Paper: ASAT-14-136-CM

14th International Conference on
AEROSPACE SCIENCES & AVIATION TECHNOLOGY,
ASAT - 14 – May 24 - 26, 2011, Email: asat@mtc.edu.eg
Military Technical College, Kobry Elkobbah, Cairo, Egypt
Tel: +(202) 24025292 –24036138, Fax: +(202) 22621908

A Verifiable New Member Joining in Threshold Key Management Scheme for Mobile Ad Hoc Networks Using Elliptic Curve Dlog-Based Cryptosystem

Ashraf D. Elbayoumy* and Hisham Dahshan†

Abstract: A mobile ad hoc network (MANET) is a self-organized wireless network where mobile nodes can communicate with each other without the use of any existing network infrastructure or centralized administration. Trust establishment and management are essential for any security framework of MANETs. However, traditional solutions to key management through accessing trusted authorities or centralized servers are infeasible for MANETs due to the absence of infrastructure, frequent mobility, and wireless link instability. In most ad hoc network applications, the number of nodes is not fixed and the network should be flexible to add new members according to different situations. In this paper, a verifiable new member joining in threshold key management scheme for MANETs is presented. The proposed scheme is implemented using elliptic curve dlog-based cryptosystem. In this scheme, the new member broadcasts a joining request and if at least k neighbors trust the new member, they cooperate to issue the new member the keying materials required to join the network. The advantages of the proposed scheme are justified through extensive simulations.

Keywords: Mobile Ad Hoc Networks, Wireless Network Security, Key Management

I. Introduction

A mobile ad hoc network MANET is an autonomous system of mobile nodes connected by wireless links. Each mobile node is able to communicate by radio waves with other nodes within its transmission range and relays on other nodes to communicate with mobile nodes outside its transmission range. The absence of centralized administration and the infrastructureless nature make MANETs good for emerging, disaster relief efforts, military and fast deployment communications. The lack of any centralized network management or certification center makes MANET vulnerable to infiltration, eavesdropping, interference, and so on. Efficient and robust key management services are central to provide MANETs with security services such as confidentiality, authentication, integrity and non-repudiation.

Traditional key management service is based on a certificate authority (CA) or a trusted third party (TTP) to issue public key certificates to all nodes in the network. However, providing key management by relying on a single TTP or CA is not applicable within the pervasive environment of MANETs. In the mobility environment of MANETs, only distributive key management schemes can work efficiently. There has been a rich literature on public-key management in MANETs.

* Egyptian Armed Forces, Egypt.
† Egyptian Armed Forces, Egypt, hishamdahshan@yahoo.com

Some schemes depend on certificate-based cryptography [1], [2], [3] in which public-key certificates are used to authenticate public keys by binding public keys to the users' identities. This approach suffers from lack of scalability with increasing the network size, and cannot handle key update in a secure and cost-effective way.

Another approach is providing keying material through a web of trust [4], [5]. In the web of trust approach, public key authentication is performed via a chain of certificates from the source node to the destination node.

In threshold cryptography (TC) [6], a group of n parties share the ability to perform a cryptographic operation by distributing the trust placed on a single user among a group of n users. In addition, there is a threshold t associated with the threshold cryptosystem such that any t out of the n users can execute the cryptographic operation. Such schemes are referred to as (t, n) TC schemes. In this case, less than t users will not be able to perform the cryptographic operation successfully. In a threshold cryptosystem, if less than t users are compromised, the security of the whole system will not be breached. In threshold key management, a set of n servers jointly generate a pair of public and private keys in a way that the public key is known to all nodes in the network while the private key is divided between the n servers via a threshold secret sharing scheme such as Shamir's (t, n) threshold cryptography [6]. Later, in order for a new node to join the network, at least t nodes (among n nodes) need to cooperate and sign a certificate for the new node.

A model for distributing trusted services (by using threshold cryptography) among a set of servers despite some servers being under control of an attacker has been proposed in [7]. A practical implementation of a distributed key generation (DKG) on a network of computers has been presented in [8]. In this scheme, secure sockets are used to create the private channels over the Internet.

