iPAK: An In Situ Pairwise Key Bootstrapping Scheme for Wireless Sensor Networks

Liran Ma, *Student Member*, *IEEE*, Xiuzhen Cheng, *Member*, *IEEE*,
Fang Liu, *Student Member*, *IEEE*, Fengguang An, and Jose Rivera

Abstract—Wireless Sensor Networks (WSNs) are characterized by resource constraints and large scalability. Many applications of WSNs require secure communication, a crucial component especially in hostile environments. However, the low computational capability and small storage budget within sensors render many popular public-key-based cryptographic systems impractical. Symmetric key cryptography, on the other hand, is attractive due to its efficiency. Nevertheless, establishing a shared key for communicating parties is a challenging problem. In this paper, we propose and analyze an **i**n situ **PA**irwise **K**ey bootstrapping scheme (iPAK) for large-scale WSNs. Our theoretical analysis and simulation study demonstrate that iPAK can achieve a high key-sharing probability between neighboring sensors and a strong resilience against node-capture attacks at the cost of low storage overhead.

Index Terms—Wireless sensor networks, in situ key establishment, key predistribution, security.

1 Introduction

As Wireless Sensor Networks (WSNs) approach widespread deployment, security provisioning has become a central concern. A significant problem in secure communication is the symmetric key establishment between neighboring sensors. Building upon a solid pairwise shared-key bootstrapping infrastructure, most security services such as confidentiality, authenticity, and privacy can be addressed.

There are three types of secret key construction techniques: key distribution using trusted third parties (TTPs), key agreement, and key predistribution. TTP-based schemes rely on a trusted server or servers organized in a hierarchical structure for key agreement between nodes (for example, Kerberos [1]). These schemes may not be practical for large-scale sensor networks, since the deployment of TTP servers is either prohibited due to environmental constraints or uneconomical because of the scalability issue.

Key agreement schemes (such as the Diffie-Hellman key agreement [2]) set up pairwise keys through successive message exchanges in a secure manner by using asymmetric cryptography. Their applicability is mainly barricaded by the limited computation and communication capabilities within sensor devices [3]. For example, the MICA2 Berkeley mote has an 8-bit 7.3828-MHz Atmega 128L processor with a 4-Kbyte static RAM (SRAM) and a 128-Kbyte ROM [4].

- *L. Ma, X. Cheng, and F. Liu are with the Department of Computer Science, The George Washington University, 801 22nd Street NW, Phillips Hall, Suite 720D, Washington, DC 20052. E-mail: {lrma, cheng, fliu}@gwu.edu.*
- *F. An is with the Institute of Computing Technology, Chinese Academy of the Sciences, Beijing 100080, P.R. China. E-mail: anfengguang@software.ict.ac.cn.*
- *J. Rivera is with the Architecture Operation Network and Space (AONS), US Army Chief Information Office, G6. E-mail: jose.rivera@us.army.mil.*

Manuscript received 5 May 2006; revised 22 Aug. 2006; accepted 25 Aug. 2006; published online 9 Jan. 2007.
Recommended for acceptance by I. Stojmenovic.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-0115-0506.
Digital Object Identifier no. 10.1109/TPDS.2007.1063.

Key predistribution schemes (KPSs), on the other hand, have attracted much attention to the sensor network research society due to their efficiency and simplicity. In KPS, keys or keying materials are preloaded into sensors before deployment. Neighboring sensors discover shared keys after deployment through partial keying information exchanges. A number of KPS protocols have been proposed in the literature [5], [6], [7], [8], [9], [10].

To compensate for the unpredictability of the network topology prior to deployment, KPS requires a large amount of keying information to be preloaded in order to achieve a desirable key-sharing probability between neighboring sensors. As a side effect, part of the keying information may never be utilized during the entire network lifetime. Such an inefficient use of the limited memory in sensors makes the current KPSs scale poorly to very large networks.

In this paper, we propose an **i**n situ **PA**irwise **K**ey bootstrapping scheme, termed as iPAK, to facilitate the shared-key establishment between neighboring sensors. The proposed scheme aims at achieving a satisfying key-sharing probability among sensors with low memory overhead in large-scale sensor networks. To achieve this goal, *service sensors* are introduced to assist the key establishment procedure of normal sensors, namely, *worker sensors*. Note that service sensors are considered as sacrificers, whose major task is to distribute the keying information to worker sensors. The keying information is delivered through a computationally asymmetric secure channel after deployment. Worker sensors discover shared keys with their neighbors after obtaining credentials from the nearby service nodes. Service sensors erase the stored security information immediately after their duty is complete to further enhance security.

The major contributions of this paper are threefold:

1. We propose iPAK, a distributed key bootstrapping protocol for large-scale WSNs. The proposed scheme explores the direction of in situ key computation instead of keying information predistribution, addressing the extreme resource constraint problem with a careful design.

2. Theoretical analysis and plausible simulation studies of iPAK are presented in detail, demonstrating that iPAK can achieve a very high key-sharing probability with low storage overhead.
3. In addition, we introduce a novel analytical model for estimating the number of multihop neighbors in a uniform random graph. This model is exploited to help analyze the performance of iPAK.

The remaining parts of this paper are organized as follows: A brief overview of the related research is given in Section 2. The assumptions and background knowledge are presented in Section 3. We propose our in situ key management scheme iPAK in Section 4. An *Effective Radius (ER)* model is derived and evaluated in Section 5. This model is employed for performance evaluation of iPAK in Section 6. Finally, we conclude our paper in Section 7.

2 Related Work

In this section, we summarize some of the key management schemes for sensor networks. For a more comprehensive literature survey, we refer the readers to [11] and the references therein.

The pioneer work on random key predistribution for sensor networks was proposed by Eschenauer and Gligor in [5]. A large key pool K is computed offline and each sensor picks k keys randomly from K without replacement before deployment. These k keys form the key ring of a sensor. After deployment, a sensor establishes a shared key with a neighbor if their key rings have at least one key in common. The security of the basic random KPS is enhanced by the q-composite scheme in [6], in which $q > 1$ common keys are required to establish a shared key. These q keys are hashed into one key to achieve better resilience against node capture.

Du et al. [7] are the first to apply Blom's scheme [12] for shared-key establishment in sensor networks. Blom's scheme is based on the computation of a symmetric matrix, which provides a key space for all nodes that possess a public share and a private share of the key space. In [7], ω key spaces instead of one key space are precomputed, and each sensor stores the private/public shares from τ key spaces. These τ key spaces are randomly selected from the ω key spaces without replacement. If two sensors share information from one common key space, they can establish a shared key after exchanging their public shares. This scheme actually combines the idea of the random key predistribution [5] with Blom's method. It has better resilience against node-capture attacks, with a reasonable number of storage and communication overheads. We will elaborate on Blom's scheme in Section 3.

Key predistribution takes advantage of the fact that a random graph is almost certainly connected if its average degree is above a threshold. However, it requires each sensor to be preloaded with a large amount of cryptographic information in order to achieve a satisfying key-sharing probability between neighboring sensors. Furthermore, the probabilistic nature of key predistribution results in serious storage wastage because much of the preloaded information may never be used during the lifetime of the sensor. Consequently, the scalability of key predistribution is poor, since the amount of required security information to be preloaded increases with the network size.

