The ISC Int'l Journal of Information Security
ISeCure

July 2010, Volume 2, Number 2 (pp. 89–105)
http://www.isecure-journal.org

An Efficient Symmetric Polynomial-based Key Establishment Protocol for Wireless Sensor Networks

Ali Fanian [a,*], Mehdi Berenjkoub [a], Hossein Saidi [a], and T. Aaron Gulliver [b]

[a] *Department of Electrical and Computer Engineering, Isfahan University of Technology (IUT), Isfahan, Iran*
[b] *Department of Electrical and Computer Engineering, University of Victoria, Victoria, BC Canada*

ARTICLE INFO.

Article history:
Received: 7 February 2010
Revised: 13 July 2010
Accepted: 17 July 2010
Published Online: 24 July 2010

Keywords:
Key Management, Sensor Networks, Probabilistic Key Sharing, Symmetric Polynomial

ABSTRACT

An essential requirement for providing secure services in wireless sensor networks is the ability to establish pairwise keys among sensors. Due to resource constraints on the sensors, the key establishment scheme should not create significant overhead. To date, several key establishment schemes have been proposed. Some of these have appropriate connectivity and resistance against key exposure, but the resources needed in the sensors are substantial. Others are appropriate from the resource consumption perspective, but have weak performance. This paper proposes a key establishment protocol based on symmetric polynomials. To improve performance, the protocol uses a new model to distribute polynomial shares to the sensors. A key feature of the proposed protocol is the tradeoff between performance, security and resource consumption. Analysis shows that our solution has good performance compared to other approaches.

1 Introduction

Wireless networks can be categorized according to their application, one of which is sensor networks. Sensor networks often comprise a number of sensors with limited resources that are used to collect environmental information [1–3]. These networks have been considered for various purposes including security monitoring, target tracking and research activities in hazardous environments [4–10]. The sensors usually communicate with each other via a wireless channel. Since the sensors may be located in a hostile environment, such as in military applications, security is a critical issue. For instance, an adversary could simply access the wireless channel to obtain information, or by capturing sensors distribute false information to the network. Therefore, security considerations, such as authentication and confidentiality, must be employed to ensure node integrity and network functionality. Since authentication and confidentiality protocols require an agreed key between the entities, key management is a very important security issue in wireless sensors networks (WSNs) [11]. In the literature, key management protocols are based on either symmetric or asymmetric cryptographic functions. Due to resource limitations in the sensors, key management protocols based on public keys are not suitable. For example, the processing time for public key cryptographic functions can be as much as 10 seconds [11, 12]. Hence, symmetric algorithm based key management protocols are considered here for use in WSNs.

Another strategy that could be used in WSNs is

[*] Corresponding author.

Email addresses: fanian@ec.iut.ac.ir (A. Fanian), brnjkb@cc.iut.ac.ir (M. Berenjkoub), hsaidi@cc.iut.ac.ir (H. Saidi), agullive@ece.uvic.ca (T.A. Gulliver).

ISSN: 2008-2045 © 2010 ISC. All rights reserved.

an online key management center via an access point. Unfortunately, this approach causes rather high overhead to establish shared keys between sensors [13]. A more promising method is to pre-distribute key material among the sensors, which can result in low cost key establishment in WSNs. In this regard, various schemes have been proposed [7–10, 13–33], but they can result in security or efficiency problems. Some of these protocols attempt to increase the efficiency of the key management by using deployment knowledge in the pre-deployment phase to classify the sensors into groups for the distribution of secret information. To increase connectivity and reduce the overhead between sensors, nodes in neighboring cells can have correlated secret information in order to easily establish a common key.

To develop a high performance key management protocol, factors such as processing overhead, resource consumption, connectivity and resilience against compromised sensors should be considered. However, some of these goals are contradictory [13, 33]. For example increasing network connectivity increases sensor memory requirements. Hence, some approaches having proper connectivity use a great deal of memory and processing power. Others use less memory but have low connectivity.

In this paper, an efficient Symmetric polynomial based Key Establishment Protocol (SKEP) is proposed which uses deployment knowledge in the predistribution phase, and employs a new model for distributing secret information. The goal is to achieve proper connectivity and security while conserving resources. This protocol includes the following:

- Partitioning the network area into hexagonal cells and allocating a subset of sensors to each cell.
- Use of a new model to distribute secret information among the sensors in each group. A group includes all sensors belonging to a cell and some sensors from neighboring cells.
- Producing a common key directly between members of each group using symmetric polynomials.
- Producing a common key indirectly among sensors belonging to different groups using a suitable intermediate sensor.
- Efficient identification of suitable intermediate sensors to produce an indirect key based on predistributed information.
- Perfect security with acceptable memory usage and connectivity.

The rest of the paper is organized as follows. Section 2 reviews current key management protocols used in WSNs. Details of our key establishment protocol are given 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 Related Work

Due to resource constraints in sensors, key management is not an easy task. Significant research has been done in this regard in an attempt to efficiently establish common keys between sensors. Besagni [34] proposed a simple scheme for key management in WSNs. In this approach, one pre-loaded key is shared among all nodes. Afterwards, all sensors can encrypt or decrypt data between them. Due to its simplicity, this method is very efficient in terms of memory usage and processing overhead, but it presents a serious security problem. If even one of the sensors is captured by an adversary, the security of the entire network will be compromised. At the other extreme, independent pair-wise keys can be established between each pair of sensors. Hence for a network with N sensors, each sensor S_i, $1 \leq i \leq N$, must store $N-1$ keys from a total of $N(N-1)/2$ keys. Although, this method provides complete security for an uncompromised sensor against any compromised sensors, it requires a large amount of memory in each sensor, particularly in large scale WSNs.

In the literature, key management protocols based on deployment methods are divided into two groups. In the first group, uniform deployment is used to distribute sensors. In the second group, the deployment is non-uniform. We next review related work which considers these key management protocols for WSNs.

2.1 Uniform Deployment Model

Eschenauer *et al.* [15] proposed a random key predistribution scheme for WSNs. In this approach, some keys from a large key-pool are selected randomly before deployment and with their identifiers are stored in the sensors. After deployment, with some probability they can establish a key between each other. If there is no common key between two sensors, a common key can be generated through an intermediate node which has common keys with both. In this method, the size of the key-pool S is an important parameter for network connectivity and security. As S grows, connectivity decreases, but security increases. Due to the distribution of random keys, it is possible that a common key cannot be established between every pair of sensors. Chan [14] proposed a key management protocol which requires that a pair of sensors have at least Q common keys for key establishment. Due to the need for these keys, the probability of establishing a common key between sensors is reduced with this protocol. Thus, from the aspect of security, this scheme is better than the previous one, but the probability of producing a common key between two sensors is decreased. Zhou [35] proposed a key management protocol based on symmetric polynomials. In this scheme, a t-degree $(K + 1)$-variate symmetric polynomial is employed and K credentials are selected for each sensor. Then, the sensor polynomial share is computed using these credentials and the polynomial, and the share and credentials are stored in the sensor. If the credentials of two sensors differ in only one dimension, they can produce a common key. Liu [16, 24] proposed another key management protocol based on symmetric polynomials. In this method, a grid based model is used to distribute and calculate the sensor polynomial share. The N sensors in the network form an $m \times m$ matrix $(N = m^2)$. For each row and column of this matrix a t-degree bivariate symmetric polynomial is selected. A sensor in position (i, j) of the matrix computes polynomial shares $f_j^c(x, j)$ and $f_i^r(i, y)$ from symmetric polynomials $f_j^c(x, y)$ and $f_i^c(x, y)$. Therefore, if two sensors are in the same row or column, they can compute a common key directly. Otherwise, they must use other sensors to compute a common key. Note that with this method, sensor deployment does not follow the polynomial share distribution model [24], and it assumes that sensors are uniformly distributed in the network. Other schemes which use symmetric polynomials for key management are given in [16, 18, 24, 36].

2.2 Non-uniform Deployment Model

Deployment knowledge can be used to improve key management performance in WSNs. In [22, 33], it is shown that if this knowledge is used before deployment, the efficiency of the protocol can be increased. Du *et al.* [22] proposed a key management protocol based on [15], which uses deployment knowledge during key distribution. Deployment was modeled using a Gaussian probability distribution function (pdf). On the other hand, methods which do not use deployment knowledge typically use a uniform pdf for the node distribution. In [22] the network area is divided into square cells and each cell is associated with one group of the sensors. The key-pool is divided into sub-key-pools, equal to the number of cells, such that each sub-key-pool has some keys which are correlated with neighboring sub-key-pools. $G(i, j)$ is defined as the sub-key-pool of cell (i, j). Each sensor in a cell stores m keys which are randomly selected from the associated sub-key-pool. Using deployment knowledge allows the selection of random keys from a smaller sub-key-pool. This can improve the protocol performance especially in large scale networks. In addition, if some sensors are compromised, other sensors have greater security than with the approach in [15]. However, there are still some problems in producing common keys between sensors. Zhou *et al.* [32] proposed another key management protocol called LAKE which is based on a symmetric polynomial and deployment knowledge. Similarly in this scheme, the network is divided into square cells and each cell is associated with a group of sensors. The polynomial in this method is a t-degree tri-variate symmetric polynomial. Each sensor in this protocol has credentials (n_1, n_2), where n_1 represents the cell identity and n_2 represents the sensor identity. Based on these credentials, a polynomial share is calculated for each sensor and stored on it. After deployment, sensors that have one mismatch in their credentials can directly compute a common key. So, in this scheme all sensors belonging to a cell can establish common key to each other directly. However, only two specific sensors belonging to different cells can establish common key to each other directly. Lin *et al.* [18] proposed another key management protocol called LPBK in which the network area is also divided into square cells. Each cell has a specific symmetric polynomial which is used to compute a polynomial share for the sensors in the cell and four adjacent (vertical and horizontal) cells. In this scheme, a sensor must store five polynomial shares in its memory.

