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

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

Abstract

Wireless Sensor Networks (WSNs) are characterized by resource constraints and large scalability. Many applications of WSN 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 efficiency. Nevertheless, establishing a shared key for communicating parties is a challenging problem. In this paper, we propose and analyze an in-situ Pairwise Key bootstrapping scheme (iPAK) for large-scale WSNs. Our theoretical analysis and simulation study demonstrate that iPAK can achieve high key-sharing probability between neighboring sensors and strong resilience against node capture attacks at the cost of a low storage overhead.

Index Terms

Wireless sensor networks, in-situ key establishment, key pre-distribution, Security.

L. Ma, X. Cheng and F. Liu are with the George Washington University.
F. An is with the Institute of Computing Technology, Chinese Academy of the Sciences.
J. Rivera is with the US Army Chief Information Office.
I. INTRODUCTION

As Wireless Sensor Networks (WSNs) approach towards a 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 (TTP), key agreement, and key pre-distribution. TTP-based schemes rely on a trusted server or servers organized in a hierarchical structure for key agreement between nodes (e.g., 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 Diffie-Hellman key agreement [2]) set up pairwise keys through successive message exchange in a secure manner using asymmetric cryptography. Their applicability is mainly barricated by the limited computation and communication capabilities within sensor devices [3]. For example, the MICA2 Berkeley mote has a 8-bit, 7.3828MHz Atmega 128L processor with 4kB SRAM and 128KB ROM [4].

Key pre-distribution schemes (KPS), 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 pre-loaded 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 literature [5]–[10].

To compensate for the unpredictability of the network topology prior to deployment, KPS requires a large amount of keying information to be pre-loaded in order to achieve 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 usage of the limited memory in sensors make the current KPS schemes scale poorly to very large networks.

In this paper, we propose an in-situ PAirwise Key bootstrapping scheme, termed as iPAK, to facilitate shared key establishment between neighboring sensors. The proposed scheme aims at achieving 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 along the direction of in-situ key computation instead of keying information pre-distribution, 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 the iPAK can achieve very high key sharing probability with low storage overhead.
3) In addition, we introduce a novel analytical model for estimating the number of multi-hop 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 II. The assumptions and background knowledge are presented in Section III. We propose our in-situ key management scheme, iPAK, in Section IV. An Effective Radius (ER) model
is derived and evaluated in Section V. This model is employed for performance evaluation of iPak in Section VI. Finally, we conclude our paper in Section VII.

II. 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 pre-distribution 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 key pre-distribution scheme 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] is 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], $\omega$ key spaces instead of one key space are pre-computed and each sensor stores the private/public shares from $\tau$ key spaces. These $\tau$ key spaces are randomly selected from the $\omega$ 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 random key pre-distribution in [5] with Blom’s method. It has better resilience against node capture attacks with a reasonable amount of storage and communication overheads. We will elaborate Blom’s scheme in next section.

Key pre-distribution 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 pre-loaded 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 pre-distribution results in a serious storage wastage because much of the pre-loaded information may never be used during the lifetime of the sensor. Consequently, the scalability of key pre-distribution is poor since the amount of required security information to be pre-loaded increases with the network size.

To improve scalability, a deployment-knowledge-based key management approach was proposed in [8]. In this scheme, multiple deployment points are identified in a sensor network. For each deployment point, a key space is pre-computed 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 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]–[7]. PIKE [13] proposes to exploit trusted intermediaries for shared key establishment between nodes. It is shown that both communication and storage overheads scale sub-linearly 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 pre-distribution protocol [16]. Location-aware key establishment schemes are proposed in [17] and [18] for better resilience against security attacks. Most recently, [19] proposes to use 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 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 ACM Wise 2005. 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 up the key-sharing probability. Compared to [8], [9], [10] release the strong requirement on deployment knowledge with a tradeoff of higher communication overhead.

As pointed out by [22], [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 pre-load 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]–[7], etc.).

The nice features of iPAK include: (i) iPAK is a truly localized scheme because the keying information is only distributed to the worker sensors in vicinity after deployment, and therefore it has no limitation on the scalability concerning the network size. (ii) iPAK introduces a reasonable amount of storage overhead in worker sensors but achieves a high key-sharing probability between neighbors. (iii) Because computation-intensive operations have been shifted to service nodes, the battery powers of worker sensors can be conserved.

