An Efficient LU Decomposition-based Key Pre-distribution Scheme for Ensuring Security in Wireless Sensor Networks

Al-Sakib Khan Pathan, Tran Thanh Dai, and Choong Seon Hong
Department of Computer Engineering, Kyung Hee University, South Korea
spathan@networking.khu.ac.kr, daitt@networking.khu.ac.kr, and cshong@khu.ac.kr

Abstract

In this paper we propose an improved and efficient key pre-distribution scheme for wireless sensor networks. Our scheme significantly increases the storage efficiency and security than that of the previously proposed scheme based on LU decomposition. It also ensures node-to-node mutual authentication in a way that an eavesdropper cannot harm the network by monitoring the traffic flows in the network. We also present detailed performance analysis and compare our scheme with the existing schemes.

1. Introduction

Wireless Sensor Networks (WSNs) are expected to serve many general purpose applications for the mass public as well as to contribute significantly to military operations, medical, disastrous and emergency situations in the near future. However, there are still a lot of unresolved issues in WSN of which security is one of the hottest research issues [1], [2]. Security becomes the prime concern when wireless sensors are deployed in hazardous, hostile or more specifically in military reconnaissance scenario. The sensory data collected from various sources are often considered as secret and not intended to be disclosed in public. Hence, there must be some sorts of mechanisms for node to node data transmission with confidentiality and authenticity of data. Efficient key management schemes could solve this issue. However, the sensors are tiny devices which have low memory, processing power and limited battery power [3], [4], [5], [6] for which, the key management schemes like trusted server, Diffie-Hellman or public-key based schemes used in other networks are not feasible to be applied directly in wireless sensor networks [7], [8], [9], [10].

To address the key agreement problem in WSNs, there are mainly three categories of schemes: key distribution center scheme, public key scheme, and key pre-distribution scheme. Constrained computation and energy resources of sensor nodes often make the first two schemes infeasible for wireless sensor networks [7], [8], [10]. The third category of key management scheme is key pre-distribution scheme, where keying materials are delivered to all sensor nodes prior to deployment [8], [9], [10]. The intent of this paper is to propose an efficient key pre-distribution scheme for ensuring better security and resilience of the network. Our scheme uses LU decomposition adopted from [11], [15] but significantly improves the security of the network. In addition to this, we also propose an efficient method to store the keying information in the sensor nodes. Main contributions of our work are:

1. A new key predistribution scheme based on LU decomposition with significantly better security than the previously proposed schemes
2. Introducing an efficient encoding mechanism for storing pre-distributed keying information in the sensor nodes to reduce network-wide storage usage
3. Rigorous guarantee of finding common keys between any two nodes in the network
4. Detailed performance analysis and comparison of our scheme with previously proposed schemes.

The organization of this paper is as follows: Following the Section 1, Section 2 mentions the related works, Section 3 presents our proposed scheme, Section 4 deals with the detailed performance analysis and comparison and Section 5 concludes the paper with future research directions. Because of the page constraints we have shortened some of the parts.

2. Related Works

Eschenauer and Gligor [9] proposed a random key pre-distribution scheme (often termed as basic probabilistic key sharing scheme) which is based on the assignment of key rings to each of the sensors. Choi and Youn [11] proposed a key pre-distribution

This work was supported by MIC and ITRC projects
Dr. C. S. Hong is the Corresponding Author

Proceedings of The Sixth IEEE International Conference on Computer and Information Technology (CIT'06)
0-7695-2687-X/06 $20.00 © 2006 IEEE

scheme guaranteeing that any pair of nodes can find a common secret key between themselves by using the keys assigned by LU decomposition of a symmetric matrix of a pool of keys. Chan et al. [7] presented three mechanisms for key establishment using the framework of pre-distributing a random set of keys to each node. Du et al. [12] proposed a method to improve the basic scheme by exploiting a priori deployment knowledge. They also proposed a pairwise key pre-distribution scheme for wireless sensor networks [8], which uses Blom's key generation scheme [13] as a building block to generate multiple key spaces. Then it utilizes the idea of the basic scheme to establish a common secret key between any pair of nodes. Liu et. al. [10] developed a general framework for pairwise key establishment based on the polynomial-based key pre-distribution protocol in [14] and the probabilistic key distribution in [7] and [9].

