TELKOMNIKA, Vol. 13, No. 4, December 2015, pp. 1113~1120
ISSN: 1693-6930, accredited A by DIKTI, Decree No: 58/DIKTI/Kep/2013
DOI: 10.12928/TELKOMNIKA.v13i4.3122

A Polynomial-Based Pairwise Key Pre-distribution and Node Authentication Protocol for WSNs

Fatemeh Banaie[1], Seyed Amin Hosseini Seno*[2], Ismat Aldmour[3], Rahmat Budiarto[4]
[1,2]Network Research Laboratory, Department of Engineering, Ferdowsi University of Mashhad
[3,4]Smart Networked Computing Research Group, College of Computer Science and Information Technology, Al Baha University, Kingdom of Saudi Arabia
*Corresponding author, email: Banaie.F@stu-mail.um.ac.ir[1], Hosseini@um.ac.ir[2], iaaldmour@bu.edu.sa[3], rahmat@bu.edu.sa[4]

Abstract

Continuous advances in the areas of sensor networks have made wireless sensor networks (WSNs) attractive for a wide variety of applications, with vastly varying requirements and characteristics. As the data sensed by nodes usually contain sensitive information, adherence to data protection requirements is vital in WSNs.This paper introduces a new and robust key pre-distribution key management scheme using random polynomial functions and a matrix. The proposed mechanism significantly increases storage efficiency and enhances network resilience against node capture. The effectiveness of the mechanism is demonstrated by security analysis and comparisontothe existing schemes.

Keywords*: Data protection; key management; pre-distribution; polynomial functions*

1. Introduction

Nowadays, Wireless Sensor Networks (WSNs) are being widely used for large range of applications such as environmental monitoring, smart environments and scientific exploration. Sensor nodes are usually scattered in unattended and adversarial environments [1]. As the data sensed by the applications of WSN often contain sensitive information, adherence to secure data transfer is vital in these applications. In WSNs, data istransmitted wirelessly, which makes itvulnerable to attacks, such as eavesdropping, impersonation and modification of data [2]. Hence, securing the links established between the nodes in the network is a critical issue.

Key management is corner stone to many security services that can be used to meet confidentiality, integrity and authentication requirements in secure communications [3]. The main goal of key management is to provide secure communications between the nodes. Existing solutionsfor conventional networks, such as the solutions based on public keys, are not suitable for use in WSNs due to resource constraints, [4]. To address this issue, most of the existing solutions use random key pre-distribution for ensuring secure communications in WSNs.

Master Key Management is a basic pre-distribution approach, in which a single key is preloaded into every sensor node before deployment. The advantages of this scheme are high scalability and minimal storage requirements. However, it suffers from very low network resiliencebecause if an adversary captures a node; the security of the entire network is compromised [5]. Localized Encryption and Authentication Protocol (LEAP) offers a better key distribution for large scale distributed sensor networks,the basic idea is to erase the master key after establishing the pair wise key that increases the resilience,however, newly added node, which still has the master key, can be compromised [1].

In *Pairwise Key Distribution* schemes, pairwise keys are loaded to all possible pairs of sensors in the network. For a network with *n* sensor nodes, every node will have to store *n*-1 keys. Although this solution provides good connectivity and very good resilience against attacks, it is impractical to store all the keys in each sensor and it is not scalable tolarger WSNs [5, 6]. In *Random Pairwise Key Pre-distribution* [7], distinct pairwise keys are pre-distributed randomly in the sensor nodes before deployment. EG scheme [8] is abasic random key pre-distribution scheme, which addresses the redundancy in the previous schemes and yet provides a

Received August 25, 2015; Revised November 3, 2015; Accepted November 18, 2015

reasonable key resilience.In this scheme, a large pool of keys*S*is first generated out of which k keys are randomly selectedfor establishing a key ring, where $k \ll S$ and S is the total number of keys. This method needs less memory to store the key ring than storing the whole pool of keys. However, if nodes are progressively captured, the attacker can discover a large portion of the pool.

