MPKMS: A Matrix-based Pairwise Key Management Scheme for Wireless Sensor Networks

Fatemeh Banaie[1], Seyed Amin Hosseini Seno[1], Reem Hajaj Aljoufi[2], and Rahmat Budiarto[2]
[1]Network Research Laboratory, Department of Engineering, Ferdowsi University of Mashhad
[2]College of Computer Science and Information Technology, Al Baha University, Kingdom of Saudi Arabia
Email: Banaie.F@stu-mail.um.ac.ir, Hosseini@um.ac.ir, abcmcba@hotmail.com, rahmat@bu.edu.sa

***Abstract*--- Due to the sensitivity of the Wireless Sensor Networks (WSN) applications and resource constraints, authentication and key management emerge as a challenging issue for WSN. In general, various approaches have been developed for the key management in WSN. This paper has come up with a new robust key pre-distribution scheme using random polynomial functions and matrix. This new proposed scheme significantly increases the storage efficiency and provides resilience to network against node capture by using random prime numbers, polynomial functions and matrix properties. The effectiveness of the scheme is demonstrated through a security analysis and comparison with the existing schemes.**

***Key words**: Resource constraint; authentication; key management; pre-distribution; polynomial functions.*

I. Introduction

Nowadays, Wireless Sensor Networks (WSNs) are increasingly used for large range of military and civilian applications, including environmental monitoring, target tracking, agricultural irrigation, battlefield surveillance and scientific exploration. In particular, most of these critical applications require sensor nodes to be deployed in unattended and adversarial environments [1]. Therefore, secure data transferring is an essential issue in these applications. In WSNs, data are transmitted by wireless communications. This makes them vulnerable to attacks, such as eavesdropping, impersonation and modification of data [2]. Hence, an efficient key management scheme should be used to achieve data confidentiality, integrity and authentication between communicating parties.

On the other hand, the sensors are tiny devices which have limited energy source, small memory footprint, limited processing power and communication capabilities. So, key management schemes like trusted server and public-key based methods used in other networks are not suitable paradigms for securing exchange in WSN [3]. As a result, key management schemes should be paid more attention in wireless sensor networks.

In this work, our aim is to propose an efficient key pre-distribution scheme for efficiently securing communication and resilience of the network, named as Matrix-based Pairwise Key Management Scheme (MPKMS). Our scheme is based on random function pre-distribution. It uses polynomial function and matrix to ensure better security.

The reminder of this paper is organized as follows. Section II presents some key management schemes in WSN. Defintions and notations are described in Section III, then followed by Section IV that discusses the proposed scheme. Section V describes the detailed performance analysis and comparison. In Section VI, we end up this paper with some conclusions.

II. Related Work

There are a lot of ongoing researches on key management in wireless sensor network. A basic approach is to preload a single master key to all sensor nodes, and any pair of nodes uses this global master key to establish a secure communication link between them. This approach has very low network resilience, because if an adversary captures a node, the security of the entire network is compromised [4].

Another approach is to use distinct pair-wise keys for all possible pairs of sensors in the network. Although this solution has very good resilience against attacks, but it is impractical to stores all the keys on each sensor and it is not scalable for large WSNs [5].

Eschenauer and Glior [6] proposed a random key pre-distribution scheme, which addresses the redundancy in previous scheme and yet provides good key resilience. The base station first generates a large pool S of keys. Then k keys is randomly selected from the key pool and stored in each node, which is called the node's key ring (Fig.1). This scheme consists of three phases as follows:

Phase I: (the initialization). Each sensor is pre-loaded with a key ring from *S*. the number of keys in the key pool and key rings are chosen such that the intersection of two key rings is not empty with some probability *P*.

Phase II: (the discovery of shared key). After deployment, sensors broadcast the list of key identifiers that they maintain. Neighboring nodes can check whether they have a common key in their key ring. Then they can use shared key to create safe communication channels.

Phase III: (The setup of path key). If sensors cannot find common keys from their key rings, they can set up path key with the nodes in their neighbors.

