Hindawi Publishing Corporation
EURASIP Journal on Wireless Communications and Networking
Volume 2009, Article ID 427492, 12 pages
doi:10.1155/2009/427492

Research Article

In Situ Key Establishment in Large-Scale Sensor Networks

Yingchang Xiang,[1] Fang Liu,[2] Xiuzhen Cheng,[3] Dechang Chen,[4] and David H. C. Du[5]

[1] *Department of Basic Courses, Rizhao Polytechnic College, Rizhao, Shandong 276826, China*
[2] *Department of Computer Science, University of Texas - Pan American, Edinburg, Texas 78539, USA*
[3] *Department of Computer Science, The George Washington University, Washington, DC, 20052, USA*
[4] *Department of Preventive Medicine and Biometrics, Uniformed Services University of the Health Sciences, Bethesda, MD 20817, USA*
[5] *Department of Computer Science and Engineering, University of Minnesota, Minneapolis, Minnesota, USA*

Correspondence should be addressed to Xiuzhen Cheng, cheng@gwu.edu

Received 1 January 2009; Accepted 11 April 2009

Recommended by Yang Xiao

Due to its efficiency, symmetric key cryptography is very attractive in sensor networks. A number of key predistribution schemes have been proposed, but the scalability is often constrained by the unavailability of topology information before deployment and the limited storage budget within sensors. To overcome this problem, three in-situ key establishment schemes, SBK, LKE, and iPAK, have been proposed. These schemes require no preloaded keying information but let sensors compute pairwise keys after deployment. In this paper, we propose an in-situ key establishment framework of which iPAK, SBK, and LKE represent different instantiations. We further compare the performance of these schemes in terms of scalability, connectivity, storage, and resilience. Our simulation results indicate that all the three schemes scale well to large sensor networks. We also notice that SBK outperforms LKE and LKE outperforms iPAK with respect to topology adaptability. Finally, observing that iPAK, SBK, and LKE all rely on the key space models that involve computationally intensive modular operations, we propose an improvement that rely on random keys that can be easily computed from a secure pseudorandom function. This new approach requires no computation overhead at regular worker sensors, therefore has a high potential to conserve the network resource.

1. Introduction

Secure communication is a critical requirement for many sensor network applications. Nevertheless, the constrained capabilities of smart sensors (battery supply, CPU, memory, etc.) and the harsh deployment environment of a sensor network (infrastructureless, wireless, ad hoc, etc.) make this problem very challenging. A secure sensor network requires a "sound" key establishment scheme that should be easily realized by individual sensors, should be localized to scale well to large sensor networks, should require small amount of space for keying information storage, and should be resilient against node capture attacks.

Symmetric key cryptography is attractive and applicable in sensor networks because it is computationally efficient. As reported by Carman et. al [1], a middle-ranged processor such as the Motorola MC68328 "DragonBall" consumes 42 mJ (840 mJ) for RSA encryption (digital signature) and 0.104 mJ for AES when the key size for both cases is 1024 bits. Therefore establishing a shared key for pairwise communication becomes a central problem for sensor network security research. Ever since the pioneer work on key predistribution by Eschenauer and Gligor [2] in the year 2002, a variety of key establishment schemes have been reported, as illustrated in Figure 1.

Key predistribution is motivated by the observation that no topology information is available before deployment. The two extreme cases are the *single master key scheme*, which preloads a master key to all sensors, and the *all pairwise keys scheme*, which preloads a unique key for each pair of sensors. The first one is weak in resilience while the second one has a high storage overhead. Other than these two extreme cases there exist a number of probabilistic-based key predistribution schemes [2–11], which attract

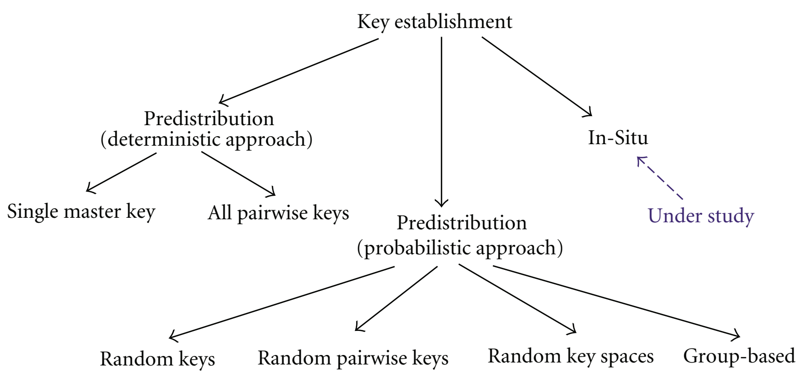


Figure 1: Existing Key Establishment Schemes - A Taxonomy.

most of the research interests in securing sensor networks. The probabilistic-based schemes require each sensor to preload keying information such that two neighboring sensors compute a shared key after exchanging part of the stored information after deployment. Generally speaking, the larger the amount of keying information stored within each sensor, the better the connectivity of the key-sharing graph, the higher the computation and communication overheads. A major drawback of the schemes in this category is the storage space wastage since a large amount of keying information may never be utilized during the lifetime of a sensor. Consequently, the scalability of key predistribution is poor, since the amount of required security information to be preloaded increases with the network size. Furthermore, many of the probabilistic-based approaches bear poor resilience as the compromise of any sensors could release the pairwise key used to protect the communications between two uncompromised sensors. In summary, probabilistic-based key predistribution could not achieve good performance in terms of scalability, storage overhead, key-sharing probability, and resilience simultaneously.

Recently, three in-situ key establishment schemes, iPAK [12], SBK [13] and LKE [14], have been proposed for the purpose of overcoming all the problems faced by key predistribution. Schemes in this category require no keying information to be predistributed, while sensors compute shared keys with their neighbors after deployment. The basic idea is to utilize a small number of service sensors as sacrifices for disseminating keying information to worker sensors in the vicinity. Worker sensors are in charge of normal network operations such as sensing and reporting. Two worker sensors can derive a common key once they obtain keying information from the same service sensor. In this paper, we first propose the in-situ key establishment framework, of which iPAK, SBK, and LKE represent different instantiations. Then we report our comparison study on the performance of these three schemes in terms of *scalability, connectivity, storage overhead* and *resilience*. Our results indicate that all the three in-situ schemes scale well to large sensor networks as they require only local information. Furthermore, we also notice that SBK outperforms LKE and LKE outperforms iPAK with respect to topology adaptability. Finally, observing that iPAK, SBK, and LKE all rely on the key space models that involve intensive computation overhead, we propose an improvement that rely on random keys that could be easily generated by a secure pseudorandom function.

This paper is organized as follows. Major key predistribution schemes are summarized in Section 2. Preliminaries, models, and assumptions are sketched in Section 3. The in-situ key establishment framework is introduced in Section 4, and the three instantiations are outlined in Section 5. Performance evaluation and comparison study are reported in Section 6. Finally, we summarize our work and discuss the future research in Section 7.

2. Related Work: Key Predistribution

In this section, major related works on key predistribution are summarized and compared. We refer the readers to [10, 15] for a more comprehensive literature survey.

The basic *random keys scheme* is proposed by Eschenauer and Gligor in [2], in which a large key pool $\mathcal{K}$ is computed offline and each sensor picks m keys randomly from $\mathcal{K}$ without replacement before deployment. Two sensors can establish a shared key as long as they have at least one key in common. To enhance the security of the basic scheme in against small-scale attacks, Chan et al. [3] propose the q-composite keys scheme in which $q > 1$ number of common keys are required for two nodes to establish a shared key. This scheme performs worse in resilience when the number of compromised sensors is large.