To improve the scalability, a deployment-knowledge-based key management approach is proposed in [8]. In this scheme, multiple deployment points are identified in a sensor network. For each deployment point, a key space is precomputed from a large key pool. Neighboring deployment points have a number of keys in common. In other words, their key spaces consist of common keys. All sensors are grouped before deployment, and each group corresponds to one deployment point. Each sensor randomly picks k keys from the key space of its group. After deployment, sensors in a close neighborhood have a high probability to share a common key. This scheme puts strong requirements on deployment knowledge, but achieves better scalability compared to those proposed in [5], [6], and [7]. Chan and Perrig [13] propose exploiting trusted intermediaries for shared-key establishment between nodes. It is shown that both communication and storage overheads scale sublinearly with respect to the number of nodes in a network.

Liu et al. [14], [15] develop a general framework for pairwise key establishment based on the polynomial-based key predistribution protocol [16]. Location-aware key establishment schemes are proposed in [17] and [18] for better resilience against security attacks. Most recently, Ren et al. [19] propose using a delicate key-generation technique such that a large number of random keys can be represented by a small number of key-generation keys.

SPINS [20] and the Lightweight Extensible Authentication Protocol (LEAP) [21] establish various keys with the assistance of a base station. Two group-based pairwise key establishment schemes [9], [10] have been proposed in the Proceedings of the Fourth ACM Workshop on Wireless Security (WiSe '05). In these two works, sensors are divided into horizontal and vertical groups, and each sensor resides in exactly one horizontal and one vertical group. A pair of sensors within the same group share a unique key, and path keys are utilized to boost the key-sharing probability. Compared to [8], [9] and [10] release the strong requirement on deployment knowledge, with a trade-off of higher communication overhead.

As pointed out in [22] and [23], the current shared-key establishment solutions are not perfect. They still have to struggle with the conflicts among memory limits, desired key-sharing probability, scalability in network size, and resilience against node compromise.

Our work is different from those mentioned above in that it is an in situ key bootstrapping protocol. Since the location of sensor nodes and network connectivity in a large deployment cannot be predetermined, we choose not to preload any key-space-related information to worker sensors. Instead, service nodes convey security information to worker nodes in their neighborhood after deployment. This is a fundamental difference compared to existing schemes ([5], [6], [7], and so forth).

The nice features of iPAK include the following: 1) iPAK is a truly localized scheme because the keying information is only distributed to the worker sensors in the vicinity after deployment and, therefore, it has no limitations on the scalability concerning the network size. 2) iPAK introduces

a reasonable amount of storage overhead in worker sensors, but achieves a high key-sharing probability between neighbors. 3) Because computation-intensive operations have been shifted to service nodes, the battery powers of worker sensors can be conserved.

3 Preliminaries

3.1 Blom's Key Management Scheme

We adopt Blom's key management scheme [12] for shared-key computation. Note that iPAK works equivalently well if the polynomial-based key space model [16] is employed.

Let G be a $(\lambda+1)\times M$ matrix over a finite field $GF(q)$, where q is a large prime.[1] The connotation of M will become clear later. G is public, with each column called a *public share*. Let D be any random $(\lambda+1)\times(\lambda+1)$ symmetric matrix. D must be kept private. D and G jointly define a *key space* (D, G).

The transpose of $D\cdot G$ is denoted by A, that is, $A=(D\cdot G)^T$. A is private, too, with each row called a *private share*. Since D is symmetric, $A\cdot G$ is symmetric as well. If we let $K=(k_{ij})=A\cdot G$, then we have $k_{ij}=k_{ji}$, where k_{ij} is the element at the ith row and the jth column of matrix K, $i,j=1,2,\cdots,M$. The basic idea of Blom's scheme is to use k_{ij} as the secret key shared by node i and node j.

The ith column of G, a public share, and the ith row of D, a private share, form the ith *keying pair*, which will be loaded into sensor i. Two sensors with keying pairs obtained from the same key space compute a shared key after exchanging their public shares. From this analysis, it is clear that M is the number of sensors supported by one key space.

Blom's key-generation scheme [16] ensures the so-called λ-secure property, which means that the network should be perfectly secure as long as no more than λ keying pairs are exposed. This requires any $\lambda+1$ columns of G to be linearly independent.

3.2 Rabin's Scheme

Rabin's scheme [26] is a public cryptosystem, which is adopted by iPAK to establish a computationally asymmetric secure channel between a worker sensor and a service sensor.

- **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 and $n=p\cdot q$ is the public key.
- **Encryption**. Let P_l be the plain text that is represented as an integer in Z_n. Then, the cipher text $c=P_l^2 \bmod n$.
- **Decryption**. Since $p\equiv q\equiv 3 \bmod 4$, we have

$$m_p = c^{\frac{p+1}{4}} \bmod p$$

and

$$m_q = c^{\frac{q+1}{4}} \bmod q.$$

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 $+r$, $-r$, $+s$, and $-s$ can be obtained:

$$r=(y_p\cdot p\cdot m_q+y_q\cdot q\cdot m_p) \bmod n, \quad (1)$$

$$-r=n-r, \quad (2)$$

$$s=(y_p\cdot p\cdot m_q-y_q\cdot q\cdot m_p) \bmod n, \quad (3)$$

$$-s=n-s. \quad (4)$$

Note that Rabin's encryption [26] requires only one squaring, which is several hundreds of times faster than RSA [24]. 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 B, 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 consisting of two types of sensors, namely, *worker nodes* and *service nodes*, randomly distributed over the deployment region. No neighborhood information is available before deployment. Worker sensors are in charge of normal network operations, whereas service sensors intend to provide keying information to facilitate shared-key computation between worker sensors in close proximity. 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.

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 [25].

There is a unique key k_0 preloaded to all sensors such that all exchanged messages are authenticated during the bootstrapping procedure. Therefore, any node deployed by an adversary can be excluded from key establishment as long as k_0 remains secure before the procedure is complete. Each service node carries one key space (as explained in Section 3.1) and two large primes p and q (as explained in Section 3.2), computed by a supercomputer before deployment. Each key space is uniquely identified by an ID. All sensors remove their stored keying information (k_0 and/or the key space) at the end of the key bootstrapping procedure.

4 iPAK Scheme

In this section, we elaborate iPAK, an in situ key bootstrapping scheme for large-scale sensor networks. This scheme contains three phases: the preloading of the key space information to each service node, the keying pair acquisition between worker sensors and service sensors, and the computation of a shared key between two neighboring worker sensors. A secure channel is utilized for a worker sensor to obtain keying information from a service sensor in the vicinity.

1. q must be large enough to accommodate a cryptographic key.

4.1 Key Space Preloading Phase

During the predeployment phase, each service node preloads with a key space (D, G), as defined in Blom's scheme [12], an integer n, and two large primes p and q such that $n = p \times q$ (see Section 3). Keying shares from the key space are to be disseminated to the worker sensors in the vicinity after deployment through a computationally asymmetric channel protected by p and q based on Rabin's public cryptosystem [26].

Note that the adoption of Blom's scheme [12] in iPAK is mainly motivated by its λ-collusion property. To break a key space in Blom's scheme, at least $\lambda + 1$ worker sensors associated with the same service sensor must be captured. This makes a successful attack much harder. If random keys (for example, a key ring) instead of keying information, as defined in Blom's scheme, are disseminated from a service sensor to a worker sensor, then the decryption of such a message releases all the keys assigned to the worker sensor.