Using elliptic curves for cryptographic protocols has been proposed in [9], [10]. Cryptosystems based on elliptic curve discrete logarithm problem (ECDLP) can use smaller key size than that is needed by discrete logarithm problem (DLP) or integer factorization problem (IFP) based cryptosystems to provide the same level of secrecy. Reducing the key size while maintaining the same security level saves memory, computation power, and communication overheads which are major concerns in the resource constrains environment such as smart cards and MANETs.

In [11], a novel threshold key management protocol for MANETs using elliptic curve dlog-based cryptosystem is proposed. In this scheme, an off-line authority is required in the network initialization phase before network deployment. After network deployment, the off-line authority has no role in the network. When a group of nodes wish to establish a secure session, a node from the session broadcasts a session formation request. Each session member collaborates with its neighboring nodes to generate its private/public key pair, other session members public keys, and the session public key.

In most ad hoc network applications, the number of nodes is not fixed and the network should be flexible to add new members according to different situations. The existence of the *CA* to issue the keying materials for each new member wishing to join the network after network deployment is not applicable and inconvenient.

In this paper, a verifiable new member joining in threshold key management protocol for MANETs using elliptic curve dlog-based cryptosystem is proposed. In the proposed scheme, the new member broadcasts a joining request and if at least *k* neighbors trust the new member they cooperate to issue the new member the required keying material. In this scheme, the new member can verify the contributions of its neighboring nodes which are necessary to produce the keying material required to join the network.

The rest of the paper is organized as follows: Section II provides a brief introduction about elliptic curve cryptography related to our proposed scheme. Section III provides the scheme description of our proposed scheme. The performance evaluation of the proposed scheme is presented in Section IV. Finally, Section V concludes the paper.

II. Elliptic Curve System Initialization

The elliptic curve cryptosystem makes use of elliptic curves where all variables and coefficients must be elements of a finite field. There are two groups of elliptic curves which are used for cryptographic applications: prime curves defined over *GF (p)* and binary curves defined over $GF(2^p)$.

a) Elliptic Curve Cryptography Domain Parameters: The operation of any cryptographic scheme involves arithmetic operations on an elliptic curve over a finite field determined by some elliptic curve domain parameters. If E is an elliptic curve over a finite field $GF(q)$, then let $E(GF(q))$ denote the points in E including the point at infinity O. The parameters of the elliptic curve are $E: \{q, F R, a, b, G, p, h\}$, where q is a prime power ($q = p$ or $q = 2^m$), an indication FR of the method used for representing field elements, a and b are the coefficients of the elliptic curve equation. G is the base point of the elliptic curve $E(Fq)$, denoted as $G = (xG, yG)$, a prime p which is the order of G, and finally the cofactor $h = (\#E(GF(q))/p)$, where $\#E(GF(q))$ denote the number of points on elliptic curve E.

III. Scheme Description

Before we proceed to the new member joining protocol, we first show the Threshold Key Management Scheme for Mobile Ad Hoc Networks Using Elliptic Curve Dlog-Based Cryptosystem [11]. This scheme consists of two algorithms; the shares matrices generation algorithm and the session key generation algorithm. We assume the following assumptions in the proposed scheme:

- Each user in the network has a unique identification number *ID* in $GF(q)$ where q is a prime or $q = 2^m$.
- Each node u_i is assumed to have a long term public/private key pair PK_i/SK_i, where the public key $PK_i = SK_i \Theta G$ is known to all mobile nodes in the network and Θ is the point multiplication by a scalar operation.

