We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

8

Key Establishment Protocol for Wireless Sensor Networks

Ali Fanian and Mehdi Berenjkoub
Department of Electrical and Computer Engineering, Isfahan University of Technology (IUT), Isfahan, Iran

1. Introduction

Wireless sensor networks usually comprise a number of sensors with limited resources. Each sensor includes sensing equipment, a data processing unit, a short range radio device and a battery [Pottie & Kaiser, 2000; Kahn et al.,1999; Akyildiz, 2002]. These networks have been considered for various purposes including border security, military target tracking and scientific research in dangerous environments [Perrig et. al., 2002; Kung & Vlah, 2003; Brooks, 2003]. Since the sensors may reside in an unattended and/or hostile environment, security is a critical issue. An adversary could easily access the wireless channel and intercept the transmitted information, or distribute false information in the network. Under such circumstances, authentication and confidentiality should be used to achieve network security. Since authentication and confidentiality protocols require a shared key between entities, key management is one of the most challenging issues in wireless sensor networks (WSNs) [Perrig et. al., 2002].

In the literature, key management protocols are based on either symmetric or asymmetric cryptographic functions [Perrig et. al., 2002]. Due to resource limitations in the sensors, key management protocols based on public keys are not suitable [Perrig et. al., 2002], [Chan et. al., 2003]. Hence, key management protocols based on symmetric cryptographic functions have been extensively investigated [Chan et. al., 2003-Fanian et.al, May 2010]. There are two types of symmetric key management schemes based on an on-demand trust center or key pre-distribution. With an on-demand trust center, the center must generate common keys for every pair of nodes that wish to establish a secure connection. Due to the lack of an infrastructure in WSNs, this scheme is not suitable. With key pre-distribution, key material is distributed among all nodes prior to deployment. In this scheme, each node carries a set of keys to establish a secure connection with other nodes.

A number of key pre-distribution schemes have been developed. A very simple approach is to have a unique pre-loaded key that is shared among the nodes. Then all sensors can encrypt or decrypt data between themselves using this key. Due to its simplicity, this method is very efficient in regards to memory usage and processing overhead, but it suffers from a very serious security problem. If even one of the sensors is captured by an adversary, the security of the entire network will be compromised. Another simple approach, called the basic scheme, is

to generate a distinct key between every pair of sensors and store these in the sensors. In this case, if N sensors are deployed in the network, each must store N-1 keys. Despite ideal resilience, this scheme is not scalable, and is not memory efficient, particularly in large networks. In addition, after node deployment, if a new node wants to join the network, none of the previously deployed sensors will have a common key with the new node. Recently, many key establishment protocols have been proposed to address this problem [Chan et. al., 2003- Fanian et. al., 2010], but as we will show most have security or performance issues. These schemes are based on random key pre-distribution, symmetric polynomials and/or the Blom scheme. As shown in the analysis section, with the protocols based on random key pre-distribution, an adversary can obtain the common key between non-compromised sensors by compromising some sensors. Thus, these schemes have a serious security problem. In the symmetric polynomial and/or Blom scheme, however, perfect security can be achieved but resource consumption is an issue. In this chapter, a key establishment protocol employing four key pre-distribution models for sensor networks with different requirements.

In this chpate, we propose a new key establishment protocol called HKey. In this protocol, both efficient resource consumption and perfect security are the goals this protocol. The approach is similar to that of the basic scheme where every pair of sensors has a unique common key. In the proposed protocol, each sensor has a secret key and a unique identity. The common key between two sensors is generated using the secret key of one node and the identity of the other. This key is stored only in the latter node. For example, suppose sensors A and B want to generate a common key. Before deployment, the key distribution center (KDC) generates a key, for example, using the secret key of A and the identity of B, and stores this key only in B. When these sensors want to establish a common key, sensor A can generate the key with its own secret key and the identity of B. Sensor B just retrieves this key from its memory. Hence the memory usage in the proposed scheme is half that of the basic scheme.

In HKey, we propose several different models based on the WSN requirements. In some of these models, the aim is low memory consumption in the sensors. In others, network connectivity and memory usage are equally important. In the last model, the goal is high connectivity. The models are deterministic, so every sensor knows whether or not it can establish a direct common key with another sensor. Since, every pairwise key between two sensors is unique, the security of the protocols is perfect. Also, it this protocol, only one node needs to store a common key, the common key can be generated between a new node and an old one based on the proposed protocol and the key stored in the new node. Therefore, this protocol is scalable. As we will show, this protocol is efficient in comparison other proposed protocols.

The rest of the chapter is organized as follows. Section 2 reviews some required primitives including related work. Details of our key establishment protocol are discussed in Section 3. Performance evaluation and security analysis of the proposed protocol are presented in Section 4. Finally, some conclusions are given in Section 5.

2. Background

Most of the proposed key establishment protocols in WSNs are based on random key pre-distribution, symmetric polynomials and/or the Blom scheme. In this section, we review some well known protocols based on these techniques.

2.1 Key establishment protocols based on random key pre-distribution

Eschenauer et al. [Eschenauer & Gligor, 2002] proposed a random key pre-distribution scheme for WSNs. In this approach, before deployment some keys from a large key pool are selected randomly and stored in the sensors. After deployment in the network, a pair of nodes may have a shared common key to establish a secure connection. If there is no common key between two sensors, they have to establish a key through an intermediate sensor node which has common keys with both sensors. In this method, there is a tradeoff between connectivity and security. Network connectivity is determined from the probability of direct key generation between two adjacent sensors. As the size of the key pool increases, connectivity decreases, but protocol security increases. Due to the distribution of random keys, it may not be possible to establish a common key between every pair of sensors.

Du et al. [Du, 2004] proposed a deployment knowledge key management protocol (denoted Du-1), based on the approach in [Eschenauer & Gligor, 2002]. In this case, deployment knowledge is modeled using a Gaussian probability distribution function (pdf). Methods which do not use deployment knowledge such as in [Eschenauer & Gligor, 2002], assume a uniform pdf for the node distribution in the network. In this case, sensors can reside anywhere in the network with equal probability. In [Du, 2004], the network area is divided into square cells and each cell corresponds to one group of sensors. The key pool is divided into sub key pools of size S, one for each cell, such that each sub key pool has some correlated keys with its neighboring sub key pools. Each sub key pool has αSc common keys with the horizontal and vertical neighboring sub key pools, and βSc common keys with the diagonal neighboring sub key pools, such that $4\alpha + 4\beta = 1$, with $\alpha > \beta$. Each sensor in a cell randomly selects m_R keys from its associated sub key pool.

2.2 Key establishment protocols based on symmetric polynomials

A symmetric polynomial [Borwein & Erde´lyi, 1995; Zhou & Fang, Apr. 2006; Zhou & Fang, Oct. 2006] is a t-degree (K+1)-variate polynomial defined as follows

$$f(x_1, x_2, ..., x_{K+1}) = \sum_{i_1=0}^{t} \sum_{i_2=0}^{t} ... \sum_{i_{k+1}=0}^{t} a_{i_1, i_2, ..., i_k, i_{k+1}} x_1^{i_1} x_2^{i_2} ... x_K^{i_K} x_{K+1}^{i_{K+1}} \quad (1)$$

All coefficients of the polynomial are chosen from a finite field F_q, where q is a prime integer. The polynomial f is a symmetric polynomial so that [Zhou & Fang, Apr. 2006]

$$f(x_1, x_2, ..., x_{K+1}) = f(x_{\partial(1)}, x_{\partial(2)}, ..., x_{\partial(K+1)}) \quad (2)$$

where ∂ denotes a permutation. Every node using the symmetric polynomial based protocol takes K credentials ($I_1, I_2, \ldots, I_K$) from the key management center, and these are stored in memory. The key management center must also compute the polynomial shares using the node credentials and the symmetric polynomial. The coefficients b_i stored in node memory as the polynomial share are computed as follows

$$f_u(x_{K+1}) = f(I_1, I_2, ..., I_K, x_{K+1}) = \sum_{i=0}^{t} b_i x_{K+1}^i \quad (3)$$

Every pair of nodes with only one mismatch in their identities can establish a shared key. Suppose the identities of nodes u and v have one mismatch in their identities $(c_1,c_2,\ldots,c_{i-1},u_i,c_{i+1},\ldots,c_K)$ and $(c_1,c_2,\ldots,c_{i-1},v_i,c_{i+1},\ldots,c_K)$, respectively. In order to compute a shared key, node u takes v_i as the input and computes $f_u(v_i)$, and node v takes u_i as the input and computes $f_v(u_i)$. Due to the polynomial symmetry, both nodes compute the same shared key. In [Zhou & Fang, Apr. 2006] it was shown that in order to maintain perfect security in the WSN, the polynomial degree must satisfy

$$\begin{cases} 0 \le N_i - 2 \le t \\ N_i K + 1 \sqrt{\dfrac{K(K+1)!}{2}} \le t \end{cases} \quad i = 1,2,...,K \tag{4}$$

where N_i is the number of nodes in group i.

Zhou et al. [Zhou & Fang, Apr. 2007] proposed another key management protocol named LAKE which is based on symmetric polynomials and deployment knowledge. In this scheme, the network is also divided into square cells and each cell is allocated to one group of sensors. A t-degree tri-variate symmetric polynomial is employed. Each sensor in this protocol has credentials (n_1, n_2), where n_1 and n_2 represent the cell identity and the sensor identity, respectively. According to Section 2-1, the sensor polynomial share is calculated and stored in the sensor. After deployment, sensors that have one mismatch in their credentials can directly compute a shared key.

Lin et al. [Liu & Ning, 2003] proposed another key management protocol called LPBK in which the network area is divided into square cells. Each cell has a specific symmetric polynomial which is used to compute the polynomial share for the sensors in the corresponding cell and four adjacent vertical and horizontal cells. Therefore, each sensor must store five polynomial shares in its memory. Then each sensor can directly compute a common key with the sensors in these five cells.

