uses much number of sequential scalar multiplications than TGECDH. Our protocol is much simpler comprising only two steps with very simple operations (See Figure 1).

Join protocol. The number of sequential scalar multiplications for new member join group key in our protocol is fewer than GECDH and TGECDH protocol, In fact only six scalar multiplications independent of group size (See Figure 2).

Leave protocol. The number of sequential scalar multiplications for new member leave group key in our protocol is fewer than GECDH and TGECDH protocol, In fact only three scalar multiplications independent of group size (See Figure 3).

Communication Complexities.

Number of messages. DCGKA protocol is the best in terms of communication for updating the group key whenever a new member joins or existing member leaves. For initialization of group key our protocol uses $2m - 1$ messages which is nearly same as TGECDH and higher than GECDH.

Storage cost. As per the memory to store the keys at member nodes, the ECC makes the process as easy as possible, since the key sizes are small with ECC. In tree based approaches each node has to maintain the keys of its leaf nodes and so on. So DCGKA consumes very low memory storage cost than tree based approaches.

In view of the above observations, DCGKA is optimal in terms of low communication and computation costs and also it provides same security level with smaller key sizes. Thus it is relatively a better protocol for secure group key distribution over ad-hoc networks among the DLP and ECDLP based schemes discussed in this paper.

6 Conclusion and Future Work

In this paper, we proposed ECDLP based Dynamic contributory Group key agreement (DCGKA) protocol for secure group communication over ad-hoc networks. The theoretical analysis shows that DCGKA is certainly a better protocol in overall performance among the DLP and ECDLP based schemes discussed in this paper. Also it provides secure key attributes such as key secrecy, forward secrecy and backward secrecy.

The performance of DCGKA over ad-hoc networks can be Summarized as follows:

- It has relatively low communication overheads and lesser computational expense.
- It consumes very low memory storage cost than the tree based approaches.
- Most importantly, it is quite simple to implement in the sense that it uses only two steps which involve very simple operations.
- It uses dynamic updating of key without a re-run of the protocol anew as soon as a member joins or leaves the existing group.
- It uses smaller keys.

Therefore it may be apt for secure group key agreements over mobile ad-hoc networks.

In continuation of this paper, there remain some items for future work. Our protocol do not provide authentication of the participants. It should be possible to argument them to provide authentication using public Key Infrastructure (PKI), with out increasing computational and communication load. Also to address most of the active attacks, such as key impersonation and forgery attack etc..

References

[1] J. Alves-Foss, "An efficient secure group key exchange algorithm for large and dynamic groups," in *Proceedings of the 23rd National Information Systems Security Conference*, pp. 254–266, Oct. 2000.

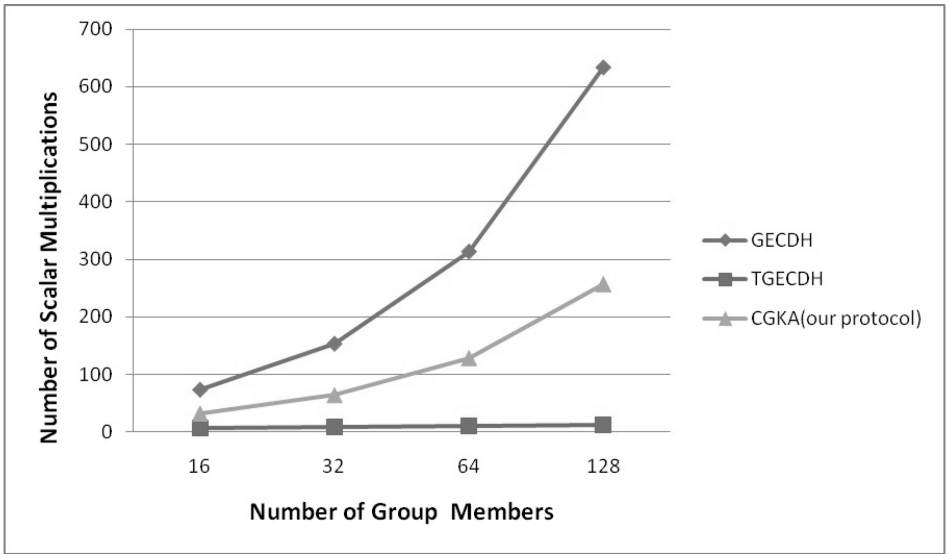

Figure 1: Comparative analysis of ECDLP based protocols for initialization of group key