3. Our Key Pre-distribution Scheme

In this section we propose our scheme. Before stating the details and the analysis part, we briefly mention the terms and preliminaries for our scheme.

3.1. LU Decomposition

LU decomposition [15] is a procedure for decomposing a square matrix A (dimension $N \times N$) into a product of a lower triangular matrix L and an upper triangular matrix U, such that,

$$A = LU \tag{1}$$

Where, lower triangular matrix L and upper triangular matrix U have the forms,

$$L_{ij} = \begin{cases} l_{ij} & for \quad i \geq j \\ 0 & for \quad i < j \end{cases} \quad and \quad U_{ij} = \begin{cases} u_{ij} & for \quad i \leq j \\ 0 & for \quad i > j \end{cases}$$

So, for a square matrix of 4 × 4 equation (1) looks like:

$$\begin{bmatrix} l_{11} & 0 & 0 & 0 \\ l_{21} & l_{22} & 0 & 0 \\ l_{31} & l_{32} & l_{33} & 0 \\ l_{41} & l_{42} & l_{43} & l_{44} \end{bmatrix} \cdot \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ 0 & u_{22} & u_{23} & u_{24} \\ 0 & 0 & u_{33} & u_{34} \\ 0 & 0 & 0 & u_{44} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \tag{2}$$

Now, according to the definition, elementary matrix E is an $N \times N$ matrix if it can be obtained from the identity matrix I_n by using one and only one elementary row operation (e.g., elimination, scaling, or interchange) [16]. Elementary row operations are, $R_i \leftrightarrow R_j$, $cR_i \leftrightarrow R_i$, $R_i + cR_j \leftrightarrow R_i$. If the elementary matrices corresponding to the row operations that we use are, E_1, E_2 ..... E_k, then, $E_k....E_2E_1A = U$. Hence, $A = (E_k....E_2E_1)^{-1}U$ or $L = E_k^{-1} ... E_2^{-1} E_1^{-1}$

3.2. Details of Our Scheme

3.2.1. Before deployment key-predistribution. Key-predistribution is done in four distinct but interrelated steps.

Step 1 (Key pool generation). In this step, a large pool of keys P with size *s* is generated along with their identifiers. This pool of huge number of keys is used for the symmetric matrix formation step.

Step 2 (Symmetric key matrix formation). In this step, a symmetric key matrix A ($N \times N$) is generated where N is the maximum number of sensor nodes that can be deployed and has to satisfy condition (if distinct key is to be used along a row or a column in the symmetric matrix),

$$\frac{N^2 - N}{2} + N = s \tag{3}$$

Hence,

$$N = \left\lfloor \frac{-1 + \sqrt{1 + 8s}}{2} \right\rfloor \tag{4}$$

Equations (3) and (4) denote that, if the size of the key pool P is known, the number of sensor nodes that P can support could be calculated using (4) and vice versa using (3). Each element A_{ij} of A is assigned a distinct key from key pool P such that $A_{ij} = A_{ji}$ for, $i,j = \overline{1, N}$

Step 3 (LU decomposition). LU decomposition is applied on the matrix A in this step. As the results of the decomposition are two matrices (L, U), both of them are $N \times N$ matrices and follow equation (1) or (2).

Step 4 (LU key pre-distribution and encoded storage). Now, every sensor node is randomly assigned one row from the L matrix and one column from the U matrix, following the condition that, the *i*th row of L, $L_r(i)$ and the *i*th column of U, $U_c(i)$ always go together when assigned to a sensor node. Moreover, for storing the row and column information in the sensor, we apply an encoding scheme to increase the efficiency of memory usage throughout the network. It could be noticed that, each row of L or each column of U matrices has two parts: one non-zero-element part and another zero-element part. The zero-element part might contain (N-1) elements at the best while in the worst case, zero-element part could be absent in a row or in a column. Figure 1 shows the scenario.