In these two schemes [2, 3], increasing the number of compromised sensors increases the percentage of compromised links shared by uncompromised sensors. To overcome this problem, in the same work Chan et al. [3] propose to boost up a unique key for each link through multi-path enhancement. For the same purpose, Zhu et al. [16] propose to utilize multiple logic paths. To improve the efficiency of key discovery in [2, 3], which is realized by exchanging the identifiers of the stored keys, or by a challenge-response procedure, Zhu et al. [16] propose an approach based on the pseudo-random key generator seeded by the node id. Each sensor computes the key identifiers and preloads the corresponding keys based on its unique id. Two sensors can determine whether they have a common key based on their ids only. Note that this procedure does not improve the security of the key discovery procedure since an attacker can still Figure out the key identifiers as long as the algorithm is available. Further, a smart attacker can easily beat the pseudo-random key generator to compromise the network faster [17]. Actually for smart attackers, challenge-response is an effective way for key discovery but it is too computationally intensive. Di Pietro et al. [17] propose a pseudo-random key predeployment scheme that supports a key discovery procedure that is as efficient as the pseudo-random key generator [16] and as secure as challenge-response.

To improve the resilience of the random keys scheme in against node capture attacks, *random pairwise keys schemes* have been proposed [3, 4], in which a key is shared by two sensors only. These schemes have good resilience against node capture attacks since the compromise of a sensor only affects the links incident to that sensor. The difference between [3] and [4] is that sensors in [3] are paired based on ids while in [4] are on virtual grid locations. Similar to the random keys schemes, random pairwise keys schemes do not scale well to large sensor networks. Neither do they have good key-sharing probability due to the conflict between the high keying storage redundancy requirement and the memory constraint.

To improve the scalability of the random keys schemes, two *random key spaces schemes* [5, 7] have been proposed independently at ACM CCS 2003. These two works are similar in nature, except that they apply different key space models, which will be summarized in Subsection 3.1. Each sensor preloads several keying shares, with each belonging to one key space. Two sensors can establish a shared key if they have keying information from the same key space. References [7] also proposes to assign one key space to each row or each column of a virtual grid. A sensor residing at a grid point receives keying information from exactly two key spaces. This realization involves more number of key spaces. Note that these random key spaces schemes also improve resilience and key-sharing probability because more key spaces are available, and because two sensors compute a unique key within one key space for their shared links.

Compared to the works mentioned above, *group-based schemes* [6, 8, 9, 11] have the best performance in scalability, key-sharing probability, storage, and resilience due to the relatively less randomness involved in these key predistribution schemes. Du et al. scheme [6] is the first to apply the group concept, in which sensors are grouped before deployment and each group is dropped at one deployment point. Correspondingly, a large key pool $\mathcal{K}$ is divided into subkey spaces, with each associated with one group of sensors. Subkey spaces overlap if the corresponding deployment points are adjacent. Such a scheme ensures that close-by sensors have a higher chance to establish a pairwise key directly. But the strong assumption on the deployment knowledge (static deployment point) renders it impractical for many applications. Also relying on deployment knowledge, the scheme proposed by Yu and Guan in [9] significantly reduces the number of potential groups from which neighbors of each node may come, yet still achieves almost perfect key-sharing probability with low storage overhead. Two similar works [8, 11] have been proposed at ACM Wise 2005 independently. In [8], sensors are equally partitioned based on ids into disjoint *deployment groups* and disjoint *cross groups*. Each sensor resides in exactly one deployment group and one cross group. Sensors within the same group can establish shared keys based on any of the key establishment schemes mentioned above [2–4, 18, 19]. In [11], the deployment groups and cross groups are defined differently and each sensor may reside in more than two groups. Note that these two schemes inherit many nice features of [6], but release the strong topology assumption adopted by [6]. A major drawback of these schemes is the high communication overhead when path keys are sought to establish shared keys between neighboring sensors.

Even with these efforts, the shared key establishment problem still has not been completely solved yet. As claimed by [20, 21], the performance is still constrained by the conflict between the desired probability to construct shared keys for communicating parties and the resilience against node capture attacks, under a given capacity for keying information storage in each sensor. Researchers have been actively working toward this to minimize the randomness [22, 23] in the key predistribution schemes. Due to space limitations, we could not cover all of them thoroughly. Interested readers are referred to a recent survey [15] and the references therein.

Architectures consisting of base stations for key management have been considered in [24] and [25], which rely on base stations to establish and update different types of keys. In [1], Carman et al. apply various key management schemes on different hardware platforms and evaluate their performance in terms of energy consumption, for and so forth. Authentication in sensor networks has been considered in [24–26], and so forth.

The three in-situ key establishment schemes [12–14] are radically different from all those mentioned above. They rely on service sensors to facilitate pairwise key establishment between worker sensors after deployment. The service sensors could be predetermined [12], or self-elected based on some probability [13] or location information [14]. Each service sensor carries or computes a key space and distributes a unique piece of keying information to each associated worker sensor in its neighborhood via a computationally asymmetric secure channel. Two worker sensors are able to compute a pairwise key if they obtain keying information from the same key space. As verified by our simulation study in Section 6, in-situ schemes can simultaneously achieve good performance in terms of scalability, storage overhead, key-sharing probability, and resilience.

3. Preliminaries, Models, and Assumptions

3.1. Key Space Models. The two key space models for establishing pairwise keys, one is polynomial-based [19] and the other is matrix-based [18], have been tailored for sensor networks at [7] and [5], respectively. These two models are similar in nature.

The polynomial-based key space utilizes a bivariate λ-degree polynomial $f(x,y) = f(y,x) = \sum_{i,j=0}^{\lambda} a_{ij}x^j y^j$ over a finite field F_q, where q is a large prime number (q must be large enough to accommodate a cryptographic key) . By pluging in the id of a sensor, we obtain the keying information (called a *polynomial share*) allocated to that sensor. For example, sensor i receives $f(i,y)$ as its keying information. Therefore two sensors knowing each other's id can compute a shared key from their keying information as $f(x,y) = f(y,x)$. For the generation of a polynomial-based key space $f(x,y)$, we refer the readers to [19].

The matrix-based key space utilizes a $(\lambda+1)\times(\lambda+1)$ public matrix (Note that G can contain more than $(\lambda+1)$ columns.) G and a $(\lambda+1)\times(\lambda+1)$ private matrix D over a finite field $GF(q)$, where q is a prime that is large enough to accommodate a cryptographic key. We require D to be symmetric. Let $A = (D\cdot G)^T$. Since D is symmetric, $A\cdot G$ is symmetric too. If we let $K = A\cdot G$, we have $k_{ij} = k_{ji}$, where k_{ij} is the element at the ith row and the jth column of K, $i,j = 1,2,\dots,\lambda+1$. Therefore if a sensor knows a row of A, say row i, and a column of G, say column j, then the sensor can compute k_{ij}. Based on this observation, we can allocate to sensor i a keying share containing the ith row of A and the ith column of G, such that two sensors i and j can compute their shared key k_{ij} by exchanging the columns of G in their keying information. We call (D,G) a matrix-based key space, whose generation has been well-documented by [18] and further by [5].

Both key spaces are λ-collusion-resistent [18, 19]. In other words, as long as no more than λ sensors receiving keying information from the same key space release their stored keying shares to an attacker, the key space remains perfectly secure. Note that it is interesting to observe that the storage space required by a keying share from either key space at a sensor can be very close ($(\lambda+1)\cdot\log q$ for the polynomial-based key space [19] and $(\lambda+2)\cdot\log q$ for the matrix-based key space [18]) for the same λ, as long as the public matrix G is carefully designed. For example, [5] proposes to employ a Vandermonde matrix over $GF(q)$ for G, such that a keying share contains one row of A and the seed element of the corresponding column in G. However, the column of G in a keying share must be restored when needed, resulting in $(\lambda-1)$ modular multiplications.