Yu *et al.* [13] proposed another key management protocol based on the approach by Blom [37]. Blom developed a key establishment protocol that allows each pair of nodes to establish a common key. With this scheme, if no more than t nodes are compromised, all common keys of the uncompromised nodes remain secure. A $(t + 1) \times N$ matrix G is defined as public information where N is the size of the network. During the key generation phase, the key management server creates a random $(t + 1) \times (t + 1)$ symmetric matrix, D, for each group which must be secret. In the Yu *et al.* protocol [13], the network area is divided into hexagonal cells and the associated G and D matrices are stored in the sensors based on deployment knowledge. The matrices are assigned to cells so that a confidential exclusive matrix, called A_i (equivalent to matrix D in the Blom method), is allocated to each cell. The sensors in each cell are given a row from the corresponding matrix A_i. Hence, the sensors belonging to a cell can produce a common key directly. To generate a common key between sensors in different cells, another confidential matrix B is constructed for use by sensors in neighboring cells. To allocate these matrices, two parameters b and w are defined where b indicates the number of B matrices allocated to a group and w indicates the maximum number of rows selected by each sensor from this allocation. The analysis in [13] shows that the best results are obtained with $w = 2$ and $b = 2$. In this case, the cells are divided into 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_i is allocated to each base cell together with its six neighbors. The Blom scheme is used on this matrix to produce the required information for the sensors. Since each normal cell is a neighbor with two base cells, their sensors receive information from two matrices B_i and B_j. Although the connectivity of this scheme is close to one, the memory consumption is extremely high.

3 The Proposed Scheme

In sensor networks, in contrast to ad-hoc networks that have mobile nodes, the assumption is that sensors are static after deployment [13, 18, 22, 32]. In other words, after distribution in the network, the sensors are fixed at their resident points. Therefore, deployment knowledge can be quite useful in producing common keys among sensors. Moreover, in most WSN applications, a secure peer-to-peer connection between distant sensors is unnecessary [13, 18, 22]. Therefore, the most important goal is establishing secure connections between adjacent sensors, so knowledge of probable neighbors can be useful in key predistribution. 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 randomness of sensor distribution, it is impossible to specify the exact location of each sensor. Knowing the probable neighbors is much more realistic. In this paper, we exploit deployment knowledge in producing key material in the pre-deployment phase. We next present a brief description of authentication based on symmetric polynomials. Then the proposed scheme, SKEP, is described in detail.

3.1 Authentication Using Symmetric Polynomials

A symmetric polynomial [35, 38, 39] is a t-degree $(K+1)$-variate polynomial defined as (1).

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

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 in the sense of (2) [35].

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

This means that for any permutation, ∂, we obtain the same polynomial. 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 in (3) [19].

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

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 given by $(c_1, c_2, \ldots c_{i-1}, u_i, c_{i+1}, \ldots, c_K)$ 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 [21] it was shown that in order to maintain perfect security in the sensor network, the polynomial degree must satisfy (4).

$$\begin{cases} 0 \le N_i - 2 \le t \\ \\ N_i \sqrt[K+1]{\frac{K(K+1)!}{2}} \le t \end{cases} \qquad i = 1, 2, \ldots, K \tag{4}$$

where N_i is the number of nodes in group i.

3.2 SKEP Model

In this section, we present the network and the key generation model in SKEP.

3.2.1 Network and Deployment Model

In key management protocols based on deployment knowledge, two distinct points are defined, namely the deployment point and the resident point [22]. The desired sensor location is the deployment point. This point is determined before the distribution of sensors in the network. The actual location of a sensor is the resident point. It is unlikely that the resident point will be the deployment point. If the sensors are distributed from a vehicle, such as an airplane or helicopter, the distance from the deployment point to the resident point will follow a certain probability distribution (pdf). With a uniform distribution, the resident point is equally likely to be anywhere within the deployment area, and in this case no information is available. However, if the distribution is non-uniform, some information is available. For example, with a Gaussian distribution, the distance between the resident and deployment points is less than 3σ, with probability 0.9987, where σ is the standard deviation [22]. In addition, if the distance between deployment points in two cells is more than 6σ, there is little chance that two sensors, each belonging to one of these cells, are adjacent [22, 33]. Since the probability of two sensors being adjacent is almost zero if their deployment points are far apart, there is no need to allocate correlated secret information to produce a common key between them. Therefore, this model can be used to determine which sensors are likely to be close and they can be allocated appropriate information to produce a common key. All previous schemes based on deployment knowledge have used a Gaussian distribution for the sensor deployment in the network [13, 18, 20, 22, 27, 28, 32, 33].

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 [18, 22, 32] the network is divided into square cells, but in [13] and SKEP hexagonal cells are used. As sensors are typically distributed circularly around the deployment point, using cells with a shape closer to a circle is a better approximation for protocol design. Figure 1 shows the division of the network into hexagonal cells. The distance between adjacent cell deployment points is L. Each cell in SKEP 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 can be formulated as (5).

$$f_k^{ij}(x, y \mid k \in C_{i,j}) = f(x - x_i, y - y_j) = \frac{1}{2\pi\sigma^2} e^{-\left[(x-x_i)^2+(y-y_j)^2\right]/2\sigma^2} \quad (5)$$

where $f(x, y) = \frac{1}{2\pi\sigma^2} e^{-\left[x^2+y^2\right]/2\sigma^2}$.

Assuming identical pdfs for every group of sensors, we can use $f_k(x, y \mid k \in C_{i,j})$ instead of $f_k^{ij}(x, y \mid k \in C_{i,j})$. As in [14–28, 32, 33], and with SKEP it is assumed that the routing protocol delivers the transmission data to the correct destinations.

3.2.2 The Key Generation Model

In SKEP, each cell has a distinct t-degree bi-variate symmetric polynomial given by $f(x_1, x_2) = \sum_{i_1=0}^{t} \sum_{i_2=0}^{t} a_{i_1,i_2} x_1^{i_1} x_2^{i_2}$. 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 in (6).

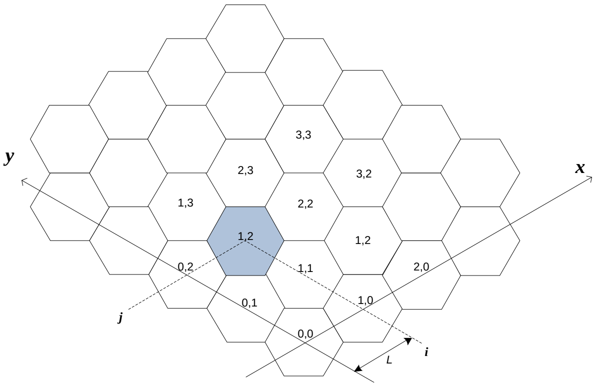


Figure 1. A two-dimensional sensor network with adjacent deployment point distance L.

$$f_k(x) = f_{i,j}(k, x) = \sum_{i_1=0}^{t} \sum_{i_2=0}^{t} a_{i_1,i_2} k^{i_1} x^{i_2} b_{i_2} = \sum_{i_1=0}^{t} a_{i_1,i_2} k^{i_1} \quad (6)$$

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 and computing (7).

$$\left.\begin{aligned} K_{ij} &= f_{k_i}(k_j) = \sum_{i2=0}^{t} b_{i_{i2}} k_j^{i2} \\ K_{ji} &= f_{k_j}(k_i) = \sum_{i2=0}^{t} b_{j_{i2}} k_i^{i2} \end{aligned}\right\}, K_{ij} = K_{ji} \quad (7)$$

3.2.3 The Threat Model

Since a large number of sensors are distributed in WSNs, the cost of producing a sensor should not be high. Therefore, sensors are typically not tamper proof. Hence, an adversary can obtain secret information by capturing one or more sensors. By compromising some sensors, it may be possible to derive a common key between uncompromised sensors. Moreover, an adversary can eavesdrop on transmissions over wireless channels. Hence, the key management protocol should be designed so that it is resistant to these threats.