We proposed a key establishment protocol for large scale sensor networks based on both symmetric polynomials and random key pre-distribution called HKEP [Fanian et. al., Apr 2010]. In HKEP, both symmetric polynomials and random key pre-distribution are used to improve efficiency. In this scheme, key information is distributed to the sensors during the pre-deployment stage. Once the sensors have been deployed, they can produce a common key either directly or indirectly. Due to the use of two methods in HKEP, two types of information must be stored in the sensors. One is the sensor polynomial shares generated using the symmetric polynomials and finite projective plan, while the other is a set of random keys. Finite projective plane is a subset of symmetric BIBDs. There are two types of key generation in HKEP. In the first type, a common key between near sensors is generated via their polynomial shares. The polynomial shares for each sensor are distributed by a finite projective plan which is a symmetric design discus in combinatorial design theory. In the second type, a common key between far sensors is generated using the pre-distributed random keys. Since in this case a key may be selected by several sensors, the common key between two far sensors may also be used by other pairs of sensors. Conversely, the common key between near sensors is unique. As we will show, the proposed end to end key establishment protocol between every pair of sensors can be supported without significant overhead.

Another proposed key establishment protocol, called SKEP, is based on symmetric polynomial [Fanian et. al., 2011]. In SKEP, the network area is divided into non-overlapping hexagonal cells, and the sensors are allocated in groups to these cells. The center of a cell is defined as the deployment point of the sensors allocated to that cell. In SKEP, each cell has a distinct t-degree bi-variate symmetric polynomial given by

$$f(x_1, x_2) = \sum_{i_1=0}^{t} \sum_{i_2=0}^{t} a_{i_1,i_2} x_1^{i_1} x_2^{i2}$$

Each sensor has a triplet of credentials, (*i,j,k*). The first two credentials specify the deployment point of the sensor, while the last uniquely identifies each sensor in the cell. The polynomial share of a sensor, $f_k(x)$, can be computed from the symmetric polynomial assigned to cell *C*(*i,j*) and the sensor credential *k* as follows

$$f_i(x_2) = f_i(A, x_2) = \sum_{i_1=0}^{t} \sum_{i_2=0}^{t} a_{i_1,i_2} A^{i_1} x_2^{i2} = \sum_{i_2=0}^{t} b_{i_2,A} x_2^{i2}$$

$$, b_{i_2,A} = \sum_{i_1=0}^{t} a_{i_1,i_2} A^{i_1}$$

If the polynomial share of two sensors is generated from the same symmetric polynomial, these sensors can create a common key by exchanging their credentials. Before distributing sensors in the network, the secret information is placed in the sensors. Given the sensor distribution in the network, some sensors in neighboring cells or even non-neighboring cells can be adjacent to each other. In order to connect to the network, these sensors must be able to generate a common key. Therefore, some correlated secret information should be given to these sensors in order to generate this key. However, this should not consume a significant amount of sensor memory. To meet this requirement, SKEP generates a polynomial share from the symmetric polynomial allocated to each cell for a portion of the sensors in neighboring cells. The sensors containing this additional polynomial share can operate as agent nodes to indirectly generate common keys between sensors in neighboring cells. In order to generate this additional polynomial share, we divide each hexagonal cell into six virtual regions. The division of cell (*i,j*) into virtual regions is shown in figure 1.

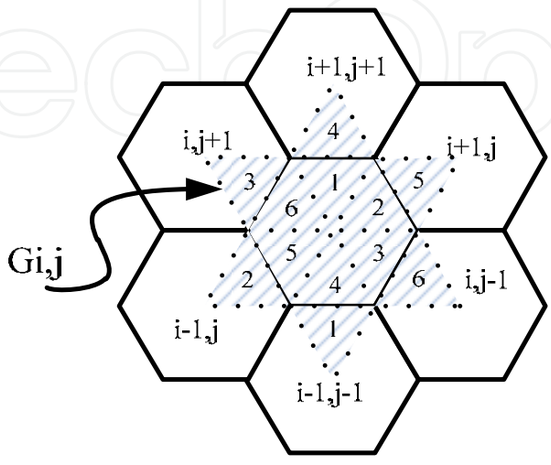


Fig. 1. Dividing each cell into six virtual regions.

Each sensor will belong to one of these virtual regions according to its credential. After deployment, a sensor may not reside in the virtual region it is allocated to. However, each sensor can infer adjacent sensors which have a suitable polynomial share, and can also find suitable agents to generate an indirect common key with the other sensors. As mentioned previously, each sensor has three credential (*i*,*j*,*k*), so two sensors can easily verify whether they are in a common group via their credentials.

2.3 Key establishment protocols based on blom's scheme

Blom proposed a key establishment protocol that allows each pair of nodes to establish a common key [Blom, 1985]. In this method, if no more than *t* nodes are compromised, the link between non-compromised nodes will remain secure. We refer to this as a *t-secure* method. To guarantee perfect security in a network with *N* nodes, an (*N*-2)-secure Blom scheme should be used. In the initialization phase, the key management center constructs a $(t+1)\times N$ matrix *G* over a finite field F_q, where *N* is the size of the network and *q* is a large prime. The matrix *G* is known by all nodes. Then the center constructs a random $(t+1)\times(t+1)$ symmetric matrix *D* over *GF*(*q*), and an $N\times(t+1)$ matrix $P=(D\cdot G)^T$, where *T* denotes transpose. The matrix *D* is kept secret in the center and is not revealed to any user. If *D* is symmetric, then $K=P\cdot G$ is also symmetric since

$$K=P\cdot G=(D\cdot G)^T\cdot G=G^T\cdot D^T\cdot G=G^T\cdot D\cdot G=G^T\cdot P^T=(P\cdot G)^T \tag{5}$$

Therefore, K_{ij} is equal to K_{ji}, where K_{ij} is the element in the *i*th row and *j*th column of *K*. In the Blom scheme, K_{ij} is used as a secret key between the *i*th and *j*th sensors. To generate this common key, the key management center assigns the *i*th row of *P* and *i*th column of *G* to user *i*, *i* = 1, 2, …, *N*. When nodes *i* and *j* want to establish a common key, they first exchange their columns of *G*. Then they can compute K_{ij} and K_{ji}, respectively, using their private row of *P* according to

$$K_{ij}=\begin{bmatrix}P_{i,1} & P_{i,2} & \dots & P_{i,t+1}\end{bmatrix}\cdot\begin{bmatrix}G_{1,j}\\ G_{2,j}\\ \cdot\\ \cdot\\ G_{t+1,j}\end{bmatrix} \tag{6}$$

As mentioned previously, *G* is public information, so the nodes can freely transmit their columns of *G*. It has been shown that if any *t*+1 columns of *G* are linearly independent then the Blom scheme is *t*-secure [Blom, 1985]. In this scheme, each sensor must store a row of *P* and a column of *G*. Therefore, the memory required is 2*t*+2. However, the structure of *G* can be exploited to reduce this memory requirement [Zhou et. al., 2005].

Du [Du et. al., 2006] proposed another key management protocol (denoted Du-2) using the Blom scheme. In this case, many pairs of matrices *G* and *D*, called the key spaces, are

produced, and some key spaces are assigned to each cell. Adjacent cells have some common key space as in [Du, 2004], where adjacent cells have correlation between their sub key pools. Each sensor selects τ key spaces randomly, and according to the Blom scheme, the required information is stored in the sensors. As a result, sensors with a common key space can produce a common key. As in [Du, 2004], two vertical or horizontal neighboring cells have αSc common key spaces, and two diagonal neighboring cells have βSc common key spaces, where Sc is the number of key spaces assigned to a cell.

Yu and Guan [Yu & Guan, 2008] also proposed a key management protocol based on the Blom scheme. In this protocol, the network area is divided into hexagonal cells and information on the associated matrices is stored in the sensors based on deployment knowledge. The matrices are assigned to the cells such that a confidential exclusive matrix, denoted A_i (equivalent to matrix D in the Blom method), is allocated to cell i. The sensors in a cell, according to their identities, take a row from the corresponding matrix so they can produce a common key directly. To generate a common key between sensors belonging to different cells, another confidential matrix B is employed. The B matrices are allocated to the cells based on two parameters b and w, where b is the number of matrices allocated to a group, and w is the number of rows selected by each sensor from these matrices. The analysis in [Yu & Guan, 2008] shows that the best results are obtained with w=2 and b=2. In this approach, the cells are divided into two categories, base cells and normal cells. Base cells are not neighbors, but normal cells are neighbors with two base cells. To produce a common key between sensors in neighboring cells, a confidential matrix B is allocated to each base cell together with its six neighbors. Using the Blom scheme with this matrix, the necessary information is stored in the sensors. Then the sensors in the seven neighboring cells can produce a common key directly. Since each normal cell is a neighbor with two base cells, normal cell sensors receive information from two B matrices. Although the connectivity of this scheme is close to one, the memory consumption is extremely high.

We proposed another key establishment protocol for low resource wireless sensor networks based on the Blom scheme and random key pre-distribution called KELR [Fanian et. al., May 2010]. In this protocol, the Blom scheme is used to establish common keys between sensors in a cell. Therefore, the key distribution center constructs distinct matrices $G_{i,j}$ and $D_{i,j}$ for each cell $C(i,j)$. Each sensor has a triplet of credentials (i,j,k). The first two credentials specify the deployment point of the sensor and the last is the unique ID of the sensor in the cell. The center uses this unique identifier to construct $G_{i,j}$. Since the sensors belonging to a cell use the same matrix $D_{i,j}$, they can directly generate a common key.

Given the sensor distribution in the network, some sensors belonging to neighboring cells or even non-neighboring cells can be deployed adjacent to each other. In order to connect the network, these sensors should be able to generate a common key, so secret information must be allocated to enable this. However, this should not consume a lot of memory. To meet this requirement, KELR employs random key pre-distribution. We could also use the Blom scheme to establish common keys among sensors in neighboring cells, but this would result in high memory consumption.

3. The new key establishment protocol

Nodes are typically mobile in ad-hoc networks while in sensor networks they are assumed to be static after deployment. Therefore, deployment knowledge can be quite useful in producing common keys among sensors. In addition, in most WSN applications, a secure peer-to-peer connection between remote sensors is unnecessary [Chan et. al., 2003-Fanian et.al, May 2010]. Therefore, the main goal is establishing secure connections between adjacent sensors, so knowledge of probable neighbors can be beneficial in key pre-distribution. In fact, if one can predict the adjacency of sensors in the network, a key management protocol can be developed with high efficiency and low cost. However, due to the inherent randomness of sensor distribution, it is impossible to specify the exact location of each sensor; knowing the probable neighbors is much more realistic. Deployment knowledge is exploited to generate key material in the pre-deployment phase. We first present our key pre-distribution protocol and then consider its use with different models.

