HGKM: A Group-based Key Management Scheme for Sensor Networks using Deployment Knowledge

Ngo Trong Canh, Young-Koo Lee*, Sungyoung Lee
Dept of Computer Engineering, Kyung Hee University, Korea.
ntcanh@oslab.khu.ac.kr, yklee@khu.ac.kr, sylee@oslab.khu.ac.kr

Abstract

Key establishment plays a central role in authentication and encryption in wireless sensor networks, especially when they are mainly deployed in hostile environments. Because of the strict constraints in power, processing and storage, designing an efficient key establishment protocol is not a trivial task. Compare with traditional public key cryptography, symmetric key cryptographic with key predistribution mechanism is more suitable for large-scale wireless sensor networks. Most of previous solutions have some issues on performance and security capabilities. In this paper, we propose a novel key predistribution model using pre-deployment knowledge to take advantage in terms of network connectivity, resilience against node compromised, memory requirement and energy for transmission.

Keywords: Key predistribution, network security, sensor networks.

1. Introduction

Sensor networks have a numerous applications such as home security monitoring, military reconnaissance, target tracking… [1]. Typical sensor networks normally consist of large number of small devices. Such devices are sensor nodes, having limited battery power, data processing and often communicate with each others by short-range radio signal. In many applications, sensor nodes are often spread out randomly over specific regions to sense and collect information.

One of the most basic security requirements for sensor networks is to guarantee the confidentiality and integrity in sending messages between sensor nodes. Environments in which sensor networks are exploited are regularly hostile areas. In these spaces, attackers could eavesdrop on messages or disable the networks by launching physical attacks to sensor nodes, or even using logical attacks to different communication protocols [2], [3]. Thus, to get rid of above problems, sensor networks need encryption and authentication services. Due to resource constraints, implementation an efficient key establishment mechanism is not a trivial task. Beside advantage of elliptic curve cryptography recently, symmetric key algorithms are the feasible solutions to solve this problem.

The random key predistribution was first proposed by Eschenauer and Gligor [4]. Chan, Perrig and Song [5] improved with q-composite and random pairwise key predistribution. Du, Deng, Han and Varshney applied deployment knowledge to basic random pairwise key in their scheme [8]. Polynomial-based proposals relied on Blundo's approach [10] are in [11], [12], [13]. The key matrix schemes, developed from Blom's solution [6], are multiple-space key predistribution scheme [7] and DHDV-D [9] of Du, Deng, Han and Varshney. Although some models exploited prior deployment knowledge, they still didn't take advantage of this information.

In this paper, we introduce Hexagonal Group-based Key Management model (HGKM) for wireless sensor networks which uses deployment knowledge to improve the security and performance questions. With the advantages of deployment knowledge, we distribute polynomial information to a limited number of sensor nodes over specific area in a hexagonal grid. So it will decrease the probability to reveal a polynomial when the adversary compromised some nodes. Our scheme is shown to have better security than solutions in [4],[5],[12],[13],[14].

The rest of the paper is organized as follows: In Section 2, we briefly describe related work. Next, Section 3 gives an overview of Blundo's polynomial key predistribution technique. Section 4 presents our proposal in detail. Afterward, we show the analysis and estimation of our scheme compared with others in Section 5. Finally, in Section 6, we conclude the paper and point out further research directions.

2. Related Work

The first scheme is proposed by Eschenauer and Gligor [4]. In this system, a large key pool is generated

* Corresponding author.

off-line and each sensor picks a random subset of keys from the key pool. Any two nodes in the communication range can talk to each other only if they share a common key. Depending on the size of the key pool and the number of sensor nodes in the network, this design may achieve different connectivity and resilience. Chan, Perrig and Song [5] later proposed an approach using the similar idea, but increased the intersection sharing keys between key-rings from one key to some *q>1* keys. It is shown that, by increasing the value of q, network resilience against node capture is improved. Du, Deng, Han and Varshney suggested a key predistribution model by applying deployment knowledge [8]. In their design, entire network was divided into groups. Each group implements the basic random key predistribution as in [4]. The key pool of a group shared α keys with horizontal groups' key-pools and β keys with diagonal groups' key-pools.

