15th International Conference on
AEROSPACE SCIENCES & AVIATION TECHNOLOGY,
ASAT - 15 – May 28 - 30, 2013, Email: asat@mtc.edu.eg ,
Military Technical College, Kobry Elkobbah, Cairo, Egypt,
Tel: +(202) 24025292 –24036138, Fax: +(202) 22621908

An Elliptic Curve Threshold Group Signature Scheme

{A. Kamal, H. Dahshan, A. Rohiem}*

Abstract: In a $(\boldsymbol{t},\boldsymbol{n})$ threshold group signature scheme, only t or more signers of the group can sign messages on behalf of the group. Elliptic Curve Cryptography (ECC) was introduced by Victor Miller and Neal Koblitz in 1985. ECC proposed as an alternative to established public-key systems such as DSA and RSA. The main reason for the attractiveness of ECC is the fact that there is no sub-exponential algorithm known to solve the discrete logarithm problem on a properly chosen elliptic curve. This means that significantly smaller parameters can be used in ECC than in other competitive systems such RSA and DSA, but with equivalent levels of security. Some benefits of having smaller key sizes include faster computations, and reductions in processing power, storage space and bandwidth. This makes ECC ideal for constrained environment such as pagers, PDAs, cellular phones and smart cards. In this paper, we propose a threshold group signature scheme based on elliptic curve discrete logarithm problem. The advantages of the proposed scheme are justified through extensive simulations.

Keywords*:* Threshold group signature; elliptic curve cryptography

1. Introduction

Using elliptic curves for cryptographic protocols has been proposed in [1], [2]. Cryptosystems based on ECDLP can use smaller key size than that is needed by DLP or 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. Table 1 gives approximate comparable key sizes for symmetric systems, Elliptic Curve Cryptography (ECC) systems, and DH/DSA/RSA systems. The estimates are based on the running times of the best algorithms known today [3, 4]. Thus, for example, when securing a 192-bit symmetric key, it is prudent to use either 409-bit ECC or 7680-bit DH/DSA/RSA.

Table 1. Key sizes in bits for equivalent security levels

Strength	ECC	DH/DSA/RSA
804	163	1024
112	233	2048
128	283	3072
192	409	7680
256	571	15360

[*] Egyptian Armed Forces, Egypt.

Group signatures, first introduced by Chaum and van Heyst in [1], allow any member of a group to sign messages anonymously on behalf of the group such that anyone can verify the signature but no one except the group manager can find out which group member creates the signature. Following the first scheme, a number of new group signature schemes have been put forward. However, most of the proposed schemes face two problems which are: how to make the private key more safe? and how to keep the private key effectively?

To solve the two problems, secret sharing was introduced in group signature schemes [2, 5] and therefore forms a new class of group signature scheme, called as the threshold signature scheme. In 2003, M. Bellare proposed a simplified form of the definition of threshold group signature [6].

Then, Guilin Wang first presents a definition of threshold group signature [7]. After that, a lot of threshold signature schemes were proposed. In a (*t*, *n*) threshold group signature scheme, the technique of sharing a secret among *n* users is used. Such a scheme allows only *t* or more users to reconstruct the secret, and then they can generate a group signature. But any *t-1* users have no access to the secret. Threshold group signatures have all the properties of both threshold signatures and group signatures. A secure threshold group signature should meet the following requirements: correctness, unforgeable, anonymity, traceability, robustness, threshold characteristics, coalition-resistance, revocable.

Generally, there is a group manager (GM) in a threshold group signature scheme which accomplishes the final group signature. Recently, *Tong Lu and Baoyuan Kang* proposed a new threshold group signature [8], which is actually a threshold group signature with designated verifiers. And the verifiers are also belong to a group with threshold value “k” , which means only k or more members in this group can verify the signature.

In this paper, we will show the same scheme proposed by *Tong Lu and Baoyuan Kang but* applied with *Elliptic Curve Cryptography (ECC).*

The remainder of this paper is organized as follows. In section 2, we present our improvement scheme. The analysis is described in section 3. Then we discuss the security of the improvement in section 4. At last, we give our conclusions in section 5.

2. Our Improvement Scheme

GM acts to authenticate every participant’s identity, including the sign group and the verify group, keeps and renovates the current Public Key State List (PKSL) of every group. Their scheme is mainly divided into four phases: the Key-Gen phase, the Join phase, the Sign phase, the Verify phase, the Open phase and the Revoke phase.