For each row of L or each column of U, the first portion consists of nonzero elements. The second portion consists of zero elements (might be absent). Therefore, to store one row of L and one column of U in each sensor, we only need to store each element in the nonzero portion and one value specifying the number of following zeros in the zero-element portion. The non-zero part could actually contain one or more

zeros which are treated as non-zero values and stored accordingly. This storing method is extremely effective if the size of network is large or the dimension of the matrix A is large. Later, we examine the effectiveness and efficiency of this encoding mechanism.

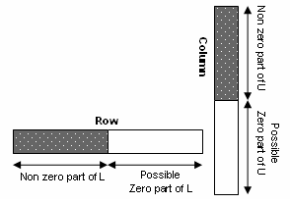

Figure 1. Structure of L's row and U's column

3.2.2. After Deployment Pairwise Key derivation. Let us assume that sensor with id, S_x contains $[L_r(i), U_c(i)]$ and with id, S_y contains $[L_r(j), U_c(j)]$. When S_x and S_y need to find a common secret key between them for communication, they first exchange their columns, and then compute vector products as follows:

$$S_x: L_r(i) \times U_c(j) = A_{ij}$$
$$S_y: L_r(j) \times U_c(i) = A_{ji}$$

As A is the symmetric matrix, $A_{ij} = A_{ji}$

A_{ij} (or A_{ji}) is then used as a common key between S_x and S_y. For the secure exchange of the column information, we apply the following procedure. Let's suppose given A is,

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 5 & 6 & 7 \\ 3 & 6 & 8 & 9 \\ 4 & 7 & 9 & 10 \end{bmatrix}$$

As the procedure stated earlier we calculate the elementary matrices, $E_1, E_2 \ldots E_k$ and eventually get,

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 3 & 0 & 1 & 0 \\ 4 & -1 & 3 & 1 \end{bmatrix} \text{ and } U = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & -1 & -3 \\ 0 & 0 & 0 & 2 \end{bmatrix} \quad (5)$$

Let us consider that, S_x contains 2nd row of L and 2nd column of U from equation (5). For S_y, it contains 3rd row of L and 3rd column of U. Hence, the sample scenario is:

S_x: $L_r(i)$ = [2 1 0 0] $U_c(i)$ = [2 1 0 0]
S_y: $L_r(j)$ = [3 0 1 0] $U_c(j)$ = [3 0 -1 0] (6)

When two nodes S_x and S_y want to communicate with each other, they need to regenerate the zeros in the rows and columns assigned to them. The following procedure is followed in our scheme,

1. S_x sends $U_c(i)$ to S_y.

2. S_y re-generates the zeros in $L_r(j)$ and $U_c(i)$, computes the possible pairwise key, K_{ji} as, $L_r(j) \times U_c(i) = K_{ji}$ (In our example, $K_{ji} = 6$)

3. S_y sends $U_c(j)$ and its id S_y to S_x.

4. Now S_x re-generates the required zeros in the zero-element portion and computes the possible pairwise key K_{ij} as, $L_r(i) \times U_c(j) = K_{ij}$ (In our example, $K_{ij} = 6$. At this stage both of the nodes have calculated the possible pairwise key locally)

5. S_x sends $K_{ij}(S_y)$ to S_y which is the encrypted id of S_y by possible key K_{ij}.

6. S_y uses the key K_{ji} to decrypt $K_{ij}(S_y)$ and eventually finds the value S_y which is its own id. Once, S_y could find the value S_y, it could be sure that S_x is a valid node as the calculated key of S_x matches with the calculated key of S_y. Now, S_y uses its key K_{ji} to generate MAC (Message Authentication Code) and sends K_{ji}(CLR, MAC(S_y, CLR)) that is, it agrees with the key of S_x. This CLR message is the confirmation that the node S_y agrees with S_x for the locally computed keys. RC5 [18] could be used to calculate the MAC using the key K_{ji}. Now as A_{ij} is symmetric, $K_{ij} = K_{ji}$. (For example, following the procedure, in case of equation (6) the common key is 6).