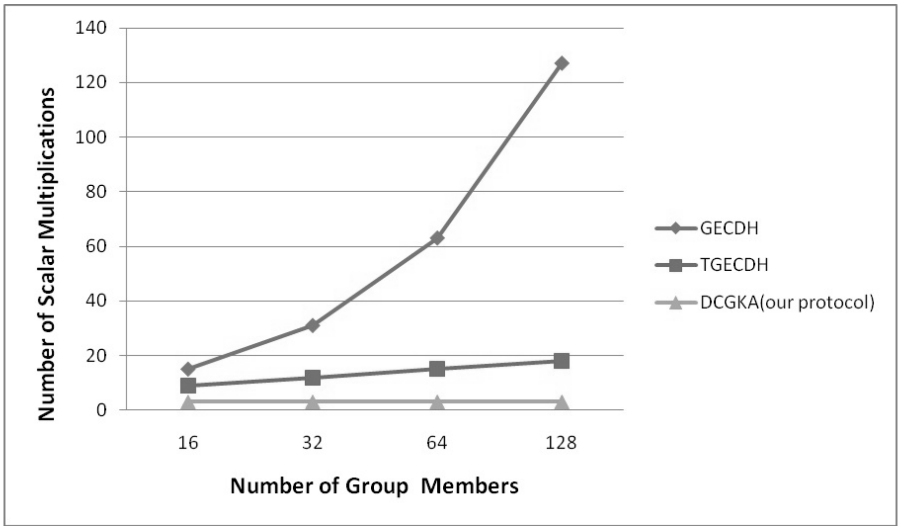

Figure 2: Comparative analysis of ECDLP based protocols for member join group key

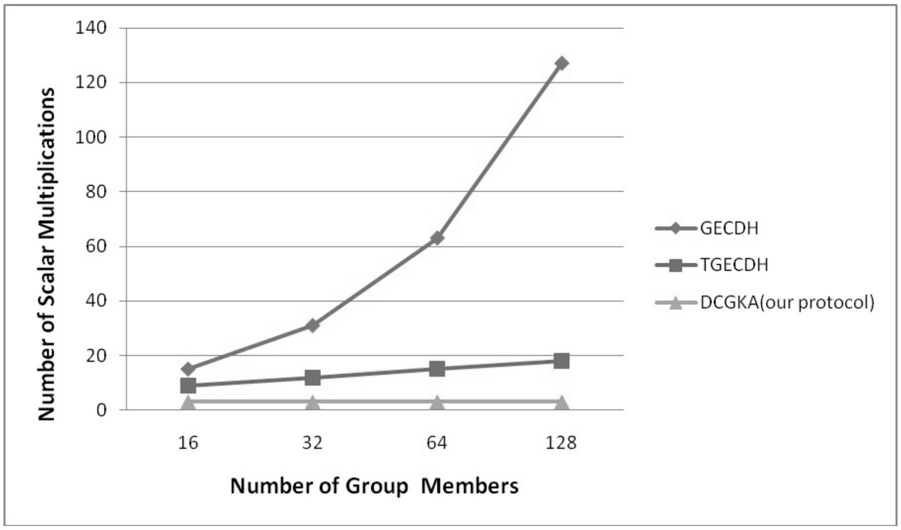

Figure 3: Comparative analysis of ECDLP based protocols for member leave group key

[2] Y. Amir, Y. Kim, C. Nita-Rotaru, and G. Tsudik, "On the performance of group key-agreement protocols," *ACM Transactions on Information and System Security*, vol. 7, no. 3, pp. 457–488, 2004.

[3] K. Becker and U. Wille, "Communication complexity of group key distribution," in *5th Conference on Computer and Communication Security*, pp. 1–6, 1998.