3.1 The new key pre-distribution protocol

As mentioned in Section 2, most key establishment protocols are based on symmetric polynomials, the Blom scheme and/or random key pre-distribution [Chan et. al., 2003-Fanian et.al, May 2010]. In this section, a High performance Key establishment protocol, HKey, is proposed which is not based on these techniques. The goal of this new protocol is efficient resource consumptions and perfect security. The approach is similar to that of the basic scheme where every pair of sensors has a unique common key. As mentioned in Section 1, in this case a distinct key must be generated and stored for every pair of sensors, so memory consumption will be excessive in a large scale WSN. Thus while this scheme is quite simple, it has poor scalability. The goal with HKey is to retain the simplicity of the basic scheme while providing scalability and memory efficiency. In the proposed protocol, each sensor has a secret key and a unique identity. The common key between two sensors is generated using the secret key of one node and the identity of the other. This key is stored only in the latter node. For example, suppose sensors A and B want to generate a common key. Before deployment, the key distribution center (KDC) generates a key, for example, using the secret key of A and the identity of B, and stores this key only in B. When these sensors want to establish a common key, sensor A can generate the key with its own secret key and the identity of B. Sensor B just retrieves this key from its memory. Hence the memory usage in the proposed scheme is half that of the basic scheme.

In a group of Ng sensors, each sensor must store $Ng/2$ keys to establish a secure connection with all sensors in the group. To generate common keys, the KDC establishes a key map matrix. This map is an $Ng \times Ng$ matrix which determines whether the secret key or the identity of the corresponding sensor is used to generate the common key. The KDC may generate the key map randomly such that the memory usage for each sensor is not more than $\left\lceil \frac{Ng-1}{2} \right\rceil$. In Table 1 the key map for a group of 8 sensors is shown. We assume the sensor identities are 1 to 8. In Table 1, '√' in location (i,j) indicates that the common key between sensors i and j is generated by the secret key of the jth sensor and the identity of the ith sensor, so the key must be stored in the ith sensor. Conversely, '–' indicates that the common key between sensors i and j is generated from the secret key of the ith sensor and

the identity of the *j*th sensor, so the key must be stored in the *j*th sensor. Each sensor stores a maximum of only 4 keys.

Sensor Identity	1	2	3	4	5	6	7	8
1	×	–	√	–	√	–	√	–
2	√	×	–	√	–	√	–	√
3	–	√	×	–	√	–	√	–
4	√	–	√	×	–	√	–	√
5	–	√	–	√	×	–	√	–
6	√	–	√	–	√	×	–	√
7	–	√	–	√	–	√	×	–
8	√	–	√	–	√	–	√	×

Table 1. Key Map for a Group with 8 Sensors

The common keys are generated based on the key map generated in the pre-deployment phase. As mentioned previously, each sensor has a unique identity which is a number between 1 and *Ng*. To generate the common key K_{ij}, the KDC uses a one-way hash function with the secret S_i and identity N_j as follows

$$K_{ij} = H(S_i \| N_j)$$

Unlike the basic scheme, the proposed protocol is scalable. In the basic scheme, since deployed nodes do not have a common key with a new node, they cannot establish a secure connection. In the proposed protocol, since only one node needs to store a common key, it can be generated between a new node and an old one based on the proposed protocol and the key stored in the new node. Therefore, all nodes can establish a secure connection.

3.2 Network and deployment model

In HKey, the network area is divided into non-overlapping hexagonal cells, and the sensors are allocated in groups to these cells. The center of a cell is defined as the deployment point of the sensors allocated to that cell. Figure 2 shows the division of the network into hexagonal cells. Each cell in HKey has a pair of credentials (*i,j*) which is the cell position. Using two-dimensional Cartesian coordinates and assuming that the deployment point of cell *C*(*i,j*) is (x_i,y_i), the pdf of the sensor resident points is

$$f_k^{ij}(x,y|k\in C(i,j)) = f(x-x_i, y-y_j) = \frac{1}{2\pi\sigma^2} e^{-\left[(x-x_i)^2+(y-y_j)^2\right]/2\sigma^2}$$
$$\text{where } f(x,y) = \frac{1}{2\pi\sigma^2} e^{-\left[x^2+y^2\right]/2\sigma^2} \quad (7)$$

And δ is the standard deviation. Assuming identical pdfs for all group of sensors, we can use $f_k(x,y \mid k \in C(i,j))$ instead of $f_k^{ij}(x,y|k \in C(i,j))$. As in [Liu & Ning, 2003-Du, 2006], in HKey it is assumed that the routing protocol delivers transmitted data to the correct destinations. A typical distribution for the sensors belonging to cell *C*(1,2) is shown in figure 2.

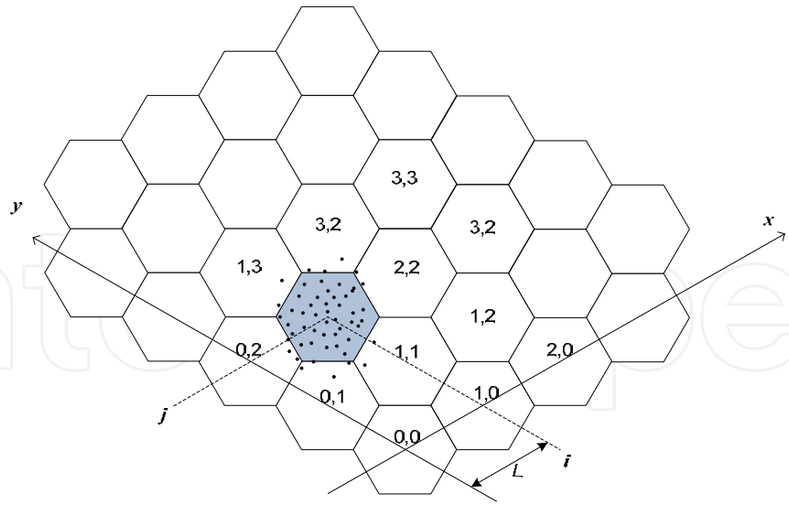


Fig. 2. A sensor network with distance L between adjacent deployment points. The center of each cell is defined as the deployment point.

3.3 Pre-distribution of secret information

The secret information for generating a common key should be produced before deployment. As mentioned in Section 3-1, in the proposed protocol, each sensor has a secret key, a unique identity, and some common keys with other sensors stored in its memory. In HKey, each sensor is able to establish a common key with any sensor in the same cell. If only common keys between sensors in a cell are generated, adjacent sensors belonging to different cells will not be able to establish a secure connection. Since sensor deployment follows a Gaussian distribution, it may be that two sensors from neighboring cells are adjacent to each other. Thus some sensors must be able to establish a common key with sensors in neighboring cells. For this purpose, in HKey the pre-distributed common keys are generated in two phases. In the first phase, the KDC generates the common keys for sensors belonging to a cell based on the proposed protocol in Section 3-1. In the second phase, common keys are distributed so that some sensors belonging to neighboring cells can establish a direct common key.

The percentage of sensors that have second phase keys has a great influence on network connectivity and memory consumption. Thus in HKey, we propose several different models based on the WSN requirements. In some of these models, the aim is low memory consumption in the sensors. In others, network connectivity and memory usage are equally important. In the last model, the goal is high connectivity. The models are deterministic, so every sensor knows whether or not it can establish a direct common key with another sensor. In figure 3, the four models are depicted.

First, a Low Resource consumption (HKey-LR) model is proposed for very low resource sensors. In HKey-LR, each cell is divided into two virtual regions. Virtual regions are also used in some of the other proposed models. In this case, cells are divided into regions, and in the pre-deployment stage each sensor in a cell is assigned to one of these regions. For example, if a cell with Nc sensors is divided into two virtual regions, as shown in figure 3(a), sensors with identities from 1 to $Nc/2$ are assigned to the first virtual region, and the

remainder to the second region. After deployment, a sensor may not reside in their virtual region. However, during the key generation process, each sensor will know which adjacent sensors they can establish a common key with. In HKey-LR, a group consists of virtual regions from three neighboring cells. In figure 3(a), the network is divided into triangular areas. In Phase 2, the common keys are generated based on the proposed scheme with a small change. Since common keys among sensors belonging to a cell are generated in Phase 1, we should not produce any more of these keys in Phase 2. Therefore, in the second phase, the KDC only generates common keys among sensors in a group which belong to different cells. For instance, suppose each cell has 6 sensors, so a group has 9 sensors from three neighboring cells. Assume these sensors are A_1, A_2, A_3 in cell A, B_1, B_2, B_3 in cell B, and C_1, C_2, C_3 in cell C. The common keys among sensors in this group are shown in Table 2. In this Table, K_{ij} is the common key between sensors i and j generated using the secret key of sensor i and the identity of sensor j.

A_1	A_2	A_3	B_1	B_2	B_3	C_1	C_2	C_3
K_{B1-A1}	K_{B1-A2}	K_{B1-A3}	K_{C1-B1}	K_{C1-B2}	K_{C1-B3}	K_{A1-C1}	K_{A1-C2}	K_{A1-C3}
K_{B2-A1}	K_{B2-A2}	K_{B2-A3}	K_{C2-B1}	K_{C2-B2}	K_{C2-B3}	K_{A2-C1}	K_{A2-C2}	K_{A2-C3}
K_{B3-A1}	K_{B3-A2}	K_{B3-A3}	K_{C3-B1}	K_{C3-B2}	K_{C3-B3}	K_{A3-C1}	K_{A3-C2}	K_{A3-C3}

Table 2. Common Keys Generated in Phase 2

Second, a Medium Resource consumption (HKey-MR) model is proposed for low resource sensors. Cells in HKey-MR are divided into two types called base cells and normal cells. As shown in figure 3(b), cells C(i,j) and $C(i+1,j+2)$ are base cells. Note that base cells are not neighbors of each other. Each normal cell is the neighbor of two base cells, and is divided into two virtual regions. A group consists of a base cell and virtual regions in the six neighboring cells. Each virtual region belongs to one group. The common keys among sensors belonging to different cells in a group are generated in Phase 2.