4.2 Keying Pair Acquisition Phase

The applicability of iPAK relies on the availability of a secure channel between a worker sensor and the corresponding service sensor because the keying pair of each worker sensor needs to be transferred securely. In this section, we propose a public-key-assisted key exchange protocol to establish a secret key K_s between a worker sensor and a service sensor. Since worker sensors are supposed to operate for years, whereas service nodes can die after their duty is complete, cryptographic algorithms that shift a large amount of the computation overhead to the service node are preferred. Based on this consideration, iPAK adopts Rabin's public cryptosystem [26].

4.2.1 Secure-Session Establishment

Based on Rabin's scheme [26], described in Section 3.2, we propose the following secure-session establishment protocol between a worker sensor and a service sensor:

- Each service sensor broadcasts n, announcing its existence to worker sensors within one T_0-hop away (called *forwarding bound*). Note that the hop count T_0 is a design parameter that greatly affects the performance of the scheme in terms of *key sharing probability* and *storage overhead*.
- After receiving an announcement from a service node I, a worker sensor picks K_s and computes $E_n(K_s \| B) = (K_s \| B)^2 \bmod n$, where B is a predefined bit pattern to resolve the ambiguity in Rabin's decryption [26] and transmits $E_n(K_s \| B) \| B$ to I. Note that B is transmitted as plain text. K_s is the shared key between the worker sensor and the service sensor I.
- Upon receiving the $E_n(K_s \| B) \| B$ from a worker sensor, the service sensor computes $D_{(p,q)}(E_n(K_s \| B))$ based on Rabin's decryption algorithm [26].

Note that, in this protocol, each worker sensor executes one Rabin's encryption [26] 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. Also, note that, even though it is computationally favorable to adopt a simple scheme such as computing a secret key by XORing k_0 (Section 3.3) and service node ids to secure the disseminated keyshare, we believe that this method is not strong enough in a harsh environment. However, the security of Rabin's scheme [26] is comparable to that of RSA [24], since both rely on the hardness of factoring large primes.

4.2.2 Keying Pair Acquisition

The secure channel established based on Rabin's public cryptosystem (Section 4.2.1) is employed for keying pair acquisition:

- Each worker sensor sends a request to the service sensor, asking for a keying pair containing a public share and a private share. This message is optional, since a service node can treat the message containing K_s from the worker sensor as a request.
- Upon receiving a request from sensor i, the service sensor selects an unused keying pair and transmits it to i together with the key space id. This message must be encrypted by K_s.

Keying pair acquisition can be further secured with the introduction of nonces to avoid replay attacks.

4.3 Shared-Key Discovery Phase

After obtaining keying pairs from service sensors in the vicinity, each worker sensor broadcasts the tuple $< \text{key space id}, \text{public share} >$ for each key space once.[2] Two neighboring sensors are able to compute a shared key based on Blom's scheme [12] if they have acquired keying information from at least one common service node. Blom's λ-secure key management scheme [12] has been well tailored for lightweight sensor networks in [7].

5 The ER Model

To study the performance of iPAK, we need the *ER* model presented in this section to identify the expected number of t-hop neighbors[3] that a sensor may have in a randomly distributed network, where $t = 1, 2, \cdots$. Note that this is a nontrivial problem due to the randomness of the node positions.

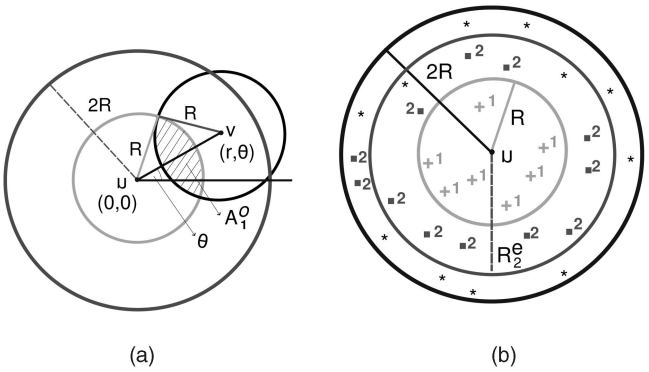
5.1 Model Derivation

We assume that there are N nodes randomly distributed in the deployment region with an area of A. Furthermore, we assume that all the nodes have the same radio transmission range R. Thus, an arbitrary node u can cover $d_1 = \pi R^2 \frac{N}{A} - 1$ nodes within one hop on the average. Now, how do we derive d_t, where $t = 2, 3, \cdots$, that is, the expected number of t-hop neighbors (in the shortest path) that an arbitrary node may have? In the following, we will introduce our ER model to recursively compute these values.

In the first place, we consider the case of $t = 2$. Let v be another arbitrary node whose euclidean distance to u is denoted by r. If $r \leq R$, then v is a one-hop neighbor of u. On

2. The broadcastings for all associated key spaces can be combined into a larger message.

3. In the shortest path.



(a) (b)

Fig. 1. Overlapping regions. (a) v is a two-hop neighbor of u. (b) The effective two-hop radius R_2^e of u.

the other hand, if $r > 2R$, then v cannot be reached by u via two hops. Therefore, v is a two-hop neighbor of u if and only if 1) $R < r \leq 2R$ and 2) u and v share at least one immediate neighbor.[4]

Let E_1 be the event that the distance $r \in (R, 2R]$ and E_2 be the event that u and v have at least one common immediate neighbor. Let A_1^o be the overlapping area of u and v, as shown in Fig. 1a. We have

$$Pr[E_2|E_1] = 1 - \left(1 - \frac{A_1^o}{\pi R^2}\right)^{d_1}, \tag{5}$$

where A_1^o is defined as

$$A_1^o = 2R^2 \arccos\left(\frac{r}{2R}\right) - \frac{r}{2}\sqrt{4R^2 - r^2}. \tag{6}$$

Based on (5), the expected value of $Pr[E_2|E_1]$ throughout the annulus region from R to $2R$ (see Fig. 1a), denoted by P_2^a, can be represented by

$$P_2^a = \frac{\int_0^{2\pi} d\theta \int_R^{2R} \left(1 - \left(1 - \frac{A_1^o}{\pi R^2}\right)^{d_1}\right) r dr}{\pi((2R)^2 - R^2)}. \tag{7}$$

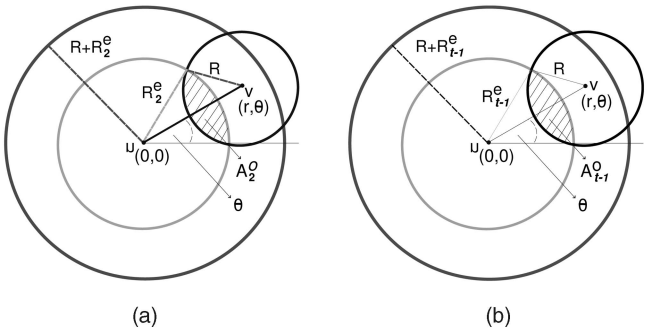
As a result, the number of u's two-hop neighbors, denoted by d_2, is given as follows:

$$d_2 = (3\pi R^2)\frac{N}{A} P_2^a. \tag{8}$$

Directly computing the exact number of t-hop neighbors is a difficult problem when t is greater than 2. Therefore, we introduce the *ER* model to facilitate this computation. Let D_t be the expected number of neighbors that are at most one t-hop away. In our ER model, the ER of the t-hop coverage of a node u is defined as the radius of a virtual disk centered at u, which can cover D_t number of nodes.