3.3 Pre-distribution of Secret Information

Before distributing sensors in the network, the secret information is placed in the sensors. As mentioned in the previous section, a t-degree bi-variate symmetric polynomial is generated for each cell. A unique cre-

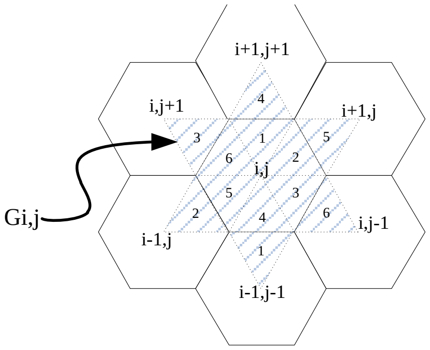


Figure 2. Dividing a cell into six virtual regions.

dential is also assigned to each sensor in order to generate a sensor polynomial share according to (6). Sensors which have a share of the same symmetric polynomial can directly generate a common key. 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 lot 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 2.

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. If the number of sensors in each cell is N_c, credentials between 1 and $N_c/6$ will be allocated to virtual region 1 of each cell. Sensors in the other regions are allocated credentials in a sequential manner. Therefore, the symmetric polynomial allocated to each cell is used to generate a polynomial share for the sensors belonging to the cell and one sixth of the sensors from the six neighboring cells. A set of sensors containing a polynomial share from a symmetric polynomial is called group. As shown in Figure 3, group $G(i, j)$ includes all sensors of cell (i, j) and the sensors belonging to virtual region 4 in cell $(i + 1, j + 1)$, virtual region 5 in cell $(i + 1, j)$, etc. Hence, each group includes $2N_c$ sensors and each sensor is a member of two groups. Therefore, each sensor has two polynomial shares, one derived from its own cell and another from a neighboring cell.

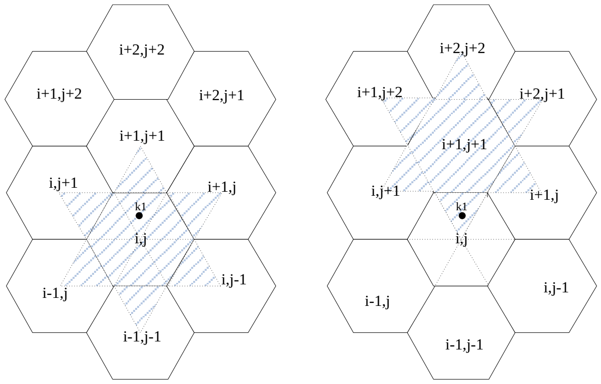


Figure 3. The groups to which sensor $k1$ belongs.

3.4 Indirect Key Calculation

Sensors in two distinct cells may be adjacent to each other. However, they may not be able to generate a common key directly, so a suitable agent should be used. A proper agent node is one that can directly generate a common key with both sensors. Since a number of sensors from two neighboring cells can directly generate a common key with all the sensors of both cells, one agent node suffices to generate a common key between any pair of sensors in these cells. Moreover, sensors in distinct cells a distance of no more than two cells apart can generate a common key with a maximum of two agents. Note that if the resident point distance between the agents is less than the wireless transmission range, they can communicate with each other directly; otherwise, a routing protocol is required to connect them as in [11–28, 32, 33]. The performance and security of indirect common key generation is greatly affected by the number of agent nodes. In SKEP, the number of usable agents is 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. From Section 2, these protocols are LPBK [18], LAKE [32], Du *et al.* [22] and Yu *et al.* [13].

4.1 Evaluation Metrics

We first review the metrics that will be used to compare the WSN key management schemes.

4.1.1 Local Connectivity

Local connectivity determines the probability of directly producing a common key between adjacent sensors. A high local connectivity indicates that adjacent sensors can generate a common key with little energy consumption. It is possible that even with high local connectivity, some sensors are isolated, and so cannot establish a common key with other sensors outside the isolated components. The mechanism used in the distribution of secret information among the sensors affects the local connectivity.

4.1.2 Resilience against Node Capture

Since sensors are likely not to be tamper-proof, an adversary may be able to capture secret information through a physical attack on some sensors. Obviously, if a sensor is captured, its common key(s) with other sensors can be derived. However, the critical question is the effect of a number of compromised sensors on overall network security. Resilience is defined as the fraction of the secure links that are compromised after a certain number of nodes are captured by an adversary. In other words, the effect of capturing X sensors on the exposure of keys between uncompromised sensors.

4.1.3 Memory Usage

We will calculate the required memory units for different schemes. Each memory unit can accommodate a cryptographic key in the random key pre-distribution schemes, a polynomial coefficient in the symmetric polynomial based schemes or an element of the Blom matrix in protocols based on the Blom scheme. Since sensors nodes have limited memory, low memory consumption for the keys is desirable.

4.1.4 Communication Overhead

Since the probability of direct key generation between two adjacent sensors is less than one, agent sensors should be used when two adjacent sensors do not establish a common key. To evaluate the communication overhead with a key establishment protocol, we need to determine the required number of hops between two adjacent sensors. When more sensors are used, the communication overhead is higher.

4.1.5 Computation Overhead

We will determine the number of computations required for each sensor to establish a direct common key. Due to resource constraints on the nodes, establishing this key should not involve a large number of computations.

4.2 Network Configuration

We consider a WSN with the following parameters similar to those in [22]:

- The number of sensors in the network is 10,000.
- The network area is 1000m × 1000m.
- Sensors have a two dimensional Gaussian distribution with standard deviation σ.
- The wireless transmission range is 40m.
- The number of sensor in each cell is 100.

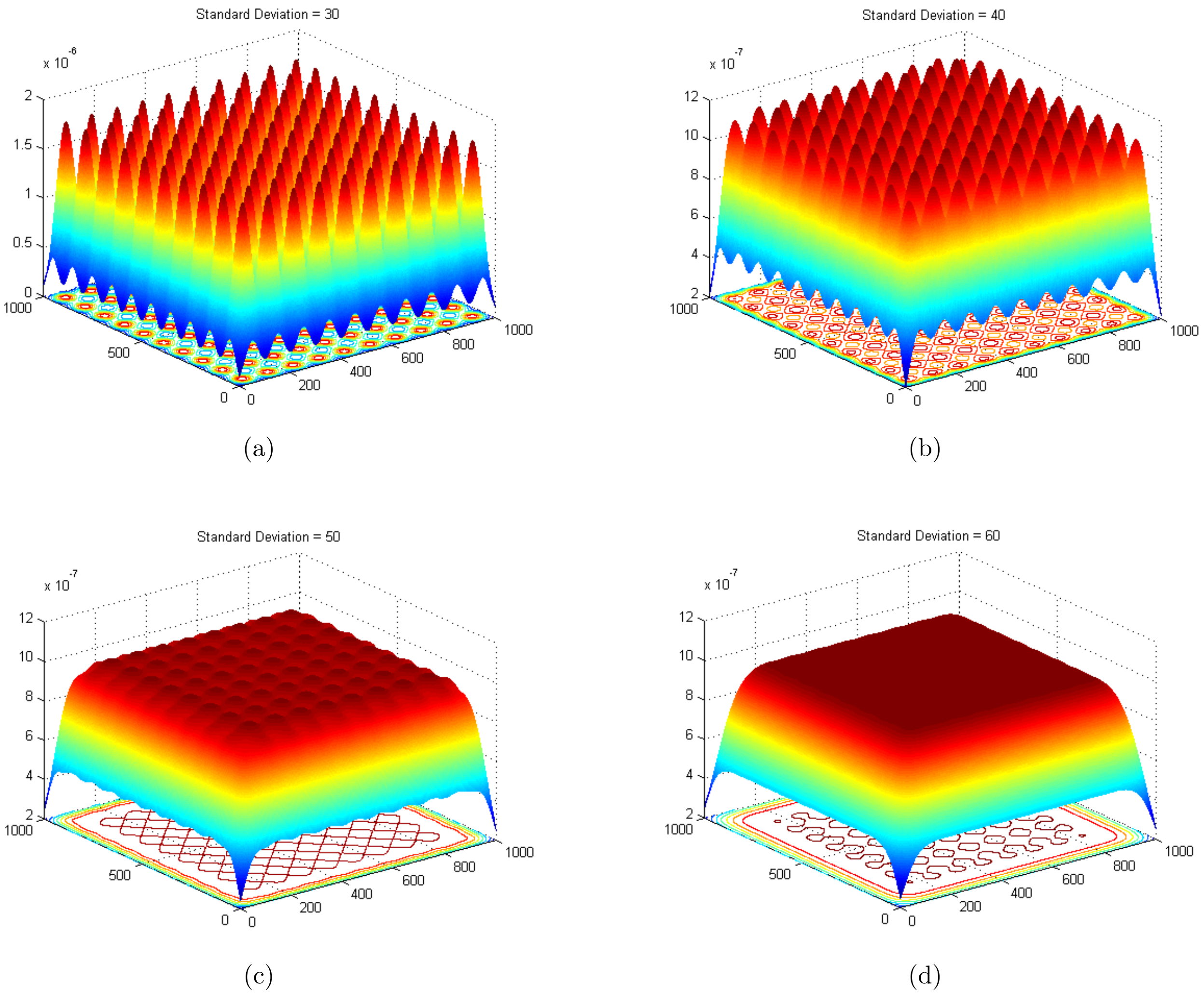
The value of σ depends on several factors, including the deployment height and environmental conditions such as wind. As stated in Section 3.2.1, the distance between the resident point and the deployment point of a sensor is less than 3σ with a probability of 0.9987. Hence, if the distance between deployment points of neighboring cells is more than 6σ, the number of sensors assigned to a cell but reside in neighboring cells will be very low. In this case, the connectivity of these cells will also be low. Note that although the distribution of sensors in each group is non-uniform, we still require that the sensors be distributed throughout the entire network with some uniformity in order to provide suitable coverage. In Figure 4, the deployment distribution of the sensors in the entire region is shown for the above parameters with four different values of σ. In Figure 4(d) with $\sigma = 60$m, the distribution of the sensors is nearly uniform except near the boundaries. Thus we choose σ equal to $0.6 \times L$, where L is the distance between neighboring deployment points.

