The Performance of Elliptic Curve Based Group Diffie-Hellman Protocols for Secure Group Communication over Ad Hoc Networks

Yong Wang
University of Nebraska-Lincoln, ywang@cse.unl.edu

Byrav Ramamurthy
University of Nebraska-Lincoln, bramamurthy2@unl.edu

Xukai Zou
Indiana University-Purdue University Indianapolis, xkzou@cs.iupui.edu

ICC '06. IEEE International Conference on Communications, 2006.
Volume: 5 Digital Object Identifier: 10.1109/ICC.2006.255104
Publication Year: 2006 , Page(s): 2243 - 2248

The Performance of Elliptic Curve Based Group Diffie-Hellman Protocols for Secure Group Communication over Ad Hoc Networks

Yong Wang, Byrav Ramamurthy
Department of Computer Science and Engineering
University of Nebraska-Lincoln
Lincoln, NE 68588 USA
Email: {ywang, byrav}@cse.unl.edu

Xukai Zou
Department of Computer and Information Science
Indiana University-Purdue University Indianapolis
Indianapolis, IN 46202 USA
Email: xkzou@cs.iupui.edu

***Abstract*— The security of the two party Diffie-Hellman key exchange protocol is currently based on the discrete logarithm problem (DLP). However, it can also be built upon the elliptic curve discrete logarithm problem (ECDLP). Most proposed secure group communication schemes employ the DLP-based Diffie-Hellman protocol. This paper proposes the ECDLP-based Diffie-Hellman protocols for secure group communication and evaluates their performance on wireless ad hoc networks. The proposed schemes are compared at the same security level with DLP-based group protocols under different channel conditions. Our experiments and analysis show that the Tree-based Group Elliptic Curve Diffie-Hellman (TGECDH) protocol is the best in overall performance for secure group communication among the four schemes discussed in the paper. Low communication overhead, relatively low computation load and short packets are the main reasons for the good performance of the TGECDH protocol.**

I. Introduction

Secure group communication (SGC) refers to a scenario in which a group of participants can send and receive messages to/from 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 DLP-based Diffie-Hellman as the basic key agreement protocol [1].

The DLP-based Diffie-Hellman key agreement protocol 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 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.

Elliptic Curve Cryptography (ECC) is a public key cryptosystem based on elliptic curves [2], [3]. The attraction of ECC is that it appears to offer equal security for a far smaller key size, thereby reducing processing overhead. However, the methods for computing general elliptic curve discrete logarithms are much less efficient than those for factoring or computing conventional discrete logarithms and it indicates that more computation time is required for ECC. Thus, the overall performance of ECDLP-based applications needs to be evaluated.

The recent work on performance evaluation of group Diffie-Hellman protocols can be found in [4] and [5]. In [4], the authors evaluated five notable group key agreement protocols: Centralized Group Key Distribution (CKD), Burmester Desmedt (BD), Steer et al. (STR), Group Diffie-Hellman (GDH) and Tree based Group Diffie-Hellman (TGDH) concluding that TGDH exhibits the best average performance in high-delay Wide Area Network (WAN). The study in [5] evaluated three Diffie-Hellman based shared key agreement protocols: Group Diffie-Hellman (GDH), Tree-based and Hypercubic Diffie-Hellman demonstrating that GDH is good in ad hoc networks containing less than 100 nodes and TGDH is attractive for larger networks. These papers provide an elegant evaluation of the performance of group Diffie-Hellman protocols. However, few studies have been conducted in literature on the performance of ECDLP-based group Diffie-Hellman protocols.

In this paper, we propose and evaluate the performance of ECDLP-based group Diffie-Hellman protocols for secure group communication under different channel conditions. 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 ECDLP-based group schemes. Section 4 analyzes the communication overhead of each protocol and Section 5 describes the experiments and results. Finally, Section 6 concludes the paper.

II. Background

The two party Diffie-Hellman algorithm was presented by Whitfield Diffie and Martin E. Hellman in 1976 [6]. The Diffie-Hellman algorithm depends on the difficulty of computing discrete logarithms. It assumes that all participants know a prime number p and a primitive root g of p $(g < p)$.