The key-matrix solutions are based on the idea of Blom [4]. He recommended a key predistribution scheme making certain that any pair of members in a group is able to calculate the common sharing key. Denote N is the number of sensor nodes in the network, let G be a generator matrix of size (t+1)×N over finite field and let D be a secret random matrix (t+1)×(t+1) with elements in . From the matrix G and D, construct a N×N symmetric K whose entries will be the pairwise keys between nodes. The matrix K is equal to $K = (D \cdot G)^T \cdot G$. Each node *i* stores a corresponding row *i* of private matrix $A = (D \cdot G)^T$. If node *i* want to communicate with node j, then it computes the inner product of row vector it stores with the j-th column of G to obtain the common key $K_{i,j}$. Multiple-space key predistribution of Du, Deng, Han and Varshney [7] combined the Blom's method with the basic random key predistribution of Eschenauer and Gligor [4] for applying to sensor networks. In this approach, they denoted the set of keys that each tuple <*D,G*> can generate a key space. Each node in the network stored randomly τ spaces from ω pre-generated spaces. Based on probabilistic, any two nodes could share a common space, which may compute a common secret key. Later, Du, Deng, Han and Varshney also applied pre-deployment knowledge to propose DDHV-D scheme in [9]. It is the combination of multiple-space key predistribution [7] with the random predistribution scheme applied deployment knowledge [8]. All the key-matrix solutions have threshold t-secure property. It means that no more than t nodes are compromised by attackers then the communications between non-compromised nodes are still secured.

The basic idea of polynomial key generation was proposed by Blundo et al [10]. It uses symmetric polynomial evaluations to obtain a pairwise key. The detail of this method will be described in the next section. This proposal is t-collusion resistant against node captured with property: compromise of less than t+1 node doesn't reveal any information about keys of other nodes. Derived from above method and basic random key predistribution [4], Liu and Ning introduced random subset assignment key predistribution model [11]. Instead of generating large key-pools and creating key-rings, this scheme creates a large polynomials pool and assigned each node a subset of polynomials from the pool. Then two nodes can only communicate to each other when they shared at least one common polynomial. It is shown that this solution increased the resilience comparing with Eschenauer and Gligor's model [4]. Further solution using predeployment knowledge is Closet Polynomials Predistribution Scheme (CPPS) of Liu and Ning [12], [13], 8-Square Grid-based Polynomial Predistribution [14]. Most of these solutions still have some limited on using pre-deployment knowledge to improve performance and security.

3. Blundo's key predistribution scheme

Blundo's scheme in [10] uses n variables polynomials with t-degree to establish key distribution for t-secure n-conference. Applied to pairwise key between two entities, key predistribution server randomly generates a bivariate t-degree polynomial $f(x,y) = \sum_{i,j=0}^{t} a_{ij} x^i y^j$ over a finite field F_q, where q is a large enough prime number that could accommodate a cryptographic key. The function $f(x,y)$ is symmetric meaning that $f(x,y) = f(y,x)$. Each node having unique integer ID i loads the information of $f(i,y)$ from the polynomial $f(x,y)$. Then any two nodes i and j can compute the key $k_{i,j} = f(i,j)$ at node i and $k_{j,i} = f(j,i)$ at node j. Because of symmetric property, we have $k_{i,j} = k_{j,i}$ so that two nodes have a common pairwise key.

Each node must store *t+1* coefficients, each coefficient costs $\log_2 q$ bits. So the memory storage requirement for each node in this model is $(t+1)\log_2 q$ bits. The analysis in [6] shows that, this scheme is unconditionally secure and t-collusion resistant. It means that as long as no more than t nodes are compromised, the attacker knows nothing about the

pairwise key between any two non-compromised nodes.

This basic proposal is not able to apply directly to sensor networks due to its memory overhead for storing keys. The size of memory depends exponentially on the size of the network, so it is not useful for such resource-constraint devices like sensor nodes using only this model. We will focus on this problem by using predeployment knowledge and showing that it will take more advantages than other polynomial-based schemes applied expected location knowledge.