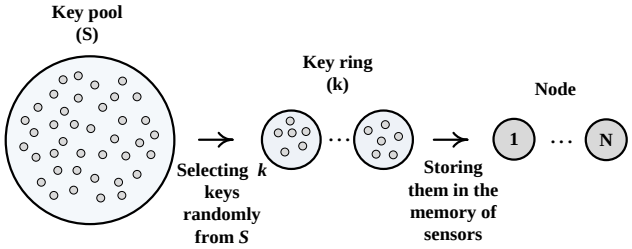

Figure 1. The process of key distribution.

In this approach, if the sensor nodes are progressively captured, the attacker may discover a large part of the pool.

Chan et al. [7] proposed a key pre-distribution scheme called *Q-composite*, in which the amount of key overlap required in *the discovery of shared key* phase. It means that two neighboring nodes require at least *Q* keys to establish a secure link. This approach improves resilience against node capture, because the attacker needs to compromise many more nodes in order to break a secure link. But it is not guarantee to find shared key between any two nodes because of the need for sharing *Q* common keys instead of one.

In [8] and [9], the authors applied combinatorial design for key pre-distribution based on Symmetric Balanced Incomplete Block Design (SBIBD). The system generates $k = m + 1$ key rings from key pool of $m^2 + m + 1$ keys. In this scheme each pair of nodes share exactly one common key. However, it is not scalable in large networks.

Liu and Ning in [10] proposed to use symmetric bivariate polynomials ($f(x, y) = f(y, x)$) instead of keys. Sensor nodes can generate a common group key by evaluating these polynomials. This approach enhances resilience against node captured by computing distinct secret keys, but it increases the memory usage and the computational overhead.

Blom [11] proposed a λ-secure symmetric key generation that uses a public matrix $G_{(\lambda+1)\times n}$ and a private symmetric matrix $D_{n\times(\lambda+1)}$. The scheme keys are secure if no more than λ nodes are compromised. Then, the matrix of pairwise keys of node n is $k = G^T DG$. Yu and Guan [12] used Blom's scheme and introduced a key pre-distribution scheme for WSN. The introduced scheme, sensor nodes are deployed into a grid and a distinct secret matrix is assigned to each group.

III. Preliminaries

This section gives a brief description of preliminaries and various concepts used in this paper.

1. *Network Model:* We assume that deployment region is logically divided into N_c equal-size sub areas. The whole network is composed of a large number of static wireless sensor nodes, cluster heads and a single base station. Sensor nodes are deployed into sensor field uniformly. They also partitioned into t groups (clusters), with $N = \frac{M}{N_c}$ nodes in each cluster (Fig. 2).

Sensor nodes have constrained resources such as limited computational, memory, bandwidth and short radio transmission range. They only can communicate with base station and nodes in other clusters via cluster heads in order to reduce energy consumption in the network. Cluster heads are equipped with more energy resources than sensors.

Once an adversary captures a node, it can obtain all of the nodes credential information like keys, identity, etc.

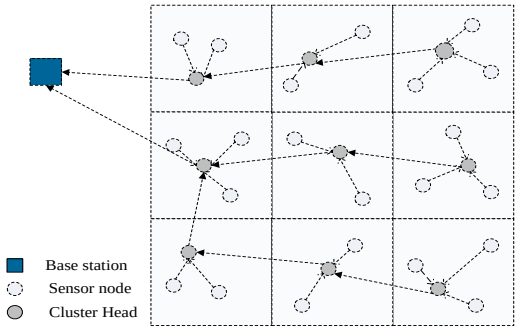

Figure 2. The structure of the network.

2. *Notations:* Table 1 gives notations used in this paper.

Table 1. Notation in this paper.

Notation	Description
M	The number of nodes in a WSN.
N	The number of sensor nodes in one cluster.
N_c	The number of clusters in network.
ID_j	Identity of node *j*.
R_j	Random number generated by node *j*.
f_no_k	The number of function *k*.
K_{ij}	Session key between node *i* and *j*.
MAC	Massage Atentication Code.

r_{ik}	The *i*th row of matrix Q_k.

IV. Definition of Proposed Pre-distribution Key Management scheme

In this section, we present a pairwise polynomial-based key management scheme using matrix for WSN, which is done in 3 phases.

Phase. 1 System Setup

In the following, we describe the Initial information distribution process of the proposed scheme.