The Group Diffie-Hellman (GDH) key distribution protocols were first presented in [7]. There are three different versions and we consider GDH.2 in this paper which involves fewer number of rounds and messages than GDH.1 and GDH.3. The GDH protocol consists of two stages: upflow and downflow. The upflow stage collects contributions from all group members. The downflow stage broadcasts the intermediate values to all group members for calculating the shared group key. The

1-4244-0355-3/06/$20.00 (c) 2006 IEEE

the one i read already

authors in [8] introduced a tree based group Diffie-Hellman protocol which uses a binary tree to minimize the total number of two party exchanges when computing a group key.

A. Elliptic Curve Cryptography

An Elliptic Curve is usually defined over two finite fields: the prime finite field F_p containing p elements and the characteristic 2 finite field containing 2^m elements. This paper focuses on the prime finite field.

Let F_p be a prime finite field so that p is an odd prime number, and let $a, b \in F_p$ satisfy $4a^3 + 27b^2 \neq 0 (mod\ p)$, then an elliptic curve $E(F_p)$ over F_p defined by the parameters $a, b \in F_p$ consists of the set of solutions or points $P = (x, y)$ for $x, y \in F_p$ to the equation:

$$y^2 = x^3 + ax + b (mod\ p) \tag{1}$$

The equation $y^2 = x^3 + ax + b(mod\ p)$ is called the defining equation of $E(F_p)$. For a given point $P = (x_P, y_P)$, x_P is called the x-coordinate of P, and y_P is called the y-coordinate of P.

Cryptographic schemes based on ECC rely on scalar multiplication of elliptic curve points. Given an integer k and a point $P \in E(F_p)$, scalar multiplication is the process of adding P to itself k times. The result of this scalar multiplication is denoted $k \times P$ or kP. Scalar multiplication of elliptic curve points can be computed efficiently using the addition rule together with the double-and-add algorithm or one of its variants.

Consider the equation $Q = kP$, where $Q, P \in E_p(a, b)$ and $k < p$. It is relatively easy to calculate Q given k and P, but it is relatively hard to determine k given Q and P. This is called the discrete logarithm problem for elliptic curves.

Elliptic Curve domain parameters over F_p are a sextuple,

$$T = (p, a, b, G, n, h)$$

consisting of an integer p specifying the finite field F_p, two elements $a, b \in F_p$ specifying an elliptic curve $E(F_p)$ defined by equation 1, a base point $G = (x_G, y_G)$ on $E(F_p)$, a prime n which is the order of G, and an integer h which is the cofactor $h = \#E(F_p)/n$.

SEC2 [9] lists some defined elliptic curve domain parameters over F_p, which include 112-bit, 128-bit, 160-bit, 192-bit, 224-bit, 256-bit, 384-bit and 521-bit elliptic curve domain parameters. Our simulation is based on these parameters.

B. Two Party Elliptic Curve Diffie-Hellman Protocol

Similar to DLP-based Diffie-Hellman key exchange agreement, a key exchange between users A and B using Elliptic Curve Diffie-Hellman (ECDH) can be accomplished as follows:

1. A selects an integer n_A less than p, this is A's private key. A then generates a public key $P_A = n_A \times G$; the public key is a point in $E_p(a, b)$.
2. B similarly selects a private key n_B and computes a public key $P_B = n_B \times G$.
3. A generates the secret key $K = n_A \times P_B$. B generates the secret key $K = n_B \times P_A$.

The two calculations in step 3 produce the same result because $n_A \times P_B = n_A \times (n_B \times G) = n_B \times (n_A \times G) = n_B \times P_A$.

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 in 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.

III. Proposed ECDLP-based Schemes

In this section, we propose two ECDLP-based schemes for secure group communication.

A. Group Elliptic Curve Diffie-Hellman Protocol

The Group Elliptic Curve Diffie-Hellman (GECDH) protocol is an extension of GDH based on ECDLP. GECDH can also be divided into two stages: upflow and downflow. The upflow stage collects contributions from all group members. The downflow stage broadcasts the intermediate values to all group members for calculating the shared group key.