Third, an Advanced Medium Resource consumption (HKey-AMR) model is proposed. In HKey-AMR, as shown in figure 3(c), the network cells are divided into even and odd rows. Each cell located in an odd row is divided into two virtual regions. In this model each cell along with its six neighboring cells establishes a group. Each cell along with its neighboring cell that is in the same row establishes one group. In other words, the sensors in cell $C(i,j)$ belong to a distinct group with the sensors in cell $C(i-1,j)$ and also to a group with the sensors in cell $C(i+1,j)$, for j even or odd. If a cell is located in an even row, it will establish four distinct groups with neighboring cells located in odd rows. In figure 3(c), $C(i,j)$ is in an even row, so its sensors along with the sensors belonging to a virtual region of cells $C(i,j+1)$, $C(i+1,j+1)$, $C(i-1,j-1)$ and $C(i,j-1)$ establish four distinct groups. Common keys among sensors belonging to different cells in the groups are generated according to Phase 2 of the proposed protocol.

Finally, a High Performance (HKey-HP) model is proposed. In HKey-HP, similar to HKey-MR, the cells are divided into normal cells and base cells. Each group consists of one base cell and two neighboring normal cells, as shown in figure 3(d) for cell $C(i,j)$. In this case, each cell is a member of three groups. The common keys among sensors within groups are generated in Phase 2.

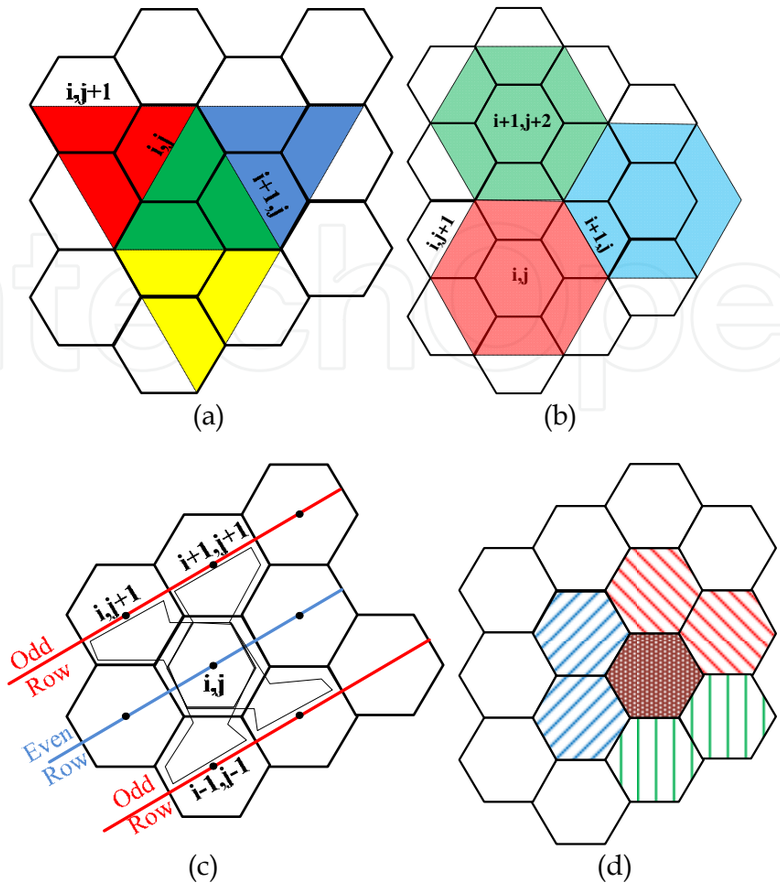


Fig. 3. The proposed key distribution models. a) Low Resource consumption (HKey-LR) b) Medium Resource consumption (HKey-MR) c) Advanced Medium Resource consumption (HKey-AMR) d) High Performance (HKey-HP). In a), b) and c), half of cell (i,j+1) is a virtual region. In b) cells (i,j) and (i+1,j+2) are base cells and their neighbors are normal cells. The shading denotes a group.

3.4 Direct key calculation

In the four proposed models, two sensors belonging to a cell can establish a shared key directly. If they do not belong to the same cell, they may be located in a common group. Based on the model employed, the sensors can easily determine if they are in the same group or not. Sensors in a group can also establish a common key directly.

3.5 Indirect key calculation

Considering the Gaussian distribution for sensor deployment, sensors in two distinct groups may be adjacent to each other. In this case, they cannot generate a common key directly and suitable agents must be found. A proper agent node is one that can generate common keys directly with both sensors. Since a number of sensors in two neighboring cells can directly generate common keys with sensors in both cells, at most two agent nodes suffice to generate a common key between any two adjacent sensors that cannot establish a common

key directly. If the distance between sensor deployment points is large, establishing a common key between them may require several agents. Note that if the resident point distance between agents is less than the wireless transmission range, they can communicate with each other directly; otherwise, a routing protocol is needed to connect them such as in [Du, 2004 - Yu & Guan, 2008]. The performance and security of indirect common key generation is greatly affected by the number of agent nodes. In HKey, the number of usable agents is typically high to ensure efficient key generation.

4. Security analysis and performance evaluation

In this section the security analysis and performance evaluation of the proposed protocol are presented and compared with similar protocols including Du-1 [Du, 2004], LAKE [Zhou & Fang, Apr. 2007], LPBK [Liu & Ning, 2003], Du-2 [Du et. al., 2006] and Yu and Guan [Yu & Guan, 2008]. Using the threat model in [Su et. al, 2010], we assume that an adversary can obtain all secret information from compromised sensors, and their goal is to obtain the common keys between non-compromised sensors. An adversary may be an insider or outsider. An outsider does not have any prior information about the network or the relationship between sensors. In contrast, an insider can have significant information about the network such as the deployment model, sensor groups, etc., and thus may know which sensors can establish a common key with a given sensor directly. This information is very useful for some attacks such as network discovery, false route injection, common key compromising, etc [Su et. al, 2010]. Nevertheless, when the common key between every pair of sensors is distinct and independent from any other keys such as in the basic scheme and the proposed protocol, a brute-force attack is the only option for an adversary (insider/outsider) to compromise common keys. In other words, additional information about the sensors and the network does not help in finding a common key between two non-compromised sensors. Unlike the basic scheme and our proposed protocol, with the other protocols based on random key pre-distribution, the Blom scheme and/or symmetric polynomials, insider information can be exploited to compromise the common key between non-compromised sensors. The adversary is assumed to be an outsider who compromises sensor nodes randomly.

4.1 Network configuration

We consider a WSN with the following parameters similar to those in [Du et. al., 2006]:

- The number of sensors in the network is 10,000.
- The network area is $1000m \times 1000m$.
- Sensors have a two dimensional Gaussian distribution with standard deviation $\sigma = 50\ m$.
- The number of sensors in each cell is 100.
- The wireless transmission range is 40 m.

4.2 Local connectivity

Local connectivity is defined as the probability of direct key generation between two adjacent sensors. Each sensor can establish a common key with some adjacent sensors

directly, and with other adjacent sensors indirectly. Three parameters are most relevant to the local connectivity. The first is the sensor distribution, which can be uniform or non-uniform. With a uniform distribution, each sensor may reside anywhere in the network. Since the available memory for each sensor is restricted, each sensor can establish a common key with a limited number of sensors. Therefore in this situation the local connectivity is often low [Eschenauer & Gligor, 2002]. With a non-uniform distribution (typically Gaussian), there is a greater chance that two adjacent sensors are members of the same cell or group. Thus in this case the local connectivity can be much higher. The second parameter is the shape and size of the groups. This influences how the correlated key information is distributed among sensors in neighboring cells. The last parameter is the average number of accessible sensors for each sensor. This is related to the sensor wireless transmission range and the density of the sensor distribution.

In this section, we use deployment knowledge to distribute secret information to the sensors. We compare our proposed models with those given above using the specified network parameters. In the scheme in [Du, 2004], each cell has a sub key pool with S keys, and there are some common keys between neighboring cell sub key pools. As mentioned in Section 2-3, the two horizontal or vertical neighboring cell sub key pools have αS common keys, and the two diagonal neighboring cells have βS common keys. The probability of common key establishment between two sensors in a cell with this technique is

$$P_C = 1 - \frac{\binom{S-m}{m}}{\binom{S}{m}} \tag{8}$$

where m is the number of keys in each sensor. The sub key pools of two neighboring cells have ωS shared keys where ω is equal to α for horizontal or vertical neighboring cells and β for the other neighboring cells. Two sensors belonging to neighboring cells can establish a common key if they select at least one common key from the shared keys in their sub key pools. The probability of having i common keys for these sensors is

$$\frac{\binom{\omega S}{i}^2 \binom{S-\omega S}{m-i}^2}{\binom{S}{m}^2}$$

The probability of common key establishment between two sensors belonging to neighboring cells is then

$$P_C = \sum_{i=1}^{\min(\omega.S,m)} \frac{\binom{\omega S}{i}^2 \binom{S-\omega S}{m-i}^2}{\binom{S}{m}^2} \tag{9}$$

With the approach in [Du et. al., 2006], *Sc* key spaces are assigned to a cell, where each key space is a Blom matrix. Similar to [Du, 2004], two neighboring cell key spaces have some common matrices. In this scheme each sensor selects τ matrices from the corresponding key spaces. Thus the probability of establishing a common key between two adjacent sensors can be computed similar to (8) and (9).

In LAKE, a tri-variate *t*-degree symmetric polynomial is used to generate a polynomial share for each sensor. With this scheme, every pair of sensors belonging to a cell can establish a common key, but only one sensor belonging to each cell can establish a direct key with a given sensor in another cell. In LPBK and the approach of Yu and Guan [Yu & Guan, 2008], two adjacent sensors belonging to a cell or neighboring cells can establish a direct common key.

In the proposed models, every pair of sensors belonging to a cell can establish a direct common key. However, only some sensors belonging to neighboring cells can establish a direct common key. In HKey-LR the probability of establishing a common key for these sensors is 1/6. In HKey-MR, if one sensor is in a base cell and another is in a neighboring normal cell, the probability is 0.5. If the sensors are in different normal cells around a base cell, the probability of a common key existing is 0.25. In HKey-AMR, every pair of sensors belonging to neighboring cells in the same row can establish a common key directly. However, the probability for nodes belonging to other neighboring cells is 0.5. In HKey-HP, every pair of sensors belonging to neighboring cells can establish a direct common key.