For example, Fig. 1b depicts the ER for the case of two hops. In this figure, the virtual disk centered at u, with a radius of R_2^e, covers $d_1 + d_2$ number of nodes in total. These covered nodes include all the one-hop neighbors (labeled with plus signs), a number of two-hop neighbors (labeled with dots), and a few other nodes (labeled with star signs). Note that the number of two-hop neighbors that fall out of the virtual disk equals the number of nodes that cannot be reached from u within two hops, but fall into this virtual disk.

4. A one-hop neighbor can also be called an immediate neighbor.



(a) (b)

Fig. 2. ER for t-hop neighbors. (a) v is a three-hop neighbor of u. (b) v is a t-hop neighbor of u.

Accordingly, the ER R_2^e for the two-hop case can be calculated as follows:

$$\pi(R_2^e)^2 \frac{N}{A} = d_1 + d_2 + 1. \tag{9}$$

By plugging in $d_1 = \pi R^2 \frac{N}{A} - 1$ and (8) into (9), we obtain

$$R_2^e = \sqrt{R^2 + 3R^2 P_2^a}. \tag{10}$$

Now, we are ready to derive the number of three-hop neighbors for u. In our ER model, v's transmission range remains to be R, whereas u's transmission range is set to be R_2^e. In other words, the virtual disk with a radius R_2^e centered at u represents u's two-hop coverage. In this case, v is a three-hop neighbor of u if and only if 1) $R_2^e < r \leq R_2^e + R$ and 2) u's two-hop virtual disk covers at least one of v's immediate neighbors, where r is the euclidean distance between u and v.

With a similar analysis, we obtain P_3^a, the probability that v is a three-hop neighbor of u, given $R_2^e < r \leq R_2^e + R$:

$$P_3^a = \frac{\int_0^{2\pi} d\theta \int_{R_2^e}^{(R_2^e+R)} \left(1 - \left(1 - \frac{A_2^o}{\pi R^2}\right)^{d_1}\right) r dr}{\pi((R_2^e + R)^2 - (R_2^e)^2)}, \tag{11}$$

where A_2^o, the overlapping area covered by both u and v, as shown in Fig. 2a, is regulated by

$$\begin{aligned} A_2^o = {} & R^2 \arccos\left(\frac{r^2 + R^2 - (R_2^e)^2}{2rR}\right) \\ & + (R_2^e)^2 \arccos\left(\frac{r^2 + (R_2^e)^2 - R^2}{2rR_2^e}\right) \\ & - \frac{1}{2}\sqrt{4r^2(R_2^e)^2 - (r^2 - R^2 + (R_2^e)^2)^2}. \end{aligned} \tag{12}$$

Thus, the number of u's three-hop neighbors can be approximated by

$$d_3 = \pi((R_2^e + R)^2 - (R_2^e)^2)\frac{N}{A} P_3^a. \tag{13}$$

Furthermore, the effective radius for three hops is

$$R_3^e = \sqrt{(R_2^e)^2 + ((R_2^e + R)^2 - (R_2^e)^2)P_3^a}. \tag{14}$$

By recursively applying this procedure, we get the probability P_t^a of v being u's t-hop neighbor, the expected

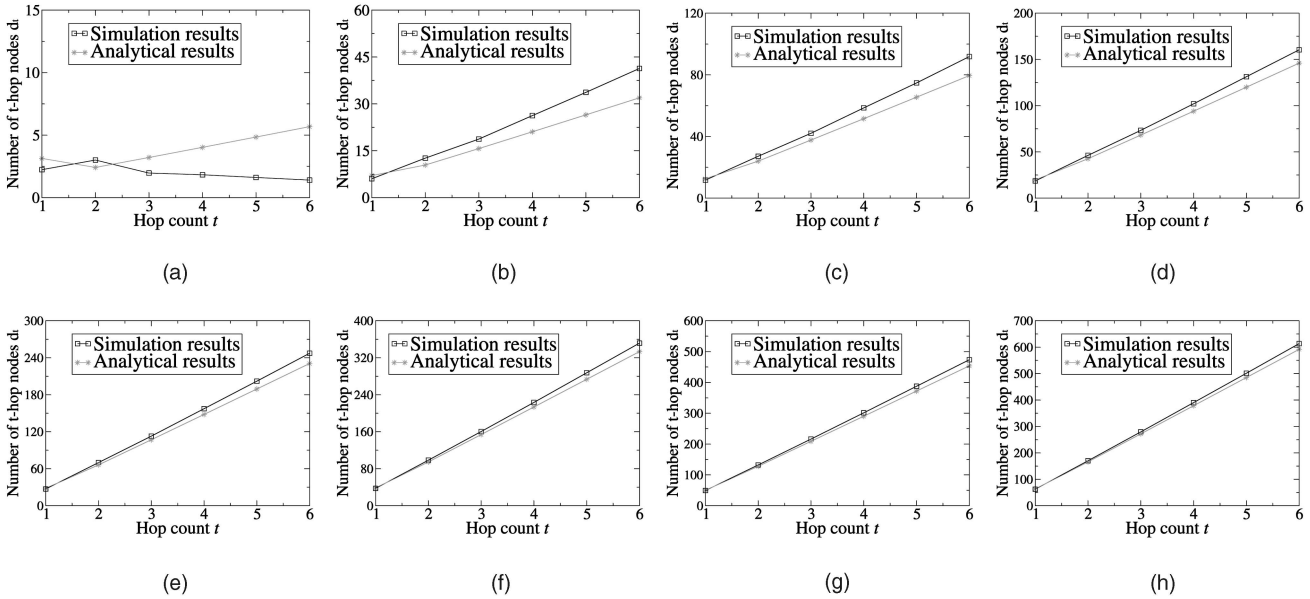


Fig. 3. Analytical results versus simulation results. (a) $R = 1.0,\ d_1 = 2.1$. (b) $R = 1.5,\ d_1 = 6.1$. (c) $R = 2.0,\ d_1 = 11.5$. (d) $R = 2.5,\ d_1 = 18.6$. (e) $R = 3.0,\ d_1 = 27.2$. (f) $R = 3.5,\ d_1 = 37.5$. (g) $R = 4.0,\ d_1 = 49.3$. (h) $R = 4.5,\ d_1 = 62.6$.

number of u's t-hop neighbors d_t, and the equivalent radius R_t^e as follows:

$$P_t^a = \frac{\int_0^{2\pi} d\theta \int_{R_{t-1}^e}^{(R_{t-1}^e+R)} \left(1 - \left(1 - \frac{A_{t-1}^o}{\pi R^2}\right)^{d_1}\right) r dr}{\pi((R_{t-1}^e + R)^2 - (R_{t-1}^e)^2)}, \quad (15)$$

$$d_t = \pi((R_{t-1}^e + R)^2 - (R_{t-1}^e)^2)\frac{N}{A}P_t^a, \quad (16)$$

$$R_t^e = \sqrt{(R_{t-1}^e)^2 + ((R_{t-1}^e + R)^2 - (R_{t-1}^e)^2)P_t^a}. \quad (17)$$

Our ER model will be validated through a simulation study in Section 5.2.

5.2 Evaluation of the ER Model

5.2.1 Validation Settings

- There are 6,400 nodes randomly distributed in a field with a length and width of 80×80. Therefore, the node density $\phi = 1$; that is, there is one node in a unit square on the average.
- By varying the transmission range R from 1.0 to 4.5, with a step of 0.5, we achieve the average node degree (the number of immediate neighbors) of 2.1, 6.1, 11.5, 18.6, 27.2, 37.5, 49.3, and 62.6, respectively.
- The results are averaged over 1,000 runs.