III. PRELIMINARIES

A. 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. The connotation of $M$ will become clear latter. $G$ 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 $A$, i.e., $A = (D \cdot G)^T$. $A$ is private too, with each row called a private share. Since $D$ is symmetric, $A \cdot G$ is symmetric as well. If we let $K = (k_{ij}) = A \cdot G$, we have $k_{ij} = k_{ji}$, where $k_{ij}$ is the element at the $i$th row and the $j$th column of matrix $K$, $i,j = 1,2,\ldots,M$.

The basic idea of Blom’s scheme is to use $k_{ij}$ as the secret key shared by node $i$ and node $j$.

The $i$th column of $G$, a public share, and the $i$th row of $D$, a private share, form the $i$th 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 ensures the so-called $\lambda$-secure property, which means that the network should be perfectly secure as long as no more than $\lambda$ keying pairs are exposed. This requires that any $\lambda + 1$ columns of $G$ be linearly independent.

B. Rabin’s Scheme

Rabin’s scheme 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 $p \equiv q \equiv 3 \mod 4$. $(p,q)$ is the private key while $n = p \cdot q$ is the public key.

Encryption: Let $P_i$ be the plain text that is represented as an integer in $Z_n$. Then the cipher text $c = P_i^2 \mod n$.

Decryption: Since $p \equiv q \equiv 3 \mod 4$, we have

$$m_p = c^{p+1} \mod p$$

$q$ must be large enough to accommodate a cryptographic key.
and
\[ m_q = c^{q+1} \mod q. \]

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 \(+r, -r, +s, -s\) can be obtained:
\[
\begin{align*}
  r &= (y_p \cdot p \cdot m_q + y_q \cdot q \cdot m_p) \mod n \quad (1) \\
  -r &= n - r \quad (2) \\
  s &= (y_p \cdot p \cdot m_q - y_q \cdot q \cdot m_p) \mod n \quad (3) \\
  -s &= n - s \quad (4)
\end{align*}
\]

Note that Rabin’s encryption requires only one squaring, which is several hundreds of times faster than RSA [24]. But 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 pre-specified redundancy, a bit string \( B \), is appended to the plain text before encryption for ambiguity resolution.

C. 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 while 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, etc.) in order to complete the key bootstrapping procedure before 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 Subsection III-A) and two large primes \( p \) and \( q \) (as explained in Subsection III-B), computed by a super computer 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.

IV. 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 pre-loading 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 vicinity.
A. Key Space pre-loading Phase

During the pre-deployment phase, each service node pre-loads with a key space $(D, G)$ as defined in Blom’s scheme, an integer $n$, and two large primes $p$ and $q$ such that $n = p \times q$ (see Section III). Keying shares from the key space are to be disseminated to worker sensors in 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 in iPAK is mainly motivated by its $\lambda$-collusion property. To break a key space in Blom’s scheme, at least $\lambda + 1$ number of worker sensors associated with the same service sensor must be captured. This makes a successful attack much harder. If random keys (e.g. a key ring) instead of keying information as defined in Blom’s scheme is disseminated from a service sensor to a worker sensor, the decryption of such a message releases all the keys assigned to the worker sensor.

B. 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 Subsection, 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 while service nodes can die after their duty is complete, cryptographic algorithms that shift large amount of the computation overhead to the service node are preferred. Based on this consideration, iPAK adopts Rabin’s public cryptosystem.

1) Secure Session Establishment: Based on Rabin’s scheme described in Subsection III-B, 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 $T_0$-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 $E_n(K_s || B) = (K_s || B)^2 \mod n$, where $B$ is a predefined bit pattern to resolve the ambiguity in Rabin’s decryption, 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.

Note that in this protocol each worker sensor executes one Rabin’s encryption for each service sensor from which an existence announcement is received while the computationally intensive decryption of Rabin’s system is performed only at service sensors. This can conserve 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 $k_0$ (Subsection III-C) and service node ids to secure the disseminated keying share, we believe this method is not strong enough in a harsh environment. However, the security of Rabin’s scheme is comparable to that of RSA since both rely on the hardness of factoring large primes.