Step I (polynomial pool generation). The base station first randomly generates a large pool of bivariate *t*-degree polynomials over the finite field F_q with unique *Id* for each of polynomials, where *t* is the degree of polynomial and *q* is a large prime number to accommodate a cryptographic key as shown in Eq.1.

$$F_k(x,y) = \sum_{i,j=0}^{t} a_{ij} x^i y^j \qquad \text{(Eq.1)}$$

These polynomials have a property that the number of compromised nodes is less than *t* [13].

Each cluster head receives a subset of polynomials from the pool before deployment. The distribution should be in such a way that any pair of clusters can discover at least one common function.

Step II (symmetric matrix formation). In this step, a symmetric matrix A with a size of $(N+1) \times (N+1)$ is generated, where N is the maximum number of nodes that are deployed in the cluster. We calculate this matrix as follow:

1. At first, a prim number is generated for each sensor node with unique identity *i*, using *Euler's function*. It gets a random generated number P_n as input and produces a prim number.
2. Then the cluster heads calculate a share of $f_k(x,y)$, that is $f_k(x_i,y_j) = \sum_{i,j=0}^{t} a_{ij} x_i^{\,i} y_j^{\,j} \in F_q$, *i, j= 1,..., N,* where H($f_k(x_i,y_j)$) is the element in the *i*th row and *j*th column of matrix Q_k.

$$Q_k = \begin{bmatrix} H(f_k(x_1,y_1)) & \dots & H(f_k(x_1,y_n)) \\ \vdots & \dots & \vdots \\ H(f_k(x_i,y_1)) & \dots & H(f_k(x_i,y_n)) \\ \vdots & \dots & \vdots \\ H(f_k(x_n,y_1)) & \dots & H(f_k(x_n,y_n)) \end{bmatrix}$$

As $f_k(x,y) = f_k(y,x)$, the generated matrices are symmetric. Therefore $a_{ij} = a_{ji}$, where a_{ij} is the element in the *i*th row and *j*th column of the matrix.

Step III (key pre-distribution). For every sensor node a subset of generated matrices is selected. Then one row from each matrix $Q_1, \dots, Q_k$ is pre-loaded to that node based on its identity.

$$\left(i, k, \left[H(f_k(x_i,y_1)) \quad \dots \quad H(f_k(x_i,y_n))\right]\right)$$

Here *i* is the index of matrix row and *k* is the function number. Matrix distribution should be in such a way that any pair of nodes can discover at least one common function.

The description can be better completed with the help of an example. Fig. 3 illustrates an example of a simple network. Suppose the calculated matrices for the first cluster are as follow:

$$Q_2 = \begin{bmatrix} 5 & 3 & 7 \\ 3 & 1 & 4 \\ 7 & 4 & 2 \end{bmatrix} \quad Q_5 = \begin{bmatrix} 1 & 10 & 6 \\ 10 & 3 & 5 \\ 6 & 5 & 1 \end{bmatrix} \quad Q_7 = \begin{bmatrix} 9 & 1 & 2 \\ 1 & 4 & 7 \\ 2 & 7 & 10 \end{bmatrix}$$

We choose a subset of matrices and load to nodes. For example, r_{22} and r_{25} the second rows of them to node 1, where r_{ij} is the *i*th rows of matrix *j*. Then, r_{35} *and* r_{37} are assigned to node 2. The first row in each matrix is allocated to the cluster head.

$$[(2, 2, [3 \quad 1 \quad 4]), (2, 5, [10 \quad 3 \quad 5])] \rightarrow node1$$

$$[(3, 5, [6 \quad 5 \quad 1]), (3, 7, [2 \quad 7 \quad 10])] \rightarrow node2$$

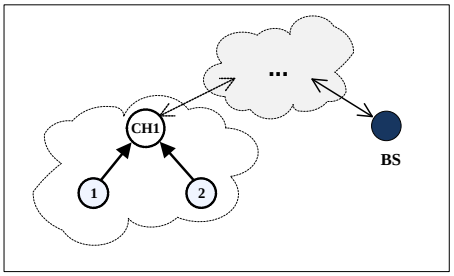

Figure 3. An example of key Pre-distribution.

Phase. 2 Session Key Establishment

For pairwise key establishment between two sensor nodes, the nodes perform key discovery phase to find out with which of their neighbors they share a key. This is done according to the following steps.