A. Shares Matrices Generation:

Let U be the set of nodes in the network initialization phase, where

$$U = \{ u_i, 1 \leq i \leq n \}$$

The set of nodes U has a cardinality equal to n and nodes that $\in U$ are identified by their unique identities $(u_1, u_2, \ldots, u_n)$, where $u_i \in GF(q)$. Shares matrices generation phase includes the following steps:

- Step 1: *CA* selects *B* secret keys *S*, where

$$S = \{S_1, S_2, \quad , S_B\}$$

- Step 2: *CA* picks *B* secret polynomials $f_1(x)$, $f_2(x)$,… ,$f_B(x)$:

$$f_r(x) = \left(S_r + \sum_{i=1}^{t-1} f_{ri} x^i \right) mod\ p, \tag{1}$$

where $1 \leq r \leq B$

- Step 3: *CA* computes secret polynomial shares:
  $s_{ri} = fr(u_i)$, where $1 \leq r \leq B$, and $1 \leq i \leq n$.
- Step 4: *CA* rearranges the shares for node u_i, establishes, and preloads the following $m \times d$ shares matrix to node u_i before network deployment:

$$\begin{bmatrix} s_{11}^i & \cdots & s_{1d}^i \\ \vdots & \vdots & \vdots \\ s_{m1}^i & \cdots & s_{md}^i \end{bmatrix}_{m \times d} \tag{2}$$

Shares of the same indices in the shares matrix of every mobile node represent shares for the same secret S_r. The parameter *t* chosen by the *CA* determines the number of neighboring nodes needed to collaborate with a session member u_l in order to construct its private/public key pairs, session members public keys, and the session public key and this parameter cannot be changed henceforth.

B. Session Key Generation

When a group of nodes $W = \{ u_l, 1 \leq l \leq L \}$ wish to generate session keys to be used in securing communications for this session (for example generate a signature for a message *m*), they follow the following steps:

- Step 1: the mobile node u_l which has the largest index over all other session members, broadcasts a session formation request as follows:

$$u_l \rightarrow Broadcast : \{REQ,\ Nonce, k, u_1, \dots, u_L\}_{Sig_{ul}}$$

where *k* is the number of shares to be picked from the shares matrix by each node in order to construct its secret sharing polynomial, and Nonce is a large random number used to randomize the session keys generation process.

- Step 2: when a neighboring node u_j (which is not necessary to be a session member and not limited to the 1-hop neighbors) receives the request, it authenticates the origin of the request by verifying the signature of the sender node u_l using the mobile node u_l*'s* long term public key PK_{ul} .
- Step 3: When the verification succeeds, mobile node u_j maps the session member *IDs* and the Nonce by using a *k* recursive hash functions $H^k(x)$, where $H^2(x) = H(H(x))$, to obtain shares indices to be picked from its shares matrix as follows:

1) calculate *k* row numbers as follows:

$$R_i = H^i(u_1|u_2|\dots|u_L|Nonce)\ mod\ M,$$

where $1 \leq i \leq k$

2) calculate *k* column numbers as follows:

$$C_i = H^i(Nonce) \bmod D,$$

where $1 \le i \le k$

- Step 4: mobile node *uj* picks *k* shares from its shares matrix according to the rows and columns indices obtained from the previous step such that $s_{R_1C_1}$, $s_{R_2C_2}$,…, $s_{R_tC_t}$ and constructs its secret sharing polynomial as follows:

$$f_j(x) = \left(a_1 + \sum_{i=2}^{k} a_i x^{i-1}\right) mod\ p,$$

where $a_i = s_{R_iC_i}$, $(1 \le i \le k)$

- Step 5: using its secret sharing polynomial, node u_j calculates a share for each session member $u_l \in W$ as follows:

$$Z_{jl} = f_j(u_l) = a_1 + \sum_{i=2}^{k} \left(a_i\left((u_l)^{i-1}\right)\right) mod\ p$$

where $1 \le l \le L$

- Step 6: node u_j calculates the public shares $\{PV_{j1}, \dots, PV_{jL}\}$, where $PV_{jl} = Zjl \odot G$ and sends out the following message to the session member node u_l:

$$u_j \rightarrow u_l : \left\{REP, Enc_{PK_l}\left[Z_{jl}\right], PV_{j1}, \dots, PV_{jL}\right\}_{Sig_{u_j}}$$