2) Keying Pair Acquisition: The secure channel established based on Rabin’s public cryptosystem (Subsection IV-B.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 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 $i$, 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.
C. Shared Key Discovery Phase

After obtaining keying pairs from service sensors in 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 if they have acquired keying information from at least one common service node. Blom’s \( \lambda \)-secure key management scheme \([12]\) has been well-tailored for light-weight sensor networks by \([7]\).

V. THE EFFECTIVE RADIUS MODEL

To study the performance of iPAK, we need the Effective Radius (ER) model presented in this section to identify the expected number of \( t \)-hop neighbors\(^3\) a sensor may have in a randomly distributed network, where \( t = 1, 2, \cdots \). Note that this is a non-trivial problem due to the randomness of the node positions.

A. Model Derivation

We assume that there are \( N \) nodes randomly distributed in the deployment region with an area of \( A \). Furthermore, we assume 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 to derive \( d_t \), where \( t = 2, 3, \cdots \), the expected number of \( t \)-hop neighbors (in the shortest path) an arbitrary node may have? In the next, 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 Euclidian distance to \( u \) is denoted by \( r \). If \( r \leq R \), \( v \) is a one-hop neighbor of \( u \); On the other hand, if \( r > 2R \), \( v \) cannot be reached by \( u \) via two hops. Therefore, \( v \) is a two-hop neighbor of \( u \) if and only if (i) \( R < r \leq 2R \) and (ii) \( u \) 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 \( A_1^u \) be the overlapping area of \( u \) and \( v \), as shown in Fig. 1(a). We have

\[
Pr[E_2|E_1] = 1 - (1 - \frac{A_1^u}{\pi R^2})^{d_1},
\]

where \( A_1^u \) is defined as

\[
A_1^u = 2R^2 \arccos(\frac{r}{2R}) - \frac{r}{2} \sqrt{4R^2 - r^2}.
\]

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

\[
P_2^u = \int_{0}^{2\pi} d\theta \int_{R}^{2R} (1 - (1 - \frac{A_1^u}{\pi R^2})^{d_1}) r dr
\]

As a result, the number of \( u \)'s two-hop neighbors, denoted by \( d_2 \), follows

\[
d_2 = (3\pi R^2) \frac{N}{A} P_2^u
\]

Directly computing the exact number of \( t \)-hop neighbors is a difficult problem when \( t \) is larger than two. Therefore, we introduce the Effective Radius (ER) model to facilitate this computation. Let \( D_t \) be the expected number of neighbors that are at most \( t \)-hop away. In our ER model, the effective radius of the \( t \)-hop coverage of a node \( u \) is defined as the radius of a virtual disk centered at \( u \) that can cover \( D_t \) number of nodes.

\(^2\)The broadcasting for all associated key spaces can be combined into a larger message.

\(^3\)in the shortest path.

\(^4\)A one-hop neighbor can also be called an immediate neighbor.
For example, Fig. 1(b) depicts the effective radius for the case of two hops. In this figure, the virtual disk centered at \( u \) with a radius of \( R_u^e \) covers \( d_1 + d_2 \) number of nodes in total. These covered nodes include all the one-hop neighbors (labelled with plus signs), a number of two-hop neighbors (labelled with dots), and a few other nodes (labelled with star signs). Note that the number of two-hop neighbors that fall out of the virtual disk equals to the number of nodes that can’t be reached from \( u \) within two hops but fall into this virtual disk.

Accordingly, the effective radius \( R_u^e \) for the two-hop case can be calculated as follows.

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

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

\[
R_u^e = \sqrt{R^2 + 3R^2 P_A^1}. \tag{10}
\]

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 \( R \) while \( u \)’s transmission range is set to be \( R_u^e \). In other words, the virtual disk with a radius \( R_u^e \) centered at \( u \) represents \( u \)’s two-hop coverage. In this case, \( v \) is a three-hop neighbor of \( u \) if and only if (i) \( R_u^e < r \leq R_u^e + R \) and (ii) \( u \)’s two-hop virtual disk covers at least one of \( v \)’s immediate neighbors, where \( r \) is the Euclidian 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 \( R_u^e < r \leq R_u^e + R \):

\[
P_3^a = \frac{\int_0^{2\pi} \int_0^{R_u^e + R} \left( 1 - \frac{A_u^1}{\pi R^2} \right)^2 r dr d\theta}{\pi((R_u^e + R)^2 - (R_u^e)^2)},
\]

where \( A_u^1 \), the overlapping area covered by both \( u \) and \( v \) as shown in Fig 2(a), is regulated by Eq. (12).