The system parameters and the notations are defined as follows:

A. Notations and system parameters

U_{s_i} , U_{v_i}: the identity of members of sign group verify group respectively.
U_s = { U_{s_1} , U_{s_2} , ………, U_{s_n}}: the sign group.
U_v = { U_{v_1} , U_{v_2} , ………, U_{v_n}}: the verify group.
(g_s , p_s) : parameters of the sign group.

B. Key-Gen

- According to the given threshold value *t*, every member U_{s_i} randomly selects a secret nonzero integer d_{s_i} in $GF(p_s)$ and selects a (*t-1*) degree polynomial as follows: $f_{s_i}(x) = f_{s_{i.0}} + f_{s_{i.1}}x + f_{s_{i.2}}x^2 + \cdots + f_{s_{i.t-1}}x^{t-1}$, where $f_{s_{i.l}} \neq 0, l = 0,1,2,\cdots,t-1$
- Every member U_{s_i} computes and broadcasts:

$$D_{s_i} = d_{s_i} \odot g_s \quad , \quad f_{s_{i,l}} \odot g_s$$

- Simultaneously, every member computes $f_{s_i}\left(u_{s_j}\right)$ and sends it to U_{s_j} where $j = 0,1,2,\cdots,n$
- After U_{s_j} receives all $f_{s_i}\left(u_{s_j}\right)$, he can verify its validity via the equation:

$$\boldsymbol{f_{s_i}\left(u_{s_j}\right) \odot g_s = \sum_{l=0}^{t} f_{s_{i,l}} \odot g_s \odot u_{s_j}^{l}} \qquad \textbf{(1)}$$

- Every member U_{s_i} computes the followings:

$\mathrm{X}_{\mathrm{S}_i} = \mathrm{d}_{\mathrm{S}_i} + \sum_{j=1}^{n} f_{s_j}\left(u_{s_i}\right)$ which is his private key
$\mathrm{Y}_{\mathrm{S}_i} = \mathrm{X}_{\mathrm{S}_i} \odot \mathrm{g}_\mathrm{S}$ which is his public key
$\mathrm{Y}_{\mathrm{U}_s} = \sum_{i=1}^{n} \mathrm{D}_{\mathrm{S}_i} + \sum_{i=1}^{n} f_{s_{i,0}} \odot \mathrm{g}_\mathrm{S}$ which is the group public key.
$\mathrm{D}_{s_n} = \sum_{i=1}^{n} {D_s}^{-1}$ which is a part of signature

It is the same way in the verify group U_v, every U_{v_i} gets his own private key:

$$\mathrm{X}_{v_i} = \mathrm{d}_{v_i} + \sum_{j=1}^{n} f_{v_j}\left(u_{v_i}\right)$$

and his public key: $Y_{v_i} = X_{v_i} \odot g_v$
and the group public key:

$$Y_{U_v} = \sum_{i=1}^{n}{}^{\oplus} D_{v_i} + \sum_{i=1}^{n} f_{v_{i,0}} \odot g_v$$

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

- The group manager GM selects a (*t-1*) degree polynomial such that:

$J(x) = a_0 + a_1x + a_2x^2 + \cdots + \mathrm{a}_{\mathrm{t-1}}\, x^{t-1}$,
where $a_i \neq 0, i = 0\,,1\,,2\,, \ldots \ldots \ldots \ldots \ldots, t-1$

- Simultaneously, GM computes $J(u_{s_i})$ and sends it to U_{s_i}:
- GM computes and broadcasts:

$$Y = J(0) \odot g_s \ , \ Y_i = J\left(u_{s_i}\right) \odot g_s, where\ i = 1,2,\cdots,n$$

C. Join

After communication keys have been negotiated in the sign group U_s and the verify group U_v, every member $U_{s_{new}}$ and $U_{v_{new}}$ who wants to join sends the following to GM to register via a secure channel.

- GM computes $J\left(u_{s_{new}}\right)$ and sends it to the new member via a secure channel.
- GM computes $Y_{s_{new}} = J(u_{s_{new}}) \odot g_s$.
- GM publishes $\left(new, u_{s_{new}}, D_{s_{new}}, Y_{s_{new}}, T_{new-start}, T_{new-end}\right)$ in the PKSL.