4. Proposed key predistribution model

Before presenting our proposed scheme, we define a key-space as a set of all keys that a t-degree bivariate polynomial $f(x, y)$ in Blundo's model could generated. The number of keys in a key-space is denoted as key-space size. We assume that a node will pick a key-space if it carries the information generated from $f(x, y)$. Any two nodes picking a common key-space always compute their pairwise key.

Our scheme has totally two phases: key predistribution, direct key establishment. The key predistribution phase is carried out to preload the credential information to each sensor node before deployment. After set up, two sensor nodes can establish a direct key between them if they share at least a common key-space. At the beginning, we will handle with the deployment model of sensor networks.

4.1. Hexagonal grid-based deployment model

In our proposal, the target area is divided in hexagonal grid. This geometry provides the best approximation to circle and covers the biggest area than other two in three geometries that can be repeated over a continuous field: triangle, rectangle and hexagon. Also, a hexagon has the least (six) neighbor cells comparing to eight for rectangle or twelve for triangle. Sensor nodes are partitioned and distributed into groups on cells. This model is practical in realistic, when sensor nodes in each group are delivered together, such as using airplane to drop out groups in sequence, so expected adjacent groups have better chance of being close to each other on the ground.

Normally, the arrangement of sensor nodes relies on some probability distribution function (pdf function). Let's assume that target deployment area is two-dimension with size $X \times Y$. The pdf for the location of node i, with $i = 1,..,N$, over the two-dimensional region is $f_i(x,y)$, where $x \in [0, X]$ and $y \in [0, Y]$. N sensor nodes are divided into G equal size groups. The pdf may be uniform distribution, as in [11],[12],[13],[14] or more realistic like Gaussian distribution in [7],[8],[9]. In this proposal, we use two-dimensional Gaussian distribution for a group G_i as following:

$$f_i(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x-x_i)^2+(y-y_i)^2}{2\sigma^2}} \quad (1)$$

where (x,y) is the coordinate of a node in the group G_i, the deployment point of the group is (x_i,y_i) and σ is the standard deviation of distribution. The value of σ depends mainly on the height of plane when dropping out sensor groups.

At a cell, the distribution function is non-uniform, we could choose a proper distance between deployment points to get the overall distribution is nearly uniform. The hexagonal group-based deployment model could be seen in Figure 1.

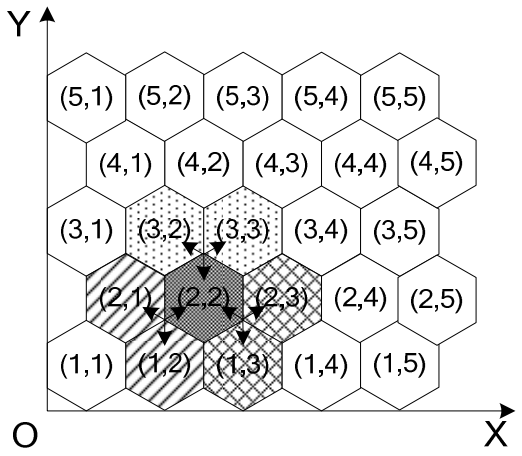


Figure 1. Hexagonal group-based deployment model.

Before describing the organize of our proposal, we define a cluster is a set of neighbor groups. There are three types of cluster which a group belongs to: the *1-cluster, 2-cluster* and *3-cluster*. From point of view of any group *(i, j)*, the *1-cluster* consists of this group and two groups *(i+1, j)* and *(i+1, j+1)*. The *2-cluster* consists of group *(i, j)* and two groups *(i, j-1)* and *(i-1, j)*. The *3-cluster* consists of group *(i, j)* and two groups *(i-1, j+1)* and *(i, j+1)*.

For example, from group *(2, 2)* in Figure 1, group *(3, 2)* and *(3, 3)* belong to *1-cluster(2, 2)*. Group *(2, 1)* and group *(1, 2)* belong to *2-cluster(2, 2)*. And *3-cluster(2, 2)* consists of group *(2, 3)* and group *(1, 3)*.

Therefore, each group consists of three cells, and every cell belongs to three groups.

4.2. Key predistribution phase

In this phase, we need to assign key information to each node. After deployment, neighboring nodes can compute a pairwise key between themselves.