We simulated the local connectivity of the techniques considered, and the results are given in figure 5. Our simulation program was developed using the C++ language in the Visual Studio .Net environment. This table shows that HKey-LR has lower local connectivity compared to the other proposed models. However, as we will discuss in the next section, the memory usage with this model is lower than with the others. The local connectivity with LAKE is less than the other schemes. Moreover, the local connectivity for HKey-HP, the approach by Yu and Guan [Yu & Guan, 2008], and LPKB are approximately equal to one. However as we will show, the memory usage in our proposed model is lower compared to these schemes. In the simulations, we assumed $\alpha = 15\%$ and $\beta = 10\%$ for the approaches in [Du, 2004] and [Du et. al., 2006].

4.3 Memory usage

Memory in sensors is often very restricted, so the key establishment protocol should use memory efficiently. In this section, the memory usage is determined for the proposed HKey models and the other schemes considering perfect security. With perfect security, an adversary who wants to compromise the common key between two non-compromised sensors cannot do better than a brute-force attack or capturing at least one of these sensors. To achieve perfect security when symmetric polynomials are used, the polynomial degree, *t,* must satisfy (4). The Blom scheme is a *t*-secure scheme, so that if no more than *t* nodes are compromised, the link between non-compromised nodes will remain secure. In this case, *t* specifies the size of the Blom matrix and memory usage. Since in our protocol the common keys among sensors are generated using their secret keys, it intrinsically has perfect security.

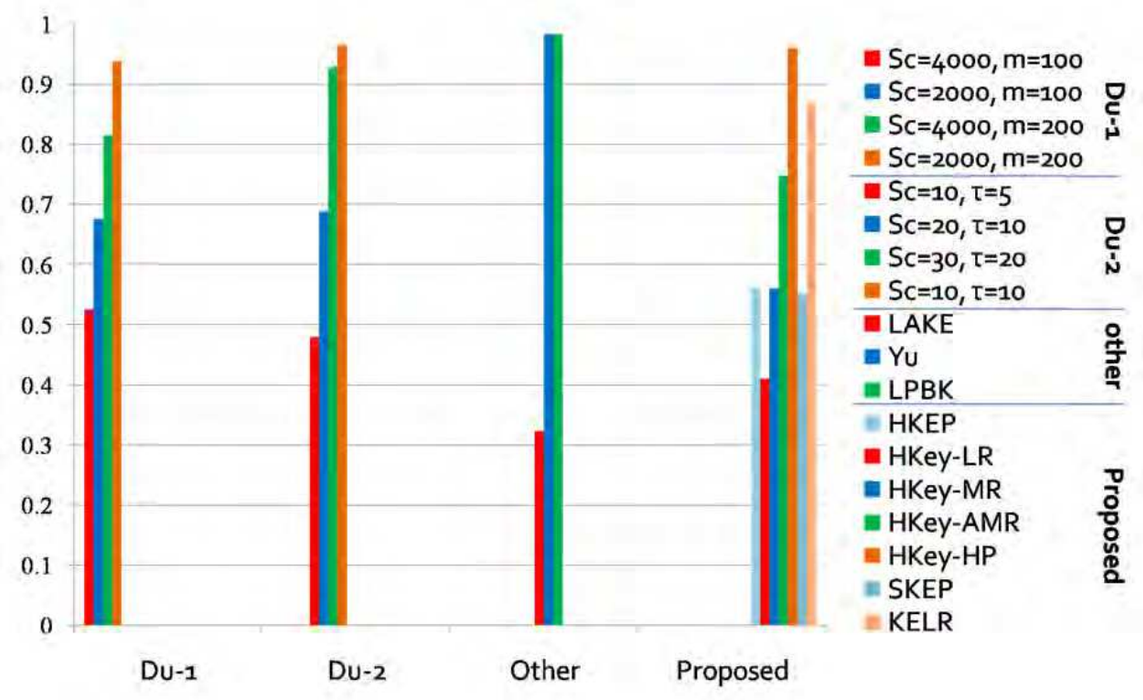


Fig. 4. Comparison of Local Connectivity with Different Techniques

4.3.1 Memory usage for HKey-LR

In HKey-LR, as shown in figure 3(a), each cell is divided into two virtual regions. In this model, a group consists of virtual regions from three neighboring cells. In Phase 2, only the common keys for sensors belonging to different cells are generated. The total memory usage comprises that required for keys generated in Phase 1 ($M_{LR,1}$) and Phase 2 ($M_{LR,2}$). As mentioned previously, in Phase 1 the KDC generates common keys for all sensors in a cell. If the number of sensors in a cell is Nc, the memory usage for each sensor in this phase will be $\left\lceil \frac{N_C}{2} \right\rceil$. In Phase 2, the KDC generates common keys for the groups. Since there are different regions in a group, the KDC generates common keys for the pairs of regions in the group. As shown in figure 3(a), there are three virtual regions in a group each from a neighboring cell. Let the virtual regions in a group be v_1, v_2, and v_3. In Phase 2, the KDC generates common keys among sensors in pairs (v_1, v_2), (v_1, v_3), and (v_2, v_3) independently such that the memory usage for all sensors is similar. If the number of sensors in v_1, v_2, and v_3 is n_1, n_2, and n_3, respectively, the number of common keys among sensors in (v_i, v_j) is $n_i \times n_j$. These keys must be distributed among n_i+n_j sensors. In HKey-LR, since each virtual region has $Nc/2$ sensors, the number of common keys for every pair of virtual regions is $Nc^2/4$. Since the number of sensors in each region is identical and the group structure is the same, the number of common keys distributed to the sensors in every pair of regions is the same. Therefore each sensor in the pair (v_i, v_j) must store $Nc/4$ keys out of $Nc^2/4$ keys. As a result, the memory usage for this model is

$$
\begin{aligned}
M_{\mathrm{LR},1} &= \left\lceil \frac{N_C}{2} \right\rceil \\
M_{\mathrm{LR},2} &= 2 \times \left\lceil \frac{N_C}{4} \right\rceil \\
M_{\mathrm{LR}} &= M_{\mathrm{LR},1} + M_{\mathrm{LR},2} \approx N_C
\end{aligned}
\tag{10}
$$

4.3.2 Memory usage for HKey-MR

In HKey-MR, as shown in figure 3(b), cells are divided into base cells and normal cells, and each normal cell is divided into two virtual regions. Each group consists of a base cell (v_1) and six virtual regions from neighboring cells (v_2 to v_7). Unlike HKey-LR, in HKey-MR the size of the regions is not equal. Although the number of sensors in six regions of a group is identical, the number of sensors in one region of that group is different. The KDC should generate and distribute common keys such that the memory usage for all sensors is close to identical. In this model, the number of sensors in v_1 is Nc and in the other regions is $Nc/2$. Thus, the number of common keys between v_1 and v_i (i = 2, 3, ..., 7) is $Nc^2/2$ and the number of common keys between v_i and v_j (i,j : 2, 3, ..., 7) is $Nc^2/4$. To balance the memory usage, we assume fractions α and β of the common keys are stored in v_1 and v_i , i = 2, 3, ..., 7, respectively. Since the sensors in v_1 have common keys with all sensors in v_2, v_3, …, v_7, the memory usage for each sensor in v_1 is $3\alpha Nc$ in Phase 2. On the other hand, the common keys for the other regions can be equally distributed among them. Therefore, each sensor in v_i (i = 2, 3, ..., 7) has βNc common keys with v_1 and $Nc/4$ common keys with v_j (j=2, 3, ..., 7, $i \neq j$), so that in total there are $\beta Nc + 5Nc/4$ keys in Phase 2. To ensure the same memory usage for all sensors, α and β must satisfy

$$
\begin{cases}
3\alpha Nc = \beta Nc + 5Nc/4 \\
\alpha + \beta = 1
\end{cases}
\qquad \text{giving } \alpha = \frac{9}{16} \text{ and } \beta = \frac{7}{16}
\tag{11}
$$

Therefore, the memory usage for HKey-MR is

$$
\begin{aligned}
M_{\mathrm{MR},1} &= \left\lceil \frac{N_C}{2} \right\rceil \\
M_{\mathrm{MR},2} &= \left\lceil \frac{27N_C}{16} \right\rceil \\
\text{giving } M_{\mathrm{MR}} &= \left\lceil \frac{35N_C}{16} \right\rceil \approx 2.2Nc
\end{aligned}
\tag{12}
$$

4.3.3 Memory usage for HKey-AMR

In HKey-AMR, as shown in figure 3(c), each cell and its neighboring cells establish six distinct groups. As described in Section 3-3, each cell constructs a group with each of its

neighboring cells in the same row. Since the groups are similar, the common keys are distributed equally among the sensors in the group. Therefore, the number of common keys which should be stored in each sensor is $Nc/2$. In addition, each cell located in an even row constructs four separate groups with the virtual regions located in neighboring cells in odd rows. In this case, the two regions have different sizes, and the number of common keys generated for each group is $Nc^2/2$. In order to use the same memory in all sensors, similar to HKey-MR, we assume fractions α and β of the keys are stored in the sensors of even row cells (full-cells) and virtual regions (half-cells). Therefore the number of common keys stored in each sensor belonging to full and half cells is $\alpha Nc/2$ and βNc, respectively. Each virtual region located in an odd row belongs to two groups with neighboring cells in an even row. Since each full cell constructs four distinct groups with four half cells, the Phase 2 memory usage for a sensor belonging to an even row cell is $Nc + 2\alpha NC$, and for a sensor belonging to an odd row cell is $Nc + 2\beta NC$. Thus $\alpha = \beta = 1/2$ provides the required balanced memory usage in all sensors. The required memory for this model is then

$$\begin{aligned} M_{\mathrm{AMR},1} &= \left\lceil \frac{N_C}{2} \right\rceil \\ M_{\mathrm{AMR},2} &= \left\lceil \frac{N_C}{2} \right\rceil + \left\lceil \frac{N_C}{2} \right\rceil + 4 \times \left\lceil \frac{N_C}{4} \right\rceil \\ &\text{giving } M_{\mathrm{AMR}} \approx 2.5 N_C \end{aligned} \tag{13}$$