Another approach is the *Polynomial Key Pre-distribution*based schemes. Liu and Ning in [9] proposed to use symmetric bivariate polynomials in the key generalization phase that have the property *f(x, y) = f(y, x)*.This approach enhances resilience against node capture by computing distinct secret keys, but it increases the memory usage and the computational overhead.Authors in [10] proposed a novel polynomial based Q composite approache which combines robustness of Q composit scheme with polynomial based key generation scheme.

Blom in [11] proposed a λ-secure symmetric key generation system that uses a public matrix $G_{(\lambda+1)\times n}$ and a private symmetric matrix $D_{n\times(\lambda+1)}$. The keys are secure if no more than λ nodes are compromised. Then, the matrix of the pairwise keys of node n is $k = (DG)^T G$. In [12] a pre-distribution scheme using LU matrix is proposed, which uses LU matrix and Polynomial based key pre-distribution to achieve high resilience and connectivity.

Theworkin this paper is an extension of previous work in [13].It is based on random function pre-distribution and uses polynomial function and matrix to ensure a better security. The main aim of this work is to propose an efficient key pre-distribution scheme for efficiently more security and resilience of the network.

2. Proposed Pre-Distribution Key Management Scheme

The whole network consists of large number of static sensor nodes distributed randomly in the field, cluster heads and a single base station.Without loss of generality, it is assumedthat the sensing field is divided into Clogical regions, denoted by $C_i, i = 1, 2, \ldots, c$, (as shown in Figure 1). It follows that there are around $\varphi = \left\lceil \frac{NM}{c} \right\rceil$ nodes and one cluster head in each region.To reduce energy consumption in the network, sensor nodes communicate with base station and nodes in other regions through cluster heads, which have more energy resources than sensor nodes. Once an adversary captures a node, it can obtain all of the node's credential information like keys and identity.The following assumptions have been made for the proposed protocol:

- Base station is trusted and has a tamper resistant hardware. It has information about all the keys and nodes in the network.
- Cluster heads are assumed to have more resources than sensor nodes and the base station is a powerful node and more powerful than the cluster heads.
- Every sensor node has a unique global identity in the whole network and a local identity in its own cluster.
- Each matrix Q_l is generated by the key seeds of region C_l and its cluster head.

Table 2 summarizes some of the necessary notations in this paper.

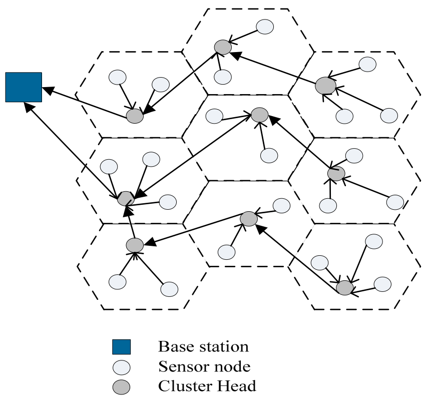

Figure 1. The structure of the network.

Table II. Notation in this paper

Notation	Description
a, b, c, …	Cluster heads identities
NM	Number of sensor nodes in the WSN
N_i	Number of sensor nodes in cluster *i*
C	Number of regions in the sensing field
ID_j	Identity of node *j*
R_j	Random nonce generated by node *j*
K_{ij}	Session key between node *i* and *j*
r_{ik}	*i*th row of matrix Q_k
$H(m)$	One-way hash function applied to string *m*

The proposed key pre-distribution and node authentication protocol for WSNsconsists of 3 phases as follows:

2.1. Initial Setup of System

Theinitial setup of the network and the key pre-distribution process of the proposed protocol is composed of 3 steps as follows.

Step I (polynomial pool generation): At first, the key setup server generates a large pool of bivariate *t*-degree polynomials over the finite field F_q with unique *Id* for each of the polynomials. $F_k(x,y)=\sum_{i,j=0}^{t} a_{ij}x^iy^j \ ,(k=1,\dots,n)$, where k is the total number of polynomials. These polynomials have a property that the number of compromised nodes is less than t [14] and $f_k(x,y)=f_k(y,x), forallx,y \in R^n$.Each cluster head randomly picks a subset of polynomials from the pool before deployment.The distribution must be in such a way that any pair of regions can discover at least one common function.