We generate a *t*-degree bivariate polynomial and distribute to all sensor nodes in each cluster. Because each cell belongs to three clusters, every node has to store knowledge of 3 *t*-degree bivariate polynomials. In other words, each node needs to pick 3 key-spaces. The algorithm to distribute polynomials is that at each group $G_{i,j}$, the center server checks groups in 1-cluster of $G_{i,j}$ whether sharing key-spaces with $G_{i,j}$. If not, the server generates a new polynomial and assigns to all nodes in the 1-cluster. The same steps are also taken place for 2-cluster and 3-cluster. The detail algorithm for polynomial pre-distribution is following:

Table 1. The Polynomial Predistribution Algorithm

```
For each group G_{i,j} {
    For each group G_{u,v} in 1-cluster(G_{i,j}) {
        If not is_polynomial_sharing(G_{u,v}, G_{i,j}) {
            Geneate a f(x,y).
            Assign f(x,y) to 1-cluster(G_{i,j})
        }
    }
    For each group G_{u,v} in 2-cluster(G_{i,j}) {
        If not is_polynomial_sharing(G_{u,v}, G_{i,j}) {
            Geneate a f(x,y).
            Assign f(x,y) to 2-cluster(G_{i,j})
        }
    }
    For each group G_{u,v} in 3-cluster(G_{i,j}) {
        If not is_polynomial_sharing(G_{u,v}, G_{i,j}) {
            Geneate a f(x,y).
            Assign f(x,y) to 3-cluster(G_{i,j})
        }
    }
}
```

4.3. Direct key establishment phase

After set up, each node must discover whether it shares certain key-space with its neighbors. To do this, each node broadcasts a message containing the following information: (i) the node's ID, (ii) the IDs of key-spaces it carries.

Suppose that nodes A and B are neighbors, with ID are N_a and N_b respectively. They receive the above broadcast messages from each other. If they find out a common sharing key-space f_c, they could compute the pairwise key as shown in Blundo's scheme: Node N_a computes they key $K_{A,B} = f_c(N_a, N_b)$. Node N_b computes the key $K_{B,A} = f_c(N_b, N_a)$. Because of the symmetric property of bivariate polynomial f_c, we have $K_{A,B} = K_{B,A}$. This key is used as the secret pairwise key between node A and node B.

After above phase, the sensor network forms a key graph *G(V,E)* as follow:

We define a key graph *G(V, E)* with V is the set of vertices, which are equivalent to all sensor nodes, E is set of edges, like secure links in the network. Existing an edge between any two vertices u & v if and only if: (i) u, v are in transmission range of each other; (ii) both have a common sharing key-space ID, meaning that they could establish a pairwise key.

4.4. Sensor addition and revocation

To add a new sensor, the key setup server only needs to predistribute the related polynomial shares to the new node, similar to predistribution phase. Since the size of key-space is limited, the more sensors are added, the lower the security in that cell becomes.

The revocation method is also straightforward. Each sensor node only needs to store a black list IDs of compromised sensors that share at least one bivariate polynomial with itself. If there are more than t compromised nodes sharing the same polynomial, the non-compromised nodes that have this polynomial will remove this polynomial and all related compromised nodes.

5. Analysis

We discuss about the following measurements:

- *Network connectivity*: including local connectivity and global connectivity. Local connectivity is the probability a node could connect with neighbor nodes in its transmission range. Global connectivity is the ratio of the number of sensor nodes forming the largest isolated connected component in the final key graph *G* to the size of the whole network.
- *Communication overhead*: is the energy a node needs to make communication in proposed model.
- *Memory overhead*: that is the memory requirement for storing key materials at nodes in our model.

- *Resilience against node captured*: we estimate the impacts of nodes compromised attacks to the remaining network. In this analysis, we evaluate the probability the adversaries can discover a bivariate polynomial, meaning that they can reveal all secure connections encrypted by derived keys from this polynomial.

5.1. System configuration

We use the setup in Table 2 for our simulation and numerical analysis.

Table 2. Simulation setup

Symbol	Value	Description
N	10,000	Number of nodes in the network.
S	1000×1000 (m^2)	Network deployment area.
r	40m	The communication range.
M	200 keys	The memory size of nodes for storing key materials.
σ	50m	The standard deviation in Gaussian distribution.