D. Sign

- Every U_{s_i} of the supposed signing participants $\{U_{s_1}, U_{s_2}, \ldots\ldots, U_{s_t})$ randomly selects a secret integer w_{s_i} from [1, q_s - 1] , and computes and broadcasts:

$$W_{s_i} = w_{s_i} \odot g_s \qquad , \qquad z_{s_i} = \left(w_{s_i} * d_{s_i}^{-1}\right) \odot g_s$$

- Then U_{s_i} computes :

$$Z = \sum_{i=1}^{t\ \oplus} z_{s_i} \ , \quad r = Z[1] \qquad (2)$$

$$a_{s_i} = \prod_{j=1, j\neq i}^{n} \frac{u_{s_j}}{u_{s_j} - u_{s_i}}$$

$$s_i = X_{s_i} * a_{s_i} * h(M) - r * J(u_{s_i}) * a_{s_i}$$

- And sends (u_{s_i}, r, s_i) to the appointed group signature generator U_{s_1} .
- After the appointed group signature generator U_{s_1} received the individual signatures (u_{s_i}, r, s_i) he first check its validity by the equation:

$$s_i \odot g_s + (r * a_{s_i} \odot Y_i) = a_{s_i} * h(M) \odot Y_{s_i}$$

- Then he computes :

$$D_{s_A} = \sum_{i=1}^{t} (a_{s_i} \odot D_{s_i}) \qquad (3)$$

$$S = \sum_{i=1}^{t\ \oplus} s_i \qquad (4)$$

- Then he randomly selects an integer w ∈ [1, q_s - 1] and computes :

$$B = w \odot g_v \ , W_{v_i} = -w \odot D_{v_i} \ , \ R_{v_i} = (w \odot Y_{v_i})$$

- At last he broadcasts the computing values and the group signature on the message M: (S , r , Y , Y_{U_S} , D_{S_A} , D_{S_N}, B , W_{v_i} , R_{v_i})

E. Verify

Any k of m members of group U_v can verify the signature on behalf of the group. Firstly, every verifier U_{v_i} ($i = 1, 2, \ldots\ldots, k$) computes:

$E_{v_i} = X_{v_i} \odot B$ and sends (u_{v_i} , E_{v_i}) to the appointed group signature verifier U_{v_1} , U_{v_1} firstly checks its validity by the equation:

$E_{v_i} \odot g_v = R_{v_i}$ and computes:

$e_{v_i} = a_{v_i} \odot E_{v_i}$ (i = 1 , 2 , ……, k)

where $a_{v_i} = \prod_{j=1, j\neq i}^{n} \frac{u_{v_j}}{u_{v_j} - u_{v_i}}$

At last he accepted it as a valid signature by verifying the equation:

$$(S \odot g_s) + (r \odot Y) = (Y_{U_S} \oplus D_{S_A} \oplus D_{S_N}) \odot h(M')$$

3. Security Analysis

In this section, the security of our scheme is discussed on the basis of characteristics of the threshold group signature.

A. Correction

Proof:

$S \odot g_s + \mathrm{r} \odot Y = (\sum_{i=1}^{t} (X_{s_i} * a_{s_i} * h(M)) - \sum_{i=1}^{t} (r * J(U_{s_i}) * a_{s_i})) \odot g_s + (J(0) * r) \odot g_s$

$= (\sum_{i=1}^{t} (X_{s_i} * a_{s_i} * h(M)) - (r * J(0))) \odot g_s + (J(0) * r) \odot g_s$

$= \sum_{i=1}^{t} (X_{s_i} * a_{s_i} * h(M)) \odot g_s$

$= \sum_{i=1}^{t}((d_{s_i} + \sum_{j=1}^{n} f_{s_j}(U_{s_i})) * a_{s_i} * h(M)) \odot g_s$

$= \left(\sum_{i=1}^{t} d_{s_i} * a_{s_i} * h(M)\right) \odot g_s + \sum_{i=1}^{n} f_{s_{i,0}} * h(M) \odot g_s$

$= \left(Y_{U_S} \oplus D_{S_A} \oplus D_{S_N}\right) \otimes h(M')$

B. Unforgeable

In this scheme, the appointed group signer is the most powerful forger. Due to this, we give us attack which can be mounted by the appointed group signer, In this scheme we let GM distributes the secret value (0) , and only t or more members can restrict $f(0)$. Moreover, we put $f(0) \odot g_s$ into the verify equation , which means that if the appointed group signer wants to forge a correct signature he must know $f(0)$ in advance . while , if one wants to get $f(0)$ from $f(0) \odot g_s$, he will face the difficulty of solving elliptic curve discrete logarithm problem, similarly if he wants to forge s_i, he will face with the same problem , because he doesn't know the user U_{s_i} 's private key X_{s_i} . at the same time , as for the other adversary , he doesn't have a valid membership certificate , so he can't forge a group signature satisfying the verification procedure.