\[
A_u^1 = R^2 \arccos\left( \frac{r^2 + R^2 - (R_u^e)^2}{2rR} \right)
+ (R_u^e)^2 \arccos\left( \frac{r^2 + (R_u^e)^2 - R^2}{2rR_u^e} \right)
- \frac{1}{2} \sqrt{4r^2(R_u^e)^2 - (r^2 - R^2 + (R_u^e)^2)^2}.
\]

Thus, the number of \( u \)’s three-hop neighbors can be approximated by

\[
d_3 = \pi((R_u^e + R)^2 - (R_u^e)^2) \frac{N}{A} P_3^a.
\]

Fig. 1. Overlapping regions.
And the equivalent radius for three hops is

\[ R_3^e = \sqrt{(R_2^e)^2 + ((R_2^e + R)^2 - (R_2^e)^2)P_3^a}. \]  \hspace{1cm} (14)

By recursively applying this procedure, we get the probability \( P_t^a \) of \( v \) being \( u \)'s \( t \)-hop neighbor, the expected number of \( u \)'s \( t \)-hop neighbors \( d_t \), and the equivalent radius \( R_t^e \) as follows.

\[ P_t^a = \frac{\int_{2\pi}^{0} d\theta \left( \frac{(R_{t-1}^e + R)}{\pi(R_{t-1}^e)^2} \right) (1 - \frac{1}{\pi(R_{t-1}^e)^2} d_t)}{\pi((R_{t-1}^e + R)^2 - (R_{t-1}^e)^2)}, \]  \hspace{1cm} (15)

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

and

\[ R_t^e = \sqrt{(R_{t-1}^e)^2 + ((R_{t-1}^e + R)^2 - (R_{t-1}^e)^2)P_t^a}. \]  \hspace{1cm} (17)

Our ER model will be validated through simulation study in the following section.

**B. Evaluation of the Effective Radius Model**

1) **Validation Settings:**

- There are 6400 nodes randomly distributed in a field with length and width of 80 × 80. Therefore, the node density \( \phi = 1 \), i.e., 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 1000 runs.

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 certain threshold, as shown in Fig. 3(a). 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 towards those of the simulation when the node degree becomes larger, as illustrated by Fig. 3(b) to Fig. 3(h).
- The higher the node degree, the more accurate the ER model. When the transmission range \( R \geq 3.0 \), as shown in Fig. 3(e) to Fig. 3(h), the difference of the results obtained from the ER model and those of the simulation is less than 7%. In the best case, as shown in Fig. 3(h), the maximum difference between the results of simulation and analysis is less than 3.5%.
- The ER model is accurate and suitable for sensor networks that is densely deployed.
VI. 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 previous works in [5]–[7], etc. In the following, we describe our simulation settings.

A. Settings

- There are $N_w$ worker sensors. In simulation, $N_w$ is set to be 1000.
- The number of service sensors, denoted as $N_s$, is determined by $\rho * N_w / \lambda$, where $\rho$ is a parameter to compensate for the non-uniformity of the deployment. In the ideal case, when the deployment of worker sensors is a perfect uniform, $\rho = 1.0$ suffices to ensure that a service node serves $\lambda$ worker nodes. In our simulation, $\rho$ is set to 1.0, 1.5, 2.0, 2.5, and 3.0.
- All service and worker sensors are randomly and independently deployed over an area of $A$, where $A$ is set to be either $10 \times 10$ or $15 \times 15$, mimicking a high density 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 $\lambda$ is set to be either 50 or 100, to be consistent with that in [7], [14].
- The forwarding bound $T_0$ can be 1, 2, or 3.
- All simulation results are obtained by averaging over 100 runs.

B. Storage Requirements

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

Let $S_i$ be an arbitrary service node, where $i = 1, 2, \cdots, N_s$. By applying the ER model derived in Section V, 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 keying information to $\sum_{t=1}^{T_0} d_t$ nodes in total on the average. We have

$$\tau = \frac{N_s}{N_w} \sum_{t=1}^{T_0} d_t = \frac{\rho}{\lambda} \sum_{t=1}^{T_0} d_t.$$  (18)
**Fig. 4.** $d_t$ vs. $t$ when the node degree is 13 or 31.