Step II (symmetric matrix formation): In this step, a symmetric matrix Q with a size of $(N_i+1)\times(N_i+1)$ is generated for each region, where N_i is the total number of sensor nodes that are deployed in the region *i*. This matrix is calculated as follow:

1. At first, a prim number is generated for each sensor node with unique identity *i*, using *Euler's function*.
2. The server calculates 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 x_i is the generated prime number for node *i* using *Euler's function* and y_j is the generated prime number of sensor node *j*.
3. The server generates matrix $Q_k\,,(k=1,\dots,l)$ for each of the cluster heads, where H($f_k(x_i,y_j)$) is the element in the *i*th row and *j*th column of matrixQ_k, l is the number of polynomials assigned to each cluster head and H is a one-way hash function.

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 each sensor node a subset of the generated matrices from matrices that are assigned to its cluster head, is selected. Then the *i*th row of selected matrix for node *i*, index of matrix row, and identity of matrices is pre-loaded to it.$\left(i+1,k,\left[H\left(f_k(x_i,y_1)\right)\dots H\left(f_k(x_i,y_n)\right)\right]\right)$. The description can be better completed with the help of an example. Figure 2 illustrates an example of a simple network. Suppose the calculated matrices for the first region are as follow:

$$=\begin{bmatrix}5 & 3 & 7 & 12\\ 3 & 1 & 4 & 9\\ 7 & 4 & 2 & 20\\ 12 & 9 & 20 & 5\end{bmatrix} \qquad Q_5=\begin{bmatrix}1 & 10 & 6 & 17\\ 10 & 3 & 5 & 2\\ 6 & 5 & 1 & 9\\ 17 & 2 & 9 & 7\end{bmatrix}$$

$$=\begin{bmatrix}9 & 1 & 2 & 8\\ 1 & 4 & 7 & 12\\ 2 & 7 & 10 & 3\\ 8 & 12 & 3 & 20\end{bmatrix} \qquad Q_8=\begin{bmatrix}4 & 14 & 3 & 5\\ 14 & 8 & 2 & 11\\ 3 & 2 & 7 & 20\\ 5 & 11 & 20 & 1\end{bmatrix}$$

$$[(2,2,[3\ \ 1\ \ 4\ \ 9]),(2,5,[10\ \ 3\ \ 5\ \ 2])]\rightarrow 1$$
$$[(3,5,[6\ \ 5\ \ 1\ \ 9]),(3,7,[2\ \ 7\ \ 10\ \ 3])]\rightarrow 2$$
$$[(4,7,[8\ \ 12\ \ 3\ \ 20]),(4,8,[5\ \ 11\ \ 20\ \ 1])]\rightarrow 3$$

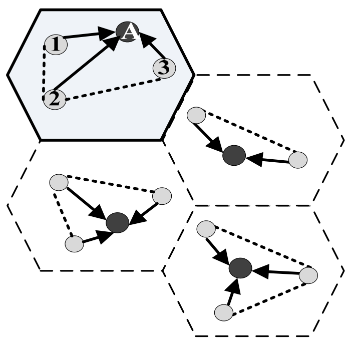

Figure 2. An example of key Pre-distribution.

Suppose that function 2 and 5 is assigned to node 1, 2 and 7 tonode 2 and 7 and 8 tonode 3. Hence, the second row of matrices Q_2 and Q_5 are pre-loaded to node 1, while the third row of Q_5, and Q_7 and forth row of Q_7 and Q_8 are pre-loaded to node 2 and node 3, respectively. The first row in each matrix is allocated to the cluster head.

2.2. Pairwise Key Establishment

Pairwise key establishment between two sensor nodes is done according to the following steps.