4. Performance Analysis and Comparison

We analyze our scheme from the perspectives: (a) what is the storage-efficiency of this scheme, (b) what is the computational overhead, (c) what level of security it could ensure (d) connectivity of the network and other related analysis. For each of these parameters we mention how our scheme performs in comparison with the previously proposed scheme [11] and then with some of the other schemes. We also mention the limitations of our proposal.

4.1. Storage Analysis

Our scheme outperforms Choi-Youn scheme [11] in terms of network-wide storage efficiency. We propose an efficient method to store the row and column information of L and U matrices which was not present in [11]. In fact in that scheme, when the network size is large, in the worst case, the sensors in the network initially have to memorize the full length of all keying information in rows and columns of L and U matrices respectively. Here, we compare memory usage for keying information storage in [11] and in our proposed scheme. The followings are some notations that will be used later:

- k – length or the number of bits for each keying information in L or U matrix
- N – maximum number of sensor nodes that are to be deployed in the network
- w_i – the total number of nonzero elements in a row of L and in a column of U stored in sensor node with id i
- z – the number of bits needed so that the largest number of zero elements in the zero-element

part in a row of L or in a column of U could be represented

In [11], the memory needed to store keying information in the network-wide scale is,

$$\Gamma_{\text{Choi-Youn}} = 2 \times N^2 \times k \quad (7)$$

In our scheme, the memory usage to store keying information in each individual sensor node is,

$$\gamma_i = (w_i \times k + 2 \times z) \quad (8)$$

It follows that the network-wide memory usage is (Figure 2),

$$\Gamma_{Our} = \sum_{i=1}^{N} \gamma_i = k \times \sum_{i=1}^{N} w_i + N \times (2 \times z)$$

$$= k \times \frac{N \times (N+1)}{2} + N \times (2 \times z)$$

$$= k \times \frac{N \times (N+1)}{2} + N \times (2 \times ceil\,(\log_2 (N-1))) \quad (9)$$

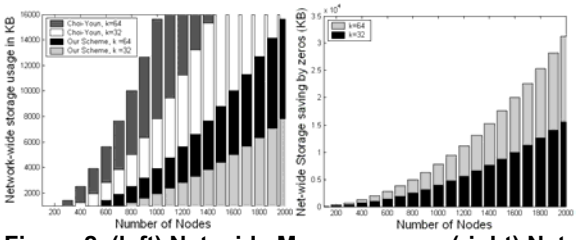

Figure 2. (left) Net-wide Memory usage (right) Net-wide storage saving by encoding (zero-elements) for variable length of k (value of k could be smaller)

In our scheme, major memory saving is done by encoding the zeros in the zero-element parts of the L and U matrices. Hence, Network-wide memory saving considering only the encoding of the zero-element portions, could be obtained by (for [11] equation (10) becomes zero),

$$\Gamma_{\text{Saving\_zero\_elements}} = 2 \times k \times \frac{N \times (N-1)}{2} - N \times (2 \times z)$$

$$= k \times N \times (N-1) - N \times (2 \times ceil\,(\log_2 (N-1))) \quad (10)$$

4.2. Computational Resource Utilization

Re-generation of the zeros in the exchanged rows and columns incurs some computational overhead which was not present in the previous scheme but this encoding mechanism saves memory resources. The encryption operations and MAC generation also incur extra computations but if there is no encryption of the messages during pairwise key establishment, the eavesdropper could get important information just by listening the traffic flow. In our proposal, both the initiator node and responder nodes perform one encryption and one decryption operations. Besides these, two nodes need to generate and verify MAC using their keys. So, our scheme requires more computation at the time of pairwise secret key computation than that of the previously proposed [11], but we show that, for ensuring secrecy of the key and better security of node-to-node communication, this trade-off is necessary.