Note that iPAK, SBK and LKE work with both key space models. In these schemes, service sensors need to generate or to be preloaded with a key space and then distribute to each worker sensor a keying share. Two worker sensors can establish a shared key as long as they have keying information from the same key space. Note that for a polynomial-based key space, two sensors need to exchange their ids while for a matrix-based key space, they need to exchange the columns (or the seeds of the corresponding columns) of G in their keying shares.

3.2. Rabin's Public Cryptosystem. Rabin's scheme [27] is a public cryptosystem, which is adopted by the in-situ key establishment schemes to set up a computationally asymmetric secure channel through which keying information can be delivered from a service sensor to a worker sensor.

3.2.1. Key Generation. Choose two large distinct primes p and q such that $p \equiv q \equiv 3 \bmod 4$. (p,q) is the private key while $n = p\cdot q$ is the public key.

3.2.2. Encryption. For the encryption, only the public key n is needed. Let P_l be the plain text that is represented as an integer in Z_n. Then the cipher text $c = P_l^2 \bmod n$.

3.2.3. Decryption. Since $p \equiv q \equiv 3 \bmod 4$, we have

$$\begin{aligned} m_p &= c^{p+1/4} \bmod p, \\ m_q &= c^{q+1/4} \bmod q. \end{aligned} \tag{1}$$

By applying the extended Euclidean algorithm, y_p and y_q can be computed such that $y_p\cdot p + y_q\cdot q = 1$.

From the Chinese remainder theorem, four square roots $+r, -r, +s, -s$ can be obtained:

$$\begin{aligned} r &= \left(y_p\cdot p\cdot m_q + y_q\cdot q\cdot m_p\right) \bmod n \\ -r &= n - r \\ s &= \left(y_p\cdot p\cdot m_q - y_q\cdot q\cdot m_p\right) \bmod n \\ -s &= n - s. \end{aligned} \tag{2}$$

Note that Rabin's encryption [27] requires only one squaring, which is several hundreds of times faster than RSA. However, its decryption time is comparable to RSA. The security of Rabin's scheme is based on the factorization of large numbers; thus, it is comparable to that of RSA too. Since Rabin's decryption produces three false results in addition to the correct plain text, a prespecified redundancy, a bit string R, is appended to the plain text before encryption for ambiguity resolution.

3.3. Network Model and Security Assumptions. We consider a large-scale sensor network with nodes dropped over the deployment region through vehicles such as aircrafts. Therefore no topology information is available before deployment. Sensors are classified as either *worker nodes* or *service nodes*. Worker sensors are in charge of sensing and reporting data, and thus are expected to operate for a long time. Service sensors take care of key space generation and keying information dissemination to assist in bootstrapping pairwise keys among worker sensors. They may die early due to depleted energy resulted from high workload in the bootstrapping procedure. In this sense, they are sacrifices. Nevertheless, we assume service sensors are able to survive the bootstrapping procedure.

In our consideration, sensors are not tamper resistant. The compromise or capture of a sensor releases all its security information to the attacker. Nevertheless, a sensor deployed in a hostile environment must be designed to survive at least a short interval longer than the key bootstrapping procedure when captured by an adversary; otherwise, the whole network can be easily taken over by the opponent [28].

We further assume that a cryptographically secure key k_0 is preloaded to all sensors such that all communications in the key establishment procedure can be protected by a popular symmetric cryptosystem such as AES or Triple-DES. Therefore k_0 is adopted mainly to protect against false sensor injection attacks, and any node deployed by an adversary can be excluded from key establishment. Note that k_0 is strong enough such that it is almost impossible for an adversary to recover it before the key establishment procedure is complete, and the release of k_0 after the key establishment procedure does not negatively affect the security of the in-situ key establishment schemes since all sensitive information involved in the key establishment procedure is protected via a different technique. All sensors should remove their stored keying information (k_0 and/or the key space/pool) at the end of the key bootstrapping procedure.

4. The In-Situ Key Establishment Framework

Compared to the predistribution schemes, in-situ key establishment schemes distribute keying information for shared key computation after deployment.

All the in-situ key establishment contains three phases: *service node determination and key space construction*, *service node association and keying information acquisition*, and *shared key derivation*. iPAK, SBK, and LKE mainly differ from each other in the first phase, which will be detailed afterwards. Now we sketch the framework for in-situ key establishment in sensor networks.

4.1. Service Node Determination and Key Space Construction. In the first phase, service nodes are either preselected (in iPAK[12]), or self-elected with some probability (in SBK[13]) or based on sensors' physical location (in LKE[14]). A λ-collusion resistent key space (either polynomial-based [19] or matrix-based [18]) is allocated to [12] or generated by [13, 14] each service sensor.

Before deployment, each sensor randomly picks up two primes p and q from a pool of large primes without replacement. The prime pool is precomputed by high-performance facilities, which is out of the scope of this paper. Primes p and q will be used to form the private key such that Rabin's public cryptosystem [27] can be applied to establish a secure channel for disseminating keying information in the second phase.

4.2. Service Node Association and Keying Information Acquisition. Once a service sensor finishes the key space construction, it broadcasts a beacon message notifying others of its existence after a random delay. A worker node receiving the beacon will acquire keying information from the service sensor through a secure channel established based on Rabin's cryptosystem between the two nodes. As illustrated in Figure 2, the service node association and keying information acquisition is composed of the following three steps.

4.2.1. Key Space Advertisement. A service node S announces its existence through beacon broadcasting when its key space is ready. The beacon message should include: (i) a unique key space id, (ii) the public key n, where $n = p \times q$ and (p, q) is the corresponding private key preloaded before deployment, and (iii) the coverage area of the service sensor, which is determined in LKE by a grid size L, and specified in iPAK and SBK by a forwarding bound H, the maximum distance in hop count over which the existence of a key space can be announced. The message will be forwarded to all sensors within S's coverage area.

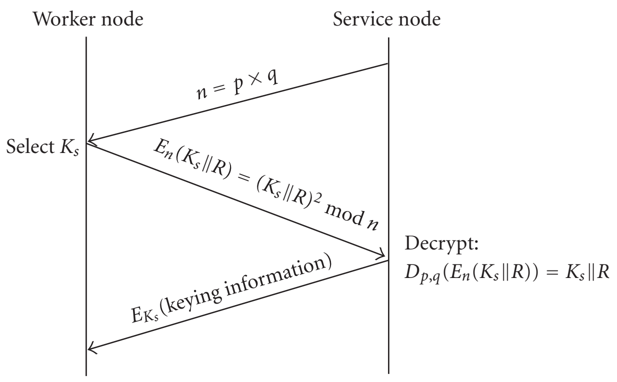

Worker node
Service node
$n = p \times q$
Select K_s
$E_n(K_s \| R) = (K_s \| R)^2 \bmod n$
Decrypt:
$D_{p,q}(E_n(K_s \| R)) = K_s \| R$
E_{K_s}(keying information)

Figure 2: Service sensor association. A worker node associates itself to a service sensor to obtain the keying information through a secure channel established based on Rabin's public cryptosystem.

4.2.2. Secure Channel Establishment. Any worker node requesting the keying information from a service node needs to establish a secure channel to the associated service node. Recall that we leverage Rabin's public key cryptosystem [27] for this purpose. After obtaining the public key n, a worker sensor picks up a random key K_s and computes $E_n(K_s \| R) = (K_s \| R)^2 \bmod n$, where R is a predefined bit pattern for ambiguity resolution in Rabin's decryption. $E_n(K_s \| R)$, along with the location information, is transmitted to the corresponding service sensor. After Rabin's decryption, the service sensor obtains $D_{p,q}(E_n(K_s \| R)) = K_s \| R$, where K_s will be utilized to protect the keying share transmission from the service sensor to the work sensor.