Step I: After sensor nodes are deployed in the field, they initiate the protocol by generating a random nonce R_i and exchanging a HELLO message of the following format: $i \rightarrow * :< Hello,\ ID_i, r_i, R_i, f\_no_1, \dots, f\_no_k >$. The message contains the identity of node *i*, the nonce R_i, index of the matrix row $r_i$and the identity of matrices that are preloaded to it.

Step II: On recipient of such a message, node *j* calculates the pairwise key 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 node*i*.

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

2. After obtaining *R*, it calculates *Vir* by the production of the two vectors V_{jk} and $R_m (m = k)$. V_{jk} is vector of shared matrices'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 matrices with node *i*. Then it refers to the r_i*th* element of stored row r_{ik}, for all $k\ (k = 1, \dots, n)$ that *k* is the shared matrices between them. Let $f\_no_1, \dots, f\_no_n$ be the identities of shared matrices, 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, and 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: Upon receiving the message by node*i*, it can calculate R_m having the value of R_j. Then node*i*obtains Vir' by 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 calculateV_{jk} , consider the previous example shown in Figure 2. After sensor nodes are deployed, the nodes begin to exchange HELLO messages as follow:

$1 \rightarrow * : [Hello, 1, 2, 15, 2, 5]$, $2 \rightarrow * : [Hello, 2, 3, 11, 5, 7]$, $3 \rightarrow *: [\,Hello, 3, 4, 8, 7, 8\,]$

When node *2* receives the message from node 1, it checks the number of functions in common with it. In this example the only function is *5*; therefore vector V_{jk} has one element. As the index of node *i* 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 response to the HELLO message: $2 \rightarrow 1\ :\ [2, 3, 11, 5, 7, H(75)]$. After this message is received by node i, it can calculate $(15 \parallel 11)$to obtain 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, $\mathrm{H}(Vir) = H(Vir') = H(75)$. $\mathrm{H}(Vir)$is used as session key K_{ij} for establishing a secure communication between node i and node j.

Step IV: Now, node i should authenticate itself to node *j*. So, it sends a MAC message containing its identity, a random generated key *K* and R_i encrypted by K_{ij}. This message allow *j* to verify the validity of node *i*. $i \rightarrow j\ :\ MAC_{K_{ij}}(\,ID_i,\ R_i, K)$.

2.3. Path Key Establishment

In establishment of the pairwise key between sensor nodes that are not in the same cluster or do not have any shared matrix, the following procedures are carried out:

Case I: Both nodes are in the same cluster

Suppose that node i in cluster a wants to send a data to node j in the same cluster. Node $i$broadcasts a HELLO message as follow:

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

After receiving the message by the nodes that has a common matrix with j, it calculates vir according to the mentioned steps in section 2 and responds to node $i$with a message as shown in the following: $n \rightarrow i: < ID_n, R_n, H(Vir) >$.After receiving this message by node i, it calculates Vir' and authenticates itself by sending a MAC message to n:$i \rightarrow n$: $MAC_{K_{in}}(ID_i, R_i, K)$. Now node $i$encrypts data using K_{ij} and sends it to n After decrypting the message by n it first authenticates node j and sends the message to it through the following steps:
$i \rightarrow n$: $E_{K_{in}}(ID_j, Data)$, $n \rightarrow j$:$< Hello, ID_n, R_n >$, $j \rightarrow n$:$< ID_j, R_j, r_i, R_i, f\_no_1, \dots, f\_no_k, H(Vir) >$, $n \rightarrow j$: $MAC_{K_{jn}}(ID_n, R_n, K)$, $j \rightarrow n$: $E_{K_{jn}}(ID_n, R_n, K+1)$, $n \rightarrow j$: $E_{K_{jn}}(Data)$

Thus, a path is established between i and j through node n.