- Step 7: mobile node u_l computes and broadcasts the following public values:

$$P_{val_{ji}} = \left(a_{ji} \odot G\right) \qquad \text{where } (0 \le i \le k)$$

- Step 8: mobile node u_l collects at least *t* replies from its neighboring nodes in addition to its own calculations from the previous steps to reconstruct its private/public key pair, session members public keys and the session public key as follows:

1) The private key of the mobile node u_l in this session is calculated as follows:

$$u_{l_{SK}} = \sum_{j=1}^{t} Z_{jl} \prod_{\substack{i=1 \\ i \ne j}}^{t} \frac{i}{i-j} mod\ p$$

2) The public key of the mobile node u_l in this session is $u_{l_{PK}} = u_{l_{SK}} \odot G$.

3) The session member u_l calculates the public key for other session members u_h, where $(1 \le h \le L, h \ne l)$ as follows:

$$u_{h_{PK}} = \sum_{j=1}^{t}{}^{\oplus} PV_{jh} \prod_{\substack{i=1 \\ i \ne j}}^{t} \frac{i}{i-j} mod\ p$$

where $\sum^{\oplus}$ is the point summation under point add operation $\oplus$.

4) The session member u_l calculates the session public key as follows:

$$SS_{PK} = \sum_{l=1}^{k} u_{l_{PK}} \prod_{\substack{i=1 \\ i \ne l}}^{k} \frac{i}{i-1} mod\ p$$

where *k* is the threshold number included in the session formation request which determines the minimum number of mobile nodes required to form the session.

C. The Proposed New Member Joining Algorithm

Although the process of joining a new member to the network is an important issue for a mobile ad hoc network, it should not be at the expense of changing the shares matrices of the old members. Taking into consideration the previous requirements, the process of joining a new member to the mobile ad hoc network in our proposed scheme can be summarized as follows: the new member node u_j broadcasts a joining request that contains its identity and the trust evidence to its neighboring nodes as follows:
$u_j \rightarrow Broadcast$: {u_j, *Trust Evidence*}, where *Trust Evidence* is a credential that existing nodes can verify. If a group of at least t old members (We call this group henceforth the issuing group *IG*) believe that the requesting node is trustworthy according to its trust evidence, they cooperate to issue a shares matrix for the requesting node by using the Lagrange interpolation rule on every share of their shares matrices.

An overview of the new member joining process is shown in Figure 1

The new member joining process for each element of the shares matrix can be summarized as follows:

- Step 1: each node $u_i \in IG$ calculates a contribution $Q_i(j)s_{kh}^{(i)}$ by using the shares in its shares matrix (refer to equation 2) where $1 \leq k \leq m$, $1 \leq h \leq d$ and the Lagrange coefficient $Qi(j)$ in this case can be calculated as follows:

$$Q_i(j) = \prod_{\substack{l=1 \\ l \neq i}}^{t} \frac{j-l}{i-l} \tag{3}$$

New Member Joining

New member broadcasts a joining request and if at least *k* neighbors trust the new member, they do the following steps for each matrix element:
Step 1: *u* divides its contribution of the new member share among the *k* neighbors
Step 2: Neighboring node *u*, collects *k* contributions from neighbors
Step 3: *u*, sends the encrypted sum of contributions to new member
Step 4: New member decrypts contributions it received from all *k* neighbors and forms its shares matrix

Figure 1 New Member Joining

- Step 2: node $u_i \in IG$ divides randomly its contribution $(Q_i(j)s_{kh}^{(i)} mod\ p)$ among the t members of the *IG* group and sends an encrypted message includes β_{il} (where l =1,…,t) to each *IG* member such that:

$$\sum_{i=1}^{t} \beta_{il} = Q_i(j)s_{kh}^{(i)} mod\ p \tag{4}$$

- Step 3: node u_i broadcasts the public value $(Qi(j)s_{kh}^{(i)}) \odot G$.
- Step 4: node $u_i \in IG$ collects $t-1$ contributions from the other members of the *IG* group in addition to its own contribution, encrypts them with its private key and sends the following message to the new member:

$$u_i \rightarrow u_j : \left\{ Enc_{PK_i} \left[\sum_{i=1}^{t} \beta_{li} \right] \right\}$$