Note that in this protocol, each worker sensor executes one Rabin's encryption for each service sensor from which an existence announcement is received, whereas the computationally intensive decryption of Rabin's system is performed only at service sensors. This can conserve the energy of worker sensors to extend the operation time of the network. In this aspect, service nodes work as sacrifices to extend the network lifetime.

4.2.3. Keying Information Acquisition. After a shared key K_s is established between a worker node and a service node, the service sensor allocates to the node a keying share from its key space. The keying information, encrypted with K_s based on any popular symmetric encryption algorithm (AES, DES, etc.), is transmitted to the requesting worker node securely. Any two worker nodes receiving keying information from the same service node can derive a shared key for secure data exchange in the future.

After disseminating the keying information to all worker sensors in the coverage area, *the service sensor should erase all stored key space information for security enhancement*.

4.3. Shared Key Derivation. Two neighboring nodes sharing at least one key space (having obtained keying information from at least one common service sensor) can establish a shared key accordingly. The actual computation procedure is dependent on the underlying key space model. We refer the readers for the details to Subsection 3.1. Note that this procedure involves the exchange of either node ids, if polynomial-based key space model is utilized [19], or columns (seeds) of the public matrix, if matrix-based key space model is utilized [18]. To further improve security, nonces can be introduced to protect against replay attacks.

5. Service Sensor Election for the In-Situ Key Establishment Schemes

All the in-situ key establishment schemes rely on service sensors for keying information dissemination after deployment. As stated before, the major difference among the three schemes lies in how service sensors are selected, which is sketched in this section.

5.1. iPAK. Service node election in iPAK is trivial. They are predetermined by the network owner. iPAK considers a heterogeneous sensor network consisting of two different types of sensors, namely, worker nodes and service nodes. Since the number of service sensors is expected to be much smaller than that of the worker sensors, service sensors are assumed to have much higher capability (computational power, energy, and so forth) in order to complete the key bootstrapping procedure before they run out of energy.

Each service node is preloaded with all the necessary information, including one key space and two large primes. Worker sensors and service sensors are deployed together, with the proportion predetermined by ρ, where $\rho = \lambda \cdot N_s/N_w$, and $N_s(N_w)$ is the number of service nodes (worker nodes). The serving area of a service node is predetermined by the forwarding bound T_0, the utmost hop distance from the service node that the keying information can be disseminated.

5.2. SBK. Compared to iPAK, SBK does not differentiate the roles of worker sensors and service sensors before deployment. Instead, sensors determine their roles after deployment by probing the local topology of the network. In SBK, service sensors are elected based on a probability P_s, which is initialized as $P_s = 1/\lambda$. Once elected, a service sensor constructs a λ-collusion-resistent key space and serves worker sensors within its coverage area that is determined by the forwarding bound T_0. T_0 is defined according to the expected network density, which should satisfy $N_{T_0} \leq \lambda$ where N_{T_0} is the average number of neighbors within T_0 hops in the network.

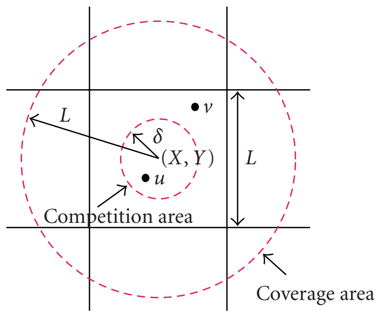


Figure 3: LKE: A virtual grid, with each grid size of L, is computed based on location information. Sensor u is selected from the competition area and will take care of key establishment for nodes residing in the coverage area.

In SBK, the service node election is conducted for several rounds. At the beginning of each round, a non-service sensor that does not have any service node within $T_0 - 1$ hops decides to become a service node with the probability P_s. If a sensor succeeds in the self-election, it sets up a key space, announces its status to T_0-hop neighbors after a random delay, and then enters the next phase for keying information dissemination. Otherwise, it listens to key space advertisements. Upon receiving any new key space announcements from a service node that is at most $T_0 - 1$ hops away, the sensor becomes a worker node, erases its primes, and enters the next phase for service sensor association and keying information acquisition. Note that the reception of a service node announcement also suppresses sensors who have self-elected as service nodes but have not broadcasted their decisions to broadcast their status. If no service node within $T_0 - 1$ hops is detected in the current round, the sensor participates in the next round.

To speed up the key bootstrapping procedure, an enhanced scheme, iSBK, is also proposed in [13], which achieves high connectivity in less time by generating more service sensors. In iSBK, the service sensor election probability P_s is initialized as $P_s = 1/N_{T_0 - 1}$, and is doubled in each new round until it reaches 1.

5.3. LKE. Similar to SBK, LKE [14] is a self-configuring key establishment scheme. However, the role differentiation is based on location information instead of a probability P_s. Right after deployment, each sensor positions itself and computes a virtual grid with the grid size of L. As illustrated in Figure 3, each grid contains a *competition area*, the disk region within a radius of δ from the grid center. At most one service sensor will be selected from the competition area.

An eligible sensor first waits a random delay. If it receives no competition message from others, it announces its decision to be a service sensor. Otherwise, the sensor self-configures as a worker sensor. Note that all the eligible sensors are within δ-distance from the grid center with $\delta = R/\sqrt{5}$, where R is the nominal transmission range. The setting of δ ensures that all eligible sensors within a grid can communicate with each other directly.

Each service sensor will establish a λ-collusion-resistent key space and serve those worker sensors residing in the *coverage area*, the disk region centered at the grid center with a radius of L. The setting of L satisfies $\pi L^2 = \lambda \times A/N$, where A is the deployment area, and N is the total number of nodes to be deployed. Thus, each service node is expected to serve λ nodes in a uniformly distributed network. To improve performance, iLKE is proposed, which adaptively generates service nodes based on a hierarchical virtual grid structure such that each service sensor will serve at most λ worker sensors.

6. Performance Evaluation

In this section, we study the performance of iPAK, SBK, and LKE via simulation. Note that we focus on worker sensors only, as service sensors are sacrifices that will not participate in the long-lasting networking operations. We will evaluate the in-situ key establishment schemes in terms of the following metrics via simulation: *Scalability, Resilience, Connectivity, Storage*, and *Cost*. These performance metrics will be defined at which our corresponding simulation results are reported.

6.1. Simulation Settings. We consider a sensor network of 300 or 500 nodes deployed over a field of 100 by 100. The sensors are uniformly distributed in the network, with each node capable of a fixed transmission range of 10. All the results are averaged over 100 runs.

In SBK and LKE, the two system parameters that affect the performance are the node density and λ, the security parameter of the λ-collusion-resistant key spaces. In iPAK, two more system parameters to be specified are ρ and T_0, where ρ determines the fraction of service nodes to be deployed, and T_0 determines the serving area of a service node. In our simulation study, we measure the performance of the three schemes under the same node density and security parameter λ, and conFigure the other parameters (T_0 and ρ) accordingly for a fair comparison.

In iPAK, the serving area of a service sensor is specified by the preconfigured parameter T_0. While in SBK and LKE, a service sensor determines its coverage area according to λ and the node density. Specifically, a service sensor serves worker sensors within T_0-hop (in SBK) or L-distance (in LKE), respectively, where $N_{T_0} \leq \lambda$ and $\pi L^2 = \lambda \times A/N$, T_0 is the maximum number satisfying $N_T \leq \lambda$ and N_T is the average number of neighbors within T hops in the network, N is the number of sensors in the network, and A is the deployment area. In the simulation, we select T_0 (for SBK and iPAK) and L (for LKE) that satisfy

$$N_{T_0} \leq \lambda = \frac{N}{A} \times \pi L^2. \quad (3)$$

Specifically, we consider $N = 300$ or 500 sensors in the network, estimate N_T, the average number of neighbors within T-hop using the ER model [12] (see Table 1), decide the forwarding bound T_0 for a given security parameter λ (see Table 2), and measure the performance accordingly.