To describe this in more detail, let $(M_1, M_2, ..., M_n)$ be a group of users, the i-th round of upflow stage is as follows:

1. M_i, where $0 < i \leq n$, receives a sequence of $(i-1)$ intermediate key values $\{\frac{N_1 ... N_{i-1}}{N_k} G | k \in [1, i-1]\}$ and one cardinal value $K_{i-1} = N_1 ... N_{i-1} G$.
2. M_i generates its own contribution N_i.
3. M_i computes the new cardinal value $K_i = N_i K_{i-1}$.
4. The old cardinal value becomes one of the intermediate values.
5. Multiply each old intermediate value with N_i thus producing a set of new intermediate values.
6. If $i < n$, M_i sends K_i and the new intermediate values to M_{i+1}.

In the upflow stage, the intermediate key values and the cardinal value are all points in the elliptic curve and thus they have double the length of the private key. The last cardinal value of the highest-indexed group member M_n is the secret key. In the last downflow round, M_n broadcasts $n-1$ intermediate values to the entire group. Each receiving M_i identifies its intermediate value and multiples it with N_i thus computing the secret key. The shared group session key can be derived from the secret key. In our simulation, we use the x-coordinate of the secret key as the shared group key.

B. Tree-based Group Elliptic Curve Diffie-Hellman Protocol

The proposed protocol, Tree-based group Elliptic Curve Diffie-Hellman (TGECDH), is a variant of TGDH based on ECDLP. In TGECDH, a binary tree is used to organize group members. The nodes are denoted as $< l, v >$, where $0 \leq v \leq 2^l - 1$ since each level l hosts at most 2^l nodes. Each

exactly the same idea, as normal DH
--> generate private, compute public, share public, compute shared secrete using your privet and their public

TABLE I

PROTOCOL COMPARISON (*SEE DETAILS IN SECTION IV)

		Rounds	Total messages	Combined message size in bits*
GDH / GECDH	Join	n	n	$\frac{k_u}{2}n^2 + (\frac{3k_u}{2} + r)n - 3k_u$
GDH / GECDH	Leave	n-1	n-1	$\frac{k_u}{2}(n-1)^2 + (\frac{3k_u}{2} + r)(n-1) - 3k_u$
TGDH / TGECDH	Join	2	3	$2((n+1)h - 2^h + 1)k_u + (n^2-1)c + 3(n-1)r$
TGDH / TGECDH	Leave	1	1	$hk_u + r$

node $< l, v >$ is associated with the key $K_{<l,v>}$ and the blinded key $BK_{<l,v>} = \mathbb{F}(K_{<l,v>})$ where the function $\mathbb{F}()$ is scalar multiplication of elliptic curve points in prime field. Assuming a leaf node $< l, v >$ hosts the member M_i, the node $< l, v >$ has $M_i's$ session random key $K_{<l,v>}$. Furthermore, the member M_i at node $< l, v >$ knows every key in the key-path from $< l, v >$ to $< 0, 0 >$. Every key $K_{<l,v>}$ is computed recursively as follows:

$$\begin{aligned} K_{<l,v>} &= K_{<l+1,2v>}BK_{<l+1,2v+1>} \ mod \ p \\ &= K_{<l+1,2v+1>}BK_{<l+1,2v>} \ mod \ p \\ &= K_{<l+1,2v>}K_{<l+1,2v+1>}G \ mod \ p \\ &= \mathbb{F}(K_{<l+1,2v>}K_{<l+1,2v+1>}) \end{aligned}$$

It is not necessary for the blind key $BK_{<l,v>}$ of each node to be reversible. Thus, we simply use the x-coordinate of $K_{<l,v>}$ as the blind key. The group session key can be derived from $K_{<0,0>}$. Each time when there is member join/leave, the sponsor node calculates the group session key first and then broadcasts the new blind keys to the entire group and finally the remaining group members can generate the group session key. Next, we analyze the communication time for each protocol.