5.2.2 Validation Results

The results are reported in Fig. 3. Based on this study, we draw the following conclusions:

- The ER model does not give accurate results when the node degree is below a certain threshold, as shown in Fig. 3a. As a matter of fact, when the node degree is less than 6, the whole network tends to become disconnected in simulation, which is consistent with [27].
- The results of the ER model approach those of the simulation when the node degree becomes greater, as illustrated in Figs. 3b, 3c, 3d, 3e, 3f, 3g, and 3h.
- The higher the node degree, the more accurate the ER model. When the transmission range $R \geq 3.0$, as shown in Figs. 3e, 3f, 3g, and 3h, the difference of the results obtained from the ER model and those of the simulation is less than 7 percent. In the best case, as shown in Fig. 3h, the maximum difference between the results of the simulation and the analysis is less than 3.5 percent.
- The ER model is accurate and suitable for sensor networks that are densely deployed.

6 Performance Analysis

In this section, we evaluate the performance of iPAK in terms of storage requirement, key sharing probability between neighboring sensors, resilience against node-capture attacks, and computation and communication overheads. We will conduct both theoretical analysis and simulation studies. This evaluation approach is consistent with the previous work in [5], [6], [7], and so forth. In the following sections, we describe our simulation settings.

6.1 Settings

- There are N_w worker sensors. In the simulation, N_w is set to be 1,000.
- The number of service sensors, denoted as N_s, is determined by $\rho * N_w/\lambda$, where ρ is a parameter to compensate for the nonuniformity of the deployment. In ideal cases, when the deployment of worker sensors is a perfect uniform, $\rho = 1.0$ suffices to ensure that a service node serves λ worker nodes. In our simulation, ρ is set to 1.0, 1.5, 2.0, 2.5, and 3.0.

d_t \ t / Node degree	1	2	3	4
13	13	28	44	60
31	31	75	122	168

Fig. 4. d_t versus t when the node degree is 13 or 31.

- All service and worker sensors are randomly and independently deployed over an area of A, where A is set to be either 10×10 or 15×15, mimicking a high-density network and a medium-density network, respectively. The corresponding average node degrees are 31 and 13.
- The transmission range of all service and worker sensors is set to be 1 unit.
- The security parameter λ is set to be either 50 or 100 to be consistent with those in [7] and [14].
- The forwarding bound T_0 can be 1, 2, or 3.
- All simulation results are obtained by averaging over 100 runs.

6.2 Storage Requirements

The average number of keying pairs stored in a single worker sensor, denoted as τ, depends on the node degree, the forwarding bound T_0, and λ.

Let S_i be an arbitrary service node, where $i = 1, 2, \cdots, N_s$. By applying the ER model derived in Section 5, we can easily compute d_t, the expected number of t-hop sensors covered by S_i. Fig. 4 reports the results of d_t for $t = 1$, 2, 3, and 4. Therefore, S_i will distribute the keying information to $\sum_{t=1}^{T_0} d_t$ nodes in total on the average. We have

$$\tau = \frac{N_s}{N_w} \sum_{t=1}^{T_0} d_t = \frac{\rho}{\lambda} \sum_{t=1}^{T_0} d_t. \tag{18}$$

Fig. 5 illustrates the relationship of τ versus ρ for a different forwarding bound T_0. It can be easily observed that τ grows as T_0 grows because a worker sensor can detect more service nodes for keying pair acquisition. For the same reason, τ increases as ρ increases. Furthermore, the higher the node degree, the higher the τ value because more worker nodes can be covered by a service node on the average.

In Blom's key space model [12], a keying pair takes $2 * (\lambda + 1)$ units of memory, since its private share is a row of D, and its public share is a column of G, where (D, G) is a λ-secure key space. Therefore, each worker sensor needs $m = \tau(2(\lambda + 1))$ units of memory on average to store the keying information.

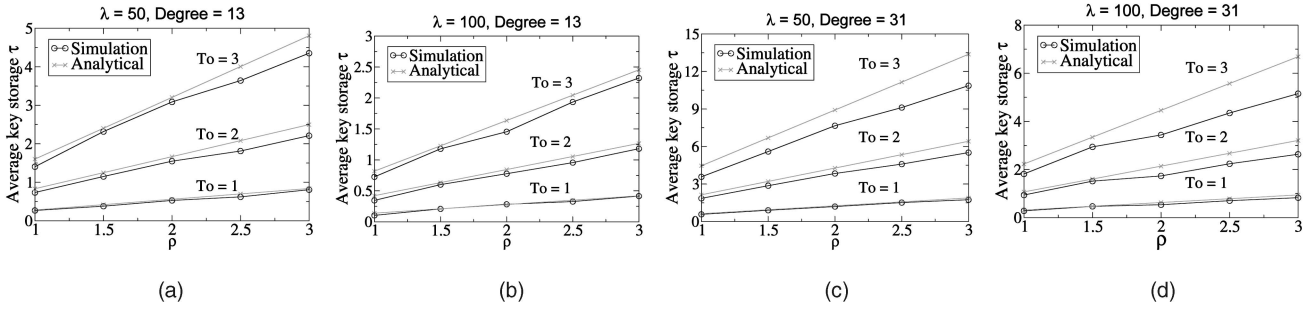


Fig. 5. Average storage τ at worker sensors: $\rho = \frac{\lambda * N_s}{N_w}$.

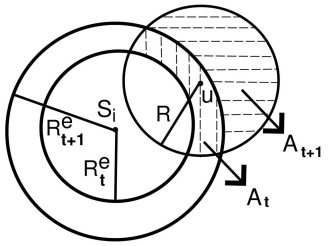


Fig. 6. Case E_1.

6.3 Keysharing Probability

In this section, we study the key-sharing probability between two neighboring sensors.

Let E_0 be the event that two worker sensors, say, u and v, are immediate neighbors. If u gets a keying pair from a service node S_i, then we say that u *is associated with* S_i. If both u and v are associated with S_i, then one of the following cases must happen:

- *Case* E_1. u is a t-hop neighbor of S_i, where $t = 1, 2, 3, \cdots, (T_0 - 1)$, and v is a t-hop neighbor (in the region A_t in Fig. 6) or a $(t+1)$-hop neighbor (in the region A_{t+1} in Fig. 6) of S_i.
- *Case* E_2. Both u and v are the T_0-hop neighbors of S_i.

Therefore, the probability P_{s_i} that both u and v are associated with the service node S_i can be expressed as

$$P_{s_i} = Pr[E_1|E_0] + Pr[E_2|E_0]. \tag{19}$$

According to the ER model proposed in Section 5.1, the probability that u is a t-hop neighbor of S_i is $\frac{d_t}{N_w}$, where d_t is the expected number of t-hop neighbors of S_i. Let p_{A_t} be the probability that v falls into the region A_t. We have

$$Pr[E_1|E_0] = \sum_{t=1}^{T_0-1} \left(\frac{d_t}{N_w} \left(p_{A_t} + p_{A_{t+1}} \right) \right), \tag{20}$$

where p_{A_t} can be computed based on the region A_t.