Step I: After sensor nodes are deployed, they first exchange HELLO message containing their identity.

$$i \rightarrow *: Hello,\ ID_i, r_i, R_i, f\_no_1, \dots, f\_no_k$$

Step II: The node *j* receives the row vector from its neighbors, and then it calculates K_{ij} that is computed as follow:

1. Node *j* first generates a random number R_j. Then it calculates the m-dimensional vector *R* by combining R_j with the receiving R_i from its neighbor.

$$R = H(R_i \parallel R_j) \rightarrow R_m = (R_1, \dots, R_m)$$

2. After obtaining *R*, it calculates *Vir* by operating dot product of two vector V_{jk} and R_m $(m = k)$. V_{jk} is vector of shared function's values between two nodes.

$$vir = V_{jk} . R_m = (V_{j1}, \dots, V_{jk}). (R_1, \dots, R_m)$$

To obtain the values of vector V_{jk}, node *j* should find the common functions with node *i*. Then it refers to the *i*th element of its r_{ik}, for all k $(k = 1, \dots, n)$ that *k* is the shared functions between them. Let $f\_no_1, \dots, f\_no_n$ be the shared functions, then V_{jk} is:

$$V_{jk} = (H\left(f_1(x_j, y_i)\right), \dots, H\left(f_n(x_j, y_i)\right))$$

As $Q_1, \dots, Q_k$ are symmetric matrices, we always have $H\left(f_k(x_j, y_i)\right) = H\left(f_k(x_i, y_j)\right)$. So, the *i*th element of V_{jk} $\{H\left(f_k(x_j, y_i)\right)\}$ is always equal to the *j*th element of $V_{ik}\{H\left(f_k(x_i, y_j)\right)\}$.

Now, node *j* responds with an ACK message as shown in the following:

$$j \rightarrow i : ID_j, r_j, R_j, f\_no_1, \dots, f\_no_k, H(Vir)$$

Step III: After receiving the message by node *i*, it can calculate R_m with having the value of R_j. Then node *i* obtains Vir' by performing a dot production of V_{ik} and R_m. If $Vir' = Vir$, then *j* is an authorized node and $K_{ij} = Vir' = Vir$.

To illustrate how to calculate V_{jk}, consider the previous example shown in Fig. 3. After sensor nodes are deployed, node *1* and node *2* begin to exchange HELLO messages as follow:

$$1 \rightarrow *: [Hello, 1, 2, 15, 2, 5]$$
$$2 \rightarrow *: [Hello, 2, 3, 11, 5, 7]$$

When node *2* receives the message from node *1*, it checks the number of function in common with it. In this example the only function is 5; therefore vector V_{jk} has one element in this example. As the index of node *1* in matrix Q_5 is *2*, node *2* should refer to the second element of its vector r_{35} $(r_{35} = a_{32} = 5)$.

$$Vir = 5 \times (15 \parallel 11) = 75$$

Node *2* sends a message in responding the HELLO message:

$$2 \rightarrow 1 : [2, 3, 11, 5, 7, H(75)]$$

After this message is received by node *1,* it can calculate $(15 \parallel 11)$ and obtains the value of V_{ik}, which is the third element of vector r_{25} $(r_{35} = a_{23} = 5)$. As mentioned previously, $Q_1, \dots, Q_k$ are symmetric matrices and $a_{23} = a_{32}$.

$$V_{15} = 5, \; Vir' = 5 \times (15 \parallel 11) = 75$$

So,

$$\mathrm{H}(Vir) = H(Vir') = H(75).$$

Step IV: Now, node *i* should authenticates itself to node *j*. So, it sends a message authentication code (*MAC*) message containing its identity, a random generated key *K* and R_i encrypted by K_{ij}. This message allows *j* to verify the validity of node *i*.

$$i \rightarrow j : MAC(ID_i, R_i, K)$$

V. Performance Analysis

In this section, we evaluate the performance of the proposed scheme and compare it with other methods.

1. Security Analysis

In fact, sensor nodes may be deployed into hostile environment, where an adversary may physically capture one or more sensor nodes. If a node is captured by an adversary, the credential key information kept in the node might be exposed. As a result, adversaries may compromise the connection of these nodes and some extra connections secured by these keys. The number of connections that are affected by node capturing is defined as the resilience against node capture.