Table 1: N_T, the number of neighbors within T hops, computed from ER model, used in Tests 1, 2, and 5.

T	1	2	3	4	5
$N_T(N = 300)$	9	26	55	101	164
$N_T(N = 500)$	16	48	106	194	310

Table 2: T_0, the forwarding bound, used in Tests 1 and 2.

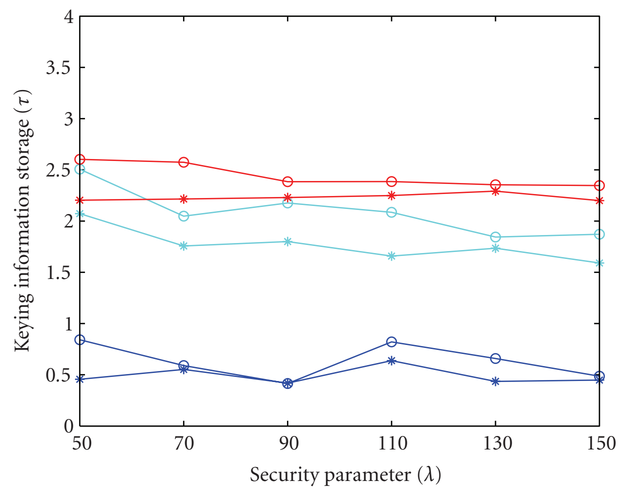
λ	50	70	90	110	130	150
$T_0(N = 300)$	2	3	3	4	4	4
$T_0(N = 500)$	2	2	2	3	3	3

Another parameter to be considered in iPAK is ρ, where $\rho = \lambda \times N_s/N_w$ and $N_s(N_w)$ is the number of service sensors (worker sensors). iPAK specifies the proportion of the two different sensors before deployment. While in SBK and LKE, service sensors are elected based on probability or location after deployment. In SBK, a service sensor is elected with the probability $P_s = 1/\lambda$, with the expectation that each service sensor serves only λ worker sensors. Thus, N_s/N_w is expected to be $1/\lambda$ in SBK. While in LKE, the network is divided into grids, and one service sensor is elected from each grid. Hence, $N_s/N_w \approx (\lceil \sqrt{A}/L \rceil)^2/N \approx A/NL^2 = \pi/\lambda$, where L is the grid size which satisfies $\pi L^2 = \lambda \times A/N$. Therefore, we consider two settings in the simulation: one is to compare iPAK and SBK with $\rho = 1$, the other is to compare iPAK and LKE with $\rho = \pi$.

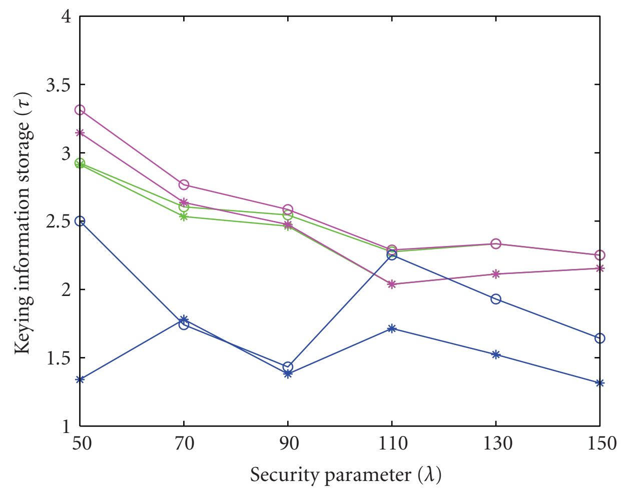
6.2. Comparison on Scalability, Storage, Connectivity and Cost. Given a series of λ values, we first measure the performance of iPAK, SBK and LKE in terms of *storage*, measured by τ, the number of keying information units (polynomial shares [19] or crypto shares [18]) obtained by a worker sensor; *connectivity*, measured by the key sharing probability P_0, the fraction of communication links that are secured by shared keys; and *cost*, measured by the percentage of service nodes generated [13, 14] or allocated [12] by the in-situ schemes.

We consider a network of 300 or 500 nodes, and employ the ER model to estimate N_T, the number of nodes within T hops in the network. The derived N_T values are given in Table 1. Then for each given λ, we set T_0 which is the maximal number satisfying $N_T \leq \lambda$. The T_0 values used in iPAK and SBK are reported in Table 2. According to the analysis in Section 6.1, we conduct three experiments: one is to compare SBK and iPAK, with $\rho = 1$ in iPAK; one is to compare LKE and iPAK, with $\rho = \pi$ in iPAK; one is to compare SBK and LKE under the same λ and node density. The results are presented in Figures 4, 5, and 6, respectively.

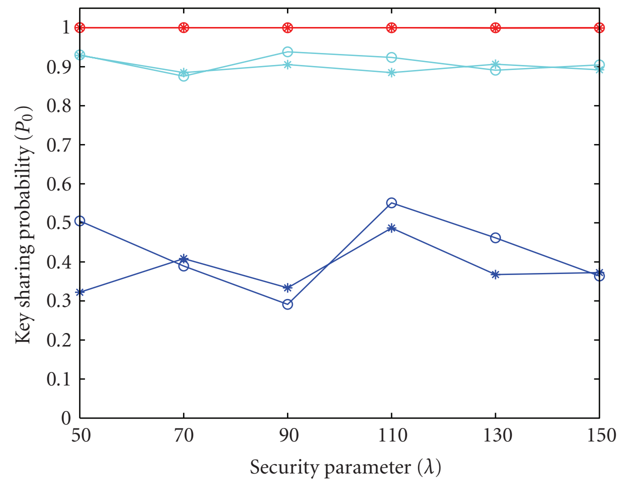
As illustrated in Figures 4, and 5, SBK and LKE can reach better connectivity than iPAK. By adjusting the number of service nodes to be generated, SBK and LKE respond actively to different network conditions with a high key sharing probability. However, iPAK has no such self-adjustability due to the predetermined ρ and T_0 values. Hence, iPAK requires that the system parameters should be carefully planned beforehand for specific network conditions. Nevertheless,