Similarly, we obtain

$$Pr[E_2|E_0] = \frac{d_{T_0}}{N_w} p_{A_{T_0}}. \tag{21}$$

Therefore, the key-sharing probability between neighboring worker sensors, denoted by P_{local}, can be regulated by

$$P_{local} = 1 - (1 - P_{s_i})^{N_s}. \tag{22}$$

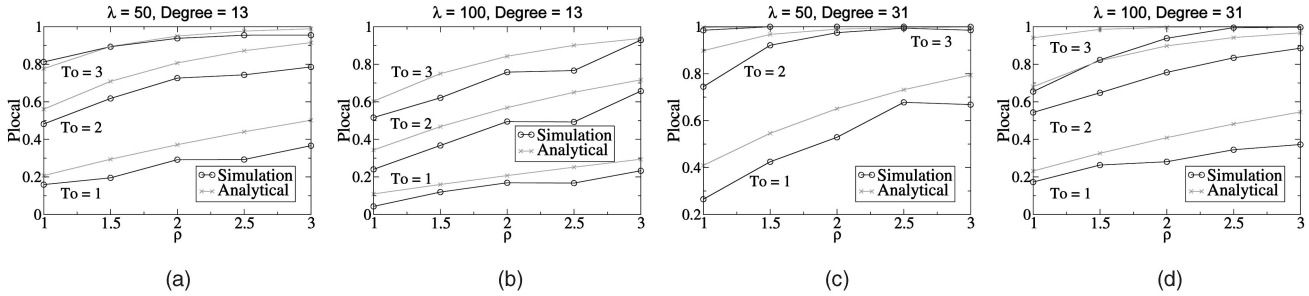


Fig. 7. Keysharing probability P_{local} : $\rho = \lambda * N_s/N_w$.

Fig. 7 demonstrates the relationship between P_{local} and ρ under a different forwarding bound T_0. Similar to the case of storage overhead, increasing T_0 or ρ improves P_{local} because of the availability of a larger number of service sensors from which a worker node requests a keying pair. Moreover, a higher node degree leads to a higher P_{local} because more neighbors are associated with the same service node.

By combining the analytical results in Section 6.2 with this study, we obtain the relationship between P_{local} and τ in iPAK. As reported in Fig. 8, a worker node can establish pairwise keys with over 90 percent of its immediate neighbors if it is associated with about two service nodes on average. If it is associated with a little more than one service node, then P_{local} approaches 70 percent. If a worker node is associated with more than five service nodes, then it can securely communicate with almost all its neighbors.

iPAK is superior compared with the two random key spaces schemes [7], [14], the two most related works in which a sensor is preloaded with keyshares from τ key spaces randomly selected from a key space pool of size ω. Since each service node in iPAK contains a key space, N_s in iPAK is equivalent to the ω in [7] and [14]. In our comparison study, the tuple of (ω, τ) for the random key spaces schemes [7], [14] is set to be (10, 1), (20, 2), (30, 3), (40, 4), (50, 5), or (60, 6), whereas the (N_s, τ) pair for iPAK is set to be (10, 1.1), (20, 2.2), (30, 3.2), (40, 4.3), (50, 5.4), or (60, 6.5). Fig. 8 manifests that the random key spaces schemes [7], [14] require the preloading of six keying pairs in order to achieve a 50 percent key-sharing probability between neighbors.

This difference is attributed to the fact that the keying pairs from a single service node are distributed to the vicinity of the service node in iPAK, whereas in [7] and [14], they may reside in any sensor at any location. Note that

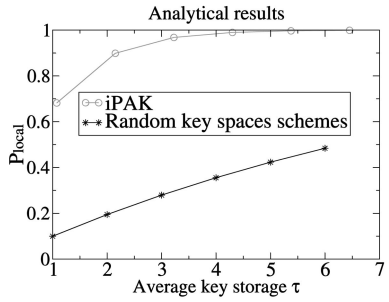


Fig. 8. P_{local} versus τ.

P_{local} in iPAK is solely determined by the triplet (T_0, d_1, ρ), which has nothing to do with the network size N_w. Therefore, our scheme scales well to large sensor networks. On the other hand, ω and τ in the two random key spaces schemes [7], [14] must grow with the network size for better resilience and key-sharing probability.

6.4 Resilience against Node-Capture Attacks

In this section, we study the resilience of iPAK against node-capture attacks. In our security model, a captured node releases all stored information. Let N_c denote the number of sensors that have been captured. Since Blom's key space [12] is λ-secure, N_c has to be at least $\lambda + 1$ in order for one key space to be broken.

We consider two types of attacks:

- **Oblivious attack**. The compromised nodes are independently and randomly selected from the entire deployment region.
- **Smart attack**. All compromised nodes reside in a circular region A_c, with a radius

$$R_c = \sqrt{N_c/(\pi * N_w/A)}.$$

The center of A_c is randomly selected from the deployment region A.[5]

Let P_{1-kc} be the probability that an arbitrary key space is compromised. In other words, P_{1-kc} is the probability that at least $\lambda + 1$ compromised worker sensors are associated with the same service sensor, say, S_a. Let p_{a-kc} denote the probability that a compromised worker node carries security information from S_a. We have

$$P_{1-kc} = \sum_{j=\lambda+1}^{N_c} \binom{N_c}{j} p_{a-kc}^{j} (1 - p_{a-kc})^{N_c - j}. \quad (23)$$

Therefore, the probability of at least one key space being compromised, denoted as P_{kc}, can be expressed as

$$P_{kc} = 1 - (1 - P_{1-kc})^{Ns}. \quad (24)$$

Next, we will study p_{a-kc} for each case.

Oblivious attack. From the ER model derived in Section 6.2, S_a provides keying pairs to $\sum_{t=1}^{T_0} d_t$ number of worker sensors on average. Thus, we have $p_{a-kc} = \frac{\sum_{t=1}^{T_0} d_t}{N_w}$.

5. If A_c is not an ideal disk due to the boundary effect, we increase R_c until N_c is reached.

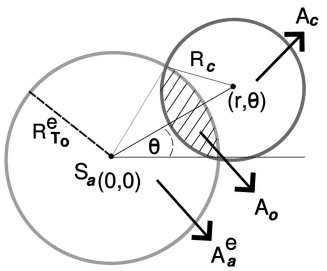


Fig. 9. The N_c captured nodes reside in the region A_c. Those nodes that fall into the overlapping region A_o obtain keying information from the service node S_a.

Smart attack. For simplicity, we assume that the service sensor S_a is located at (0, 0). Based on the ER model, S_a can cover a circular region, denoted as A_a^e (see Fig. 9), with an ER $R_{T_0}^e$. Assume that the center of the compromised area is located at (r, θ).

If the overlapping region of A_a^e and A_c, denoted as A_o in Fig. 9, contains more than $\lambda + 1$ worker nodes, then the key space S_a will be compromised. In other words, the area of A_o should be at least $\frac{(\lambda+1)*A}{N_w}$. Therefore, p_{a-kc} is regulated by

$$p_{a-kc} = \begin{cases} 1, & \text{if} \quad A_o \geq \frac{(\lambda+1)*A}{N_w} \\ 0, & \text{if} \quad 0 < A_o < \frac{(\lambda+1)*A}{N_w}. \end{cases} \tag{25}$$

In this study, we fix $\lambda = 50$. Note that similar results can be obtained for the case of $\lambda = 100$. Based on Fig. 4, we choose $T_0 = 3$ when the node degree is 13 and $T_0 = 2$ when the node degree is 31 such that, in general, a service node covers more than $\lambda + 1$ worker sensors.