C. Anonymity

Since ($s_1 , s_2 , \dots \dots \dots . . , s_t , U_{s_1}$) contains random numbers and no information about signer is revealed. Therefore, it is computationally hard for everyone except the appointed group signer to identify the actual signers.

D. Traceability

($s_1 , s_2 , \dots \dots \dots . . , s_t , U_{s_1}$) And, the malicious signer who sent a invalid values (u_{s_i} , r , s_i) can be identified by the equation:

$$s_i \odot g_s + \left(r * a_{s_i} \odot Y_i \right) = a_{s_i} * h(M) \odot Y_{s_i}$$

E. Threshold characteristics

Only t or more members can reconstruct the secret value (0) , so any $t - 1$ members can not sign a valid signature on behalf of the group.

F. Revocable

GM can revoke any member just by modifying the end time T_{i-end}, and any member revoked cannot take part in the sign procedure due to the PKSL.

4. Efficiency Analysis

In this section, we present an efficiency analysis of our proposed scheme in terms of the computational cost. We denote by *Ω* the computation cost of point multiplication by a scalar in *G*, by *Ψ* a point add operation in *G*. The subtraction operation is considered the same computation cost as the add operation in this analysis. Table 2 shows a summary of the computation performed in each phase in terms of the number of users n, the threshold t, point addition, and point multiplication by a scalar operation in *G*. on the number of users and the number of the users assigned to recover the secret (Threshold number).

Ψ..... point addition.
Ω.... point multiplication.
n.... number of users.
t.... number of signers.

Table 2. Complexity analysis of the proposed scheme

PHASES	COMPUTATION
KEY-GEN PHASE	6nΩ+3nΨ
SIGN PHASE	(5t+1)Ω+3tΨ
VERIFY PHASE	3Ω+3Ψ
TOTAL	(6n+5t+4)Ω+3(n+t+1)Ψ

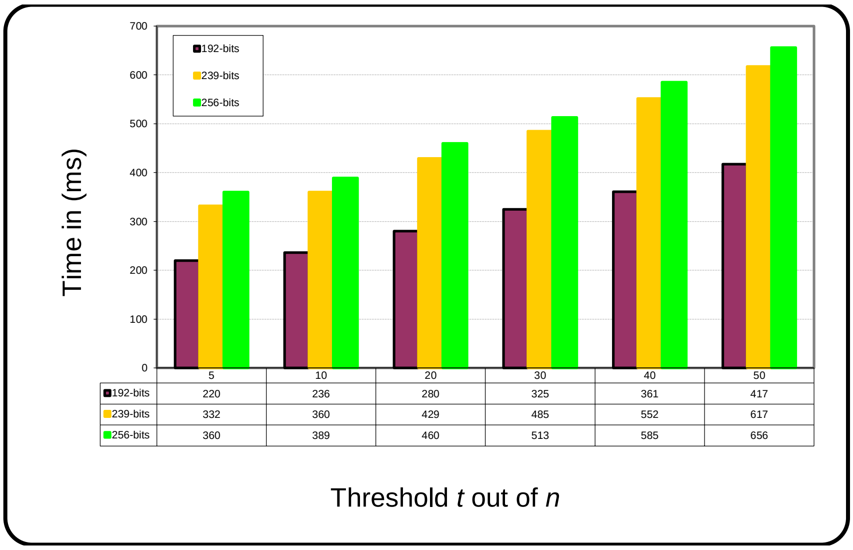
5. Simulation Results

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 [9] and would thus lead to lower performance. On a desktop with an Intel core 2 Duo 2.6 GHz processor and 4GB memory, PARI/GP [9] is used to evaluate the performance of our proposed scheme.

The performance evaluation of the proposed scheme will be given in terms of the minimum threshold t users (out of the total n users) required to collaboratively recover the session secret keys. All results are the average of 10 runs and the total number of users is set to 50 users. The performance of the proposed scheme is evaluated for three different key sizes: 192 bits, 239 bits, and 256 bits. Values that remain constant between different scheme runs (for example, the inner parts of the Lagrange coefficients or the verification at different points) can be precomputed and are therefore not included in the evaluation. The y-axis in the graphs below represents timings in milliseconds while the x-axis represents the threshold *t*.