IV. COMMUNICATION ANALYSIS

The group key distribution schemes discussed above are summarized and compared in Table I.

According to Table I, we can calculate the total message size in bits for each protocol. Assume we have n ($n \geq 2$) participants and let k_i, k_u indicate the private key and the blind key length and r be the overhead of each message. We use $f(Join, n)$ to denote the total message length when n participants establish a group key and use $f(Leave, 1)$ to denote the total message length when the remaining $n-1$ participants rebuild the group key after an existing member leaves.

Then, in GDH or GECDH, the total message length for n participants to generate the shared key can be calculated as follows:

$$\begin{aligned} f(Join, n) &= (\frac{(n+3)n}{2} - 3)k_u + nr \\ &= \frac{k_u}{2}n^2 + (\frac{3k_u}{2} + r)n - 3k_u \end{aligned}$$

When a member leaves the group, the remaining *n-1* participants need to rebuild the group key as *n-1* parties build the group key. Thus,

$$\begin{aligned} f(Leave, 1) &= f(Join, n-1) \\ &= \frac{k_u}{2}(n-1)^2 + (\frac{3k_u}{2} + r)(n-1) - 3k_u \end{aligned} \quad (2)$$

In tree based group Diffie-Hellman protocol, join and leave have different processing loads. When a new participant joins a group of size *i*, three messages are required:

- The new participant broadcasts its blind key
- The sponsor node broadcasts the key tree and the blind keys of the nodes which are the siblings of the nodes from the joining member to the root.
- The sponsor node broadcasts the new blind keys to rest of participants

Assuming we need *c* bits to represent a key tree node when a sponsor node broadcasts the key tree, then, the message size for a new participant to join a group of size *i* is equal to:

$$\begin{aligned} m(i) &= (1 + h + h - 1)k_u + (2i-1)c + 3r \\ &= 2hk_u + (2i-1)c + 3r \end{aligned}$$

where h is the height of the binary tree and thereby $h = \lceil \log_2 i \rceil$.

Therefore, the total message length to build a group of n participants from initial state (including 1 group member) to final state when all participants can generate the group key can be calculated as:

$$f(Join, n) = \sum_{i=2}^{n} m(i) = 2s_n k_u + (n^2-1)c + 3(n-1)r$$

where $s_n = (n+1)h - 2^h + 1$ and $h = \lceil \log_2 n \rceil$.

When a member leaves the group in tree based group protocols, the sponsor needs to generate a new session key, recalculate the agreed keys and blinds key along the key path and broadcast the new blind keys. Thus, the message size for one member leave is equal to

$$f(Leave, 1) = hk_u + r \quad (3)$$

Let *B* indicate the bandwidth of the ad hoc network, *d* be the maximum distance between two participants, *s* be the number of messages to build a group key for *n* parties and *p* be the probability of frames in errors. The communication time can be calculated as:

$$t = \frac{f}{B}\frac{1}{1-p} + \frac{sd}{3 \times 10^8}$$

TABLE II

COMPARABLE 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

Compared with the transmission time, the propagation delay $\frac{sd}{3\times10^8}$ is very small. Thus, we can approximately estimate the communication time as:

$$t \approx \frac{f}{B}\frac{1}{1-p} \tag{4}$$

Table II shows the comparable key sizes of the same security level for an ECDLP-based group scheme and DLP-based scheme [10]. It shows that ECDLP-based schemes can use a much smaller key size than DLP-based group schemes. In this paper, we evaluate ECDLP-based group Diffie-Hellman schemes at the same security level as the DLP-based Diffie-Hellman schemes. We use the pair $\langle k, m\rangle$ to represent the security level, where *k* is the private key size of ECDLP-based scheme and *m* is the modular size of DLP-based scheme.