In the propsed scheme, the base station selects a subset of polynomials from the large polynomial pool randomly and assigns them to the cluster heads. Then, a prim number is generated by Euler's function and a share of these polynomials is computed for any pair of nodes in each cluster. For each node a subset of hashed shares is stored before deployment.

In MPKMS, k_{ij} is calculated locally by sensor nodes, but is not directly exchanged between nodes. Hence, an adversary is not able to listen to the traffic load and able to extract the key k_{ij}. Once a key is compromised, only one communication is affected (two nodes). Let us consider that λ denotes the number of compromised node in each cluster and φ is the average number of clusters. Thus, the number of vulnerable nodes is $2 \times \varphi \times \lambda$. Fig 4 shows the number of affected nodes for different values of.

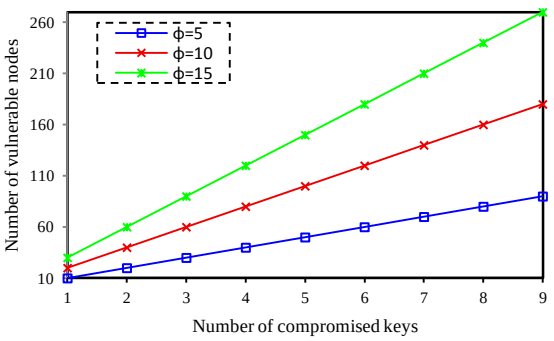

Figure 4. Number of vulnerable nodes for different φ.

2. Analysis of network connectivity:

The aim of connectivity in a network is to securely transfer data from one node to another. This secure transfer may depends on communication range and security mechanisms.

As mentioned previously, the nodes are deployed into a grid and to each cluster is assigned a distinct polynomial pools. The ddistribution is conducted in such a way that each pool has at least a function in common with other CH's pools. Also, for each node there exists at least one node that has a common function with it in the cluster. With this scheme, any pair of nodes can always compute a common key to communicate together. Therefore, probability of having two nodes with no function in common is zero. So, the proposed scheme achieves 100% connectivity.

In EG scheme, connectivity is computed as follow:

$$\Pr[6] = 1 - \frac{(p-k)!\,(p-k)!}{p!\,(p-2k)!} \qquad \text{(Eq.2)}$$

Where p is the size of key pool and each node stores k keys. In [14] the distribution is not uniform, thus Pr is obtained by the following equality.

$$\Pr[14] = 1 - \frac{(p-k)!\,(p-s)!}{p!\,(p-s-k)!} \qquad \text{(Eq.3)}$$

Where s is the number of assigned keys to the cluster heads. Fig 5 shows a comparison of network connectivity in above schemes.

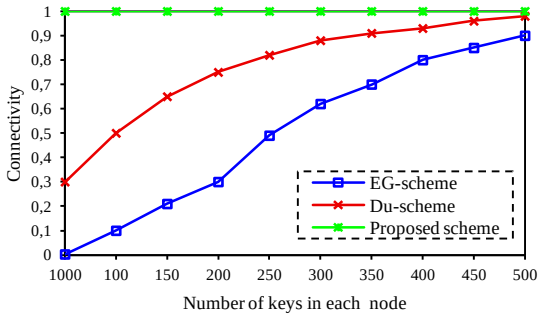

Figure 5. Connectivity of the network.

3. Memory analysis:

WSN has constraint storage resources. The proposed scheme reduces the storage cost effectively. The overall memory consumption in the network is equal to the sum of the memory usage in nodes and cluster heads.

As mentioned previously, each sensor node has $N+1$ rows of q matrixes. In each row, we have $N+1$ share of $f_k(x,y)$. Let N_c be the number of clusters in network and N be the maximum number of node that can be deployed in the cluster, the total memory usage is:

$$Mem_{node} = \eta \times (N^2 + N) \times N_c \times \Gamma_s \qquad \text{(Eq.4)}$$

Where Γ_s is the number of bits that are required to store a share of $f_k(x,y)$ and η is the average number of rows that is pre-loaded to each node.