- Step 5: the new member node u_j collects and decrypts the received t messages from the IG group to construct its share $s_{kh}^{(j)}$ as follows:

$$s_{kh}^{(j)} = \sum_{i=1}^{t}\sum_{l=1}^{t}\beta_{li} = \sum_{l=1}^{t}\sum_{i=1}^{t}\beta_{li} = \sum_{i=1}^{t} Q_l(j)s_{kh}^{(l)} mod\ p \quad (5)$$

- Step 6: the new member u_j verifies its share $s_{kh}^{(i)}$.
  The new member u_j checks if

$$s_{kh}^{(j)} \odot G = \sum_{i=1}^{t}{}^{\oplus} \left(Q_i(j)s_{kh}^{(i)}\right) \odot G \quad (6)$$

  If the verification fails, then some nodes in the IG group are sending incorrect shares. In order to ensure that the shares matrix of the new member are constructed from correct shares only, the new member node requests from each node $u_i \in IG$ to broadcast the public values $\beta_{il} \odot G$ and checks if

$$\sum_{i=1}^{t}{}^{\oplus} \beta_{il} \odot G = \left(Q_i(j)s_{kh}^{(i)}\right) \odot G \quad (7)$$

  If the verification fails, the new member u_j broadcasts a complaint against u_i.

The previous steps are repeated until the new member completes its shares matrix.

IV. Performance Evaluation

In this Section, the performance evaluation of our proposed threshold key management scheme is presented. We measure the computation complexity of our proposed scheme by presenting its timing results in comparison with that of Tang scheme [12].

One of the major factors that affect the performance of ECDLP-based cryptographic protocols is the domain parameters (refer to Subsection II-0a) and the elliptic curve selected for implementation. The fundamental operation underlying ECC is point multiplication, which is defined over finite field operations. Elliptic curves are defined over either prime fields $GF(p)$ or binary fields $GF(2^p)$. In environments in which an arithmetic processor is already available, the performance of $GF(p)$ can be improved so that in some cases it exceeds the performance of $GF(2^p)$. In the performance evaluation of our proposed scheme, we consider only prime fields $GF(^p)$ since binary field arithmetic, is insufficiently supported in PARI/GP [13] and would thus lead to lower performance. On a desktop PC with an Intel Core 2 Duo 2.6 GHz processor and 1GB memory, PARI/GP [13] is used to evaluate the performance of our proposed scheme. The prime elliptic curve over $GF(p)$ is defined by the equation

$$y^2 = (x^3 + ax + b) mod\ p \quad (8)$$

where a and $b \in GF(p)$ for p a prime and satisfy

$$4a^3 + 27b^2 \equiv 0 (mod\ p) \quad (9)$$

The domain parameters (refer to Subsection II-0a) used in our implementation are as follows:

A field size q which defines the underlying finite field $GF(q)$, where $q > 3$ is a prime number; two field elements a and b in $GF(q)$ which define the equation of the elliptic curve $E: y^2 = (x^3 + ax + b) mod\ p$; two field elements x_G and y_G in $GF(q)$, which define a point $G = (x_G, y_G)$ of prime order on E ; the order p of the point G (it must be the case that $p > 2^{160}$; and the cofactor h =(number of points on elliptic curve/p). The performance evaluation of the proposed scheme will be given in terms of the two thresholds t and k (out of the total n mobile nodes). The performance of the proposed scheme is evaluated for three different key sizes: 192 bits, 239 bits, and 256 bits [14]. The total number of mobile nodes in the network is set to 30 nodes. Values that remain constant between different scheme runs (for example, the inner parts of the Lagrange coefficients) can be precomputed and are therefore not included in the evaluation.