TABLE III

KEY SIZE PARAMETERS

	$\langle 112\ bits, 512\ bits\rangle$		$\langle 160\ bits, 1024\ bits\rangle$	
	$k_i\ bits$	$k_u\ bits$	$k_i\ bits$	$k_u\ bits$
GDH	512	512	1024	1024
GECDH	112	232	160	328
TGDH	512	512	1024	1024
TGECDH	112	112	160	160

Equation 4 is plotted in Figures 1 and 2 for each protocol to show the communication time for member join operations at the level of $\langle 112\ bits, 512\ bits\rangle$ and $\langle 160\ bits, 1024\ bits\rangle$. The bandwidth is set to 11Mbps and the message overhead is set to $r = 192\ bits$ which is the length of a TCP header and each key tree node needs $c = 24\ bits$ for storage when broadcasting. The frames error rate is set to $p = 8.70\%$. The key size parameters used in the calculations are shown in Table III.

The figures show that ECDLP-based group schemes have lower communication time than DLP-based group schemes for member join operations. For example, the communication time of GDH is 2.2 times that of GECDH and TGDH is 1.7 times that of TGECDH on average at the level of $\langle 112\ bits, 512\ bits\rangle$. Moreover, the advantages increases as the security level increases. At the level of $\langle 160\ bits, 1024\ bits\rangle$, the communication load of GDH is 3.1 times that of GECDH and TGDH is 2.4 times that of TGECDH on average.

For member leave operations, the tree-based group Diffie-Hellman schemes are far better than group Diffie-Hellman schemes as shown in equations 2 and 3.

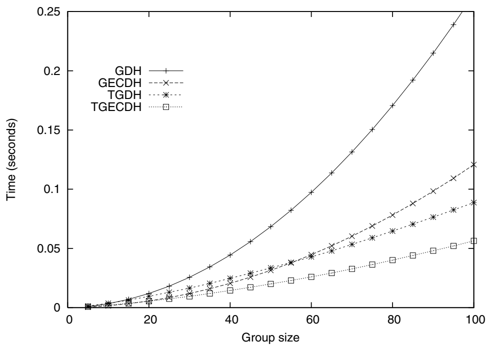

Fig. 1. Communication time: member join, level $\langle 112\ bits, 512\ bits\rangle$. The communication time of GDH is 2.2 times that of GECDH and TGDH is 1.7 times that of TGECDH on average for member join operations.

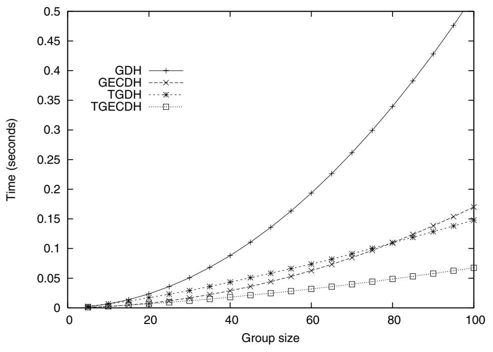

Fig. 2. Communication time: member join, level $\langle 160\ bits, 1024\ bits\rangle$. The communication time of GDH is 3.1 times that of GECDH and TGDH is 2.4 times that of TGECDH on average for member join operations.

V. EXPERIMENTS AND DISCUSSION

The experiments were conducted on a Linux box running on a 2.4 GHz Celeron(R) CPU, with 256 MB of memory. Crypo++ Library 5.2.1 [11] was used for the implementation. For each experiment, we ran the protocol ten times and calculated the average. The overall group key generation time includes communication time but does not include individual key generation time which is assumed to be completed in the initial stage. Furthermore, we assume that all the remaining group members can calculate the group key in parallel when M_n broadcasts $n-1$ intermediate values to the entire group in the downflow stage of GDH/GECDH or when the sponsor node broadcasts the new blind keys to the entire group in TGDH/TGECDH.

In this paper, we use a two state Markov model to characterize wireless channels. Markov models have been found to be an appropriate model to characterize signal to noise variations in slow fading channels [12], [13]. When used in simulations, the model is usually constructed with two states, each state representing either 'good' or 'bad' channel conditions. All frames received during a 'good' state are assumed to be error free and all frames received during a 'bad' state are assumed to be in error. Let the rate of transition from the good to the bad state be *p* and the rate of transition from the bad to the good state be *q*, this model is characterized by computing mean durations for each state (i.e. $1/p$ and $1/q$) and then using the results as the means for exponential distributed state durations. Table IV shows the mean durations of states as well as the obtained frame error rates from experiments. Two scenarios are considered in the experiments: low bit-error rate channels and high bit-error rate channels.