Since the number of sensors in each cell affects the performance, we assume that square and hexagonal cells have identical areas. This allows for a fair comparison between models, since the number of sensors in each cell is approximately equal. The relationship between the deployment point distance of two neighboring cells in grid-based (L_g) and hexagonal-based (L_h) cells is given by (8).

$$L_g = L_h \cdot \sqrt{\frac{\sqrt{3}}{2}} \qquad (8)$$

4.3 Local Connectivity

As mentioned previously, local connectivity is defined as the probability of direct key generation between two adjacent sensors. In SKEP, due to the polynomial share generation mechanism, the probability of key generation between adjacent sensors in the same cell, P_{L_C}, is equal to one. In addition, some sensors can directly generate a common key with adjacent sensors in a neighboring cell. Since the resident point of a sensor may differ from its deployment point, we must compute the probability of generating a com-



(a) (b)

(c) (d)

Figure 4. Deployment distribution of the sensors in the network for different values of σ.

mon key between two adjacent sensors. In [22, 33] the probability of local connectivity is defined as (9).

$$P_{L\_C} = \Pr\left[B(n_i, n_j) \mid A(n_i, n_j)\right] \tag{9}$$

where $B(n_i, n_j)$ is the event of directly producing a common key between sensors n_i and n_j, and $A(n_i, n_j)$ is the event that two nodes are adjacent. As shown in [22, 33], P_{L_C} is given by (10).

$$\begin{aligned} &P_{L\_C} = p_1/p_2 \\ &p_1 = \Pr\left[B(n_i, n_j) \text{ and } A(n_i, n_j)\right] = \\ &\quad \sum_{j\in\psi}\sum_{i\in\psi} \Pr\left[B(u, v) \text{ and } A(u, v) \mid u \in C_i \text{ and } v \in C_j\right] \\ &\qquad \times \Pr(u \in C_i \text{ and } v \in C_j) \\ &p_2 = \sum_{j\in\psi}\sum_{i\in\psi} \Pr\left[A(u, v) \mid u \in\ C_i \text{ and } v \in C_j\right] \\ &\qquad \times \Pr\left[u \in\ C_i \text{ and } v \in C_j\right]. \end{aligned} \tag{10}$$

In this paper we use these relations to compare the local connectivity for all methods. The probability of common key generation between two sensors n_i and n_j depends on the pre-distribution model of the keys in the sensors. Before computing the probability of common key generation between two sensors, we define the logical distance between sensors as the distance between their deployment points. For sensors u and v with deployment points (u_x, u_y) and (v_x, v_y), respectively, the logical distance is $d_l(u, v) = \sqrt{(u_x - v_x)^2 + (u_y - v_y)^2}$.

We next determine the probability of common key generation for two sensors with SKEP by considering all possible cases.

- The probability of generating a common key for two sensors with logical distance zero ($d_l(u, v) = 0$), is given by (11).

$$\Pr\left[B(u, v) \mid u \text{ and } v \in C(i, j)\right] = 1 \tag{11}$$

- Without loss of generality, to calculate the probability of common key generation for two sensors in neighboring cells ($d_l(u,v) \leq \sqrt{2}$), considering Figure 3, we assume that sensor u from cell $C(i,j)$ wants to generate a common key with sensor v from cell $C(i+1,j+1)$. As shown in Figure 5, some sensors in these cells are in the same group. Let $C(i,j,k)$ denote the k^{th} virtual region of cell $C(i,j)$. For example, in Figure 5(b), sensors belonging to $C(i,j,1)$ and all sensors belonging to $C(i+1,j+1)$ are in the same group.

The probability of generating a common key is given by (12).

$$
\begin{aligned}
&\Pr[B(u,v) \mid u \in C(i,j) \,\text{and}\, v \in C(i+1,j+1)] \\
&= \sum_{k=1}^{6}\sum_{l=1}^{6} \Pr[B(u,v) \mid u \in C(i,j,k) \,\text{and} \\
&\qquad\qquad v \in C(i+1,j+1,l)] \\
&= \sum_{l=1}^{6} \Pr[B(u,v) \mid u \in C(i,j,1) \,\text{and} \\
&\qquad\qquad v \in C(i+1,j+1,l)] \\
&\quad + \sum_{l=2}^{6} \Pr[B(u,v) \;\mid u \in C(i,j,l) \,\text{and} \\
&\qquad\qquad v \in C(i+1,j+1,4)] \\
&\quad + \Pr[B(u,v) \mid u \in C(i,j,2) \,\text{and} \\
&\qquad\qquad v \;\in C(i+1,j+1,3)] \\
&\quad + \Pr[B(u,v) \mid u \in C(i,j,6) \,\text{and} \\
&\qquad\qquad v \in C(i+1,j+1,5)] \\
&= \frac{13}{36}.
\end{aligned} \tag{12}
$$

- If the logical distance between sensor u from $C(i_1,j_1)$ and sensor v from $C(i_2,j_2)$ is $\sqrt{5}$, then these cells have two common virtual regions. Again without loss of generality, we assume sensor u from $C(i,j)$ wants to generate a common key with sensor v from $C(i+1,j+2)$. The probability of generating this key considering Figure 6 is as computed in (13).

$$
\begin{aligned}
&\Pr[B(u,v) \mid u \in C(i,j), v \in C(i+1,j+2)] \\
&= \sum_{k=1}^{6}\sum_{l=1}^{6} \Pr[B(u,v) \mid u \in C(i,j,k) \,\text{and} \\
&\qquad\qquad v \in C(i+1,j+2,l)] \\
&= \Pr[u \in C(i,j,1) \,\text{and}\, v \in C(i+1,j+2,3)] \\
&\quad + \Pr[u \in C(i,j,6) \,\text{and}\, v \in C(i+1,j+2,4)] \\
&= \frac{2}{36}.
\end{aligned} \tag{13}
$$

- If the logical distance between two sensors is $\sqrt{8}$, then only one virtual region of these cells will be in the same group. Again without loss of generality, we assume sensor u from $C(i,j)$ wants to generate a common key with sensor v from $C(i+2,j+2)$. According to Figure 3, only those sensors belonging to virtual region 1 from cell $C(i,j)$ can generate a common key with the sensors belonging to virtual region 4 from cell $C(i+2,j+2)$. Therefore, the probability of key generation in this case is given by (14).