4.3. Security Analysis

Choi-Youn [11] Scheme is severely vulnerable to man-in-the-middle attack or eavesdropping. While exchanging the information for establishing a pairwise key, there is no effective mechanism to hide the keys and messages exchanged between two nodes. Hence, it is easy for an adversary to listen to the traffic flow and collect the information to extract the key K_{ij} or K_{ji}. In this way, an adversary could capture multiple keys and if it captures the ids of the corresponding nodes, it could be able to launch sybil attack [2], [17]. Compared to this, in our scheme, the key K_{ij} or K_{ji} is never directly exchanged between the two communicating nodes rather these are calculated locally by the individual nodes. Even if the adversary knows the ids of the nodes, it could get little information by eavesdropping the message-exchange between two nodes. As encryption is used in some steps, it becomes harder for the adversary to follow and understand the exchanged messages. The adversary must know some other information like the encryption algorithm used and the information about the rows taken from the matrix L. Let us analyze what happens when a key or a node is captured in our scheme.

If a Key is Compromised. In our scheme, if a key is compromised, only one communication link is affected if distinct keys are used in a row or a column of the generating matrix (A). As two nodes are associated with each communication link (or two nodes at two ends of a communication path), number of nodes affected in such a case is two. However, rest of the links are not affected by this compromise in any way. To stop the illegal communication of the adversary with those nodes, in this scenario, the base station revokes that particular key. Let us consider that, δ ($\delta \geq 1$) denotes the number of times the same key is used along a row or a column in the generator symmetric matrix and ϕ is the number of compromised keys. Hence, if any one of the δ number of same keys (along a row or a column) is compromised, all the links and nodes associated with that particular key, could be affected. When distinct keys are used along a row or a column, δ=1 and in this case, compromise of one key (ϕ=1) could affect 1 link and 2 nodes. If δ=2 and ϕ=2, number of links that could be affected is, $\phi \times \delta$ and

number of nodes that are vulnerable is $2\times\phi\times\delta$. In Figure 3, we show the number of nodes and links that could be affected for different values of δ and ϕ. From the figure it is clear that, smaller value of δ is better for the greater resilience of the network (best case, when δ=1).

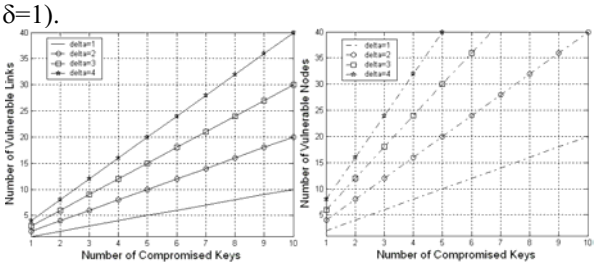

Figure 3. Number of vulnerable links and nodes for different values of ϕ and δ (delta)

If a Node is Compromised. If a node is compromised along with its id, that is when an attacker knows the id and corresponding row and column information of a legitimate node, our scheme becomes vulnerable as with these information, a key could be derived with any other node in the network. To handle such a case, there is an Intrusion Detection System (IDS) in the network which detects the abnormal behavior of any intruder and accordingly the base station gets involved to revoke that particular id and row-column information so that no other valid node communicates with the rogue node. The details of the IDS is beyond the scope of this paper and will be presented in our future works.

While sending the CLR message from the receiver node to the sender node, (see section 3.2) we encrypt the CLR and use the MAC so that the eavesdropper cannot modify the confirmation message and authenticity is ensured. If no encryption is used, the man-in-the-middle attacker could substitute CLR part from the open message with other invalid information to cause Denial-of-Service [2] attack.