TABLE IV
CHARACTERIZATION OF THE TWO-STATE MODEL

Bit error rate (BER)	Transmission rate [Mbps]-frame size [bytes]	Mean good state duration [s]	Mean bad state duration [s]	% of frames in error
Low	11-1000	0.0173	0.0015	8.70%
High	11-1000	0.0027	0.0037	58.30%

Figures 3 and 4 show the communication time of the four compared schemes under different channel conditions when the group size is 100. The figures show that TGECDH can tolerate the frame errors in wireless channels.

Figures 5 and 6 show the communication time for member join operations in the experiments as well as the data in Figures 1 and 2 at the level of $\langle 112\ bits, 512\ bits\rangle$ and $\langle 160\ bits, 1024\ bits\rangle$. The experiment results for GDH and GECDH protocols match with the communication analysis well but there is a slight difference for TGDH and TGECDH. This is because TGDH and TGECDH employ short packets and cause less packet loss in the simulation.

Figures 7, 8, 9 and 10 show the overall group key generation time for member join and leave at the level of $\langle 112\ bits, 512\ bits\rangle$ and $\langle 160\ bits, 1024\ bits\rangle$. At the level of $\langle 112\ bits, 512\ bits\rangle$, when the group size is less than 62, GDH requires the least key generation time for member join operations. When the group size is more than 62, TGECDH requires the least key generation time. At the level of $\langle 160\ bits, 1024\ bits\rangle$, GDH requires the least key generation time for member join operations when group size is less than 25 and TGECDH requires the least key generation time when group size is greater than 25. Figures 8 and 10 indicate that the tree based group schemes are far better than GDH or GECDH for member leave operations.

In Figure 9, when the group size is greater than 70, there are slight fluctuations in the curves. This may be related to the implementation details in the Crypo++ library and is under further investigation.

As for the DLP-based group key agreement protocols, the agreed group key (denoted as GK) size is at least 512 bits in order to have acceptable security level. If GK is used as the secret key to encrypt messages, it is an overkill. Thus, we can assume that from GK, a secret key SK (for example, 128 bits) is derived and the SK is used to encrypt messages.

Similarly, for ECDLP-based group key agreement protocols, we can use GK to derive a SK (128 bits) and use SK to encrypt the messages. In this way, the problem of doubling cipher text length is avoided. In addition, if the GK is used on plain text directly, the plain text needs to be translated to points on the elliptic curve (i.e., encoding), which is also not an easy task. if a SK derived from GK is used, this translation is avoided.

VI. CONCLUSIONS

In this paper, we propose and evaluate ECDLP-based Diffie-Hellman protocols for secure group communication in wireless ad hoc networks. The theoretical analysis and experiments show that TGECDH is the best protocol in terms of overall performance for secure group communication in ad hoc networks among the four schemes we discuss in this paper. Although for small group sizes, GDH performs better than TGECDH for member join operations, it takes much more time for GDH to process member leaves than TGECDH while TGECDH performs very well for both member join and leave operations. Moreover, as the key size increases for a higher security level, TGECDH gains more advantages than other schemes.

The good performance of TGECDH over wireless ad hoc networks can be summarized as follows:

• It uses smaller keys.

• It uses less computation time than the DLP-based scheme for the same security level.

• Smaller packets are used to handle high bit error rate wireless links.

ACKNOWLEDGEMENTS

This work is partially supported by NSF Grant No. CCR-0311577.