In this scenario, we assume nodes deployment follows a two dimensional Gaussian distribution with pdf function in formula (1).

5.2. Network connectivity

Denoted $d = a \times \sigma$ is the distance between two deployment points of two neighboring cells. This value has impacts on the local connectivity and global connectivity in the network. If the deployment distribution follows Gaussian distribution, there are 99.87% sensor nodes of a group reside within range 3σ from its deployment point. Therefore, if the value d is much large than 6σ, almost every nodes in a group reside in its cell area, and the neighboring nodes are from its own group. In this case, the local connectivity is very high, but the network is totally partitioned into isolated components, meaning global connectivity is very low. In case of the value d is smaller, the local connectivity may be low, but the global connectivity is high. So, choosing suitable value of d affects the network connectivity.

In the simulation, we change different values of d according to a. Along with this, the ratios of local connectivity and global connectivity also have various values as shown on Figure 2.

Table 3. Simulation result

a	Local connectivity	Global connectivity
0.4	0.0787	0.6546
0.6	0.1577	0.9290
0.8	0.2524	0.9704
1.0	0.3643	0.9921
1.5	0.6036	0.9990
2.0	0.7720	0.9994
2.5	0.8617	0.9998
3.0	0.9226	0.9999
3.5	0.9555	0.9999
4.0	0.9657	1

When the distance between two deployment points of two neighboring cells is too low (a = 0.4; 0.6; 0.8 or 1.0), at any node A, there are many nodes of non-neighbor cells distributed around it. These nodes do not share any key-space with node A. So the local connectivity and global connectivity are reduced.

From Table 3, it is easy to see that our model gains high local and global connectivity when choosing suitable value of deployment point distances. With value a=1.5, the global connectivity is 0.9990, meaning that only 0.01% number of nodes in the network are waste.

5.3. Communication and memory overhead

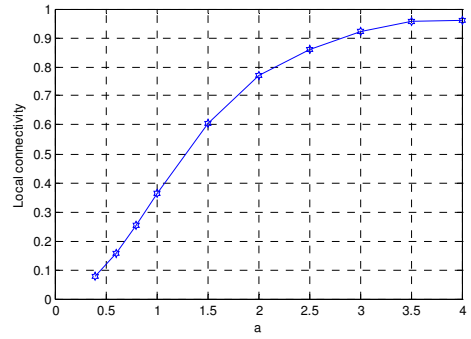
The long lived time is the critical goal in designing protocols for wireless sensor networks. In our proposal, we minimized the broadcast data requirement in discovery common key-space between neighboring nodes. Our 1-hop broadcast message length is *sizeof(node ID) + 3 ×sizeof(key-space ID)*. Comparing with other models in [4],[5], the broadcast messages in key discovery phase contain hundreds of key, to achieve high connectivity. With CPPS in [13], the length of broadcast messages is *sizeof(node ID) + 5 × sizeof (polynomial ID)*, which is higher than ours.

The memory size for storing key materials derived from polynomials is $M = 3\times(t + 1) \log_2 q$ *(bits)*. This value, along with number of nodes sharing a polynomial, affects to the resilience against node compromised attacks. We will discuss more detail in the following section.

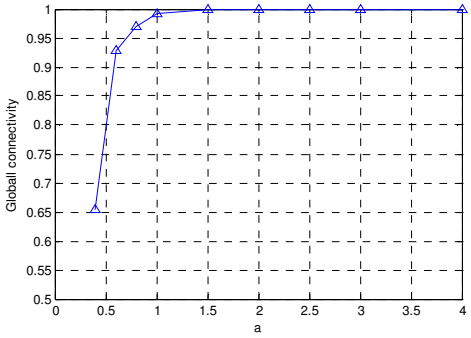
5.4. Security against node compromised

Because the working environments of sensor networks usually are hostile, it's easy for sensor nodes will be captured and revealed information. Adversaries

could get all the pairwise keys in compromised nodes, therefore they could break a number of secure links, including all links from these nodes and maybe other links between non-compromised nodes. We evaluate our model in term of the resilience against node capture. That is when x nodes are compromised, how much is probability to reveal a polynomial, meaning to disclose direct key between non-compromised sensor nodes.



a) Local connectivity



b) Global connectivity

Figure 2. Network connectivity vs deployment point distances.