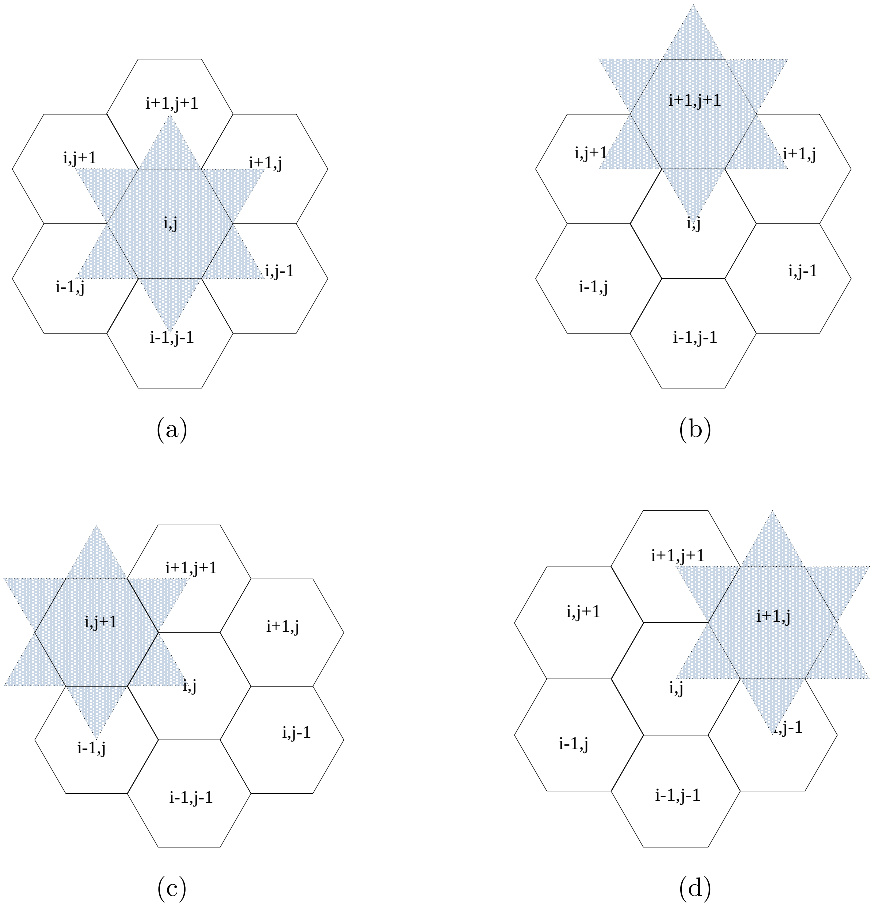
$$
\begin{aligned}
&\Pr[B(u,v) \mid u \in C(i,j) \,\text{and}\, v \in C(i+2,j+2)] \\
&= \sum_{k=1}^{6}\sum_{l=1}^{6} \Pr[B(u,v) \mid u \in C(i,j,k) \,\text{and} \\
&\qquad\qquad v \in C(i+2,j+2,l)] \\
&= \Pr[u \in C(i,j,1) \,\text{and}\, v \in C(i+2,j+2,4)] \\
&= \frac{1}{36}.
\end{aligned} \tag{14}
$$

- The probability of generating a common key in all other cases is zero.

Table 1. The Local Connectivity of Different Techniques

Technique	**Local Connectivity**
LAKE	0.2376
[22] $S = 4,000, m = 100$	0.3982
[22] $S = 4,000, m = 200$	0.6872
[22] $S = 2,000, m = 100$	0.5476
[22] $S = 2,000, m = 200$	0.8363
SKEP	0.5261
LPBK	0.9405
[13]	0.9508

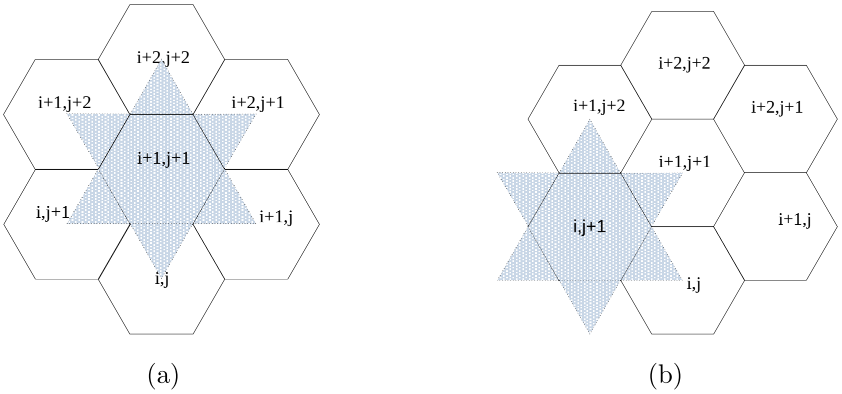
Local connectivity with the other methods has been computed by considering the probabilities of generating a common key between sensors and (10). The results are shown in Table 1. In scheme [22], each cell has a sub key-pool with S keys. Some of these keys are existence in the two neighboring cells sub key-pools. If we assume each cell shares 15% of its keys with horizontal and vertical neighboring cells and 10% with diagonal neighboring cells, then each key in a sub key-pool can select by $2N_c$ sensors. Each sensor selects m random keys. Table 1 shows that SKEP has adequate connectivity, but the local connectivity with SKEP is not as good as LPBK and the techniques in [13, 22]. However, as shown below, the memory usage, computational overhead and resilience against key exposure with SKEP is better than these schemes.



(a) (b)

(c) (d)

Figure 5. The common groups between cells $C(i, j)$ and $C(i + 1, j + 1)$.



(a) (b)

Figure 6. The common groups between cells $C(i, j)$ and $C(i + 1, j + 2)$.

4.4 Memory Usage

In SKEP, each sensor has two polynomial shares from two distinct t-degree bi-variate symmetric polynomials. Since the number of polynomial coefficients for each polynomial share is $t + 1$, memory usage is a linear function of the polynomial degree. Thus t should be kept as low as possible. The restriction on t comes from two factors. First, security between two uncompromised sensors should be guaranteed even if some sensors are compromised. Second, the security of the symmetric polynomials, which is the basis for all polynomial shares in the groups, must be guaranteed. With key management protocols based on multivariate symmetric polynomials, an attack can have different goals. An adversary may want the common key between two uncompromised sensors, or they may want to find the symmetric polynomials. The perfect security condition for both cases is given by (4). When the number of polynomial variables is greater than three, the required t to achieve perfect security in the second case is often greater than that for the first case. With SKEP, only a two variable symmetric polynomial ($K = 1$) is used for each group, therefore the required t value is the same for both cases. An adversary that wants to capture the common key between two uncompromised sensors must compromise a large number of sensors in the group. This means the adversary must reproduce the polynomial share of one of them. To obtain this polynomial share, an adversary must solve a set of linear equations with $(t+1)$ unknowns, i.e., the sensor polynomial share coefficients . Thus, t should be selected such that even if all sensors in a group except two are compromised, the adversary cannot obtain the common key between the remaining two sensors. If N_i is the number of sensors in a group, to achieve perfect security the polynomial degree t must satisfy (15).

$$N_i - 2 \leq t \tag{15}$$

As shown in [36], a $K+1$-variate t-degree symmetric polynomial has $\binom{t+K+1}{K+1}$ coefficients. Therefore, if an adversary wants to obtain the bi-variate t-degree symmetric polynomial, a set of equations with $\binom{t+2}{2}$ unknowns must be solved. If all sensors in a group are compromised, the adversary can produce the system of equations shown in (16).

$$\begin{aligned}
f_1(IDC_1) &= \sum_{i_1=0}^{t}\sum_{i_2=0}^{t} a_{i_1 i_2} IDC_1^{i_1} IDC_1^{i_2} = K_{1-1}\\
&\vdots\\
f_1(IDC_{N_i}) &= \sum_{i_1=0}^{t}\sum_{i_2=0}^{t} a_{i_1 i_2} IDC_1^{i_1} IDC_{N_i}^{i_2} = K_{1-N_i}\\
&\vdots\\
f_{N_i}(IDC_1) &= \sum_{i_1=0}^{t}\sum_{i_2=0}^{t} a_{i_1 i_2} IDC_1^{i_1} IDC_1^{i_2} = K_{N_i-1}\\
&\vdots\\
f_{N_i}(IDC_{N_i}) &= \sum_{i_1=0}^{t}\sum_{i_2=0}^{t} a_{i_1 i_2} IDC_1^{i_1} IDC_{N_i}^{i_2} = K_{N_i-N_i}
\end{aligned} \tag{16}$$

Due to the redundancy of some of the above equations, the number of independent equations is $\frac{N_i(N_i+1)}{2}$ [35].

Hence, to achieve perfect security with SKEP, inequality (17) must be satisfied.

$$\frac{N_i(N_i+1)}{2} \leq \binom{t+2}{2} \tag{17}$$

It can easily be verified that satisfying (15) results in (17), and vice versa.

Table 2. Memory Required with Different Schemes

Scheme	**Memory cost**	**Memory required to ensure secrecy of a direct key between sensors**
LAKE	$t+1$	$2N_c - 2$
LPBK	$5(t+1)$	$25N_c - 5$
[13]	$2(t+1)$	$14N_c + 1$
SKEP	$2t+3$	$4N_c - 1$

We next consider the memory required by key management protocols based on deployment knowledge. In LPBK, the sensors in each cell together with four neighboring cells have a polynomial share from a symmetric polynomial. Therefore, N_i is equal to $5N_c$, and each sensor contains five polynomial shares. Hence the memory required to achieve perfect security is $25N_c$.

In LAKE, a tri-variate t-degree symmetric polynomial is used, so that $N_i = 2N_c$. Thus, perfect security is achieved if $\frac{N_G N_c}{2}(N_G + N_c + 2) \leq \binom{t+3}{3}$ is satisfied [32], where N_G is the number of cells in the network.