4.3.4 Memory usage for HKey-HP

In HKey-HP, as shown in figure 3(d), each group consists of regions of three neighboring cells, and each cell is in three groups. The common keys among sensors in different cells are generated in Phase 2. Since groups in this model have the same structure and population, the common keys are equally distributed among the sensors. If the number of sensors belonging to the neighboring cells is n_1, n_2 and n_3, then the number of common keys generated in Phase 2 is $n_1 \times n_2 + n_1 \times n_3 + n_2 \times n_3$. These keys must be equally distributed among $n_1 + n_2 + n_3$ sensors. Since the number of sensors in each cell is N_c, the memory usage for this model is

$$\begin{aligned} M_{\mathrm{HP},1} &= \left\lceil \frac{N_C}{2} \right\rceil \\ M_{\mathrm{HP},2} &= 3N_C \\ &\text{giving } M_{\mathrm{HP}} \approx 3.5 N_C \end{aligned} \tag{14}$$

4.3.5 Memory usage for other schemes

The memory usage for the other schemes is computed based on the perfect security condition. In LPBK, as mentioned previously, the key establishment protocol is based on symmetric polynomials. As shown in (4), in order to maintain perfect security the polynomial degree must be $N_i - 2 \le t$, where N_i is the number of polynomial shares generated from a symmetric polynomial. Therefore, N_i is equal to $5Nc$. To satisfy (4), the degree, t, of each symmetric polynomial must be at least $5Nc - 2$. From Section 2-2, each

sensor receives five polynomial shares from five distinct symmetric polynomials. Since each sensor polynomial share has *t* coefficients and each sensor has five polynomial shares, the memory usage for LPBK with perfect security is $5(5Nc-2)$.

In LAKE, a tri-variate *t*-degree symmetric polynomial is used. In this scheme, each sensor can establish a direct common key with $2Nc$ nodes. According to (4), to achieve perfect security the polynomial degree and thus memory usage must be at least $2Nc-2$.

The key establishment protocols Du-2 [Du et. al., 2006] and Yu and Guan [Yu & Guan, 2008] are based on the Blom scheme. In this case, the KDC creates an $N_i \times (t+1)$ symmetric matrix. Each sensor stores a row of this matrix. As discussed in Section 2-3, the Blom scheme is *t*-secure if any subset with a maximum of *t* colluding sensors cannot compromise the common key between two other sensors. Therefore to achieve perfect security, the number of sensors which receive a row from the matrix must be less than *t*. With the approach in [Yu & Guan, 2008], in the best case, each base cell has six neighboring normal cells and each normal cell is the neighbor of two base cells. Then sensors in a base cell and the six neighboring normal cells have different rows from a Blom matrix, so *t* must be at least 7*Nc*. Each sensor in a normal cell receives two rows from two matrices corresponding to its two neighboring base cells, so the required memory with the approach in [Yu & Guan, 2008] is at least 14*Nc*. With the approach in [Du et. al., 2006], as stated in Section 2-3, each key space may exist in a maximum of two neighboring cells. Therefore, 2*Nc* sensors may select from the shared key space, so *t* must be at least 2*Nc*, and each sensor selects τ key spaces from *Sc* spaces. The minimum value for τ is one. As τ increases, the memory usage and local connectivity (as shown in figure 4), also increases.

With the approach in [Du, 2004], which is also based on random key pre-distribution, each sensor selects *m* keys from a sub key pool. In this scheme, perfect security is not possible. Based on the memory analysis above, Table 3 summarizes the required memory for each scheme. This table shows that the memory usage with HKey-LR and HKey-MR is less than with the other schemes. LAKE also has low memory consumption. Although the memory usage with LAKE is almost equal to that with HKey-MR, the corresponding local connectivity is only 0.3228 compared 0.5601. Among the schemes with local connectivity close to one, the memory required with HKey-HP is lowest.

Scheme	Memory cost	Memory required to ensure secrecy of a direct key between sensors
LAKE	*t*+1	$2Nc-1$
LPBK	5(*t*+1)	$5(5Nc-2)$
Yu and Guan	2(*t*+1)	$14N_c+1$
Du-2	$O(\tau(t+1))$	$O(\tau(t+1))$
SKEP	2(*t*+1)	4*Nc*-1
KELR	*t*+1	*Nc*-1
HKey-LR	-	N_C
HKey-MR	-	$2.2Nc$
HKey- AMR	-	$2.5N_C$
HKey- HP	-	$3.5N_C$

Table 3. Memory Required with Different Schemes

4.4 Resilience against key exposure

Since sensors are deployed in hostile environments and the sensor hardware may not be tamper proof, an adversary may be able to capture key information from one or more sensors. In some key management schemes, it is possible to obtain common keys between uncompromised sensors by compromising some sensors. In methods based on symmetric polynomials or the Blom scheme, as mentioned in Section 2, when an adversary compromises more than t sensors, access can be obtained to the common key of uncompromised sensors. Therefore, by proper selection of t, the probability of keys belonging to uncompromised sensors being compromised can be reduced to an acceptable level, or even 0. However, increasing t directly affects the sensor processing and memory overhead (as shown in Table 4), thus selecting a large value of t may not be practical. Hence, t should be chosen based on the tradeoff between memory/processing cost and the security level. Here, we assume that an adversary can uniformly capture sensors in the network. Let N_i be the number of sensors in each group. If an adversary can compromise the common key between two uncompromised sensors if at least t sensors from the group are compromised, then the probability of a direct common key between two sensors in the same group being compromised is

$$P_{Com} = \sum_{i=t+1}^{N_i} \frac{\binom{N_i}{i}\binom{N-N_i}{X-i}}{\binom{N}{X}} \tag{15}$$

where X is the number of compromised sensors in the entire network and N is the number of sensors in the network.

In [Du, 2004], each sensor selects m_R keys from a sub key pool. An adversary can get more information about the sub key pool by compromising more sensors. In this scheme, each cell has a sub key pool with S keys, and each key exists in two neighboring sub key pools. Thus, the probability of compromising the key between two sensors that have a common key is

$$1-\left(1-\frac{m_R}{S}\right)^{X_i} \tag{16}$$

where X_i is the number of compromised sensors in a group. Since there are two cells in a group, the probability of compromising X_G sensors in a group when X sensors are compromised in the entire network is

$$P(X_G = i) = \binom{X}{i}\left(\frac{2}{n}\right)^i\left(1-\frac{2}{n}\right)^{X-i} \tag{17}$$

where n is the number of cells in the network. Therefore the probability of compromising a common key based on the pre-distributed random key approach in [Du, 2004] is

$$P_{Com} = \sum_{i=1}^{2Nc}\left(1-\left(1-\frac{m_R}{S}\right)^i\right)\binom{X}{i}\left(\frac{2}{n}\right)^i\left(1-\frac{2}{n}\right)^{X-i} \tag{18}$$

In [Du et. al., 2006], each sensor selects τ key spaces from Sc available key spaces, and each key space is a Blom matrix. The value of τ is $m/t+1$, where m is the size of the available memory and t is the size of Blom matrix. Similar to [Du, 2004], each key space is shared between two neighboring cells and may be selected by $2Nc$ sensors. When t sensors having distinct rows of the same key space are compromised, the other common keys in the key space are also compromised. Therefore, assuming X_i sensors in a group are compromised, the probability of compromising a common key between uncompromised sensors is

$$\sum_{i=t+1}^{X_i} \binom{X_i}{i} \left(\frac{\tau}{Sc}\right)^i \left(1-\frac{\tau}{Sc}\right)^{X_i-i} \tag{19}$$

Then the probability of compromising the common key between two uncompromised sensors when X sensors are compromised in the entire network is

$$P_{Com} = \sum_{X_i=1}^{X} \sum_{i=t+1}^{X_i} \binom{X}{X_i} \left(\frac{2}{n}\right)^{X_i} \left(1-\frac{2}{n}\right)^{X-X_i} \binom{X_i}{i} \left(\frac{\tau}{Sc}\right)^i \left(1-\frac{\tau}{Sc}\right)^{X_i-i} \tag{20}$$

Since, our proposed key establishment protocol is not based on symmetric polynomials or the Blom scheme, capturing some sensors has no impact on compromising the common keys between uncompromised sensors. Hence, in the proposed protocol, the probability of compromising a common key between two uncompromised sensors is zero. On the other hand, if the available sensor memory is less than that required to store the Phase 1 and 2 common keys, fewer keys can be stored without any impact on security. However, the local connectivity will be reduced. Thus HKey can be employed when there are memory restrictions. In contrast, with the schemes based on symmetric polynomials or the Blom scheme, each sensor in a group can generate a common key with all other sensors in that group, so there is no flexibility to accommodate memory restrictions.

Using the above analysis, the probability of compromising a common key for all techniques except the proposed one was calculated and the results are shown in figure 5. Note that this probability is zero for all HKey models, so the security of our proposed models is not depicted. With Du-1 [Du, 2004], if the available memory increases, the number of selected random keys can be increased, but the security of this scheme based on (16) will be decreased. Conversely, techniques based on the Blom scheme or symmetric polynomials will have increased security according to (15).

4.5 Computational overhead

Key generation in HKey is based on symmetric cryptography. As mentioned in Section 3-1, a one way hash function is used to generate a common key. For a given pair of sensors, one has the common key in memory, while the other can generate the common key using the hash function. Thus in HKey, key generation is done in only one node, and the other does not require any processing. To estimate the required processing in a sensor, we assume that the key length is 128 bits and the Advanced Encryption Standard (AES) algorithm is used.