A. An Overview of Tang et al. New Member Joining Scheme

In Tang et al. scheme, a group of mobile nodes u_i (i =1, 2,…, n) cooperate to issue the new member u_j the required keying material as follows:

- Step 1: each mobile node u_i ($1 \le i \le n$), chooses a random polynomial $\tilde{f}^{(i)}(z)$ of degree $t-1$ as follows:

$$\tilde{f}^{(i)}(z) = \tilde{a}_1^{(i)} z + \cdots + \tilde{a}_{t-1}^{(i)} z^{t-1} \tag{10}$$

  where $\tilde{a}_l^{(i)} \in GF(q)$ $(0 \le l \le t-1)$.
- Step 2: u_i computes $\hat{s}_j^{(i)} = \hat{f}^{(i)}(u_j) mod\ p, (j = 1, \cdots, n)$ and sends the following message to u_j:
  $u_i \rightarrow u_j: \left\{ Enc_{PK_j}\left[\hat{s}_j^{(i)}\right] \right\}$
- Step 3: u_i computes and broadcasts the following $t-1$ public values:

$$\hat{A}_l^{(i)} = \hat{a}_l^{(i)} \odot G \tag{11}$$

  where $(0 \le l \le t-1)$.

Table I New member joining timing in (MS)

CA Threshold (t)	192-bits Curve		239-bits Curve		256-bits Curve	
	Proposed scheme	Tang's scheme	Proposed scheme	Tang's scheme	Proposed scheme	Tang's scheme
5	121.40	32.64	198.60	53.60	220.80	60.16
10	129.00	77.92	203.00	124.18	225.00	137.78
20	154.43	304.13	233.43	447.80	254.82	482
30	179.68	902	266.30	1278	289.28	1355

- Step 4: the new member u_j decrypts the received message from node u_i in Step 2 and verifies the share it received as follows:

$$\hat{s}_j^{(i)} \odot G = \sum_{i=1}^{t-1}{}^{\oplus} \left(u_j^l \hat{A}_l^{(i)} \right) \tag{12}$$

  If the verification fails, u_j broadcasts a complain against u_i.

- Step 5: each node u_i that received a complaint from the new member u_j broadcasts the value $\hat{s}_j^{(i)}$ that satisfy equation 12.
- Step 6: node u_j removes node u_i from its trusted group if it received more than t complaints against u_i in step 4, or if the reply of the node u_i in step 5 does not satisfy equation 12.
- Step 7: node u_j computes its private key $\hat{s}_{kh}^{(j)}$ as follows:

$$\hat{s}_{kh}^{(i)} = \sum_{l=1}^{n} \hat{s}_i^{(l)} \tag{13}$$

- Step 8: the new member u_j computes its public key $PK_j = \tilde{s}_{kh}^{(i)} \odot G$ and broadcasts PK_j.
- Step 9: the new member u_j receives PK_i (i =1, $\cdots, n$) and computes the session public key

$$SS_{PK} = \sum_{i=1}^{n} \overset{\oplus}{} PK_i \tag{14}$$

B. Timing Results

If the size of the shares matrix is $m \times d$, and the threshold is k, then the key space will be equal to

$$\frac{(m \times d)!}{[(m \times d) - k]!} \tag{15}$$

Results are presented for shares matrix of size 6×6 with the threshold k equal to 4 and according to equation 15, this shares matrix can produce 1.4×10^6 session keys. In Table I, we illustrate the timing results of the new member joining process of our proposed scheme which can produce 1.4×10^6 session keys in comparison with that of Tang scheme [15] which produces only one session key for the new member. Results are presented for key sizes 192 bits, 239 bits, and 256 bits. Results show that for small *CA* threshold *t*(5,10), the new member joining process in our proposed scheme takes more time than that in Tang's scheme. With increasing the *CA* threshold *t*(20,30), the timing results of the new member joining process in Tang's scheme exceed the timing results of the same process in our scheme. Table I shows that the timing of the new member joining process increases slightly with increasing the *CA* threshold *t*. It shows also that the timing of this process increases slightly with increasing the key sizes. This reflects the suitability of the proposed scheme to the dynamical nature of MANETs where the number of network nodes changes frequently. The majority of the computations in the new member joining process are consumed in verifying the shares of the shares matrix of the new member. The majority of computations in the verifications of the new member shares matrix are performed in the point multiplication and addition operations in step 5 of Subsection III-C as illustrated in equations 6 and 7.