Fig. 10 shows the results when node-capture attacks are launched. It is clear that P_{kc} increases with N_c for both oblivious attacks and smart attacks. Furthermore, the values of P_{kc} obtained from the simulation are closer to those of analysis under a higher node degree. This is because, the denser the network, the less the boundary effects. Besides, analytical results are higher compared to the simulation results in general because each run of our simulation randomly selects the center of A_c in the deployment region, whereas our theoretic analysis assumes A_c to be an ideal disk in any case.

For oblivious attacks, we observe that it is possible to break one key space with about 300 sensors captured. For smart attacks, on the other hand, compromising 300 sensors can almost certainly break at least one key space, as reported in Figs. 10c and 10d. Actually, when the number of compromised sensors reaches $\lambda + 1 = 51$, P_{kc} becomes nonnegligible, indicating that a smart attacker can relatively easily break a key space with less effort. There is a dramatic mismatch of P_{kc} between the simulation and the analysis when N_c is 51. The smaller value of P_{kc} obtained from the simulation results arises from the fact that a number of service sensors close to the boundary provide keying information to less than λ number of worker sensors and, therefore, they can never be broken no matter how many worker sensors are captured.

In addition, P_{kc} in the simulation is higher than that in the analysis when $N_c \geq 100$, as shown in Fig. 10c. This is caused by the fact that, when A_c is not an ideal disk, we increase R_c in order to capture N_c. In our theoretical analysis, however, A_c is always an ideal disk covering exactly N_c nodes.

6.5 Computation Overhead

The computation overhead of an arbitrary worker sensor mainly comes from two parts:

- **Secure-channel establishment**. In order to set up the secure channel with a service sensor, a worker sensor needs to perform Rabin's encryption once, which requires one modular multiplication (squaring) [26]. This computation occurs exactly once for each piece of keying information. As indicated in Section 6.2, each worker node obtains τ keying pairs on average. Therefore, the computation overhead for secure-channel establishment involves τ modular multiplications for each worker sensor.
- **Shared-key computation**. The computation of a shared key based on Blom's key space model [12] requires $\lambda + 1$ modular multiplications. To achieve a perfect local connectivity ($P_{local} = 1$), each worker sensor needs to conduct $(\lambda + 1) * d_1$ modular multiplications, where d_1 is the total number of one-hop neighbors.

Due to the fact that τ is usually small in our scheme, the dominant computation overhead lies in the shared-key calculation. Therefore, the computation overhead of our scheme is similar to that of [7]. Since service nodes are sacrificers, their computation overhead is not considered.

6.6 Communication Overhead

The communication overhead of a worker sensor also arises from two sources: the secret-key exchange and the shared-key discovery. Note that the communication overheads of

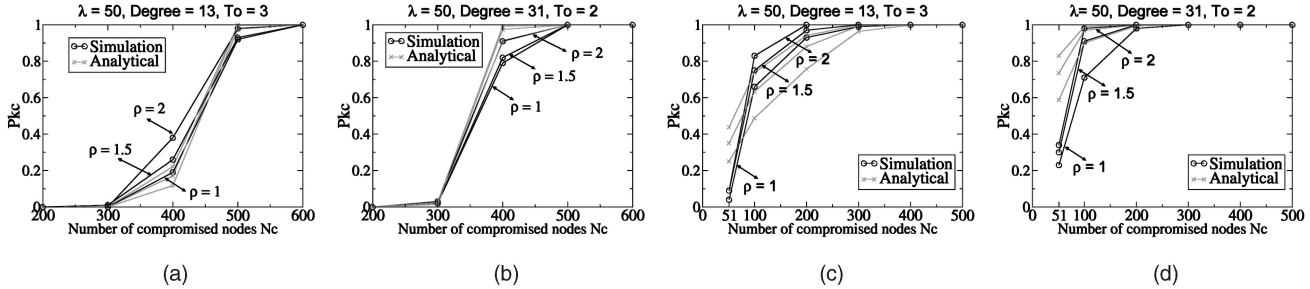


Fig. 10. Resilience against node-capture attacks. (a) and (b) Oblivious attacks. (c) and (d) Smart attacks.

the service sensors are not considered because they are sacrificers. In the following, we derive the average communication overhead of an arbitrary worker sensor:

- **Secret-key exchange**. A worker sensor is required to relay the existence announcements of nearby service sensors. Since an announcement is broadcast to T_0 hops, each worker sensor relays $(\sum_{t=1}^{T_0-1} d_t)\frac{N_s}{N_w}$ messages on average. Assume that this procedure computes a route between a worker sensor and the corresponding service sensor that it is associated with via a protocol similar to the route discovery in Ad Hoc On-Demand Distance Vector Routing (AODV) [28] or Dynamic Source Routing (DSR) [29]. This route is employed for secure-channel establishment and keying information acquisition.

  Therefore, each message from a t-hop worker sensor needs to be broadcasted t times before reaching the target service node. Thus, the communication overhead of secure-channel establishment for each worker sensor is estimated by $(\sum_{t=1}^{T_0} d_t \cdot t)\frac{N_s}{N_w}$. With a similar justification, each worker sensors needs to relay $(\sum_{t=1}^{T_0} d_t \cdot (t-1))\frac{N_s}{N_w}$ messages on average for keying information acquisition.
- **Shared-key discovery**. Since each worker sensor broadcasts the tuple $< \text{key space id}, \text{public share} >$ once for each key space that it is associated with, the average communication overhead is approximated by τ. Note that no path key is sought, since iPAK achieves a satisfying key-sharing probability already. This is another dramatic improvement compared to the two random spaces schemes in [7] and [14].

7 Conclusions

The design of iPAK targets large-scale WSNs with constrained resources (battery, memory, CPU, and so forth). In iPAK, worker sensors bear no key space information before deployment. They acquire keying pairs from service sensors in the neighborhood after deployment. To the best of our knowledge, this is the first localized key bootstrapping algorithm for shared-key establishment. The "in situ" property of iPAK significantly improves its scalability and greatly reduces the storage overhead of worker sensors. Furthermore, the probability of key-sharing in iPAK is much higher compared to those in [7] and [14] under the same storage constraint. Moreover, the introduction of the computationally asymmetric channel shifts the heavy computation overhead of Rabin's decryption [26] to service sensors, conserving the resources of worker sensors. iPAK is more favorable when high-power service nodes are available in a heterogeneous sensor network.

Acknowledgments

This research is supported by the US National Science Foundation Grant CCF-0627322.

References

[1] B.C. Neuman and T. Ts'o, "Kerberos: An Authentication Service for Computer Networks," *IEEE Comm. Magazine,* vol. 32, no. 9, pp. 33-38, 1994.

[2] W. Diffie and M.E. Hellman, "New Directions in Cryptography," *IEEE Trans. Information Theory,* vol. IT-22, no. 6, pp. 644-654, 1976.

[3] B. Atwood, B. Warneke, and K. Pister, "Preliminary Circuits for Smart Dust," *Proc. Southwest Symp. Mixed-Signal Design (SSMSD '00),* pp. 87-92, Feb. 2000.

[4] "Crossbow Mica2 Series (mpr4x0)," http://www.xbow.com/Products/productsdetails.aspx?sid=72, 2007.

[5] L. Eschenauer and V.D. Gligor, "A Key-Management Scheme for Distributed Sensor Networks," *Proc. Ninth ACM Conf. Computer and Comm. Security (CCS '02),* pp. 41-47, 2002.

[6] H. Chan, A. Perrig, and D. Song, "Random Key Predistribution Schemes for Sensor Networks," *Proc. 2003 IEEE Symp. Security and Privacy (SP '03),* p. 197, 2003.