During the pairwise key derivation, we use encryption and different formats of messages in different steps, so that an eavesdropper cannot infer whether or not the same operation is performed on the exchanged messages all the times; thus making it more puzzling for the attacker.

4.4. Key Graph Connectivity

As in our scheme any two nodes in the network can derive a common pairwise key, if the nodes are reachable via a path in the network, that path could surely be used for secure communication between them. If required, all the neighboring nodes could derive pairwise keys among them and thus, the network becomes a fully secured connected graph. However, in this case, if encryption and decryption of messages are done at each hop, the energy consumption and processing overhead of the sensors and the overall network is increased significantly. We assume that, each node does not necessarily need to derive a secret key with all the other nodes, however, this depends on the level of security needed in the network (e.g. military reconnaissance scenario where security is the top priority and some additional costs are affordable).

4.5. Further Analysis

In the L and U matrices, the last row and the last column do not have any zero-element portion. In the worst case, when N is very large, a sensor might have to store huge number of elements in the non-zero-element portion without any encoding. A sensor might not be able to store all the non-zero-part-elements for its own storage constraints. In such a case, to avoid the storage-load on a single sensor, the last row and last column could be kept unutilized. Same strategy might be applied for some of the other rows and columns with relatively less number of zeros in the zero-element parts. However, in this case, the number of nodes that could be supported by a symmetric matrix becomes less than that is estimated. This sort of decision (not using) depends on the situation at hand and the requirements for deploying the network.

Now we compare our scheme with [9] (*basic scheme*) in brief. In [9], each sensor node has to memorize the full length of each key in its key ring. It is even worse when many keys in the key ring are assigned to a particular node but they are not utilized after deployment, which is simply the wastage of storage of the tiny resource-constrained sensors. In [9], if one key is compromised by an adversary, it might lead to the compromise of another link between pairs of uncompromised nodes using the same key. In our scheme, keeping the value of δ equal to 1 could keep the rest of the links and nodes unaffected when one key is compromised.

The idea of [9] is that the shared key between any two neighboring nodes is found based on a probability of the occurrence of a key in their key rings. It implies that there still exists some probability that any two neighboring nodes cannot find a common shared key. Therefore, there is a probability that, a node might need to use a multi-link path to establish a pairwise key even with one or more of its neighbors. This surely incurs extra communication overhead on the intermediate connecting nodes. In the worst case, a

disconnected key graph might be created. Hence, it means that insecure communication links could exist in the network. On the contrary, our proposed scheme overcomes this drawback by guaranteeing that any two neighboring nodes could directly derive a pairwise key.

In our scheme, communication overhead for pairwise key establishment is considerably reduced as the traffic in the initial broadcast of each node is reduced. In the Eschenauer-Gligor scheme [9], the information need to be transmitted is the list of identifiers of the keys stored in a sensor node's key ring. Hence, the traffic is much higher in that case than that of our scheme because in our proposed scheme the only information needed to be broadcast is receiver's (S_y) identifier to query the location of the receiver. Furthermore, there is no need to include a path-key establishment phase like that is in [9].

5. Conclusions and Future Works

In this paper, we proposed an efficient key pre-distribution scheme which significantly improves the network-wide memory usage and security. The network-wide memory saved by encoding might be used for processing distributed operations in the network. We have shortened the detailed description, related works and the analysis part because of the constraint of the number of pages. In our future work, we will develop an Intrusion Detection System which will run side-by-side our scheme for successful detection of abnormal behavior or the presence of a rogue node in the network. Also we will investigate how the computational costs could be reduced to increase the efficiency of our scheme.

6. References

[1] Karlof, C. and Wagner, D., "Secure routing in wireless sensor networks: Attacks and countermeasures", Elsevier's Ad Hoc Network Journal, Special Issue on Sensor Network Applications and Protocols, September 2003, pp. 293-315.

[2] Pathan, A-S. K., Lee, H-W., and Hong, C. S., "Security in Wireless Sensor Networks: Issues and Challenges", Proceedings of the 8th IEEE ICACT 2006, , Volume II, Phoenix Park, Korea, 20-22 February, 2006, pp. 1043-1048.