V. Conclusions

In this paper, a new member joining in threshold key management scheme using elliptic curve dlog-based cryptosystem has been proposed. The process of joining a new member in our proposed scheme does not change the shares matrices of the old members. For small *CA* threshold *t*(5,10), the new member joining process in our proposed scheme takes more time than that in Tang's scheme, [12]. On the other hand, the new member joining process in our proposed scheme issues the new member the shares matrix which can produce a very large

key space compared to that of Tang scheme which issues the new member one key only. With increasing the *CA* threshold *t*(20,30), our proposed scheme has very low timings compared to that of Tang scheme, and timing does not vary significantly with changing the key size which reflects the suitability of the proposed scheme for applications where devices are resource constrained such as in the mobile ad hoc environments.

References

[1] L. Zhou and Z. J. Hass, "Securing Ad Hoc Network," IEEE network, vol. 13, no. 6, pp. 24–30, 1999.

[2] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing robust and ubiquitous security support for mobile ad-hoc networks," in Proceedings of the Ninth International Conference on Network Protocols, 2001.

[3] S. Yi and R. Kravets, "MOCA: Mobile Certificate Authority for Wireless Ad Hoc Networks," in Proceedings of the 2nd Annual PKI Research Workshop (PKI 2003), 2003.

[4] H. Mohri, I. Yasuda, Y. Takata, and H. Seki, "Certificate Chain Discovery in Web of Trust for Ad Hoc Networks," in Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops, vol. 2, pp. 479–485, IEEE Computer Society, 2007.

[5] K. Ren, T. Lib, Z. Wanb, F. Baob, R. H. Dengb, and K. Kima, "Highly reliable trust establishment scheme in ad hoc networks," The International Journal of Computer and Telecommunications Networking, ElSEVIER, vol. 45, no. 6, pp. 687–699, 2004.

[6] A. Shamir, "How to Share a Secret," Communication of the ACM, vol. 22, pp. 612–613, 1979.

[7] C. Cachin, "Distributing Trust on the Internet," in Proceeding of the 2001 International Conference on Dependable Systems and Networks, pp. 183–192, 2001.

[8] A. T. Chronopoulos, F. Balbi, D. Veljkovic, and N. Kolani, "Implementation of Distributed Key Generation Algorithms using Secure Sockets," in Third IEEE International Symposium of the Network Computing and Applications, pp. 393–398, IEEE Computer Society, 2004.

[9] V. S. Miller, "Use of elliptic curves in cryptography," Lecture notes in computer sciences; Advances in cryptology, pp. 417–426, 1986.

[10] N. Koblitz, "Elliptic Curve Cryptosystems," Mathematics of Computation, vol. 48, pp. 203–209, Jan 1987.

[11] H. Dahshan and J. Irvine, "A robust self-organized public key management for mobile ad hoc networks," Security and Communication Networks, vol. 3, no. 1, pp. 16–30, 2010.

[12] C. Tang and D. O. Wu, "An Efficient Proactive Share Refreshing Scheme for Secret Sharing in Distributed Systems," in IEEE Global Telecommunications Conference, GLOBECOM '06., December 2006.

[13] The PARI Group, Bordeaux, PARI/GP, version 2.4.3, 2008. available from http://pari.math.u-bordeaux.fr.

[14] ANSI X9.62, Public Key Cryptography For The Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA).

[15] C. Tang, A. T. Chronopoulos, and C. S. Raghavendra, "Soft-Timeout Distributed Key Generation for Digital Signature based on Elliptic Curve D-log for Low-Power Devices," in First International Conference on Security and Privacy for Emerging Areas in Communications Networks, SecureComm 2005., 2005.