With the approach in [13], which is based on the Blom scheme, in the best case, each base cell has six neighboring normal cells and each normal cell is the neighbor of two base cells. In this structure, the sensors which belong to a base cell or its six neighboring normal cells have a secret matrix B_i with dimension $(t+1) \times (t+1)$. Sensors belonging to these seven neighboring cells can produce a common key directly, so $N_i = 7N_c$. The number of sensors having different rows of B_i is $7N_c$, so t should be greater than $7N_c$. If less than t nodes are compromised, all communication links of the uncompromised nodes will remain secure. As mentioned previously, each sensor in the normal cells is a neighbor of two base cells, which receives two rows from two matrices. So, the memory required with the approach in [13] is at least $14N_c$.

In SKEP, each sensor has two polynomial shares and $2N_c$ shares are generated from each polynomial, so the memory required is $4N_c$. Table 2 presents the memory required for the different key management protocols. This shows that the memory usage of LPBK and the approach in [13] might be too high to be practical. Although, the LAKE memory usage is less than with SKEP, the local connectivity is much lower than SKEP.

4.5 Resilience against Key Exposure

By compromising some sensors, it may be possible to obtain the common key between uncompromised sensors. In this section, the effect of compromised sensors on the key generation methods is evaluated and compared.

4.5.1 The Probability of Direct Key Exposure

With methods based on a key pool [14–22, 27], an adversary can obtain more information about the key pool by compromising more sensors. In these methods if each sensor selects m keys from a key pool of size S, and X sensors are compromised, the probability of a secure connection between two uncompromised sensors is given by (18) [14, 21].

$$\left(1-\frac{m}{S}\right)^{X} \tag{18}$$

To reduce the probability of key exposure, S should be very large compared to m.

With methods based on symmetric polynomials or the Blom scheme, as mentioned above, when an adversary compromises more than t sensors, they can obtain the common key between two uncompromised sensors. Therefore, t should be chosen to minimize the probability of this occurring. However, since increasing t has a direct effect on the required processing and memory in the sensors (Table 2), t must be chosen based on the tradeoff between memory/processing cost and acceptable security. For the purposes of comparison we assume that an adversary can capture sensors in the network uniformly. As mentioned in Section 3.2, with SKEP, directly producing a common key is restricted to sensors in the same group. If X is the number of compromised sensors in the network and N_i is the number of sensors in a group, the probability of obtaining a direct common key between two sensors is given by (19).

$$P_{R-D}=\sum_{i=t+1}^{Ni}\frac{\binom{N_i}{i}\binom{N-N_i}{X-i}}{\binom{N}{X}} \tag{19}$$

N_i was computed for each protocol in Section 4.5. Based on (19), the probability of direct key exposure between two uncompromised sensors is given in Figure 7 for different protocols. It can be seen that SKEP and LAKE provide greater resistance against direct key exposure than the other schemes. In Figure 7, the memory size specifies the value of t for SKEP, LAKE, LPBK and the approach in [13]. Increasing the memory size will improve the security of these schemes based on (19). With the scheme in [22], the results are given for sub key-pool sizes of 4,000 and 2,000. With this method, the number of random keys in a sensor can be specified by the size of available memory. Increasing the memory size also increases the probability of direct key exposure based on (18).

4.5.2 The Probability of Indirect Key Exposure

To produce an indirect common key between two sensors, agent sensors are used. If agents are compromised, an adversary can access the common key produced by them. If X sensors are compromised by an adversary and the compromised sensors are not identified by other sensors, the probability of indirect key exposure, P_{R-I}, via an agent is simply given by (20).

$$P_{R-I}=\frac{X}{N} \tag{20}$$

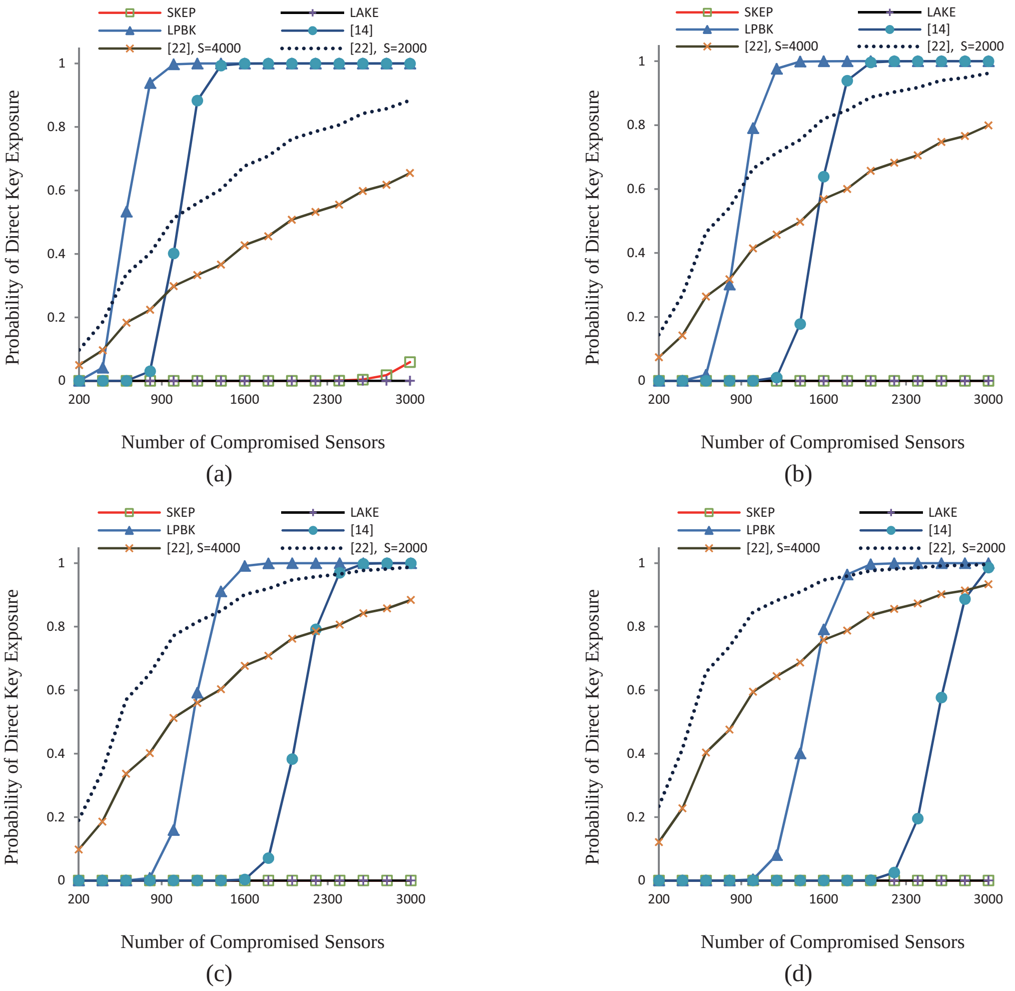
In this case, since the agents are selected randomly, the number of agents does not affect the security of indirect key generation. It should be noted that various methods to detect malicious nodes have been developed [40, 41], and sensors can use these methods to avoid using malicious nodes for transferring information. Thus we assume that uncompromised sensors can identify those that are compromised, and choose an agent among the sensors not detected as compromised for indirect key generation. In this case, if the number of sensors to be chosen as an agent increases, the possibility of compromising indirect keys will be reduced. In such a situation, it can be said that the common key will be revealed if all probable agent sensors are compromised. If N_a is the number of agent sensors, the probability of compromising an indirect key is shown in (21).

$$P_{R-I}=\prod_{i=1}^{Na}\frac{X-i}{N-i} \tag{21}$$

In LAKE scheme, there is only one agent option for a sensor trying to communicate with a sensor in another cell, hence $N_a = 1$. With SKEP, as shown in Figure 3, N_a for two sensors in neighboring cells is $N_c/3$. In LPBK and the approach of [13], the local connectivity is close to one, so an agent is not required to establish a common key in most case.

4.6 Communication Overhead

As mentioned previously, if the probability of generating a common key between two adjacent sensors is less than one, agent sensors should be used to produce a common key for these sensors. Identification of a suitable intermediate sensor could be costly. With methods based on random key distribution [22], finding an agent sensor results in additional overhead.



(a) (b) (c) (d)

Figure 7. Key resilience for different protocols for available memory (a) 100, (b) 150, (c) 200 and (d) 250.

The required number of sensor agents also affects the overhead. Since in SKEP the distribution of key material between sensors is structured, suitable agent sensors can easily be found.

To estimate the communication overhead to generate indirect common keys via an agent, the number of hops should be computed. For two adjacent sensors, if they can communicate directly there is only one hop. Thus, if the local connectivity of the key management protocol is near one, common keys can be generated with little overhead. Let the probability of generating a common key between two sensors with one hop be $P_h(1)$, the probability of generating a common key using one agent sensor be $P_h(2)$, and using two agents be $P_h(3)$. These probabilities were determined by simulation for different schemes, and the results are presented in Figure 8. This shows that the probability of generating a common key with SKEP is close to one with at most three hops. In Figure 8, [22]_a and [22]_b are the results of communication overhead with the approach in [22] and parameters $S = 4,000$, $m = 100$ and $S = 2,000$, $m = 200$, respectively.