Case II: The nodes are not in the same cluster

In this case, after establishing a pairwise key between node *i* and CH_a according to the above mentioned steps, it encrypts the data with K_{ia} and sends it to CH_a.

$$i \rightarrow CH_a \ : \ E_{K_{ia}}(ID_j, Data)$$

CH_asends ID_j to the base station and requests the identity of its cluster head to establish a secure link with it. The base station finds the node js cluster (for instance CH_b) and sends the identity of its cluster head in response to this message. By receiving its identity, CH_a generates and broadcasts a HELLO message of the following form: $CH_a \rightarrow * :< Hello, ID_a, ID_b, R_a, f\_no_1, \dots, f\_no_k >$. Where it contains the identity of two cluster heads, the nonce R_a , and the identity of the functions that are preloaded to it. When the message is received by CH_b , it acts differently in one of two cases:

Case a: It has a common function with CH_a. At first, CH_b calculates a share of its common functions $(f_k(a,b))$ with CH_a. Then it generates a random number $R_b$and calculates the m-dimensional vector *R* by combining R_b with the receiving $R_a$from CH_a.$R = H(R_a \parallel R_b) \rightarrow R_m = (R_1, \dots, R_m)$.*Vir*is calculated as follow, whereV_{bk} is vector of shared function's values between two cluster heads*:* $vir = V_{bk}.R_m = (V_{b1}, \dots, V_{bk}).(R_1, \dots, R_m)$.$CH_b$responds with an ACK message as shown in the following:

$$CH_b. \rightarrow CH_a.:< ID_b, R_b, f\_no_1, \dots, f\_no_k, H(Vir) >$$

With the receiption of this message, CH_acan calculate Vir'. If $Vir' = Vir$, they can establish a secure channel for transmitting the data.

Case b: It has not a common function with CH_a.In this case the two cluster heads can establish a secure path through their neighboring cluster heads that share a common function.Now CH_a can forward the encrypted data to CH_busing K_{ab}. After receiving the message by CH_b, it can authenticate node j and send the encrypted data to it. The process is done by exchanging the following messages:$CH_a \rightarrow CH_b : E_{K_{ab}}(ID_a, ID_i, Data)$. After authenticating j by CH_b: $CH_b \rightarrow j$: $E_{K_{jb}}(ID_i, Data)$.

3. Performance Analysis

3.1. Security Analysis

The environment in which sensor nodes are deployed is usually hostile and harsh. So, 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 proposed protocol, the key generation server randomly selects a subset of polynomials from the large polynomial pool and assigns them to cluster heads. Then, a prim number is generated by Euler's function for each node based on its identity and a share of these polynomials computed using these numbers for any pair of nodes in each cluster. For each node a subset of hashed shares is stored before deployment. On the other hand, k_{ij} is not directly exchanged between nodes calculated locally by sensor nodes. Hence, an adversary is not able to listen to the traffic load and extracts the key k_{ij}. Likewise, this protocol meets the *forward secrecy*. This means that if an adversary managed to recover a contiguous subset of session keys, no previous pairwise key can be retrieved. This property is done by hashing the keys in each step. Therefore, the proposed protocol achieves a high level of security in the network.

3.2. Analysis of Network Connectivity

Having done calculation of polynomials matrices, every node picks a subset of these matrices and stores the corresponding row of the matrices. Distribution is done in such a way that each node has at least a common key to share with other nodes in the cluster. On the other hand, two nodes that do not have a common shared key can establish a path through the cluster head. Thus, the probability of having two nodes with no path is zero. So, the proposed scheme achieves 100% connectivity.In E- G scheme, connectivity computed as follow [8, 15]: $\mathrm{P_{real}} = 1 - \frac{(p-k)!^2}{p!(p-2k)!}$, where p is the size of key pool and each node stores k keys. If the distribution is not uniform, $\mathrm{P_{real}}$ is obtained by the following equality: $\mathrm{P_{real}} = 1 - \frac{(p-k)!(p-n)!}{p!(p-n-k)!}$, where n is the number of assigned keys to the cluster heads. Figure 3 shows a comparison of network connectivity in above schemes.