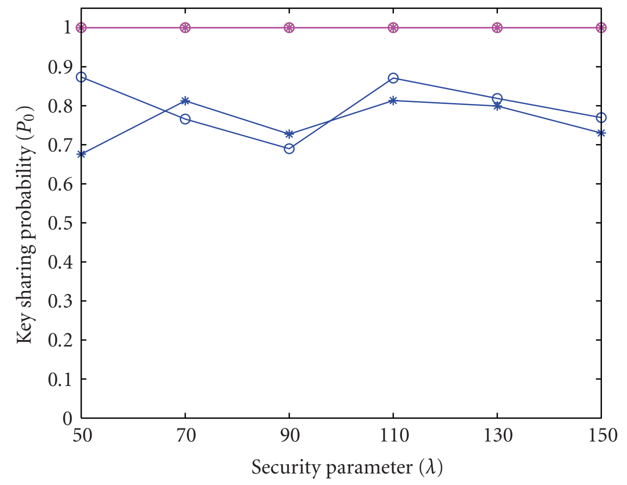
(a) Storage



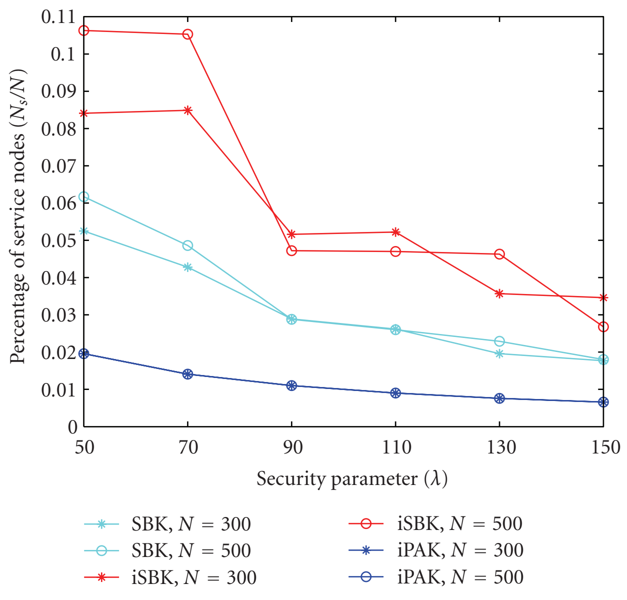
(a) Storage



(b) Connectivity

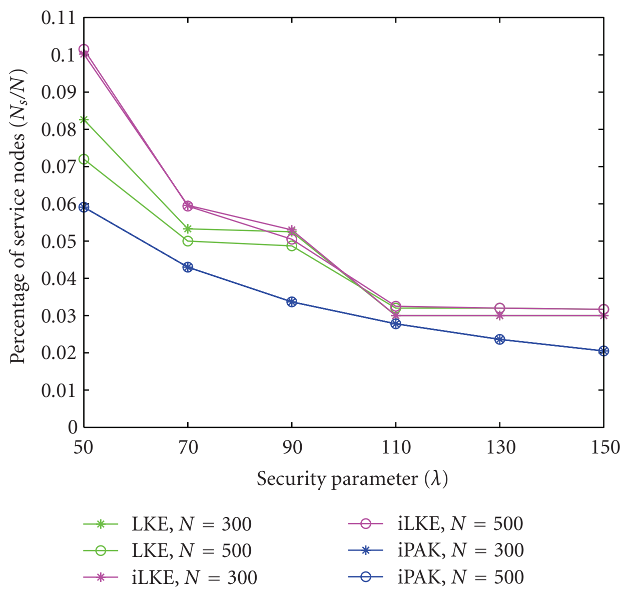


(b) Connectivity



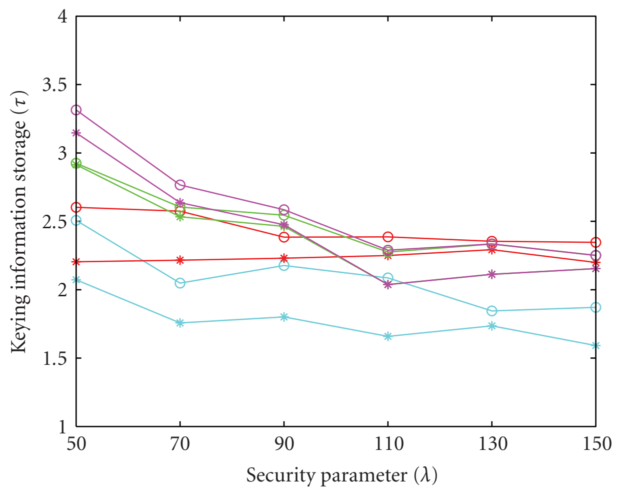
(c) Cost

Figure 4: Test 1. iPAK versus SBK (iPAK: $\rho = 1$, $N_{T_0} \leq \lambda$): Comparison on storage, connectivity, and cost.

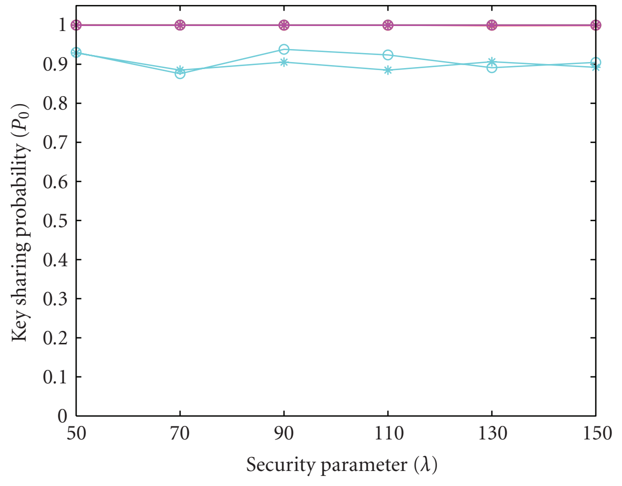


(c) Cost

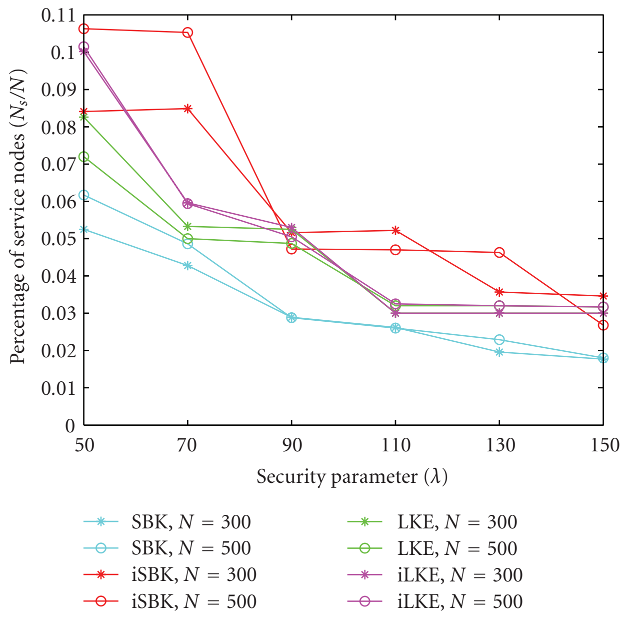
Figure 5: Test 2. iPAK versus LKE (iPAK: $\rho = \pi$, $N_{T_0} \leq \lambda$): Comparison on storage, connectivity, and cost.



(a) Storage



(b) Connectivity



(c) Cost

Figure 6: Test 3. SBK versus LKE: Comparison on storage, connectivity, and cost.

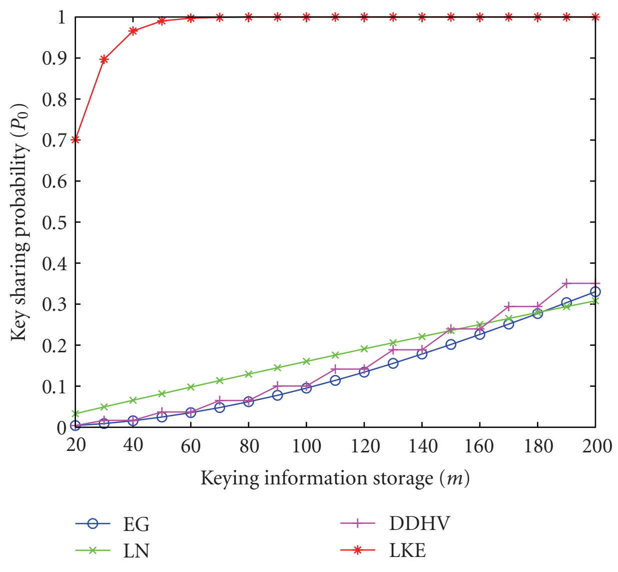


Figure 7: Test 4. Comparison of In-Situ schemes and Probabilistic-based Key Predistribution Schemes: Key Sharing Probability vs. Keying Information Storage.

iPAK has the least on-site operating complexity, since node role differentiation and key space construction are already finished before deployment.