[7] W. Du, J. Deng, Y.S. Han, and P.K. Varshney, "A Pairwise Key Predistribution Scheme for Wireless Sensor Networks," *Proc. 10th ACM Conf. Computer and Comm. Security (CCS '03),* pp. 42-51, 2003.

[8] W. Du, J. Deng, Y.S. Han, S. Chen, and P. Varshney, "A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge," *Proc. IEEE INFOCOM,* pp. 586-597, Mar. 2004.

[9] D. Liu, P. Ning, and W. Du, "Group-Based Key Predistribution in Wireless Sensor Networks," *Proc. Fourth ACM Workshop Wireless Security (WiSe '05),* pp. 11-20, 2005.

[10] L. Zhou, J. Ni, and C.V. Ravishankar, "Efficient Key Establishment for Group-Based Wireless Sensor Deployments," *Proc. Fourth ACM Workshop Wireless Security (WiSe '05),* pp. 1-10, 2005.

[11] S.A. Çamtepe and B. Yener, "Key Distribution Mechanisms for Wireless Sensor Networks: A Survey," Technical Report TR 05-07, Rensselaer Polytechnic Inst., Mar. 2005.

[12] R. Blom, "An Optimal Class of Symmetric Key Generation Systems," *Proc. Workshop Advances in Cryptology: Theory and Application of Cryptographic Techniques (EUROCRYPT '84),* pp. 335-338, 1985.

[13] H. Chan and A. Perrig, "PIKE: Peer Intermediaries for Key Establishment in Sensor Networks," *Proc. IEEE INFOCOM '05,* Mar. 2005.

[14] D. Liu and P. Ning, "Establishing Pairwise Keys in Distributed Sensor Networks," *Proc. 10th ACM Conf. Computer and Comm. Security (CCS '03),* pp. 52-61, 2003.

[15] D. Liu, P. Ning, and R. Li, "Establishing Pairwise Keys in Distributed Sensor Networks," *ACM Trans. Information Systems Security,* vol. 8, no. 1, pp. 41-77, 2005.

[16] C. Blundo, A.D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly Secure Key Distribution for Dynamic Conferences," *Proc. 12th Ann. Int'l Cryptology Conf. Advances in Cryptology (Crypto '92),* pp. 471-486, 1993.

[17] D. Liu and P. Ning, "Location-Based Pairwise Key Establishments for Static Sensor Networks," *Proc. First ACM Workshop Security of Ad Hoc and Sensor Networks,* pp. 72-82, 2003.

[18] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Location-Based Compromise-Tolerant Security Mechanisms for Wireless Sensor Networks," *IEEE J. Selected Areas in Comm.,* special issue on security in wireless ad hoc networks, vol. 24, no. 2, pp. 247-260, 2006.

[19] K. Ren, K. Zeng, and W. Lou, "A New Approach for Random Key Predistribution in Large-Scale Wireless Sensor Networks: Research Articles," *Wireless Comm. Mobile Computing,* vol. 6, no. 3, pp. 307-318, 2006.

[20] A. Perrig, R. Szewczyk, J.D. Tygar, V. Wen, and D.E. Culler, "SPINS: Security Protocols for Sensor Networks," *Wireless Networks,* vol. 8, no. 5, pp. 521-534, 2002.

[21] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks," *Proc. ACM Conf. Computer and Comm. Security,* pp. 62-72, 2003.

[22] E. Shi and A. Perrig, "Designing Secure Sensor Networks," *Wireless Comm. Magazine,* vol. 11, no. 6, pp. 38-43, Dec. 2004.

[23] W. Du, R. Wang, and P. Ning, "An Efficient Scheme for Authenticating Public Keys in Sensor Networks," *Proc. ACM MobiHoc '05,* pp. 58-67, 2005.

[24] R.L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Comm. ACM,* vol. 26, no. 1, pp. 96-99, 1983.

[25] R.J. Anderson, H. Chan, and A. Perrig, "Key Infection: Smart Trust for Smart Dust," *Proc. Int'l Conf. Network Protocol,* pp. 206-215, 2004.

[26] M.O. Rabin, "Digitalized Signatures and Public-Key Functions as Intractable as Factorization," technical report, 1979.

[27] L. Kleinrock and J.A. Silvester, "Optimum Transmission Radii in Packet Radio Networks or Why Six Is a Magic Number," *Proc. Nat'l Telecomm. Conf.*, pp. 4.3.1-4.3.5, Dec. 1978.
[28] C.E. Perkins and E.M. Royer, "Ad Hoc On-Demand Distance Vector Routing," *Proc. Second IEEE Workshop Mobile Computer Systems and Applications*, p. 90, 1999.
[29] D.B. Johnson, D.A. Maltz, and J. Broch, "DSR: The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks," *Ad Hoc Networking*, C. Perkins, ed., Addison-Wesley, chapter 5, pp. 139-172, 2001, http://monarch.cs.rice.edu/monarch-papers/dsr-chapter00.ps.

Liran Ma received the BS degree in electrical engineering from Hunan University, Changsha, China, in 1999 and the MS degree in communication and information systems from Northern Jiaotong University, Beijing, China, in 2003. He is currently working toward the PhD degree in computer science at George Washington University. His research activity focuses on the areas of ad hoc and sensor networks. His research interests include wireless networking, security, distributed systems, and approximation algorithm design. He is a student member of the IEEE.

Xiuzhen Cheng received the MS and PhD degrees in computer science from the University of Minnesota, Twin Cities, in 2000 and 2002, respectively. She is an assistant professor in the Department of Computer Science, George Washington University. Her current research interests include wireless and mobile computing, sensor networks, wireless security, statistical pattern recognition, approximation algorithm design and analysis, and computational medicine. She is an editor of the *International Journal on Ad Hoc and Ubiquitous Computing* and the *International Journal of Sensor Networks*. She has served as a TPC member for various professional conferences such as IEEE INFOCOM '05, IEEE Conference on Mobile Ad Hoc and Sensor Systems (MASS) '04 and '05, International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine) '04 and '05, IEEE Vehicular Technology Conference (VTC) '03, and so forth. She received the US National Science Foundation (NSF) Faculty Early Career Development (CAREER) Award in 2004. She is a member of the IEEE.

Fang Liu received the BS degree from the University of Science and Technology of China in 2000 and the MS degree from the Chinese Academy of Sciences in 2003, both in computer science. Since Fall 2003, she has been a PhD student at the Department of Computer Science, George Washington University. Her current research interests include wireless security, localization and tracking, and data mining and information retrieval in sensor networks. She is a student member of the IEEE.

Fengguang An received the BS and MS degrees from the National University of Defense Technology of China in 1993 and 1998, respectively. He has been a doctoral student at the Institute of Computing Technology, Chinese Academy of Sciences, since September 2002. His current research interests include key management and secure localization in sensor networks and network security in general.

Jose Rivera received the BS degree in individualized studies (computer science) from Virginia State University and the MS degree in computer science from George Washington University. He is working toward the PhD degree in computer science at George Washington University. He is currently assigned to the Army Chief Information Office (CIO/G6), Army Architecture Integration Cell (AAIC), as a program manager of the Army Service-Oriented Architecture Foundation. His research interests include ad hoc and sensor networks, wireless architecture, mobile security, Web services security, cryptography, and smart card and RFID security and architecture integration and development.