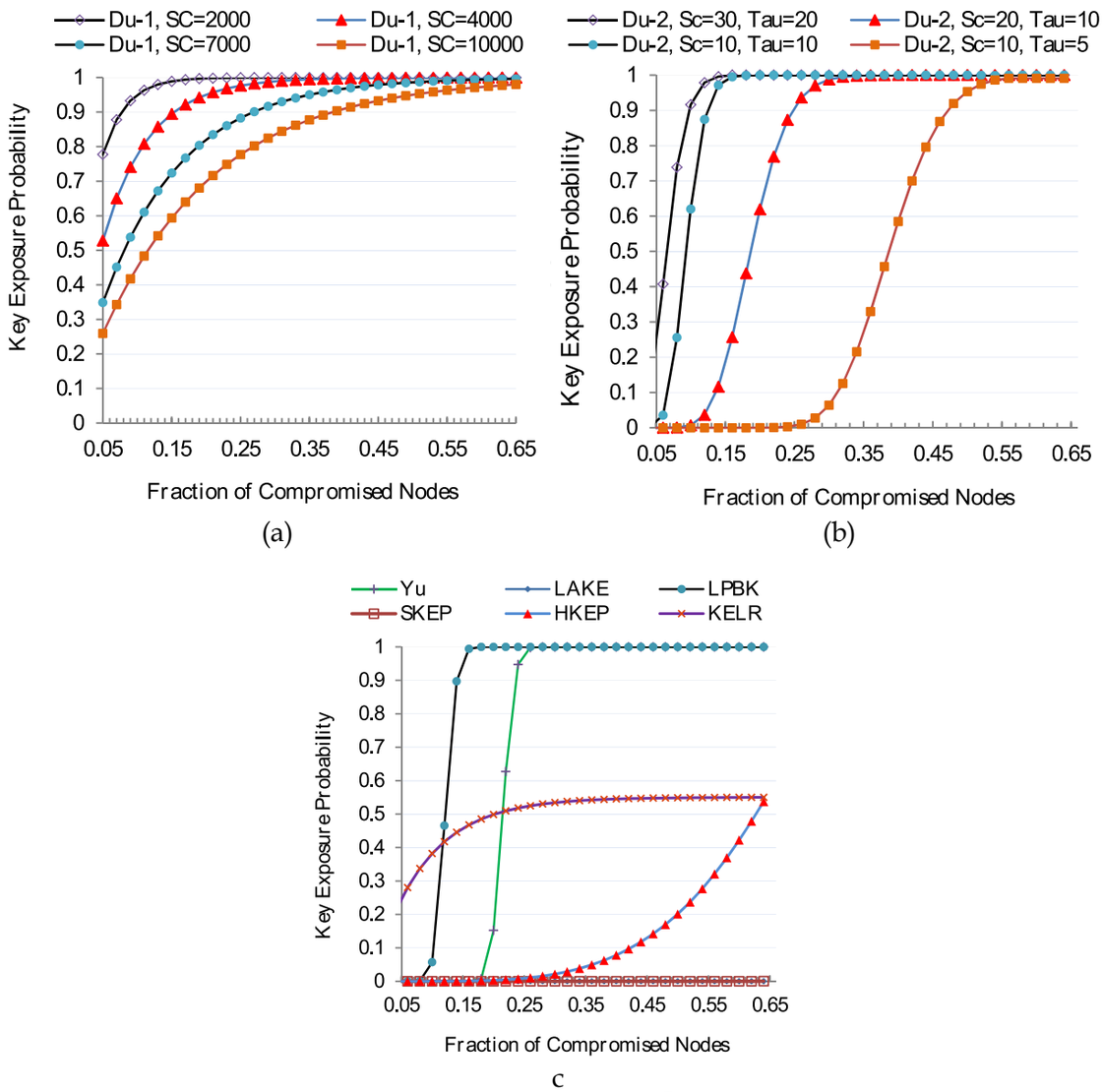


Fig. 5. Key resilience with different protocols for available memory 200 keys for each sensor.

In sensors, a low power microcontroller is typically used as the processing unit, so we assume that an 8-bit microcontroller from the 8051 family [www.atmel.com] is employed. As shown in [Nechavatal, 2000], AES has been implemented on this microcontroller using 3168 instruction cycles. Since key generation between two sensors is done in only one node, the required processing in both sensors to establish a common key is approximately 1600 instruction cycles on average, independent of network parameters. To compare with the other schemes, we have evaluated them assuming sufficient memory is available. LPBK and LAKE are based on symmetric polynomials, and if the polynomial has degree t, $2t$ modular multiplications and t summations are required. Since HKey uses a 128 bit key length, we assume these schemes use 128 bit modular arithmetic. To compute the modular multiplication of two 128 bit numbers, we use the interleaved modulo multiplication algorithm [Bunimov & Schimmler, 2004] which is implemented as follows

$$
\begin{aligned}
&\text{Inputs}: X, Y, M \text{ with } 0 \le X, Y < M \\
&\text{Output}: P = X \times Y \bmod M \\
&n: \text{number of bits in } X; \\
&x_i: i\text{th bit of } X; \\
&P := 0; \\
&\text{for}(i = n-1; i \ge 0; i--)\{ \\
&\quad P := 2 \times P; \\
&\quad I := x_i \times Y; \\
&\quad P := P + I; \\
&\quad \text{if}(P \ge M) \text{ then } P := P - M; \\
&\quad \text{if}(P \ge M) \text{ then } P := P - M; \\
&\} \\
\end{aligned}
$$

Our evaluation shows that the modular multiplication of two 128 bit numbers requires approximately 4096 instruction cycles. Hence, the number of instruction cycles is approximately 8300×t for LPBK and LAKE. The value of t in these schemes is not identical and is dependent on the group size. With the approach in [Yu & Guan, 2008], which uses the Blom scheme, multiplication of two matrices with dimensions (1,t+1) and (t+1,1) is required. For this operation, t+1 modular multiplications and t+1 summations are needed. Therefore, this scheme requires approximately 4096×t instruction cycles. The above results are summarized in Table 4 under the condition of perfect security.

Scheme	Required Threshold Value	Average Number of Instruction Cycles per Sensor
HKey (All models) with AES	-	1600
HKey (All models) with MD5	-	638
SKEP	2Nc-2	1.66×10^6
KELR	Nc-2	0.86×10^6
LAKE	2Nc-2	1.66×10^6
LPBK	25Nc-10	2.075×10^7
Yu and Guan	14Nc-2	5.734×10^6

Table 4. Computational Overhead for Different Schemes

As mentioned above, generating a common key among sensors with HKey requires a hash function. Although functions such as MD5 or SHA1 are commonly used for this purpose, we use the AES encryption algorithm. Our reason for using this algorithm is that an algorithm must be implemented in the sensors to encrypt transferred data. Due to the restricted sensor code space, it is better to avoid implementing a hash function for key generation. However, as shown in [Venugopalan et. al., 2003], the number of instruction cycles for MD5 is less than for AES. If sufficient memory space is available in the sensors, the MD5 algorithm can easily be used in our protocol. In order to compare results, we extend our analysis to the MD5 hash function as shown in Table 4.

Since in our protocol, key establishment between two sensors is not based on symmetric polynomials or the Blom scheme as in LAKE, LPBK, and the approach in [Yu & Guan, 2008], the computational overhead does not depend on the network or cell size. Thus, the proposed scheme is clearly the best in terms of processing cost.

4.6 Scalability

Wireless sensor networks typically consist of thousands of limited resource nodes which are deployed simultaneously in a network area. Since each sensor has only a small battery, it will be unusable after some period of time. One solution is to add new sensors to replace the dead ones. These new sensors must be able to establish secure connections with each other and with previously deployed sensors. Therefore an important issue for a key establishment protocol is extensibility so new sensors can establish a secure connection with other sensors while maintaining security.

All of the key establishment protocols examined in this chapter are scalable, i.e., new sensors can be added to the network such that they can communicate securely with existing nodes. For example, with the approach in [Du, 2004], if new sensors select their random keys from the current sub key pools, they should be able to establish a common key with the existing sensors. LAKE [Zhou & Fang, Apr. 2007], LPBK [Liu & Ning, 2003], and the approaches by Du [Du et. al., 2006] and Yu and Guan [Yu & Guan, 2008], are based on symmetric polynomials or the Blom scheme. If the secret information for new sensors is computed with the same symmetric polynomial or Blom matrix, these sensors can establish common keys with the old sensors. However, adding new sensors creates security concerns. As mentioned in Section 4-5, if an adversary captures a sufficient number of sensors, they can compromise common keys between uncompromised sensors. Further, once the number of captured sensors (old or new) reaches a threshold, an adversary can easily compromise the entire network. However, if the memory is allocated considering the addition of new sensors, the network will remain secure, but the sensor hardware must be able to support the needed memory.

In our proposed protocol, new sensors can join the network without affecting security. The KDC can generate common keys for the new sensors considering the deployed sensors as well as those which will be distributed in the future. In the initialization phase, the KDC generates common keys for the first group of sensors to be deployed and also the group of sensors to next be deployed. For subsequent distributions, the KDC generates three types of common keys. The first are common keys for sensors to be distributed in the current period. The second are common keys with previously deployed sensors, and the last are common keys with sensors to be distributed next. Consequently, the proposed protocol is simply extensible without any security limitations. This solution can also be used for other schemes. For example, in LPBK, LAKE, and the approach by Yu and Guan, the KDC can generate new secret information for each time interval. In this case, the KDC must generate two shares for each sensor, the first using the new secret and the second using the previous secret. This solution at most doubles the required memory.

4.7 Remarks

As discussed in the previous sections, there are several important parameters in WSN key establishment protocols such as local connectivity, memory usage, key resilience, and

computational overhead. Tradeoffs exist between these parameters. For example, to achieve better resilience as shown in figure 5, the memory usage and the computational overhead (as shown in Table 4) must be increased. In this chapter, we proposed an intrinsically secure key establishment protocol for WSNs. In this protocol, the common key between two sensors is generated based on the secret key of one sensor and the identity of the other. It differs from other schemes such as LAKE, LPBK, Du-2, and Yu and Guan in that key resilience does not depend on any other parameters. In addition, the computational overhead in the proposed scheme is independent of other parameters. In other words, in the other schemes based on symmetric polynomials or the Blom scheme, to achieve high key resilience the security threshold must be increased. However, as discussed in Section 4-5, the processing overhead is also increased. The only dependent parameters in our scheme are local connectivity and memory usage. To achieve high local connectivity, the memory usage must be increased. As shown in figure 4, the local connectivity with HKey-HP is close to one, but the memory usage is higher than with the other protocols and the proposed models. Several models were proposed for different applications. For example, when the available memory is very limited, HKey-LR is most suitable. Thus, the proposed protocol provides a number of tradeoffs between local connectivity and memory usage. Since it is sensitive to fewer parameters than the other schemes, the design and analysis of our protocol is more flexible.