[3] Perrig, A., Szewczyk, R., Wen, V., Culler, D., and Tygar, J. D., "SPINS: Security Protocols for Sensor Networks", Wireless Networks, vol. 8, no. 5, 2002, pp. 521-534.

[4] Jolly, G., Kuscu, M.C., Kokate, P., and Younis, M., "A Low-Energy Key Management Protocol for Wireless Sensor Networks", Proc. Eighth IEEE International Symposium on Computers and Communication, 2003. (ISCC 2003). vol.1, pp. 335 - 340.

[5] Rabaey, J.M., Ammer, J., Karalar, T., Suetfei Li., Otis, B., Sheets, M., and Tuan, T., "PicoRadios for wireless sensor networks: the next challenge in ultra-low power design" 2002 IEEE International Solid-State Circuits Conference (ISSCC 2002), Volume 1, 3-7 Feb. 2002, pp. 200 – 201.

[6] Hollar, S, "COTS Dust", Master's Thesis, Electrical Engineering and Computer Science Department, UC Berkeley, 2000.

[7] Chan, H., Perrig, A., and Song, D., "Random key predistribution schemes for sensor networks", Proc. 2003 IEEE Symposium on Security and Privacy, 11-14 May 2003, pp. 197-213.

[8] Du, W., Deng, J., Han, Y. S., Varshney, P. K., Katz, J., and Khalili, A., "A Pairwise Key Predistribution Scheme for Wireless Sensor Networks", ACM Transactions on Information and System Security, Vol. 8, No. 2, May 2005, pp. 228-258.

[9] Eschenauer, L. and Gligor, V. D., "A key-management scheme for distributed sensor networks", Proceedings of the 9th ACM conference on Computer and Communications, Washington, DC, USA, 18-22 Nov. 2002, pp. 41 - 47.

[10] Liu, D., Ning, P., and Li, R., "Establishing Pairwise Keys in Distributed Sensor Networks", ACM Transactions on Information and System Security, Vol. 8. No. 1, Feb. 2005, pp. 41-77.

[11] Choi, S. and Youn, H., "An Efficient Key Pre-distribution Scheme for Secure Distributed Sensor Networks", EUC Workshops 2005, IFIP International Federation for Information Processing, LNCS 3823, 2005, pp. 1088-1097.

[12] Du, W., Deng, J., Han, Y. S., Chen, S., and Varshney, P.K., "A key management scheme for wireless sensor networks using deployment knowledge", Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies. INFOCOM 2004, Volume 1, 7-11 March 2004 pp. 586-597.

[13] Blom. R., "An optimal class of symmetric key generation systems", Advances in Cryptology: Proceedings of EUROCRYPT 84 (Thomas Beth, Norbert Cot, and Ingemar Ingemarsson, eds.), Lecture Notes in Computer Science, Springer-Verlag, 209, 1985, pp. 335-338.

[14] Blundo, C., Santis, A. D., Herzberg, A., Kutten, S., Vaccaro, U., and Yung, M., "Perfectly-secure key distribution for dynamic conferences". In Advances in Cryptology – CRYPTO '92, LNCS 740, 1993, pages 471-486.

[15] Zarowski, C. J., "An Introduction to Numerical Analysis for Electrical and Computer Engineers", Hoboken, NJ John Wiley & Sons, Inc. (US), 2004.

[16] Nakos, G., and Joyner, D., Linear Algebra with Applications, Brooks/Cole USA, 1998, pp. 188-194.

[17] Newsome, J., Shi, E., Song, D, and Perrig, A, "The sybil attack in sensor networks: analysis & defenses", Proc. of the third international symposium on Information processing in sensor networks, ACM, 2004, pp. 259-268.

[18] A. Menezes, P. Oorschot, S. Vanstone, Handbook of Applied Cryptography, CRC Press, Boca Raton, 1996.