The analysis in [10] shows that the polynomial-based scheme has t-secure property: unless more than t polynomial shares of a bivariate polynomial are disclosed, adversaries would not know about the non-compromised node's pairwise keys which are established using this polynomial. Thus, the security of our model depends on the average number of sensor nodes sharing the same polynomial, which is the number of sensor nodes expected to be located in three neighboring hexagon cells.

We have described the deployment model in previous section. Denoted the average number of sensor nodes that are expected to be located in a cell is N_c, the average number of sensor nodes sharing a polynomial can be computed by:

$$N_G = 3N_c = 3\varpi S_c = \frac{3\sqrt{3}a^2\sigma^2\varpi}{2} \quad (2)$$

In this formula, ϖ is the sensor nodes density.

As described in previous section, the memory requirement for storing key materials is $M = 3\times(t + 1) \log_2 q$ *(bits)*, so the degree of bivariate polynomials is:

$$t = \left[\frac{M}{3}\right] - 1 \quad (3)$$

As long as $N_G \le t$, our scheme is perfect resistance against node captures. In other words, compromising of sensors does not lead to the compromise of direct keys shared between non-compromised sensors.

According to the analysis in [13], we consider a random attack here. We assume a fraction p_c of sensor nodes in the network have been compromised by an attacker. Among N_G sensor nodes that have polynomial shares, the probability that exactly i sensor nodes have been compromised can be evaluated by:

$$P_c(i) = \binom{N_G}{i} p_c^i (1 - p_c)^{N_s - i} \quad (4)$$

So, the probability that a bivariate polynomial is compromised can be calculated by:

$$P_c = 1 - \sum_{i=0}^{t} P_c(i) \quad (5)$$

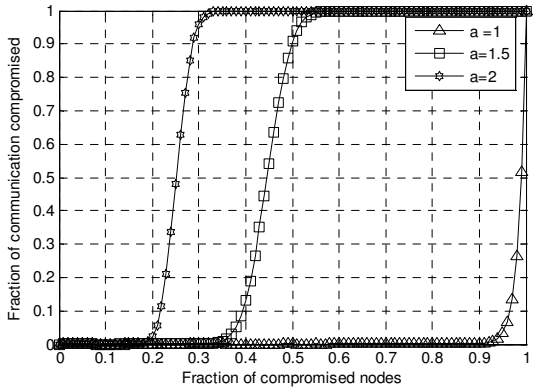


Figure 3. Network resilience against node compromised attack with different deployment point distances.

In Figure 3, we could see with longer the distance is, meaning that the cell size is larger, the more vulnerable the resilience against node compromised attacks is.

Because when the cell size is larger, there are more sensor nodes in a cell sharing a key-space, leading to lower the security.

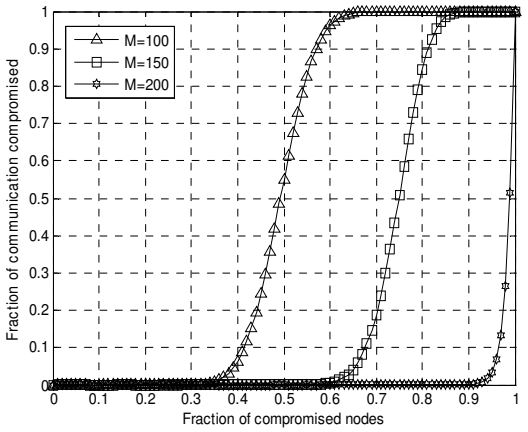


Figure 4. Network resilience against node compromised attacks with different memory sizes.

In Figure 4, with more memory, the resilience of network is strengthened, because the degree of polynomials is higher.

Comparing with other models in [4],[5],[13],[13], ours has better security in term of resilience against node compromised. In Figure 5, ours only need 150 keys storage to gain better security than CPPS [12],[13] and 8-square model [14] with 200 cryptographic keys storage.