REFERENCES

[1] X. Zou, B. Ramamurthy, and S. S. Magliveras, *Secure Group Communications Over Data Networks.* Springer, 2005.
[2] V. S. Miller, "Use of elliptic curves in cryptography," in *Lecture notes in computer sciences; 218 on Advances in cryptology—CRYPTO 85.* New York, NY, USA: Springer-Verlag New York, Inc., 1986, pp. 417–426.
[3] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203–209, 1987.
[4] Y. Amir, Y. Kim, C. Nita-Rotaru, and G. Tsudik, "On the performance of group key-agreement protocols," in *Proc. of the 22nd IEEE International Conference on Distributed Computing Systems*, Viena, Austria, June 2002.
[5] K. S. Hagzan and H.-P. Bischof, "The performance of group diffie-hellman paradigms," in *The 2004 International Conference on Wireless Networks (ICWN'04)*, Las Vegas, Nevada, USA, June 2004.
[6] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, November 1976.
[7] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman key distribution extended to group communication," in *CCS '96: Proceedings of the 3rd ACM conference on Computer and communications security.* New York, NY, USA: ACM Press, 1996, pp. 31–37.
[8] Y. Kim, A. Perrig, and G. Tsudik, "Simple and fault-tolerant key agreement for dynamic collaborative groups," in *CCS '00: Proceedings of the 7th ACM conference on Computer and communications security.* New York, NY, USA: ACM Press, 2000, pp. 235–244.
[9] *Recommended Elliptic Curve Domain Parameters*, SECG Std. SEC2, 2000, available from www.secg.org/collateral/sec2.pdf.
[10] *Elliptic Curve Cryptography*, SECG Std. SEC1, 2000, available from www.secg.org/collateral/sec1.pdf.
[11] W. Dai, "Crypto++ library 5.2.1," available from www.cryptopp.com.
[12] C. C. Tan and N. C. Beaulieu, "On first-order Markov modeling for the Rayleigh fading channel," *IEEE Transactions on Communications*, vol. 48, no. 12, pp. 2032–2040, December 2000.
[13] M. Zorzi, R. R. Rao, and L. B. Milstein, "Error statistics in data transmission over fading channels," *IEEE Transactions on Communications*, vol. 46, no. 11, pp. 1468–1476, November 1998.

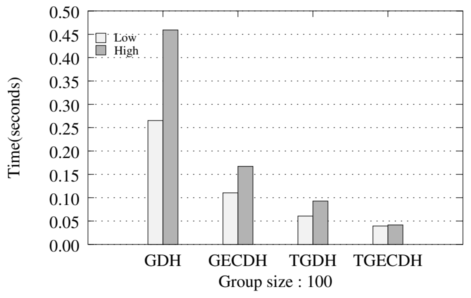

Fig. 3. Communication times under different channel conditions, member join, level $\langle 112\ bits, 512\ bits\rangle$: channel conditions have little effect on TGECDH due to the small size of packets employed.

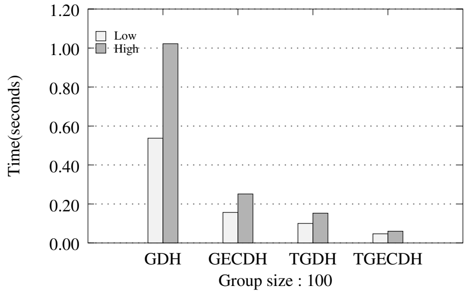

Fig. 4. Communication times under different channel conditions, member join, level $\langle 160\ bits, 1024\ bits\rangle$: channel conditions have little affection on TGECDH due to the small size of packets employed.

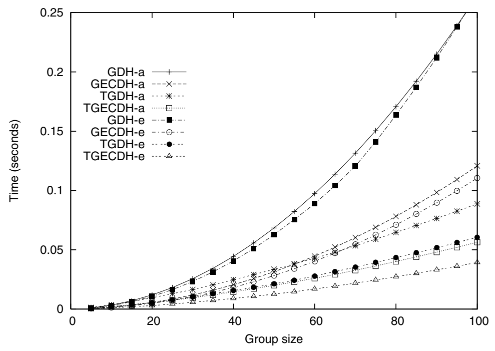

Fig. 5. Communication time from analysis and experiments: member join, level $\langle 112\ bits, 512\ bits\rangle$, channel conditions: low, 8.7% frames in error. (a stands for analysis and e stands for experiments.)