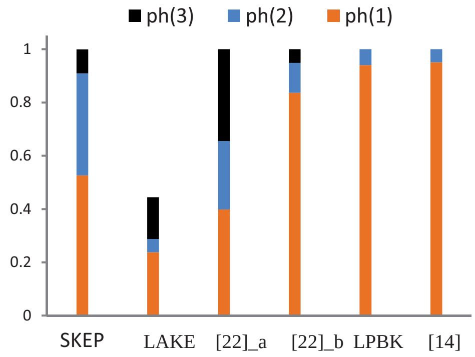


Figure 8. The probability of establishing a common key with at most three hops.

4.7 Computational Overhead

Key establishment with SKEP is based on t-degree bivariate symmetric polynomials, and each sensor has a share of two distinct polynomials. The common key

Table 3. Computational Overhead for Different Schemes

Scheme	Polynomial degree	Number of rows in the Blom matrix	Number of modular multiplications	Number of modular additions
LAKE	198	-	395	198
LPBK	498	-	995	498
[13]	-	698	1396	698
SKEP	198	-	395	198

between two sensors is calculated using (7), which requires $t-1$ modular multiplications for $x^2, x^3, \ldots, x^t$, t modular multiplications for $b_1x, b_2x^2, \ldots, b_tx^t$ and $t+1$ modular additions. All of these calculations are done in the finite field F_q. The key length depends on q, which is usually 64 or 128 bits. Therefore, keys based on symmetric polynomials require $2t-1$ modular multiplications and $t+1$ modular additions. With Blom based key establishment protocols, if a Vandermonde matrix is used [26], each sensor must calculate $2t$ modular multiplications and $t+1$ modular additions to generate a common key. The computation overhead is calculated for different schemes in perfect security condition and 100 sensors in each cell. The results are shown in Table 3. This shows that SKEP has significantly less overhead than LPBK and the technique in [13].

4.8 Discussion

A well-known sensor node is the MICA2 mote [42]. It has an 8-bit 7.3 MHz processor with 4 KB RAM and 128 KB of programmable ROM. TinyOS [43] is an operating system suitable for MICA2. If we suppose 2 KB of RAM is used for the operating system and application variables, the available space for storing keys is only 2 KB, as the keys must be stored in RAM. If the key management protocol uses a 64 bit key length, each sensor can store about 250 keys in memory. In this case, LPBK and the scheme in [13] are infeasible for use in WSNs or insecure. In other words, we can use LPBK with a low polynomial degree or the scheme in [13] with small matrices, but as shown in Figure 7 these protocols are then vulnerable to an adversary attack. SKEP, LAKE and the approach in [22] can fit within these constraints. However, as shown in Table 1 the local connectivity of SKEP is greater than with LAKE. The local connectivity in [22] can be more than SKEP, but this scheme as shown in Figure 7 is vulnerable to a key exposure attack. Finally, we note that a goal of this paper was to provide a tradeoff between performance, security and resource consumption in the key establishment for WSNs. In this context, SKEP is a unique secure protocol with reasonable resource consumption. Actually, up to now there is no similar solution in the literature.

Since each sensor uses a small battery, it will be unusable after a period of time. A reasonable solution is to add new sensors to replace the dead ones. These new sensors must be able to establish secure connections to each other and the previously deployed sensors. An important issue for a key establishment protocol is extensibility i.e. new sensors can establish a secure connection with other sensors while the security of the protocol is maintained. In this point of view, SKEP is scalable. The key distribution center assigns the new sensors to the suitable groups and generates polynomial shares based on the allocated symmetric polynomial to each group. So, the new sensors can communicate with all deployed sensors in those groups.

5 Conclusions

In this paper, we introduced a new key management protocol called SKEP which is based on symmetric polynomials. This protocol takes advantage of prior knowledge about sensor deployment, and uses a new model to generate and distribute the polynomial shares for each sensor. Using this model, sensors can simply determine whether or not they can generate a common key with other sensors. In addition, they can simply find an agent to generate an indirect common key when required. With high probability every pair of nodes with SKEP can produce a common key either directly or indirectly. The local connectivity of SKEP is less those of LPBK [13] and the approach in [22], but the memory usage, computation overhead and resistance against key exposure are better than these schemes. The results presented show that SKEP is a practical, secure and efficient (in terms of resources) key management protocol.

References

[1] Gregory J. Pottie and William J. Kaiser. Wireless Integrated Network Sensors. *Communications of the ACM*, 43(5):51–58, 2000.

[2] Joseph M. Kahn, Randy H. Katz, and Kristofer S. J. Pister. Next Century Challenges: Mobile Networking for "Smart Dust". In *Proceeding of the 5th ACM International Conference on Mobile Computing and Networking (MobiCom)*, pages 271–278, Seattle, Washington, USA, August 1999.

[3] Lan F. Akyildiz, Welljan Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. A Survey on Sensor Networks. *IEEE Communication Magazine*, 40(8):102–114, August 2002.

[4] Adrian Perrig, Robert Szewczyk, J. D. Tygar, Victor Wen, and David E. Culler. SPINS: Security Protocols for Sensor Networks. *Wireless Networks*, 8(5):521–534, September 2002.

[5] Anthony D. Wood and John A. Stankovic. Denial of Service in Sensor Networks. *IEEE Computer Magazine*, 35(10):54–62, October 2002.

[6] H.T. Kung and Dario Vlah. Efficient Location Tracking Using Sensor Networks. In *Proceeding of IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1954–1961, New Orleans, LA, USA, March 2003.

[7] Richard R. Brooks, Parameswaran Ramanathan, and Akbar M. Sayeed. Distributed Target Classification and Tracking in Sensor Networks. *Proceedings of the IEEE*, 91(8): 1163–1171, August 2003.

[8] Chris Karlof and David Wagner. Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures. In *Proceedings of the 1st IEEE Workshop on Sensor Network Protocols and Applications (SNPA)*, pages 113–127, May 2003.

[9] Ronald Watro, Derrick Kong, Sue-fen Cuti, Charles Gardiner, Charles Lynn, and Peter Kruus. TinyPK: Securing Sensor Networks with Public Key Technology. In *Proceedings of the 2nd ACM Workshop on Security of Ad hoc and Sensor Networks (SASN)*, pages 59–64, Washington DC, USA, October 2004. ACM.

[10] David J. Malan, Matt Welsh, and Michael D. Smith. A Public-Key Infrastructure for Key Distribution in TinyOS Based on Elliptic Curve Cryptography. In *Proceeding of the 1st IEEE International Conference on Sensor and Ad hoc Communication and Networks (SECON)*, pages 71–80, Santa Clara, California, USA, October 2004. IEEE.

[11] Nils Gura, Arun Patel, Arvinderpal Wander, Hans Eberle, and Sheueling C. Shantz. Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs. In *Proceeding of the International Workshop Cryptographic Hardware and Embedded System (CHES)*, pages 119–132, Boston, Cambridge, USA, August 2004. Springer.

[12] Haodong Wang and Qun Li. Efficient Implementation of Public Key Cryptosystems on Mote Sensors (Short Paper). In *Proceeding of the 8th International Conference on Information and Communication Security (ICICS)*, volume 4307 of *Lecture Notes in Computer Science (LNCS)*, pages 519–528. Springer, December 2006.

[13] Zhen Yu and Yong Guan. A Key Management Scheme Using Deployment Knowledge for Wireless Sensor Networks. *IEEE Transaction on Parallel and Distributed Systems*, 19(10):1411–1425, 2008.

[14] Haowen Chan, Adrian Perrig, and Dawn Song. Random Key Predistribution Schemes for Sensor Networks. In *Proceeding of the IEEE Symposium on Security and Privacy*, pages 197–213, Berkeley, California, USA, May 2003. IEEE.

[15] Laurent Eschenauer and Virgil D. Gligor. A Key-Management Scheme for Distributed Sensor Networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS)*, pages 41–47, Washington, DC, USA, November 2002. ACM.

[16] Donggang Liu and Peng Ning. Establishing Pairwise Keys in Distributed Sensor Networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS)*, pages 52–61, Washington D.C., USA, October 2003. ACM.

[17] Reizhong Wei and Jiang Wu. Product Construction of Key Distribution Schemes for Sensor Networks. In *Proceeding of the International Workshop on Selected Areas in Cryptography (SAC)*, volume 3897 of *Lecture Notes in Computer Science (LNCS)*, pages 280–293, Waterloo, Ontario, Canada, August 2005. Springer.

[18] Donggang Liu and Peng Ning. Location-Based Pairwise Key Establishments for Relatively Static Sensor Networks. In *Proceeding of the 1st ACM Workshop on Security of Ad hoc and Sensor Networks (SASN)*, pages 72–82, Fairfax, Virginia, USA, October 2003. ACM.