Note that the performance of iPAK can be improved by choosing the appropriate system parameters. For example, we set $\rho = 1$ in Test 1 for a fair comparison between iPAK and SBK. $\rho = 1$ indicates $N_s/N_w = 1/\lambda$, which is just the lower bound for the fraction of service sensors to ensure the desired key-sharing probability under the limitation of $N_{T_0} \leq \lambda$. Thus, the key-sharing probability of iPAK is low in Figure 4. However, by selecting $\rho = \pi$ in Test 2, iPAK can achieve a much better connectivity with a small increase in the storage overhead. Hence, we can safely claim that iPAK, as well as SBK and LKE, can be configured to reach a high connectivity with a small amount of keying information storage in worker sensors. By using service nodes as sacrifices, all of the three in-situ schemes can avoid the storage space wastage that is existent in all the probabilistic-based key predistribution schemes, since the keying information is only disseminated within the close neighborhood.

As illustrated in Figure 6, we also observe that SBK and LKE behave similarly, while SBK can always burden worker sensors with similar storage overhead while achieving high connectivity, which is attributed to SBK's excellent topology adaptability. In SBK, sensors differentiate their roles as either service nodes or worker nodes after deployment by probing the local connectivity of the network, and then service nodes disseminate the keying information according to the specific network connectivity. But in LKE, a deterministic procedure based on location information is conducted for role differentiation and keying information distribution. Thereafter, we can expect SBK to perform better than LKE in adapting to different network conditions.

To further study the scalability of the in-situ schemes, we select LKE to compare with several probabilistic-based

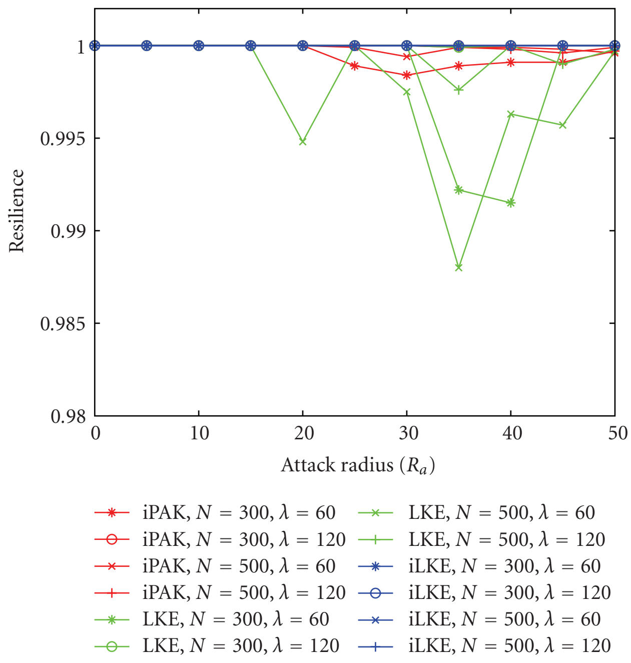


Figure 8: Test 5. iPAK vs. LKE (iPAK: $\rho = \pi$, $N_{T_0} \leq \lambda$). Comparison on Resilience Against Node Capture Attack.

Table 3: T_0, the forwarding bound, used in Test 5.

λ	60	120
$T_0(N = 300)$	3	4
$T_0(N = 500)$	2	3

key predistribution schemes. Figure 7 plots the relationship between P_0 and m, the number of memory units for keying information storage in a worker node (for a λ-collusion-resistent key space, m is determined by τ, the number of keying information units a sensor can obtain in the form of $m = (\lambda+1)\times\tau$ for the polynomial-based key space [19], and $m = (\lambda + 2) \times \tau$ for the matrix-based key space [18]) . We measure LKE's key sharing probability and compare it with that of the basic random key predistribution scheme (EG) [2], the random polynomial-based key space predistribution scheme (LN) [7] and the random matrix-based key space predistribution scheme (DDHV) [5]. The settings in EG and DDHV are the same as those in [6]. In EG, the key pool is of size 100, 000. In DDHV, we set the security parameter $\lambda = 19$ and the key pool size of 241 key spaces. For LN and LKE, both are considered in a network with 600 nodes, with each node storing 3 polynomial shares (we select 3 since it is a typical value for LKE in uniform network distribution as proved in [14]). The results show that the in-situ scheme can reach a much higher connectivity than the probabilistic-based predistribution schemes given the same amount of storage budget. Since the in-situ key establishment schemes are purely localized, they can completely remove the randomness inherent to the key predistribution schemes and hence achieve a much better scalability.

In summary, all of the three in-situ schemes obtain high scalability in network size. They can reach high connectivity with small amount of storage overhead, while SBK outperforms LKE, LKE outperforms iPAK in terms of topology adaptability.

6.3. *Comparison on Resilience.*

To evaluate the resilience of the in-situ schemes, we consider a smart attack where an adversary compromises all nodes within a disk of radius R_a, and measure the resilience with the following metric.

6.3.1. *Resilience.*

Given an attack radius R_a, the resilience against node capture attacks is defined to be the fraction of the compromised links incident to at least one compromised sensor among all the compromised links. Note that the metric resilience is in the range $(0, 1]$, where a value closer to 1 represents a better resilience.

We consider only iPAK and LKE in our simulation study, since in SBK there are at most λ worker nodes within a λ-collusion-resistent key space. Thus, the resilience of SBK remains to be 1 no matter how many nodes are captured and no matter what the network topology will be.

In the simulation, we set $\rho = \pi$ in iPAK to compare with LKE. T_0 (see Table 3) is the maximal number that satisfies $N_T \leq \lambda$, where N_T (see Table 1) is evaluated with the ER model.

As illustrated in Figure 8, both iPAK and LKE can effectively prevent the leakage of security information about uncaptured nodes, while iPAK outperforms LKE under the constraint that $N_{T_0} \leq \lambda$. We also observe that iLKE achieves the "perfect" security, which allows an adversary to learn nothing about the uncaptured sensors from those being directly attacked.

In terms of resilience, iPAK, SBK and LKE perform differently since they follow different regulations on n_s, the number of keying information to be released in a λ-secure key space. SBK requires strictly that n_s be at most λ, while iPAK has no such provision at all. In Test 4, the regulation $N_{T_0} \leq \lambda$ indicates that each λ-collusion-resistent key space is expected to cover no more than λ worker sensors, which brings about the strong resilience as illustrated in Figure 8. As for LKE, the improved scheme (iLKE) follows the same requirement as in SBK, while the basic scheme has no requirement on n_s but defines for each key space a coverage region that is expected to contain λ nodes in a uniformly distributed network. Hence, we observe that LKE and iLKE behave similarly in a uniform network distribution, while iLKE remains "perfectly" secure and LKE shows a small fluctuation in resilience. Such a fluctuation is attributed to the topology that is not perfectly uniform in our simulation.

In summary, SBK and iLKE perform the best in maintaining the security of the system. LKE can achieve a strong resilience under uniform network distribution, while iPAK must set T_0 as $N_{T_0} \leq \lambda$ to work against node capture attack.

6.4. *Discussion on Computation Overhead.*

From the in-situ key establishment framework, we know that the computation overhead of a worker sensor comes from three sources:

encrypting a shared key k_s between a service sensor and itself in secure channel establishment, decoding the keying information obtained from the associated service node in keying information acquisition, and calculating the pairwise keys shared with its neighbors in shared key derivation. The first involves one modular squaring, while the second requires a symmetric decryption operation. These operations are repeated for each service sensor with which the worker sensor associated with.

For each neighbor, a work sensor needs to compute a pairwise key if they share a common key space. In general, given the keying information, computing a shared key with one neighbor takes $(\lambda + 1)$ modular multiplications for both key space models. Furthermore, if the matrix-based key spaces are used and only a seed, instead of the whole column of the public matrix G, is included as the keying information, each worker sensor needs $(\lambda + 1)$ more modular operations in order to recover the complete matrix share for each key space.