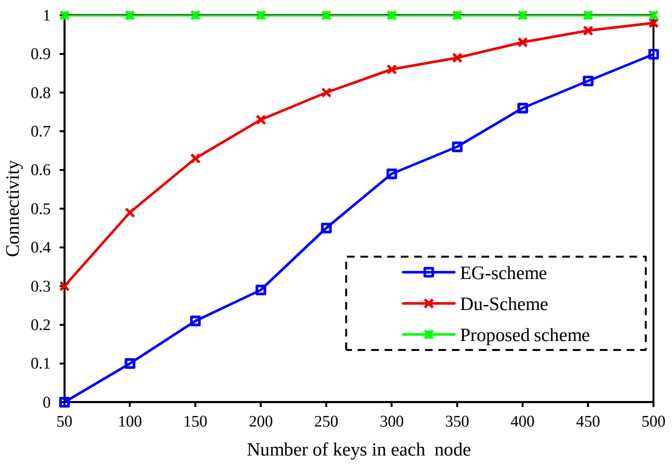

Figure 3. Connectivity of the network

3.3. Memory Analysis

WSN has constrained storage resources. The proposed protocol reduces the storage cost effectively. The storage cost of the proposed protocol is consideredas the sum of the memory usage in nodes and cluster heads. Each sensor node has $N + 1$ rows of q matrixes. In each row, there are$N + 1$ shares of $f_k(x, y)$. Let N_c be the number of clusters in the network and N be the total number of sensor nodes that can be deployed in the cluster, the total memory usage is:$Mem_{node} = \eta \times (N^2 + N) \times N_c \times \Gamma_s$,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 are pre-loaded to each node. Cluster heads need to store k matrixes, where γ is the number of the functions assigned to that cluster. The matrix has $(N + 1) \times (N + 1)$elements and each of that needs Mem_s storage resource. So, total memory needed in clusters is: $Mem_{CH} = (N + 1)^2 \times N_c \times \Gamma_s \times \gamma$.Thus, total memory required in the network is: $Mem_{total} = Mem_{node} + Mem_{CH} = ((\eta \times (N^2 + N)) + \gamma \times (N + 1)^2) \times N_c \times \Gamma_s$.

In [8], each sensor node has to store z polynomials. Ifthe 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 for their storage requirements is shown in Figure 4.

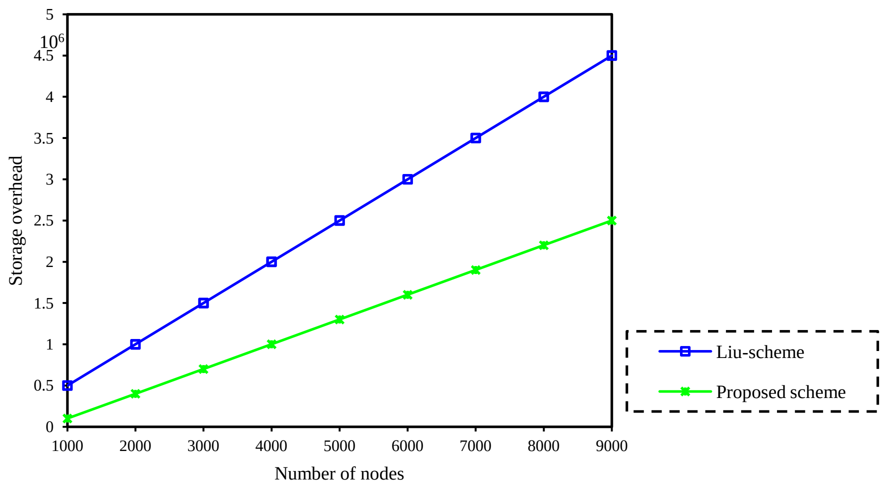

Figure 4. Effect of node number on storage overhead.

3.4. Resilience against Node Capture

As sensor nodes may be deployed into hostile environment, key information may be compromised if an adversary captures the node. In this protocol, when node i is captured, the adversary cannot retrieve information about links with non-captured nodes due to the use of several polynomials $f_k(x, y)$. In other words, retrieving other pairwise keys without having enough information about key pools and matrices is not possible. As each node holds$m$ matrices, the adversary retrieves information about m matrices by capturing one node. If the number of matrices preloaded in one node is large, the more shares of matrices are captured. The probability of resiliency against node capture can be calculated according to$P_{cN} = \frac{(NM_c + NM_s)}{NM}$. Where NM_c is the number of captured sensor nodes, NM_s is the number of sensor nodes which have common matrices with them, and NM is the total number of sensor nodes in the network. The probability of resiliency against cluster head capture is calculated in the same wayas$P_{cc} = \frac{(CH_c + CH_s)}{CH}$. Here, CH_c is the number of captured cluster heads, CH_s is the number of CHs that have common function with them, and CH is the total number of cluster heads in the network.