[19] Joengmin Hwang and Yongdae Kim. Revisiting Random Key Predistibution Schemes for Wireless Sensor Networks. In *Proceeding of the 2nd ACM Workshop on Security of Ad hoc and Sensor Networks (SASN)*, pages 43–52, Washington DC, USA, October 2004. ACM.

[20] Wenliang Du, Jing Deng, Yunghsiang S. Han, and Pramod K. Varshney. A Pairwise Key Predistribution Scheme for Wireless Sensor Net-

works. In *Proceedings of the 10th ACM conference on Computer and communications security (CCS)*, pages 42–51, Washington D.C., USA, October 2003. ACM.

[21] Roberto Di Pietro, Luigi V. Mancini, and Alessandro Mei. Random key-assignment for secure wireless sensor networks. In *Proceedings of the 1st ACM Workshop on Security of Ad hoc and Sensor Networks (CCS)*, pages 62–71, Fairfax, Virginia, October 2003. ACM.

[22] Wenliang Du, Jing Deng, Yunghsiang S. Han, Shigang Chen, and Pramod Varshney. A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge. In *Proceeding of the 23th IEEE Conference on Computer Communications (INFOCOM)*, pages 586–597, Hong Kong, March 2004. IEEE.

[23] Mahalingam Ramkumar and Nasir D. Memon. An Efficient Random Key Pre-Distribution Scheme. In *Proceeding of IEEE Global Telecommunication Conference (GLOBECOM)*, pages 2218–2223. IEEE, December 2004.

[24] Donggang Liu, Peng Ning, and Rongfang Li. Establishing Pairwise Keys in Distributed Sensor Networks. *ACM Transaction Information and System Security*, 8(1):41–77, February 2005.

[25] Haowen Chan and Adrian Perrig. Pike: Peer Intermediaries for Key Establishment in Sensor Networks. In *Proceeding of the 24th IEEE Computer and Communications Societies (INFOCOM)*, pages 524–535, Miami, Florida, USA, March 2005. IEEE.

[26] Yun Zhou, Yanchao Zhang, and Yuguang Fang. Key Establishment in Sensor Networks Based on Triangle Grid Deployment Model. In *Proceeding of the IEEE Military Communication Conference (MILCOM)*, pages 1450–1455, Atlantic City, New Jersey, USA, October 2005. IEEE.

[27] Dijiang Huang, Manish Mehta, Deep Medhi, and Lein Harn. Location-Aware Key Management Scheme for Wireless Sensor Networks. In *Proceedings of the 2nd ACM Workshop on Security of Ad hoc and Sensor Networks (SASN)*, pages 29–42, Washington DC, USA, October 2004.

[28] Yun Zhou, Yanchao Zhang, and Yuguang Fang. LLK: A Link-Layer Key Establishment Scheme in Wireless Sensor Networks. In *Proceeding of the IEEE Wireless Communication and Networking Conference (WCNC)*, pages 1921–1926, Mario Gerla, UCLA, USA, March 2005.

[29] Duncan S. Wong and Agnes Hui Chan. Efficient and Mutually Authenticated Key Exchange for Low Power Computing Devices. In *Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT)*, volume 2248 of *Lecture Notes in Computer Science (LNCS)*, pages 272–289, Gold Coast, Australia, December 2001. Springer.

[30] Donggang Liu and Peng Ning. Efficient Distribution of Key Chain Commitments for Broadcast Authentication in Distributed Sensor Networks. In *Proceeding of the 10th Annual Network and Distributed System Security Symposium (NDSS)*, pages 263–276, San Diego, California, USA, February 2003. The Internet Society.

[31] Chenyang Lu, Brian M. Blum, Tarek F. Abdelzaher, John A. Stankovic, and Tian He. Rap: A Real-time Communication Architecture for Large-scale Wireless Sensor Networks. In *Proceeding of the 8th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 55–66, San Jose, California, USA, September 2002. IEEE.

[32] Yun Zhou and Yuguang Fang. A Two-Layer Key Establishment Scheme for Wireless Sensor Networks. *IEEE Transaction Mobile Computing*, 6 (9):1009–1020, September 2007.

[33] Wenliang Du, Jing Deng, Yunghsiang S. Han, and Pramod K. Varshney. A key predistribution scheme for sensor networks using deployment knowledge. *IEEE Transaction Dependable and Secure Computing*, 3(1):62–77, March 2006.

[34] Stefano Basagni, Kris Herrin, Danilo Bruschi, and Emilia Rosti. Secure Pebblenets. In *Proceedings of the 2nd ACM international symposium on Mobile Ad hoc Networking & Computing (MobiHoc)*, pages 156–163, Long Beach, California, USA, October 2001. ACM.

[35] Yun Zhou and Yuguang Fang. A Scalable Key Agreement Scheme for Large Scale Networks. In *Proceeding of the IEEE International Conference on Networking, Sensing and Control (ICNSC)*, pages 631–636, Lauderdale, Florida, USA, April 2006. IEEE.

[36] Carlo Blundo, Alfredo De Santis, Amir Herzberg, Shay Kutten, Ugo Vaccaro, and Moti Yung. Perfectly-Secure Key Distribution for Dynamic Conferences. In *Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO '92)*, pages 471–486, Santa Barbara, California, USA, August 1993. Springer.

[37] Rolf Blom. An Optimal Class of Symmetric Key Generation Systems. In *Proceedings EUROCRYPT 84*, pages 335–338, Paris, France, April 1985. Springer.

[38] Peter Borwein and Tàmas Erdèlyi. *Polynomials and Polynomial Inequalities*, volume 161 of *Graduate Texts in Mathematics*. Springer, 1995.

[39] Yun Zhou and Yuguang Fang. Scalable Link-

Layer Key Agreement in Sensor Networks. In *Proceeding of the IEEE Military Communication Conference (MILCOM)*, pages 1–6, Washington D.C., USA, October 2006. IEEE.

[40] Vijay Bhuse and Ajay Gupta. Anomaly Intrusion Detection in Wireless Sensor Networks. *Journal of High Speed Networks*, 15(1):33–51, January 2006.

[41] Yun Wang, Xiaodong Wang, Bin Xie, Demin Wang, and Dharma P. Agrawal. Intrusion Detection in Homogeneous and Heterogeneous Wireless Sensor Networks. *IEEE Transactions on Mobile Computing*, 7(6):698–711, June 2008.

[42] Crossbow Technology. http://www.xbow.com/.

[43] TinyOS: An Open-Source OS for the Networked Sensor Regime. http://www.tinyos.net/.

Ali Fanian received the BS and MS degrees in computer engineering (Hardware and Computer Systems Architecture) in 1999 and 2001, respectively from Isfahan University of Technology (IUT), Isfahan, Iran. He is currently Ph.D student in IUT. Different aspects of computer architecture and network security are Mr. Fanian research interests; specially, adhoc networks, wireless network security and hardware design.

Mehdi Berenjkoub received the Ph.D. degree from Department of Electrical and Computer Engineering, Isfahan University of Technology in 2000. The title of his dissertation is two-party key distribution protocols in cryptography. He started his work in the same department as an assistant professor from that time. Graduate courses presented by him include Fundamentals of Cryptography, Cryptographic Protocols, Network Security, and Speech Processing. He has supervised more than a dozen M.Sc. students and a Ph.D. candidate in related areas. He also was one of the founder members for Iranian Society of Cryptology in 2001. He has continued his cooperation with the society as an active member. He along with his colleagues recently established a research group on Security in Networks and Systems in IUT. He also is responsible for a newly established academic CSIRT in IUT. His current interested research topics are wireless network security and authentication protocols.

Hossein Saidi Received B.S and M.S. degrees in Electrical Engineering in 1986 and 1989 respectively, both from Isfahan University of Technology (IUT), Isfahan, Iran. He also received D.Sc. in Electrical Engineering from Washington University in St. Louis, USA in 2004. Since 1995 he has been with the Department of Electrical and Computer Engineering at IUT, where he is currently an Associate Professor of Electrical and Computer Engineering. His research interests include high-speed switches and routers, communication networks, QoS in networks, queueing system, security and information theory.

T. Aaron Gulliver received the Ph.D. degree in Electrical Engineering from the University of Victoria, Victoria, BC, Canada in 1989. From 1989 to 1991 he was employed as a Defence Scientist at Defence Research Establishment Ottawa, Ottawa, ON, Canada. He has held academic positions at Carleton University, Ottawa, and the University of Canterbury, Christchurch, New Zealand. He joined the University of Victoria in 1999 and is a Professor in the Department of Electrical and Computer Engineering. In 2002, he became a Fellow of the Engineering Institute of Canada. He is currently an Editor for IEEE Transactions on Wireless Communications. From 2000-2003, he was Secretary and a member of the Board of Governors of the IEEE Information Theory Society. His research interests include information theory and communication theory, algebraic coding theory, MIMO systems and ultra-wideband communications.