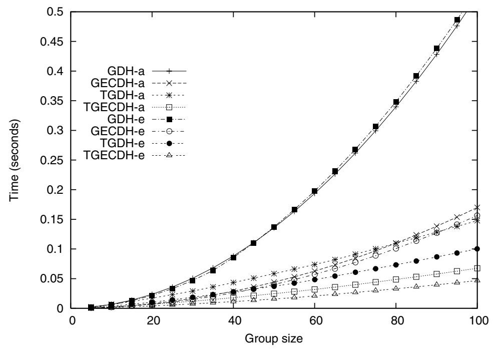

Fig. 6. Communication time from analysis and experiments: member join, level $\langle 160\ bits, 1024\ bits\rangle$, channel conditions: low, 8.7% frames in error. (a stands for analysis and e stands for experiments.)

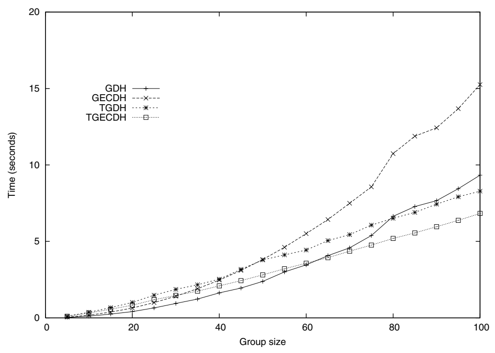

Fig. 7. Group key generation time: member join, level $\langle 112\ bits, 512\ bits\rangle$. GDH costs the least key generation time when group size is less than 62 but after that TGECDH costs the least key generation time for member join operations.

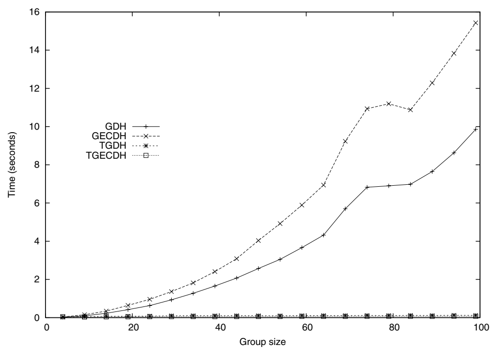

Fig. 8. Group key generation time: member leave, level $\langle 112\ bits, 512\ bits\rangle$. Tree-based group Diffie-Hellman schemes are far better than group Diffie-Hellman schemes for member leave operations.

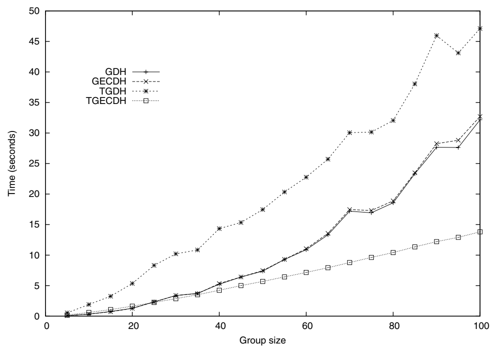

Fig. 9. Group key generation time: member join, level $\langle 160\ bits, 1024\ bits\rangle$. GDH costs the least key generation time when group size is less than 25 but after that TGECDH costs the least key generation time for member join operations.

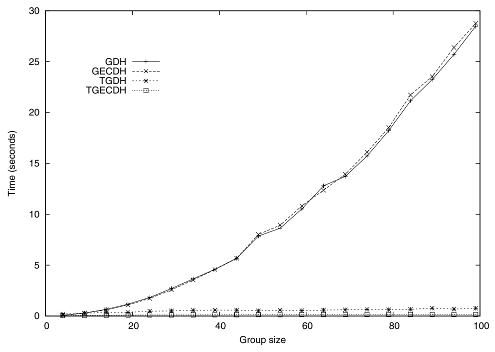

Fig. 10. Group key generation time: member leave, level $\langle 160\ bits, 1024\ bits\rangle$. Tree-based group Diffie-Hellman schemes are far better than group Diffie-Hellman schemes for member leave operations.