Modular operations are expensive in terms of energy consumption and computation time, which could make our in-situ schemes unapplicable to many practical sensor network settings. Therefore, we propose to utilize the secure pseudorandom functions (PRF) defined by the 802.11i working group and the Wi-Fi Alliance. These PRFs exploit the computationally light-weight HMAC-SHA-1, with each incorporating a different text string as input [29] to generate nonoverlapping key spaces. In our case, the text string can be the ID or the location information of the service node. Therefore in iPAK, each service node is preloaded with a PRF while in LKE and SBK, the elected service nodes run their stored PRFs to generate key spaces containing random keys. Then the service sensor securely deliver a set of pairwise keys to each associated worker sensor, as long as the worker sensor conveys the list of neighbors to the service sensor in the association phase.

Note that we can treat the PRF as another key space model, based on which each service sensor generates a random key pool that will supply pairwise keys to the associated worker sensors. It is obvious that no computation is needed at the worker sensor side. However, this zero computation overhead does not come for free: each worker sensor needs to collect the list of neighbors and send this information to all the associated service sensors. Therefore worker sensors tradeoff computation overhead with communication overhead. Furthermore, the λ-collusion resistent advantage is also lost as the PRF key space does not hold this property.

7. Conclusion

In this paper, we have studied iPAK, SBK and LKE, the three in-situ key establishment schemes proposed recently for large-scale sensor networks. We also introduce a simple improvement by exploiting a secure pseudorandom function to replace the matrix-based or the polynomial key space such that no computation is needed at the worker sensor to further conserve the resources. Our simulation results indicate that all the three in-situ key establishment schemes achieve high scalability in network size since they are purely localized. In addition, SBK and LKE outperform iPAK in terms of topology adaptability, SBK and iLKE have the best resilience against node capture attack, and iPAK has a better operating complexity. Our future research includes a more extensive performance study under different topology conditions and a comparison study with the probabilistic key predistribution schemes.

Acknowledgment

This research is supported in part by the US National Science Foundation under the CAREER Award CNS-0347674 and the Grant CCF-0627322.

References

[1] D. W. Carman, P. S. Kruss, and B. J. Matt, "Constraints and approaches for distributed sensor network security," Tech. Rep. 00-010, NAI Labs, Glenwood, Md, USA, September 2000.

[2] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS '02)*, pp. 41–47, Washington, DC, USA, November 2002.

[3] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy (S&P '03)*, pp. 197–213, Berkeley, Calif, USA, May 2003.

[4] H. Chan and A. Perrig, "PIKE: peer intermediaries for key establishment in sensor networks," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '05)*, pp. 524–535, Miami, Fla, USA, March 2005.

[5] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili, "A pairwise key predistribution scheme for wireless sensor networks," in *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS '03)*, pp. 42–51, Washington, DC, USA, October 2003.

[6] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '04)*, pp. 586–597, Hong Kong, March 2004.

[7] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS '03)*, pp. 52–61, Washington, DC, USA, October 2003.

[8] D. Liu, P. Ning, and W. Du, "Group-based key predistribution for wireless sensor networks," in *Proceedings of the ACM Workshop on Wireless Security (WiSe '05)*, Cologne, Germany, September 2005.

[9] Z. Yu and Y. Guan, "A key pre-distribution scheme using deployment knowledge for wireless sensor networks," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*, pp. 261–268, Los Angeles, Calif, USA, April 2005.

[10] Z. Yu, Y. Wei, and Y. Guan, "Key management for wireless sensor networks," in *Handbook of Wireless Mesh & Sensor Networking*, G. Aggelou, Ed., McGraw-Hill, New York, NY, USA, 2007.

[11] L. Zhou, J. Ni, and C. V. Ravishankar, "Efficient key establishment for group-based wireless sensor deployments," in *Proceedings of the ACM Workshop on Wireless Security (WiSe '05)*, pp. 1–10, Cologne, Germany, September 2005.

[12] L. Ma, X. Cheng, F. Liu, F. An, and J. Rivera, "iPAK: an in-situ pairwise key bootstrapping scheme for wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 8, pp. 1174–1184, 2007.

[13] F. Liu, X. Cheng, L. Ma, and K. Xing, "SBK: a self-configuring framework for bootstrapping keys in sensor networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 7, pp. 858–868, 2008.

[14] F. Liu and X. Cheng, "LKE: a self-configuring scheme for location-aware key establishment in wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 7, no. 1, pp. 224–232, 2008.

[15] S. A. Camtepe and B. Yener, "Key distribution mechanisms for wireless sensor networks: a survey," RPI Technical Report TR-05-07, Computer Science Department, Rensselaer Polytechnic Institute, Troy, NY, USA, March 2005.

[16] S. Zhu, S. Xu, S. Setia, and S. Jajodia, "Establishing pairwise keys for secure communication in ad hoc networks: a probabilistic approach," in *Proceedings of the 11th IEEE International Conference on Network Protocols (ICNP '03)*, p. 326, Atlanta, Ga, USA, November 2003.

[17] R. Di Pietro, L. V. Mancini, and A. Mei, "Efficient and resilient key discovery based on pseudo-random key pre-deployment," in *Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS '04)*, pp. 217–224, Santa Fe, NM, USA, April 2004.

[18] R. Blom, "An optimal class of symmetric key generation systems," in *Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques (EUROCRYPT '84)*, pp. 335–338, Paris, France, April 1984.

[19] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaroe, and M. Yung, "Perfectly-secure key distribution for dynamic conferences," in *Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO '92)*, vol. 740 of *Lecture Notes in Computer Science*, pp. 471–486, Santa Barbara, Calif, USA, August 1992.

[20] W. Du, R. Wang, and P. Ning, "An efficient scheme for authenticating public keys in sensor networks," in *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC '05)*, pp. 58–67, ACM Press, Urbana-Champaign, Ill, USA, May 2005.

[21] E. Shi and A. Perrig, "Designing secure sensor networks," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 38–43, 2004.

[22] D. Liu and P. Ning, "Location-based pairwise key establishments for static sensor networks," in *Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Security of Ad Hoc and Sensor Networks in Association with 10th ACM Conference on Computer and Communications Security*, pp. 72–82, Fairfax, Va, USA, October 2003.

[23] D. Huang, M. Mehta, D. Medhi, and L. Harn, "Location-aware key management scheme for wireless sensor networks," in *Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '04)*, pp. 29–42, ACM Press, Washington, DC, USA, October 2004.

[24] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar, "SPINS: security protocols for sensor networks," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking, (MOBICOM '01)*, pp. 189–199, Rome, Italy, July 2001.

[25] S. Zhu, S. Setia, and S. Jajodia, "LEAP: efficient security mechanisms for large-scale distributed sensor networks," in *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS '03)*, pp. 62–72, Washington, DC, USA, October 2003.

[26] R. Watro, D. Kong, S.-F. Cuti, C. Gardiner, C. Lynn, and P. Kruus, "TinyPK: securing sensor networks with public key technology," in *Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '04)*, pp. 59–64, Washington, DC, USA, October 2004.

[27] M. O. Rabin, "Digitalized signatures and public-key functions as intractable as factorization," Tech. Rep. MIT/LCS/TR-212, MIT Laboratory for Computer Science, Cambridge, Mass, USA, 1979.

[28] R. Anderson, H. Chan, and A. Perrig, "Key infection: smart trust for smart dust," in *Proceedings of the 12th IEEE International Conference on Network Protocols (ICNP '04)*, pp. 206–215, Berlin, Germany, October 2004.

[29] J. Edney and W. A. Arbaugh, *Real 802.11 Security: Wi-Fi Protected Access and 802.11i*, Addison-Wesley, Reading, Mass, USA, 2004.