4. Conclusion

Key management in wireless sensor networks is a challenging issue due to the limitations on sensor node resources. This paperhave proposed anew design of matrix based pairwise key pre-distribution using deployment knowledge to address the issue.

The deployment field is partioned into equal-size hexagons and use symmetric matrices for pre-distributing of keys. Any pair of nodes can find a common secret key or path key between themselves using the keys assigned by a pool of polynomials and matrices. 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 comparedto 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*. 2006; 2(4): 500–528.

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

[3] W Bechkit, Y Challal, A Bouabdallah, V Tarokh. "A Highly Scalable Key Pre-Distribution Scheme for Wireless Sensor Networks". *IEEE Transaction on Wireless Communication*. 2013; 12(2): 948 – 959.

[4] W Du, J Deng, YS Han, PK Varshney, J Katz and A Khalili. "A pairwise Key Pre-distribution Scheme for Wireless Sensor Networks". *ACM Transaction on Information and System Security*. 2005; 8(2): 228-258.

[5] S Bala, G Sharma, and K Verma. "Classification of Symmetric Key Management Schemes for Wireless Sensor Networks". *International Journal of Security and Its Applications*. 2013; 7(2): 117-138.

[6] A Perrig, R Szewczyk, JD Tygar, V Wen, DE Culler. "SPINS: Security Protocols for Sensor Netwo*rks*". *Journal of Wireless Networks*. 2002; 8(5): 521 - 534.

[7] H Chan, A Perrig, and D Song. "*Random key pre-distribution schemes for sensor networks*". In: Proceedings of the IEEE Symposium on Security and Privacy, Oakland, California, USA. 2003: 197–213

[8] L Eschenauer and VD Gligor. "*A key-management scheme for distributed sensor networks*". In: Proceedings of the 9th ACM Conference on Computer and Communications Security, Washington DC, USA. 2002: 41– 47.

[9] D Liu and P Ning. "*Establishing pairwise keys in distributed sensor networks*". In: Proceedings of the 10th ACM Computer and Communications Security, Washington DC, USA. 2003: 52–61.

[10] EA Mary Anita, R Gee Tha, E Kannan. "A Novel Hybrid Key Management Scheme for Establishing Secure Communication in Wireless Sensor Networks". *Wireless Personal Communications*. 2015; 82(3): 1419-1433.

[11] R Blom. "*An optimal class of symmetric key generation systems*". In Proceedings of the 1985 Eurocrypt Workshop Advances Cryptology: Theory Appl. Cryptographic Techniques, Linz, Austria. 1985: 335–338.

[12] Z Yu and Y Guan. "*A robust group-based key management scheme for wireless sensor networks*". In Proceedings of the 2005 IEEE WCNC, New Orleans, USA. 2005: 1915–1920.

[13] F Banaie, SAH Seno, RH Aljoufi, and R Budiarto. "*MPKMS: A Matrix-based Pairwise Key Management Scheme for Wireless Sensor Networks*". In: Proceedings of International Conference on Electrical Engineering, Computer Science and Informatics (EECSI 2014), Yogyakarta, Indonesia. 2014: 462-467.

[14] J Huang, GH Ou. "A Public Key Polynomial-based Key Pre-distribution Scheme for Large-Scale Wireless Sensor Networks". *Ad Hoc & Sensor Wireless Networks Journal*. 2012; 16(1-3): 45–64.

[15] X Du, Y Xiao, M Guizani, HH Chen. "An effective key management scheme for heterogeneous sensor network". *Ad Hoc Networks*. 2007; 5: 24-34.