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

![Graphs showing relationship between $\tau$ and $\rho$ for different forwarding bound $T_0$.](image)

In Blom’s key space model, a keying pair takes $2 \times (\lambda + 1)$ 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 $\lambda$-secure key space. Therefore, each worker sensor needs $m = \tau(2(\lambda + 1))$ units of memory on the average to store the keying information.

**C. Key-Sharing Probability**

In this subsection, 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 $S_i$, we say $u$ is associated with $S_i$. If both $u$ and $v$ are associated with $S_i$, one of the following two cases must happen,

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

![Case $E_1$.](image)
Therefore, the probability $P_{s_i}$ that both $u$ and $v$ are associated with the service node $S_i$ can be expressed as

$$P_{s_i} = Pr[E_1|E_0] + Pr[E_2|E_0].$$  \hspace{1cm} (19)

According to the ER model proposed in Subsection V-A, the probability that $u$ is a $t$-hop neighbor of $S_i$ is $d_t/N_w$, 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

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

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

Similarly, we obtain

$$Pr[E_2|E_0] = \frac{d_{T_0}}{N_w} p_{A_{T_0}}.$$  \hspace{1cm} (21)

Therefore, the key-sharing probability between neighboring worker sensors, denoted by $P_{local}$, can be regulated by Eq. (22).

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

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

Fig. 7 demonstrates the relationship between $P_{local}$ and $\rho$ under different forwarding bound $T_0$. Similar to the case of storage overhead, increasing $T_0$ or $\rho$ 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.

Combining the analytical results from Subsection VI-B with this study, we obtain the relationship between $P_{local}$ and $\tau$ in iPAK. As reported in Fig. 8, a worker node can establish pairwise keys with over 90% of its immediate neighbors if it is associated with about 2 service nodes on the average. If it is associated with a little more than one service node, $P_{local}$ approaches to 70%. If a worker node is associated with more than 5 service nodes, 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 work, in which a sensor is pre-loaded with keying shares from $\tau$ key spaces randomly selected from a key space pool of size $\omega$. Since each service node in iPAK contains a key space, $N_s$ in iPAK is equivalent to the $\omega$ in [7], [14]. In our comparison study, the tuple of ($\omega$, $\tau$) 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), while the ($N_s$, $\tau$) 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 pre-loading of 6 keying pairs in order to achieve a 50% key sharing probability between neighbors.

This difference is attributed to the fact that the keying pairs from a single service node is distributed to the vicinity of the service node in iPAK while in [7], [14] they may reside in any sensor at any
location. Note that the $P_{local}$ in iPAK is solely determined by the triplet $(T_0, d_1, \rho)$, which has nothing to do with the network size $N_w$. Therefore, our scheme scales well to large sensor networks. On the other hand, $\omega$ and $\tau$ in the two random key spaces schemes [7], [14] must grow with the network size for better resilience and key-sharing probability.

D. Resilience Against Node Capture Attacks

In this subsection, 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 is $\lambda$-secure, $N_c$ has to be at least $\lambda+1$ in order for one key space to be broken.

We consider the following 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, $P_{1-ke}$ is the probability that at least $\lambda + 1$ compromised worker sensors are associated with the same service sensor, say $S_a$. Let $p_{a-ke}$ denote the probability that a compromised worker node carries security information from $S_a$. We have

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

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

$$P_{kc} = 1 - (1 - P_{1-ke})^{N_s}.$$  \hfill (24)

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

**Oblivious attack:**

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

**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 $A_c^o$ (see Fig. 9), with an effective radius $R_c^{T_0}$. Assume the center of the compromised area is located at $(r, \theta)$.

\footnote{If $A_c$ is not an ideal disk due to the boundary effect, we increase $R_c$ until $N_c$ is reached.}
If the overlapping region of $A_c^c$ and $A$, denoted as $A_o$ in Fig. 9, contains more than $\lambda + 1$ worker nodes, 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

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

Fig. 9. The $N_e$ captured nodes reside in the region $A_e$. Those nodes that falls into the overlapping region $A_o$ obtain keying information from the service node $S_a$.