For cluster heads, we need to store k matrixes, where γ is the number of function assigned to that cluster. The matrix has $(N+1) \times (N+1)$ elements and each of that needs Mem_s storage resource. So, the total memory needed in clusters is:

$$Mem_{CH} = (N+1)^2 \times N_c \times \Gamma_s \times \gamma \qquad \text{(Eq.5)}$$

Therefore, total memory required in network is:

$$Mem_{total} = Mem_{node} + Mem_{CH}$$

$$= \left(\left(\eta \times (N^2 + N)\right) + \gamma \times (N+1)^2\right) \times N_c \times \Gamma_s \qquad \text{(Eq. 6)}$$

In [10], each sensor node has to store z polynomial. if sensor nodes require δ ($\delta \gg \Gamma_s$) bits to store each polynomial, the overall storage overhead is $\delta \times z$. . A graphical comparison of these schemes is shown in Fig 6.

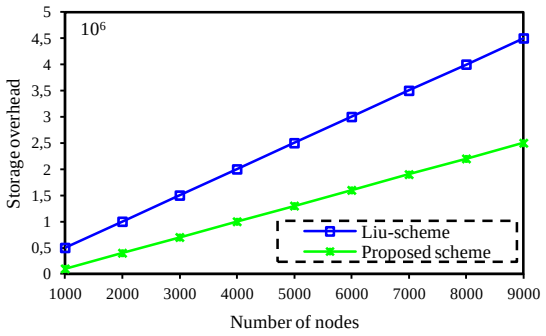

Figure 6. Effect of node number in usage overhead.

VI. Conclusion

In this paper, a new design of matrix based pairwise key pre-distribution using deployment knowledge is proposed. The deployment region is partition into equal-size grids and symmetric matrices is used for pre-distributing process of keys. Security is improved significantly by random selection of polynomials and prime number generation. This work also achieves a higher degree of connectivity and a lower storage overhead compared to existing schemes.

References

[1] S. Zhu, S. Setia, and S. Jajodia, "LEAP: efficient security mechanisms for large-scale distributed sensor networks," ACM Transactions on Sensor Networks, vol. 2, no. 4, pp. 500–528, 2006.

[2] X. Fan and G.Gong, "LPKM: A Lightweight Polynomial-Based Key Management Protocol for Distributed Wireless Sensor Networks", LNICST 111, pp. 180-195, 2013.

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

[4] S.J. Chio, K. T. Kim, H. Y. Youn, "An Energy-Efficient Key Predistribution Scheme for Secure Wireless Sensor Networks Using Eigenvector", International Journal of Distributed Sensor Networks, 2013.

[5] W. Bechkit, Y. Challal, A. Bouabdallah, V. Tarokh, " A Highly Scalable Key Pre-Distribution Scheme for Wireless Sensor Networks", IEEE Transaction on Wireless Communication, Vol. 12, No. 2, 2013.

[6] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in Proceedings of the 9th ACM Conference on Computer and Communications Security, pp. 41– 47, November 2002.

[7] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in Proceedings of the IEEE Symposium on Security and Privacy, pp. 197–213, May 2003.

[8] S. A.Camtepe and B. Yener, "Combinatorial design of key distribution mechanisms for wireless sensor network," in Proceedings of the 9th European Symposium on Research in Computer Security (ESORICS '04), pp. 293–308, 2004.

[9] J. Lee and D. R. Stinson, "On the construction of practical key predistribution schemes for distributed sensor networks using combinatorial designs," ACM Transactions on Information and System Security, vol. 11, no. 2, pp. 1–35, 2008.

[10] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in Proc. 2003 ACM CCS, pp. 52–61.

[11] R. Blom, "An optimal class of symmetric key generation systems," in Proc. 1985 Eurocrypt Workshop Advances Cryptology: Theory Appl. Cryptographic Techniques, pp. 335–338.

[12] Z. Yu and Y. Guan, "A robust group-based key management scheme for wireless sensor networks," in *Proc. 2005 IEEE WCNC*, pp. 1915–1920.

[13] J. Huang, G. H. Ou, "A Public Key Polynomial-based Key Pre-distribution Scheme for Large-Scale Wireless Sensor Networks." Ad Hoc & Sensor Wireless Networks 16(1-3), 45–64 ,2012.

[14] X.Du, Y.Xiao, M. Guizani, H.H Chen, " An effective key management scheme for heterogeneous sensor network", Ad Hoc Networks, vol. 5, pp. 24-34, 2007.