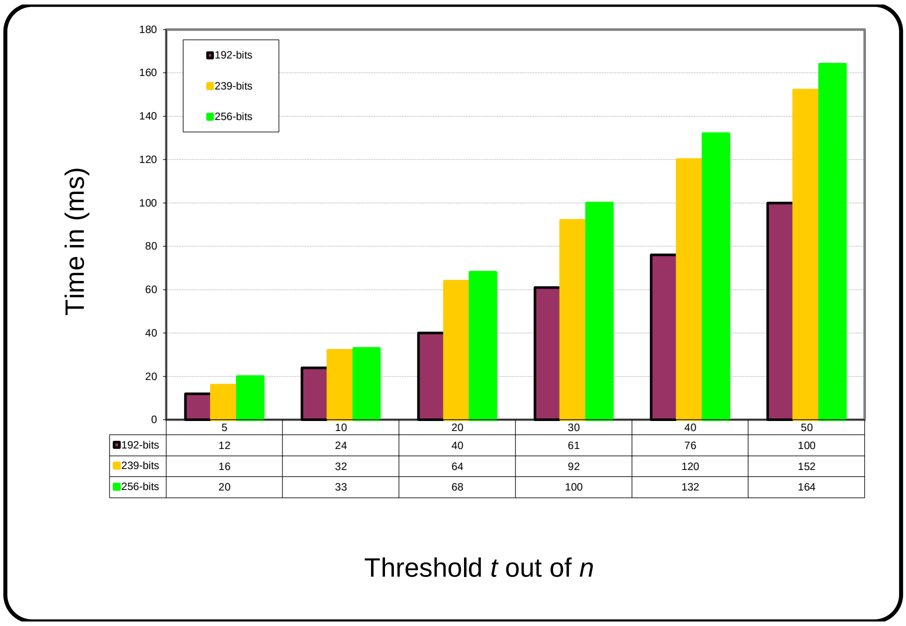
Figure 1 shows that the session key generation time increases with increasing the threshold *t* as a result of increasing the point addition and point multiplication operations of the algorithm with increasing the threshold *t* as shown in equation 1. Increasing the key length from 239 bits to 256 bits has a negligible effect on the timing of the session key generation algorithm.

Figure 2 shows that the processing time in the sign phase increases with increasing the threshold *t* as a result of increased computation overhead. The computation overhead increases as a result of increasing the point addition and multiplication operations with increasing the threshold t as shown in equations 3, and 4.



	5	10	20	30	40	50
192-bits	220	236	280	325	361	417
239-bits	332	360	429	485	552	617
256-bits	360	389	460	513	585	656

Fig. 1: Timing (msec) of Key Generation Phase



	5	10	20	30	40	50
192-bits	12	24	40	61	76	100
239-bits	16	32	64	92	120	152
256-bits	20	33	68	100	132	164

Fig. 2: Timing (msec) of Sign Phase

Figure 3 shows that Increasing threshold t increases the timing in the verify phase as a result of increasing the computation complexity. Increases the threshold t increases the number of point addition and multiplication operation. Increasing the key length from 239 bits to 256 bits has a negligible effect on the timing of the session Verify Phase of the proposed scheme.

Figure 4 shows the total timing of our proposed scheme in the key generation, sign, and verify phase.