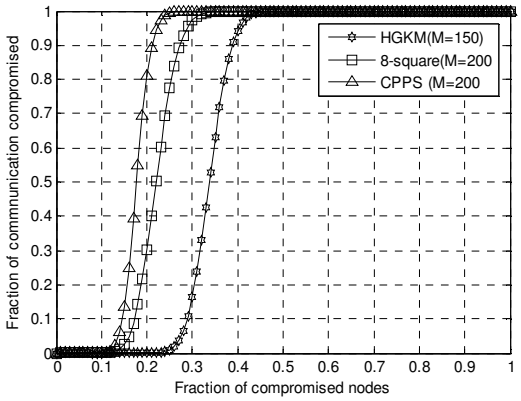


Figure 5. Comparison the fraction of communication compromised (a=1.5)

7. Conclusion

In this paper, we have described a realistic polynomial-based key predistribution approach which take advantage of pre-deployment knowledge with Gaussian distribution. We have shown that this model has more advantages in both performance (network connectivity, communication overhead, memory requirement) and security against node compromised attack. Our future work will focus on issues about privacy when broadcast message, reducing energy cost in communication and analysing the network connectivity with indirect key establishment.

Acknowledgment

This work is financially supported by the Ministry of Education and Human Resources Development (MOE), the Ministry of Commerce, Industry and Energy (MOCIE) and the Ministry of Labor (MOLAB) through the fostering project of the Lab of Excellency.

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci. “A survey on sensor networks”, In *IEEE Commun. Mag*. Vol 40, Issue 8, August 2002.

[2] A.D. Wood, and J.A. Stankovic, ”Denial of service in sensor networks”, In *IEEE Computer*. 54-62, October 2002.

[3] C. Karlof, D. Wagner. “Secure routing in wireless sensor networks: attacks and countermeasures”, In *First IEEE International Workshop on Sensor Network Protocols and Applications*, May 2003.

[4] L. Eschenauer, V. D. Gligor. "A key-management scheme for distributed sensor networks", In *Proceedings of the 9th ACM conference on Computer and communications security*, November 2002.

[5] H. Chan, A. Perrig, D. Song. "Random key predistribution schemes for sensor networks" In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, May 2003.

[6] R. Blom, “An optimal class of symmetric key generation systems”, In Proc of EUROCRYPT ’84, pp. 334-338, 1985.

[7] W. Du, J. Deng, Y. S. Han, P. K. Varshney. "A pairwise key predistribution scheme for wireless sensor networks", In *Proceedings of the 10th ACM conference on Computer and communications security*, 2003.

[8] W. Du; J. Deng; Y. S. Han; S. Chen; P.K Varshney. ”A key management scheme for wireless sensor networks using deployment knowledge”, In *23rd Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom’04)*, Hong Kong, China, March 21-25, 2004.

[9] W. Du, J. Deng, Y. S. Han, P. Varshney. "A Key Predistribution Scheme for Sensor Networks Using Deployment Knowledge". In *IEEE Transactions on Dependable and Secure Computing*, Volume 3 , Issue 1 (January 2006).

[10] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro and M. Yung. ”Perfect-secure key distribution

of dynamic conferences", In *Advances in Cryptography* – CRYPTO '92, LNCS 740, pp. 471-486, 1993.

[11] D. Liu, P.Ning. "Establishing pairwise keys in distributed sensor networks", In *ACM Transactions on Information and System Security*, February 2003.

[12] D. Liu; P. Ning. "Location-based pairwise key establishments for static sensor networks", In *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks (SASN '03)*, October 2003.

[13] D. Liu, P.Ning. "Improving Key Predistribution with Deployment Knowledge in Static Sensor Networks", In ACM Transactions on Sensor Networks, Vol. 1, No. 2, November 2005, pp. 204-239.

[14] Ngo Trong Canh, Tran Van Phuong, Young-Koo Lee, Sungyoung Lee, and Heejo Lee. "A Location-aware Key Predistribution Scheme for Distributed Wireless Sensor Networks," In The 15th IEEE International Conference on Networks (ICON '07), pp.188-193, 19-21 Nov. 2007.