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 node degree is 13, and $T_0 = 2$ when 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 $N_e$ for both oblivious attacks and smart attacks. Furthermore, the values of $P_{kc}$ obtained from 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 simulation results in general because each run of our simulation randomly selects the center of $A_e$ in the deployment region, while our theoretic analysis assumes $A_e$ 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 Fig. 10(c) and 10(d). Actually when the number of compromised sensors reaches $\lambda + 1 = 51$, $P_{kc}$ becomes non-negligible, indicating that a smart attacker can relatively easily break a key space with less effort. There is a dramatic mismatch of $P_{kc}$ between simulation and analysis when $N_e$ 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 $\lambda$ number of worker sensors, and therefore they can never be broken no matter how many worker sensors are captured.
In addition, $P_{ke}$ of simulation is higher than that of analysis when $N_c \geq 100$ in Fig. 10(c). This is caused by the fact that when $A_e$ is not an ideal disk, we increase the $R_e$ in order to capture $N_c$. In our theoretical analysis, however, $A_e$ is always an ideal disk covering exactly $N_c$ nodes.

E. Computation Overhead

The computation overhead of an arbitrary worker sensor mainly comes from two parts: the secure channel establishment and the shared key calculation.

1. **Secure channel establishment**: In order to setup 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 Subsection VI-B, each worker node obtains $\tau$ keying pairs on the average. Therefore, the computation overhead for secure channel establishment involves $\tau$ modular multiplications for each worker sensor.

2. **Shared key computation**: The computation of a shared key based on Blom’s key space model 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 1-hop neighbors.

Due to the fact that $\tau$ 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 sacrifices, their computation overhead is not considered.

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

1. **Secret key exchange**: A worker sensor is required to relay the existence announcements of nearby service sensors. Since an announcement is broadcasted to $T_0$ hops, each worker sensor relays $\frac{(\sum_{i=1}^{T_0} d_i) N_s}{N_w}$ messages on the average. Assume this procedure computes a route between a worker sensor and the corresponding service sensor it is associated with via a protocol similar to the route discovery in AODV [28] or 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 $\frac{(\sum_{i=1}^{T_0} d_i \cdot t) N_s}{N_w}$. With a similar justification, each worker sensors needs to relay $\frac{(\sum_{i=1}^{T_0} d_i \cdot (t - 1)) N_s}{N_w}$ messages on the average for keying information acquisition.

2. **Shared key discovery**: Since each worker sensor broadcasts the tuple < key space id, public share > once for each key space it is associated with, the average communication overhead is approximated by $\tau$. 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], [14].

VII. CONCLUSIONS

The design of iPAK targets large-scale wireless sensor networks with constrained resources (battery, memory, CPU, and etc.). In iPAK, worker sensors bear no key space information before deployment. They acquire keying pairs from service sensors in the neighborhood after deployment. To our best 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 that of [7], [14] under the same storage constraint. Moreover, the introduction of the computationally asymmetric channel shifts the heavy computation overhead of Rabin’s decryption 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.
REFERENCES

Liran Ma received his B.S. degree in Electrical Engineering from Hunan University, Changsha, China, in 1999 and the M.S. degree in Communication and Information System from Northern Jiaotong University, Beijing, China, in 2003. He is currently working towards his Ph.D. degree in computer science at the 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.

Xizhou (Susan) Cheng is an Assistant Professor in the Department of Computer Science at the George Washington University. She received her MS and Ph.D degrees in Computer Science from the University of Minnesota - Twin Cities in 2000 and 2002, respectively. 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 for the International Journal on Ad Hoc and Ubiquitous Computing and the International Journal of Sensor Networks. Dr. Cheng has served as a TPC member for various professional conferences such as IEEE INFOCOM 05, IEEE MASS ’04, ’05, QShine’04,05, IEEE VTC’03, etc. She received the NSF CAREER Award in 2004.

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

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

Jose Rivera received his M.S. in Computer Science from the George Washington University and B.S. in Individualized Studies (Computer Science) from Virginia State University. He is working on his Ph.D. in Computer Science Student at the George Washington University. Manny is currently assigned to the Army Chief Information Office (CIO/G6), Army Architecture Integration Cell (AAIC) as a Program Manager, Army Service Oriented Architecture Foundation. His research interests are: Ad Hoc and Sensor Networks; Wireless Architecture, Mobile Security, Web Services Security, Cryptography, SmartCard and RFID Security and Architecture Integration & Development.