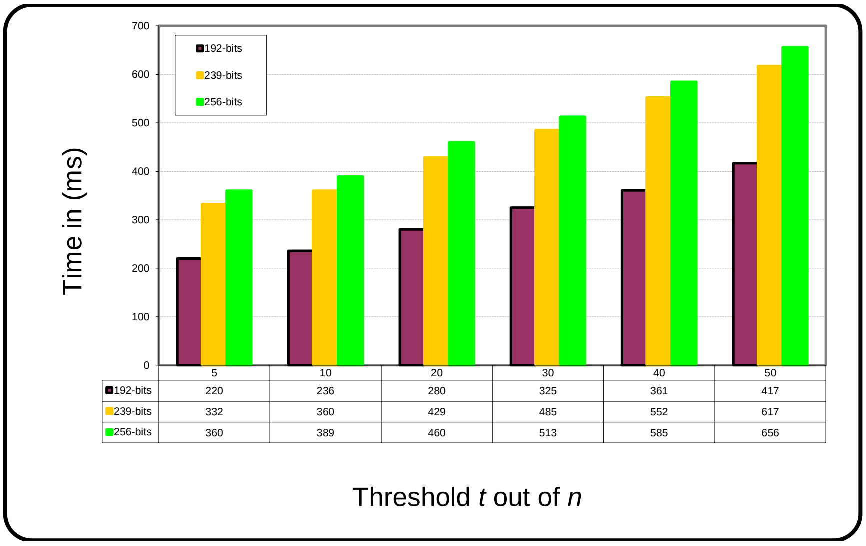


Fig. 3: Timing (msec) of Verify Phase

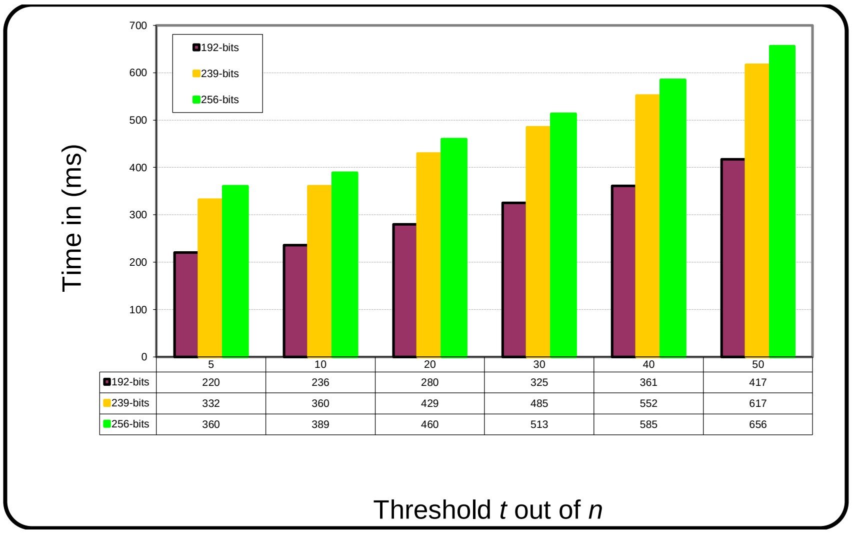


Fig. 4 Total Time

6. Conclusion

In this paper, a threshold signature protocol based on elliptic curve discrete logarithm problem has been proposed. The proposed scheme provides high level of secrecy for small key sizes compared to the Threshold Signature protocols based on discrete logarithm problem (DLP) over a finite field or an integer factorization problem (IFP). From the results, our proposed scheme has moderate timings, and timing do not vary significantly with changing the key size which reflects the suitability of the proposed scheme for applications where the devices are resource constrained such as mobile phones, PDAs, and sensor nodes.

7. References

[1] D. Chaum and E.van Heyst, “Group Signature”, Advances in Cryptology EUROCRYPT'91, vol.547 of Lecture Notes in Computer science, Springer-Verlag, 1991, pp.257-256.

[2] C.M. Li, T. Hwang, and N.Y. Lee, “Threshold-Multisignature Schemes where Suspected Forgery Implies Traceability of Adversarial Shareholders”, In: Eurocrypt'94, LNCS 950, pp. 194-204, Springer-Verlag, Berlin, 1995.

[3] K. Lauter, “The Advantages of Elliptic Curve Cryptography for Wireless Security,” IEEE Wireless Communications, vol. 11, pp. 62–67, Feb 2004.

[4] M. AYDIN and G. AYDIN, “A Survey Of Elliptic Curve Cryptography,” Journal of Electrical & Electronics Engineering, Istanbul University, vol. 6, no. 2, pp. 211–221, 2006.

[5] W.B. Lee and C.C. Chang, “(t, n) Threshold Digital Signature with Traceability Property”, JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 15, pp.669-678, 1999.

[6] M.R Bellare, D. Micciancio and B.Warinschi, “Foundation of group signatures: formal definitions, simplified requirements, and a construction based on general assumptions”, Proc of EUROCRPT 2003, LNCS2656. Berlin: Springer- Verlag, 2003, pp.614-629.

[7] G.L.Wang. “On the Security of the Li-Hwang-Lee-Tsai Threshold Group Signature Scheme”, Information Security and Cryptography (ICISC 2002), LNCS 2587, pp.75-89, Springer-Verlag, Berlin, 2003.

[8] F.Li; J.Yu; H.Ju; "A New Threshold Group Signature Scheme Based on Discrete Logarithm Problem", *Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007. SNPD 2007. Eighth ACIS International Conference on*, vol.3, no., pp.1176-1182, July 30 2007-Aug. 1 2007.

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