[4] G. P. Biswas, "Diffie hellman technique extended to multiple two party keys and one multi party key," *IET Information Security*, vol. 2, no. 1, pp. 12–18, 2008.

[5] M. Burmester and Y. Desmedt, "A secure and efficient conference key distribution system," in *Eurocrypt'94*, pp. 275–286, May 1994.

[6] K. S. Hagzan and H. P. Bischof, "The performance of group diffie-hellman paradigms," in *2004 International Conference on Wireless Networks*, Las Vegas, Nevada, USA, 2004.

[7] I. Ingemarsson, D. Tang, and C. Wong, "A conference key distribution system," *IEEE Transactions on Information Theory*, vol. 28, no. 5, pp. 714–720, 1982.

[8] Y. Kim, A. Perrig, and G. Tsudik, "Group key agreement efficient in communication," *IEEE Transactions on Computer*, vol. 53, no. 7, pp. 905–921, 2004.

[9] Y. Kim, A. Perrig, and G. Tsudik, "Tree-based group key agreement," *ACM Transactions on Information System Security*, vol. 7, no. 1, pp. 60–96, 2004.

[10] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203–209, 1987.

[11] D. Malan, M. Welsh, and M. D. Smith, "A public-key infrastructure for key distribution in tiny os based on elliptic curve cryptography," in *IEEE International Conference on Sensor and Ad Hoc Communications and Networks*, pp. 71–80, Oct. 2004.

[12] V. S. Miller, "Use of elliptic curves in cryptography," in *Crypto'85*, LNCS 218, pp. 417–426, Springer-Verlag, 1986.

[13] V. S. Naresh, N. V.E.S. Murthy, "Diffie-Hellman technique extended to efficiently and simpler group key distribution protocol," *International Journal of Computer Applications*, vol. 4, no. 11, pp. 1–5, Aug. 2010.

[14] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman key distribution extended to group communication," in *Proceedings of the 3rd ACM Conference on Computer and Communications Security*, pp. 31–37, New York, NY, USA, 1996.

[15] Y. Wang, B. Ramamurthy, and X. Zou, "The performance of elliptic curve based group Diffie-Hellman protocols for secure group communication over adhoc networks," in *IEEE International Conference on Communications*, pp. 2243–2248, 2006.

[16] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, pp. 644–654, Nov. 1976.

[17] C. Wong, M. Gouda, and S. Lam, "Secure group communication using key graphs," in *Proceeedings of the ACM SIGCOMM'98*, pp. 68–99, 1998.

[18] S. Zheng, D. Manz, and J. Alves-foss, "A communication-computation efficient group key algorithm for large and dynamic groups," *Computer Networks*, vol. 51, no. 1, pp. 69–93, 2007.

Vankamamidi Srinivasa Naresh is currently working as a Director, for the Post Graduate Department of Computer Science Courses in S.V.K.P. and Dr. K.S.R. Arts and Science College. He obtained an M.Sc. in Mathematics from Andhra University, an M.Phil. in Mathematics from Madurai Kamaraj University and an M.Tech in Computer Science and Engineering from J.N.T.University-Kakinada. He is also a recipient of U.GC.-C.S.I.R.JUNIOR RESEARCH FELLOSHIP and cleared NET for Lectureship in Mathematical sciences and also cleared UGC NET in Computer Science and Applications. He published papers in reputed journals in the area of cryptography. Presently pursuing Doctorate from JNTUK.

Nistala V.E.S. Murthy is currently working as a Professor in the department of Computer Science and Systems Engineering of Andhra University, Visakhapatnam. He developed f-Set Theory -wherein f-maps exists between Fuzzy Sets with truth values in *different* complete lattices, generalizing L-fuzzy set theory of Goguen which generalized the [0,1]-fuzzy set theory of Zadeh, the Father of Fuzzy Set Theories. He also published papers on, Representation of Various Fuzzy Mathematical (Sub) Structures in terms of their appropriate crisp cousins, Various Fuzzy Set Theories and Their Applications in Mathematics (Algebra and Topology) and Computer Science (Data Warehousing, Data Mining), Cryptography and Natural Language Modeling. He can be visited at URL: http://andhrauniversity.academia.edu/NistalaVESMurthy.