5. Conclusions

In this chapter, a new key establishment protocol in which a common key between two nodes is computed using the secret key of one node and the identity of the other has been proposed. Thus, the key is stored in only one sensor and is computed by the other sensor when necessary. Due to the unavoidable need for an offline KDC in wireless sensor networks, this simple idea yields an efficient solution to the challenging key management problem. Four different models were introduced to implement the proposed protocol. The first model, HKey-LR, is memory efficient, but has a local connectivity which is less than with the other models. Conversely, the fourth model, HKey-HP, has high local connectivity, but the memory usage is more than with the others. In all cases, the local connectivity of the proposed models is comparable to that with other well known schemes, and our models are more memory efficient under the perfect security condition. Since the proposed protocol only uses a hash function to compute the common key for one of the two nodes, it has low computational overhead compared with the other schemes. The security analysis shows that the proposed protocol is more robust in low resource situations. In summary, the proposed protocol is a practical, secure, efficient, and scalable key management protocol for wireless sensor networks. As shown in Table 4, the computational overhead for the proposed protocol is considerably less than with the other protocols, which indicates that the energy consumption should be lower.

6. References

Pottie, G.J. & Kaiser, W.J. (2000). Wireless integrated network sensors, *Communication ACM*, vol. 43, no. 5, (May 2000), pp. (51-58)

Kahn, J.M., Katz, R.H. & Pister, K.S.J. (1999). Next century challenges: Mobile networking for smart dust, *Proceeding of ACM Internation Conference on Mobile Computing and Networking (MobiCom*), (1999), pp. (217-278)

Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). A survey on sensor networks, *IEEE Communication Magazin.*, vol. 40, no. 8, (2002), pp. (102-114)

Perrig, A., Szewczyk, R., Wen, V., Culler, D.E., & Tygar, J.D. (2002). SPINS: Security protocols for sensor networks, *Wireless Networks*, vol. 8, no. 5, pp. (521-534)

Kung, H.T., & Vlah, D. (2003). "Efficient location tracking using sensor networks," *Proceeding of IEEE Wireless Communication and Networking Conference (WCNC)*, (Mar. 2003), pp. (1954-1961)

Brooks, R., Ramanathan, P., & Sayeed, A. (2003). Distributed target classification and tracking in sensor networks, *Proceeding on IEEE*, vol. 91, no. 8, (Aug. 2003), pp. (1163-1171)

Karlof, C., & Wagner, D. (2003). Secure routing in wireless sensor networks: Attacks and countermeasures, *Proceeding of IEEE Workshop on Sensor Network Protocols and Applications* (*SNPA*), (May 2003), pp. (113-127)

Chan, H., Perrig, A., & Song, D. (2003). Random key predistribution schemes for sensor networks, *Proceeding of IEEE Symposium on Security and Privacy*, (May 2003), pp. (197-213)

Eschenauer, L., & Gligor, V. D. (2002). A key-management scheme for distributed sensor networks, *Proceeding of ACM Conference on Computer and Communication Security*, (Nov. 2002), pp. (41–47)

Du, W., Deng, J., Han, Y.S., Chen, S., & Varshney, P.K. (2004). A key management scheme for wireless sensor networks using deployment knowledge, *Proceeding of IEEE Conference on Computer Communication (INFOCOM)*, (Mar. 2004), pp. (586-597)

Borwein, P. & Erde´lyi, T. (1995). Polynomials and Polynomial Inequalities, New York: Springer-Verlag.

Zhou, Y., & Fang, Y. (2006). Scalable link-layer key agreement in sensor networks, *Proceeding of IEEE Military Communication Conference* (*MILCOM*), (Oct. 2006), pp. (1-6)

Zhou, Y. & Fang, Y. (2006). A scalable key agreement scheme for large scale networks, *Proceeding of IEEE International Conference on Networking, Sensing and Control* (*ICNSC*), (Apr. 2006), pp. (631-636)

Zhou, Y., & Fang, Y. (2007). A two-layer key establishment scheme for wireless sensor networks, *IEEE Transaction on Mobile Computing*, vol. 6, no. 9, (Sept. 2007), pp. (1009-1020)

Liu, D. & Ning, P. (2003), Location-based pairwise key establishments for relatively static sensor networks, *Proceeding of ACM Workshop on Security of Ad Hoc and Sensor Networks* (*SASN*), (Oct. 2003), pp. (72-82)

Blom, R. (1985). An optimal class of symmetric key generation systems, in *Advances in Cryptology: Proc. EUROCRYPT '84*, Springer Lecture Notes in Computer Science, vol. 209, pp. (335-338).

Zhou, Y., Zhang, Y., & Fang, Y. (2005). Key establishment in sensor networks based on triangle grid deployment model, *Proceeding of IEEE Military Communication Conference* (*MILCOM*), (Oct. 2005), pp. (1450-1455)

Du, W., Deng, J., Han, Y.S., Chen, S., & Varshney, P. (2006). A key predistribution scheme for sensor networks using deployment knowledge, *IEEE Trans. Dependable and Secure Computing*, vol. 3, no. 1, (Mar. 2006), pp. (62-77)

Yu, Z., & Guan, Y. (2008). A key management scheme using deployment knowledge for wireless sensor networks, *IEEE Trans. Parallel and Distributed Systems*, vol. 19, no. 10, (Oct. 2008), pp. (1411-1425)

Wei, R. & Wu, J. (2004). Product construction of key distribution schemes for sensor networks, *Proceeding of International Workshop on Selected Areas in Cryptography* (*SAC*), Springer Lecture Notes in Computer Science, vol. 3357, (Aug. 2004), pp. (280-293)

Hwang, J., & Kim, Y. (2004). Revisiting random key predistribution schemes for wireless sensor networks, *Proceeding of ACM Workshop on Security of Ad Hoc and Sensor Networks* (*SASN*), (Oct. 2004), pp. (43-52)

Ramkumar, M., & Memon, N. (2004). An efficient random key predistribution scheme, *Proceeding of IEEE Global Telecommunication Conference* (*GLOBECOM*), (Dec. 2004), pp. (2218-2223)

Liu, D., Ning, P., & Li, R. (2005). Establishing pairwise keys in distributed sensor networks, *ACM Trans. Information and System Security*, vol. 8, no. 1, (Feb. 2005), pp. (41-77)

Chan, H. & Perrig, A. (2005). Pike: Peer intermediaries for key establishment in sensor networks, *Proceeding of IEEE Conference on Computer Communication* (*INFOCOM*), (Mar. 2005), pp. (524-535)

Zhou, Y., Zhang, Y., & Fang, Y. (2005). Key establishment in sensor networks based on triangle grid deployment model, *Proceeding of IEEE Military Communication Conference* (*MILCOM*), (Oct. 2005), pp. (1450-1455)

Huang, D., Mehta, M., Medhi, D., & Harn, L. (2004). Location-aware key management scheme for wireless sensor networks, *Proceeding of ACM Workshop on Security of Ad Hoc and Sensor Networks* (*SASN*), (Oct. 2004), pp. (29-42)

Zhou, Y., Zhang, Y., & Fang, Y. (2005). LLK: A link-layer key establishment scheme in wireless sensor networks, *Proceeding of IEEE Wireless Communication and Networking Conference* (*WCNC*), (Mar. 2005), pp. (1921-1926)

Fanian, A., Berenjkoub, M., Saidi, H., & Gulliver, T.A. (2010). A hybrid key establishment protocol for large scale wireless sensor networks, *Proceeding of IEEE Wireless Communication and Networking Conf. (WCNC)*, (Apr. 2010), pp. (1-6)

Fanian, A., Berenjkoub, M., Saidi, H., & Gulliver, T.A. (2010). A new key establishment protocol for limited resource wireless sensor networks, *Proceeding of Conference on Communication Networks and Services Research (CNSR)*, (May 2010), pp. (138–145)

Su, Z., Jiang, Y., Ren, F., Lin, C., & Chu, X.-W. (2010). Distributed KDC-based random pair-wise key establishment in wireless sensor networks, *EURASIP J. Wireless Communication and Networking*, (July 2010), pp. (1-27)

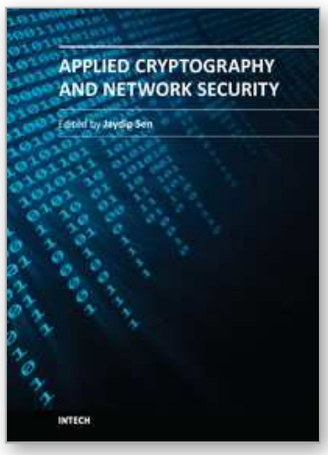
Nechavatal, J. (2000). Report on the Development of Advanced Encryption Standard (AES), *NIST*

Bunimov, V., & Schimmler, M. (2004). Area-time optimal modular multiplication, *in Embedded Cryptographic Hardware: Methodologies and Architectures*

http://www.atmel.com/dyn/products/tools_card.asp?tool_id=3422.

Venugopalan, R., Ganesan, P., Peddabachagari, P., Dean, A., Mueller, F., & Sichitiu, M. (2003). Encryption overhead in embedded systems and sensor network nodes: Modeling and analysis, *Proceeding of ACM International Conference on Compilers, Architecture and Synthesis for Embedded Systems* (*CASES*), (Nov. 2003), pp. (188-197)

Fanian, A., Berenjkoub, M., Saidi, H., & Gulliver, T.A. (2010). "An Efficient Symmetric Polynomial-based Key Establishment Protocol for Wireless Sensor Networks", *The ISC International Journal of Information Security*, pp. (89-105)



Applied Cryptography and Network Security

Edited by Dr. Jaydip Sen

ISBN 978-953-51-0218-2
Hard cover, 376 pages
Publisher InTech
Published online 14, March, 2012
Published in print edition March, 2012

Cryptography will continue to play important roles in developing of new security solutions which will be in great demand with the advent of high-speed next-generation communication systems and networks. This book discusses some of the critical security challenges faced by today's computing world and provides insights to possible mechanisms to defend against these attacks. The book contains sixteen chapters which deal with security and privacy issues in computing and communication networks, quantum cryptography and the evolutionary concepts of cryptography and their applications like chaos-based cryptography and DNA cryptography. It will be useful for researchers, engineers, graduate and doctoral students working in cryptography and security related areas. It will also be useful for faculty members of graduate schools and universities.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen