Group-Based Key Predistribution for Wireless Sensor Networks

DONGGANG LIU
The University of Texas at Arlington
PENG NING
North Carolina State University
and
WENLIANG DU
Syracuse University

Many key predistribution techniques have been developed recently to establish pairwise keys between sensor nodes in wireless sensor networks. To further improve these schemes, researchers have also proposed to take advantage of the sensors' expected locations and discovered locations to help the predistribution of the keying materials. However, in many cases, it is very difficult to deploy sensor nodes at their expected locations or guarantee the correct location discovery at sensor nodes in hostile environments. In this article, a *group-based deployment model* is developed to improve key predistribution. In this model, sensor nodes are only required to be deployed in groups. The critical observation in the article is that *the sensor nodes in the same group are usually close to each other after deployment*. This deployment model is practical; it greatly simplifies the deployment of sensor nodes, while still providing an opportunity to improve key predistribution. Specifically, the article presents a novel framework for improving key predistribution using the group-based deployment knowledge. This framework does not require the knowledge of the sensors' expected or discovered locations and is thus suitable for applications where it is difficult to deploy the sensor nodes at their expected locations or correctly estimate the sensors' locations after deployment. To seek practical key predistribution schemes, the article presents two efficient instantiations of this framework, a *hash key-based scheme* and a *polynomial-based scheme*. The evaluation shows that these two schemes are efficient

This work is supported by the U.S. Army Research Office (ARO) under grant W911NF-05-1-0247, and by the National Science Foundation (NSF) under grants CNS-0430223 and CNS-0430252.
A preliminary version of this article appeared in the *Proceedings of the 2005 ACM Workshop on Wireless Security* (WiSe 2005).

Authors' addresses: D. Liu, 300 Nedderman Hall, Department of Computer Science and Engineering, The University of Texas at Arlington, Arlingtonn, TX 76019; email: dliu@uta.edu; P. Ning, Department of Computer Science, North Carolina State University, Raleigh, NC 27695-8206; email: pning@ncsu.edu; W. Du, 3-114 CST Building, Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse, NY 13244; email: wedu@syr.edu.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or direct commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, or to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from the Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.
© 2008 ACM 1550-4859/2008/03-ART11 $5.00 DOI 10.1145/1340771.1340777 http://doi.acm.org/10.1145/1340771.1340777

and effective for pairwise key establishment in sensor networks; they can achieve much better performance than the previous key predistribution schemes when the sensor nodes are deployed in groups.

Categories and Subject Descriptors: C.2.0 [**Computer-Communication Networks**]: General—*Security and protection (e.g., firewalls)*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

General Terms: Security, Design, Algorithms

Additional Key Words and Phrases: Sensor networks, security, pairwise key establishment, key predistribution, group-based deployment

ACM Reference Format:

Liu, D., Ning, P., and Du, W. 2008. Group-based key predistribution for wireless sensor networks. ACM Trans. Sens. Netw. 4, 2, Article 11 (March 2008), 30 pages. DOI = 10.1145/1340771.1340777 http://doi.acm.org/10.1145/1340771.1340777

1. INTRODUCTION

Recent technological advances have made it possible to deploy wireless sensor networks consisting of a large number of low-cost, low-power, and multifunctional sensor nodes that communicate at short distances through wireless links [Akyildiz et al. 2002]. Such sensor networks are ideal candidates for a wide range of applications in military and civilian operations such as health monitoring, data acquisition in hazardous environments, and target tracking. The desirable features of wireless sensor networks have attracted many researchers to develop protocols and algorithms to support these various applications (e.g., [Perrig et al. 2001; Hill et al. 2000; Gay et al. 2003; Niculescu and Nath 2001; Akyildiz et al. 2002]).

Sensor networks (especially in military operations) are often deployed in hostile environments where enemies may be present. It is therefore critical to ensure the correct network operations as well as the integrity, availability, and at times confidentiality of the data collected by sensor nodes in hostile environments. However, providing security services for sensor networks is particularly challenging. In the first place, sensor nodes are usually resource-constrained, which makes it not practical to implement those well-studied but expensive algorithms on sensor nodes. Second, sensor nodes are usually deployed in an unattended manner. It is very difficult to physically protect the sensor nodes individually. A few sensor nodes could be captured by adversaries after deployment. Once a sensor node is captured, it can give way its secrets very quickly [Hartung et al. 2005]. As a result, any security algorithm has to be resilient against the compromised sensor nodes.

Key management is the cornerstone of many security services such as authentication and encryption. A critical task of key management is to establish a pairwise key between two sensor nodes in the network. Research seeking low-cost pairwise key establishment techniques that can survive node compromises in sensor networks was quite active in the past three, or four years, yielding several novel key predistribution schemes. To name a few, Eschenauer and Gligor [2002] proposed to distribute a random subset of keys from a key pool to every sensor node before deployment such that every pair of sensor nodes will have a certain probability of sharing at least one key after deployment. Chan et al. [2003] extended this scheme by having two sensor nodes share at least q predistributed keys to establish a pairwise key. Chan et al. [2003] also developed a random pairwise keys scheme. This scheme predistributes random pairwise keys between a sensor node and a random subset of other sensor nodes, and has the property that the compromise of sensor nodes does not lead to the compromise of any key shared *directly* between two noncompromised sensor nodes. To further enhance the security of key predistribution, Liu, Ning, Du, and colleagues independently developed two similar threshold-based schemes [Du et al. 2003; Liu and Ning 2003a]. Chan and Perrig [2005] also developed a protocol named PIKE for key establishment by using peer sensor nodes as trusted intermediaries.

However, due to the resource constraints (especially the limited battery power) on sensor nodes and the threat of compromised nodes, none of the above key management schemes can guarantee the security of the keying materials used for the communication between sensor nodes. It is always desirable to improve the security and performance of key management.

In many applications, long distance peer-to-peer secure communication between sensor nodes is rare. Even if it is needed, we can establish a key using a number of intermediate nodes if hop-by-hop encryption and authentication is available, and use this key to secure the long distance peer-to-peer communication. Thus one primary goal of secure communication is to provide authentication and/or encryption between neighbor sensor nodes. Therefore, the most important information that can benefit key predistribution is the knowledge about *which nodes are most likely to be the neighbors of each sensor node*. Once this information is available, we can predistribute pairwise keys for the pairs of sensor nodes that are likely to be neighbors. As a result, a sensor node will have a high probability of establishing a pairwise key with its neighbor.

Several techniques have been proposed recently to take advantage of the sensors' location information to improve key predistribution [Du et al. 2004; Liu and Ning 2003b, 2005; Huang et al. 2004; Yu and Guan 2005]. Some of them assume that *the locations of sensor nodes can be predetermined to a certain extent*, while some assume that *the locations of sensor nodes can be discovered after deployment*. However, in practice, it could be difficult, and sometimes impossible, to guarantee that the sensor nodes are deployed precisely at expected locations. Besides, the sensor's discovered location is usually not trustworthy in hostile environments [Liu et al. 2005; Lazos and Poovendran 2004; Ray et al. 2003; Li et al. 2005; Capkun and Hubaux 2005; Lazos et al. 2005].

In this article, we develop a group-based deployment model to improve the security and performance of key predistribution. In this model, the sensor nodes are required to be deployed in groups. A critical observation is that *the sensor nodes in the same group are usually close to each other after deployment*. We believe that this group-based deployment model is more practical than the aforementioned models since it requires less effort in the deployment of sensor nodes, while still providing an opportunity to improve key predistribution.

Specifically, we propose a novel *group-based key predistribution framework* based on this deployment model. Compared to the previous techniques for improving key predistribution, this framework has the following two advantages. First, the expected or discovered locations of sensor nodes are no longer required for key predistribution. This simplifies the deployment of sensor networks when it is hard to deploy sensor nodes close to their expected locations, and improves the security of key predistribution when it is hard to guarantee the correct location discovery. Second, this framework can be applied to improve any of the existing key predistribution schemes, while the previous techniques for improving key predistribution can only be used to improve certain types of key predistribution techniques.

To seek practical key predistribution techniques, we also develop two efficient instantiations in this framework, a *hash key-based scheme* and a *polynomial-based scheme*. These two schemes have a number of advantages over the previous key predistribution schemes. First, in these two schemes, the neighbor sensor nodes have a high probability of establishing direct keys with low storage overhead, and there is no additional communication overhead required for direct key establishment. Second, they are resilient to node compromise attacks even if there are a large number of compromised sensor nodes. Finally, they provide efficient alternatives to achieve better security and performance when it is difficult to deploy the sensor nodes close to their expected locations or correctly estimate the sensors' locations after deployment.

The remainder of this article is organized as follows. The next section introduces our group-based deployment model. Section 3 presents our framework. Section 4 describes our two efficient instantiations of the proposed framework. Section 5 provides the detailed analysis for the proposed approaches. Section 6 reviews related work on sensor network security. Section 7 concludes this article and points out possible future research directions.

2. GROUP-BASED DEPLOYMENT MODEL

When the sensor nodes are deployed in the target field, there are many factors that will affect where a sensor node will be finally deployed. For example, when a sensor node is dropped from an airplane, the final location of this sensor node will be at least affected by the location and the velocity of the airplane when the sensor node is dropped, and the wind speed. However, since the sensor nodes in the same group will be deployed together, it is very likely that they are affected in a similar way by the same set of factors. Therefore, the final locations of the nodes in the same group will be close to each other with a high probability after deployment.

Based on the above discussion, we present a practical deployment model in this section. In this model, the sensor nodes are only required to be deployed in groups. The deployment knowledge used to improve key predistribution is the observation that the sensor nodes in the same group are usually close to each other after deployment. This model in fact is more practical and requires less effort in the deployment of sensor nodes than the models where the sensor nodes need to be placed close to their expected locations. For example, a group

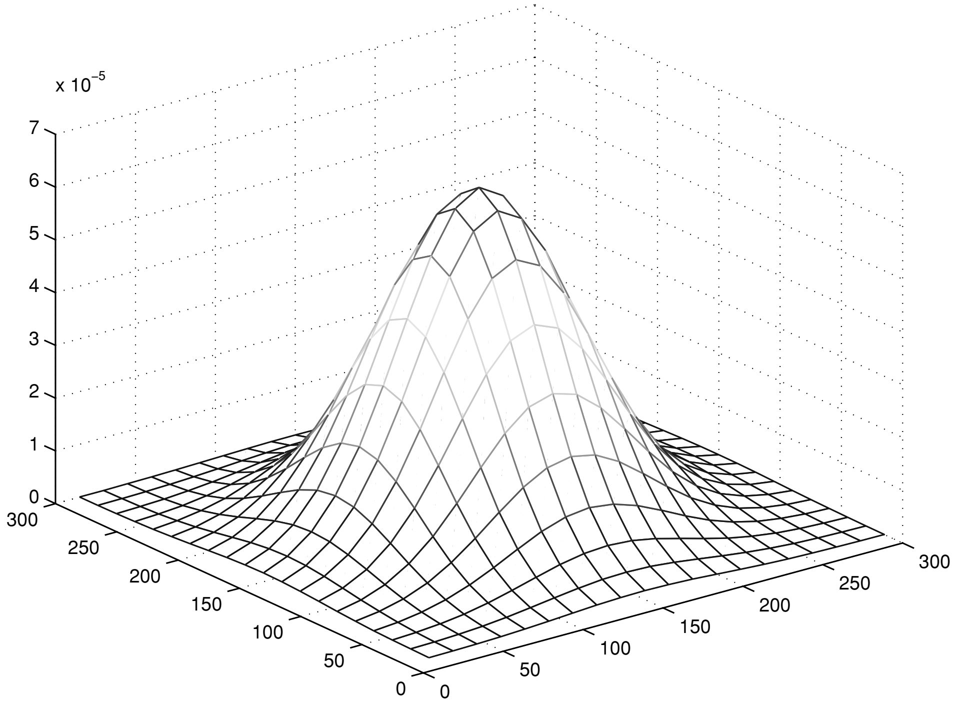

Fig. 1. Deployment distribution.

of sensor nodes may only need to be dropped from the airplane at the same time. For the sake of presentation, we call a group of sensor nodes that need to be deployed together as a *deployment group*.

We assume that the sensor nodes are static once they are deployed. We define the *resident point* of a sensor node as the point location where this sensor node finally resides. The sensors' resident points are generally different from each other even for the sensor nodes in the same group. However, in our later evaluation, we assume the resident points of the sensor nodes in the same group follow the same probability distribution function for simplicity, though the proposed method will still work under different distribution functions. The detailed description of the proposed deployment model is given below.

The sensor nodes to be deployed are divided into n deployment groups $\{G_i\}_{i=1,\ldots,n}$. We assume that these groups are evenly and independently deployed in a target field. The sensor nodes in the same deployment group G_i are deployed from the same place at the same time with the deployment index i. During the deployment of any group G_i, the resident point of any sensor node in this group follows a probability distribution function (pdf) $f_i(x, y)$, which we call the *deployment distribution* of group G_i. An example of the pdf $f_i(x, y)$ is a two-dimensional Gaussian distribution. Figure 1 illustrates a two-dimensional Gaussian distribution with center (150, 150).

The actual deployment distribution is affected by many factors. For simplicity, we model the deployment distribution as a Gaussian distribution (also called *normal distribution*) since it is widely studied and proved to be useful in practice. Although we only employ the Gaussian distribution in our evaluation, our methodology can be applied to other distributions as well.

Specifically, we assume that the deployment distribution for any node in the deployment group G_i follows a two-dimensional Gaussian distribution centered at a *deployment point* (x_i, y_i). *Different from the previous deployment models where the deployment points are predetermined [Du et al. 2004; Huang et al. 2004], we do not assume any prior knowledge of such deployment points.* In fact, we only assume *the existence of such deployment points*. The mean of the Gaussian distribution μ equals (x_i, y_i), and the pdf for any sensor node in group G_i is the following:

$$f_i(x, y) = \frac{1}{2\pi\sigma^2} e^{-[(x-x_i)^2+(y-y_i)^2]/2\sigma^2} = f(x - x_i, y - y_i),$$

where σ is the standard deviation, and $f(x, y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$.

3. A FRAMEWORK FOR GROUP-BASED KEY PREDISTRIBUTION

According to the deployment model discussed in the previous section, the sensor nodes in the same deployment group have high probability of being neighbors. To take advantage of this observation, the pairwise key predistribution techniques should at least benefit the sensor nodes in the same deployment group. Hence, we first employ an *in-group key predistribution* method, which enables the sensor nodes in the same deployment group to establish pairwise keys between each other with a high probability. To handle the pairwise key establishment between the sensor nodes in different deployment groups, we then employ a *cross-group key predistribution* method, which enables the selected sensor nodes in different deployment groups to establish pairwise keys and thus bridges different deployment groups together.

A key predistribution technique can usually be divided into three different phases, the *predistribution*, which specifies how to predistribute keying materials to each sensor node, the *direct key establishment*, which specifies how to establish a pairwise key shared between two sensor nodes *directly*, and the *path key establishment*, which specifies how to discover a sequence of sensor nodes to help two given sensor nodes to establish a temporary session key. A key established in the direct key establishment phase is called a *direct key*, while a key established in the path key establishment phase is called an *indirect key*.

For simplicity, we assume there are n equal-sized deployment groups with m sensor nodes in each of these groups. The description of our framework is given below. For simplicity, we omit the detail of the message format.

3.1 Predistribution

For each deployment group G_i, we need a key predistribution instance D_i to take care of the pairwise key establishment in G_i. For the sake of presentation, these key predistribution instances are called the *in-group (key predistribution) instances*. The in-group instance D_i can be the instance of any existing key predistribution technique.

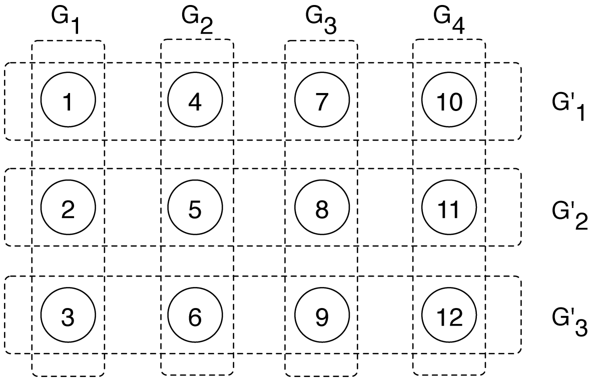

Fig. 2. Example of group construction.

To handle the pairwise key establishment between sensor nodes in different deployment groups, we construct m *cross groups* $\{G'_i\}_{i=1,\dots,m}$. The requirements on these cross groups are (1) each cross group includes exactly one sensor node from each deployment group, and (2) there are no common sensor nodes between any two different cross groups. In other words, for any i and j with $i \neq j$, we have $G'_i \cap G'_j = \phi$ and $|G'_i \cap G_j| = 1$. We also generate a key predistribution instance D'_i for the pairwise key establishment in every G'_i. These instances are called the *cross-group (key predistribution) instances*. By doing this, each cross group provides a potential link for any two deployment groups. Similarly, the cross-group instance D'_i can be the instance of any key predistribution technique.

In this article, we propose a simple way to construct the deployment groups and cross groups for pairwise key establishment. Basically, each deployment group G_i contains the sensor nodes with the IDs $\{(i-1)m+j\}_{j=1,\dots,m}$, while each cross group G'_i contains the sensor nodes with the IDs $\{i+(j-1)m\}_{j=1,\dots,n}$. By doing this, a sensor node can easily figure out which deployment group or cross group another sensor node belongs to based on the ID of the other node. Figure 2 shows an example of this group construction when $n = 4$ and $m = 3$. In this figure, G'_1 includes node 1, 4, 7, and 10; G'_2 includes node 2, 5, 8, and 11; and G'_3 includes node 3, 6, 9, and 12.

3.2 Direct Key Establishment

After the predistribution step, each sensor node has the keying materials for two key predistribution instances, an in-group instance and a cross-group instance. Hence, the direct key establishment between two neighbor sensor nodes is simple and direct. If they are in the same deployment group, for example, G_i, they can follow the direct key establishment of the in-group instance D_i. If they are not in the same deployment group but belong to the same cross group G'_j, they can follow the direct key establishment of the cross-group instance D'_j. To determine if two sensor nodes u and v are in the same deployment group or the same cross group, they only need to know the ID (either u or v) of the other party due to our group construction method. Specifically, if $\lfloor \frac{u}{m} \rfloor$ equals

$\lfloor \frac{v}{m} \rfloor$, they are in the same deployment group; if u mod m equals v mod m, they are in the same cross group. Clearly, there is not additional communication cost introduce by applying our framework.

3.3 Path Key Establishment

If two sensor nodes cannot establish a direct key, they have to go through the path key establishment to find a sequence of other sensor nodes to help them establish an indirect key. It is required that any pair of adjacent sensor nodes in this sequence can establish a direct key between them to make sure the security of key establishment. It is also assumed that every message between two sensor nodes is encrypted and authenticated by the direct key established between them.

Similar to the direct key establishment, if two sensor nodes are in the same deployment group G_i, they can follow the path key establishment in D_i. The indirect keys between the sensor nodes in the same group are called the *in-group indirect keys*.

When two sensor nodes belong to two different groups G_i and G_j, we use a different method to establish an indirect key. Basically, we need to find a "bridge" between these two deployment groups in order to establish a *cross-group indirect key*. A bridge between group G_i and G_j is defined as a pair of sensor nodes $\langle a, b\rangle$ ($a \in G_i$ and $b \in G_j$) that belong to the same cross group G'_k ($a, b \in G'_k$). A bridge is valid when the two sensor nodes involved in this bridge can establish a direct key.

According to the predistribution step, there are m potential bridges (one from each cross group) between every two deployment groups that can be used to establish an indirect key between sensor nodes in these two deployment groups. In addition, due to our group construction method, a sensor node can easily compute all possible bridges between any two deployment groups. Specifically, the possible bridges between group G_i and G_j are $\{\langle (i-1)m + k, \text{(j-1)m+k}\rangle\}_{k=1,\ldots,m}$. For example, as shown in Figure 2, there are three bridges between group G_1 and G_4: $\langle 1, 10\rangle$, $\langle 2, 11\rangle$, and $\langle 3, 12\rangle$. The path key establishment for the sensor nodes in different deployment groups thus works as follows.

(1) The source node u first tries the bridge involving itself to establish an indirect key with the destination node v. Assume this bridge is $\langle u, v'\rangle$. Node u first sends a request to v' if it can establish a direct key with v'. If node v' can also establish a (direct or indirect) key with the destination node v, node v' forwards this request to the destination node v to establish an indirect key.

(2) If the first step fails, node u will try the bridge involving the destination node v. Assume the bridge is $\langle u', v\rangle$. In this case, node u sends a request to node u' if it can establish a (direct or indirect) key with u'. If node u' can establish a direct key with node v, it forwards the request to the destination node v to establish an indirect key. Note that if nodes u and v are in the same cross group, this step can be skipped, since step 1 and step 2 compute the same bridge.

(3) When both of the above steps fail, node u has to try other bridges. Basically, it randomly chooses a bridge $\langle u', v' \rangle$ other than the above two, assuming u' is in the same deployment group with u, and v' is in the same deployment group with v. Node u then sends a request to u' if it can establish a (direct or indirect) key with u'. Once u' receives this request, it forwards the request to v' in the bridge if they share a direct key. If v' can establish a (direct or indirect) key with the destination node v, it forwards the request to node v to establish an indirect key.

We use the same example as in Figure 2 to illustrate the above algorithm. When node 1 wants to establish an indirect pairwise key with node 12, it first tries the bridge $\langle 1, 10 \rangle$. If this fails, it tries the bridge $\langle 3, 12 \rangle$. If both bridges fail, it needs to try the bridge $\langle 2, 11 \rangle$. If none of these bridges works, the path key establishment fails. In our later analysis, we will see that it is usually unlikely that none of those bridges works. When one of the bridges works, the two sensor nodes involved in this bridge will be used as the intermediate nodes by node 1 and node 12 to establish the indirect key.

In the above approach, the path key establishment in a cross-group instance has never been used. The reason is that the sensor nodes in a cross group usually spread over the entire deployment field, which makes it expensive to perform path key establishment in a cross group.

4. SEEKING EFFICIENT INSTANTIATIONS

In the proposed framework, as long as a key predistribution technique can provide pairwise key establishment between sensor nodes for a group of sensor nodes, it can be used as the basic building block to construct a group-based scheme. This implies that our framework can be applied to any existing key predistribution technique.

Since we have additional deployment knowledge in our framework, we are able to more precisely predict the possible neighbors for any sensor node than the original schemes. It is thus expected that the framework can improve the existing key predistribution schemes even if the expected or discovered locations are not available. Indeed, the preliminary version of this article have demonstrated that our proposed framework can improve the existing key predistribution schemes substantially. In this article, we are also interested in seeking more practical instantiations under the proposed framework.

4.1 Overview

The previous section has described the detail of our framework. There are two remaining issues need to be addressed properly to develop an efficient instantiation. The first issue is the technical detail of the in-group and cross-group key predistribution instances. We note that, in our framework, the size of deployment groups and the size of cross groups are usually much smaller than the network size. For small size groups, it is practical to apply those simple key predistribution schemes such as the basic polynomial-based scheme in [Blundo et al. 1993]. The reason is that a single sensor node can usually provide enough storage space for us to use very secure and efficient key predistribution schemes for a small number of sensor nodes. For example, for a group of 100 sensor nodes, we can assign a unique key between every pair of sensor nodes. This only occupies the storage space that is equivalent to 99 keys.

The second issue is the communication with the two given sensor nodes involved in the bridge identified in the path key establishment. In the proposed framework, we provide an efficient way for a sensor node to compute all the possible bridges by only looking at the IDs of the sensor nodes. However, it is still nontrivial to actually communicate with the two sensor nodes related to a given bridge. Theoretically, we are able to check every possible bridges as long as the routing protocol can support the communication between any pair of sensor nodes. However, in practice, it is not desirable to introduce too much communication overhead during the path key establishment. Besides, the routing protocol may not support the end-to-end communication between any pair of sensor nodes. We are thus more interested in the ideas that can discover a valid bridge by only contacting a few sensor nodes in the network.

With these in mind, we introduce two efficient instantiations in this subsection. For the first issue, we will look for key predistribution schemes that do not require additional communication overhead during the establishment of direct keys. This is particularly suitable for resource-constrained sensor networks since the wireless communication is one of the most expensive operations for wireless sensor networks [Perrig et al. 2001]. On the other hand, this will greatly simplify our path key establishment protocol as well as the implementation of our final group-based key predistribution technique. Indeed, it will not only save the communication overhead but also reduce the code size for the implementation. Therefore, from the whole-system point of view, we are able to allocate more storage space for keying materials, leading to a more secure and efficient group-based key predistribution technique. Additionally, the above idea is practical since every deployment group or cross group only has a small number of sensor nodes.

For the second issue, we propose a simple way to identify the working bridge between two sensor nodes. Specifically, every sensor node only needs to discover the list of the nearby sensor nodes that belong to the same deployment group as itself. Once the lists are available, two neighbor sensor nodes u and v only need to exchange their lists to identify a working bridge $\langle u', v' \rangle$ where u' and v' are actually close to the two sensor nodes u and v. For example, in Figure 2, suppose node 1 and node 12 want to establish an indirect key. Node 1 discovers that node 2 is in its neighborhood, and node 12 discovers that node 11 is in its neighborhood. In this case, after node 1 and node 12 exchanged their lists of the nearby sensor nodes, they can easily conclude that the bridge $\langle 2, 11 \rangle$ works perfectly for them. Additionally, we note that any two sensor nodes in the same deployment or cross group can always establish a direct key according to our early discussion for the requirement of the solutions for the first issue. This makes our solution for the second issue more efficient.

Note that in the above algorithm, two sensor nodes need to exchange the lists of neighbor nodes in the same group to identify the working bridge. The size of such list could be large, making it expensive for communication between sensor nodes. However, the size of such list is always no more than m. In addition, the IDs of the sensor nodes in a particular deployment group can be easily computed due to our group construction method. We can simply use a bit string to represent the list of neighbor nodes that belong to the same group. Hence, we only need $\frac{m}{8}$ bytes for such list. For example, when $m = 100$, we only need 13 bytes to represent the list. Clearly, the communication overhead introduced by such list will not be a problem for our method.

Based on the above discussion, we can see that our solution for the second issue is quite clear as long as the direct key between the sensor nodes in the same group can be established. Such a solution actually works for both of our later instantiations. Hence, the only remaining issue at this point is the in-group and cross-group key predistribution instances. Therefore, in our instantiations, we only explain the technical detail of these in-group and cross-group key predistribution instances.

4.2 Hash Key-Based Instantiation

For small groups, the simplest key predistribution method is to generate and assign a unique random key to every pair of sensor nodes. This scheme also provides *perfect security* guarantee in the presence of compromised sensor nodes, which means that the compromise of sensor nodes does not leak any information about the shared keys between noncompromised sensor nodes. However, in this article, we propose to take advantage of the one-way hash function to *reduce almost half of the storage overhead and evenly split it among sensor nodes*. The detail of the basic building block for our group-based instantiation is presented below.

Before deployment, every sensor node i is predistributed with a master key K_i, which is only known by the trusted central server (e.g., base station) and the node i. Let G be either a deployment group or a cross group. Assume that the node IDs in G have already been sorted in an ascendent order. For any sensor node $i \in G$, let $\text{Pos}(i)$ be the position of this node in the ordered group G. For any two nodes i and j ($\text{Pos}(i) < \text{Pos}(j)$) in this group, we check the value ($\text{Pos}(i) + \text{Pos}(j)$). If it is an odd value (($\text{Pos}(i) + \text{Pos}(j)) \bmod 2 = 1$), we will predistribute $H(K_i||j)$ to the node j; otherwise, we will predistribute $H(K_j||i)$ to the node i, where H is a one-way hash function. For example, as shown in Figure 2, node 1 and node 2 are in the same deployment group. In this case, we will simply predistribute the key $H(K_1||2)$ to node 2.

Because of our group construction method, we can easily calculate the positions of a sensor node in its deployment group and cross group, respectively. Specifically, given a sensor node ID i, we can easily determine that this node is in the deployment group with index $\lceil \frac{i}{m} \rceil$ and the cross group with index $((i - 1) \bmod m) + 1$. Therefore, in case of the deployment group, we have $\text{Pos}(i) = ((i - 1) \bmod m) + 1$; in the case of the cross group, we have $\text{Pos}(i) = \lceil \frac{i}{m} \rceil$.

Therefore, if a sensor node i wants to establish a key with node j after deployment, it can easily determine whether they are in the same group (either deployment group or cross group). If they are not in the same group, they will follow the path key establishment protocol that is discussed at the beginning of this section to establish an indirect key; otherwise, both of node i and node

j can easily figure out the direct key shared between them using the following algorithm:

—If $\text{Pos}(i) < \text{Pos}(j)$ and $(\text{Pos}(i) + \text{Pos}(j)) \bmod 2 = 1$, node i can compute the shared key $H(K_i||j)$, while node j has already been predistributed with this key.

—If $\text{Pos}(i) < \text{Pos}(j)$ and $(\text{Pos}(i) + \text{Pos}(j)) \bmod 2 = 0$, node i has the predistributed key $H(K_j||i)$, while node j can compute this key easily.

—If $\text{Pos}(i) > \text{Pos}(j)$ and $(\text{Pos}(i) + \text{Pos}(j)) \bmod 2 = 1$, node i has the predistributed key $H(K_j||i)$, while node j can compute this key easily.

—If $\text{Pos}(i) > \text{Pos}(j)$ and $(\text{Pos}(i) + \text{Pos}(j)) \bmod 2 = 0$, node i can compute the shared key $H(K_i||j)$, while node j has already been predistributed with this key.

From the above discussion, we can clearly see that a sensor node can immediately determine the direct key shared with another sensor node in the same group. There is no additional communication overhead. Since any two sensor nodes in the same group shared a unique key, there is no need to implement the path key establishment inside a group.

In addition, we can also see that every shared key between two sensor nodes is only known by these two related sensor nodes. Hence, the compromise of sensor nodes does not lead to the compromise of any direct key between two non-compromised sensor nodes. As a result, the above scheme guarantees the *perfect security property* in the presence of node compromise attacks.

Now let us estimate the storage space for the keying materials at sensor nodes. Consider any sensor node i in a deployment group G. There are $\text{Pos}(i)-1$ sensor nodes with smaller IDs and $n-\text{Pos}(i)$ sensor nodes with larger IDs. When $\text{Pos}(i)$ is an odd number, we can easily see that node i will get pre-distributed $\frac{\text{Pos}(i)-1}{2}$ keys for the sensor nodes with smaller IDs, and $\lfloor\frac{n-Pos(i)}{2}\rfloor$ keys for the sensor nodes with larger IDs. Therefore, the overall number of predistributed pairwise keys to node i for the deployment group G is

$$\frac{\text{Pos}(i)-1}{2} + \left\lfloor \frac{n-\text{Pos}(i)}{2} \right\rfloor \approx \frac{n}{2}.$$

Similarly, if $\text{Pos}(i)$ is even, the number of predistributed pairwise keys to node i is approximately $\frac{n}{2}$. When node i belongs to a cross group G', we can also estimate the number of predistributed pairwise keys as $\frac{m}{2}$ in a similar way. Thus the total number of pairwise keys that are predistributed to a sensor node for both of its deployment group and cross group is approximately $\frac{m+n}{2}$.

4.3 Polynomial-Based Instantiation

The idea presented in the previous subsection allows a sensor node to establish a shared key with any other sensor node in the same group easily. In addition, it also provides the perfect security guarantee. However, the hash key-based instantiation has the following two problems. First, the storage overhead for the predistributed pairwise keys increases linearly with number of groups $(n+m)$.

Although we can always allocate enough space for these keying materials, it is always desirable to allow the tradeoff between security and storage overhead given resource constraints. Second, for the hash key-based scheme, once the groups are constructed, it is nontrivial to further extend the network and deploy new groups of sensor nodes. As a result, it is also desirable to seek other instantiations where the network can be easily extended. This subsection provides an instantiation that meets these requirements.

In this instantiation, we use the basic polynomial-based pairwise key establishment in Blundo et al. [1993] as the basic building block of our group-based key predistribution. The resulting scheme is similar to the grid-based scheme [Liu and Ning 2003a]. However, it also differs from the grid-based scheme in that this instantiation exploits the node grouping idea to make sure that the sensor nodes sharing the same polynomial will be close to each other. This immediately provides some nice properties, as we will see later. Compared to the hash key-based scheme, this instantiation allows us to make tradeoffs between security and storage, while still provides an efficient way to establish a pairwise key between any two sensor nodes.

Specifically, for any group G (either a deployment group or a cross group), we generate a unique symmetric t-degree bivariate polynomial $f_g(x, y)$ with the property of $f_g(x, y) = f_g(y, x)$. Every sensor node $i \in G$ gets predistributed a *polynomial share* $f_g(i, x)$ by evaluating the bivariate polynomial $f_g(x, y)$ at $x = i$. It is assumed that the polynomial $f_g(x, y)$ is only known by a trusted server. To establish a pairwise key with node j, node i only needs to compute $f_g(i, j)$, which equals $f_g(j, i)$, the value that can be computed by node j as well. There is no additional communication involved.

For example, as shown in Figure 2, the deployment group G_1 includes nodes 1, 2, and 3. We generate a random 2-degree bivariate polynomial $f_{G_1}(x, y)$ with the property of $f_{G_1}(x, y) = f_{G_1}(y, x)$ and then predistribute $f_{G_1}(1, x)$, $f_{G_1}(2, x)$, and $f_{G_1}(3, x)$ to node 1, node 2, and node 3, respectively.

According to Blundo et al. [1993], this simple method can tolerate up to t compromised nodes in a group. In other words, the collusion of no more than t compromised nodes cannot break the shared key between any two noncompromised sensor nodes in the group. On the other hand, we also see that the overall storage overhead for key materials in the polynomial-based instantiation is $2t$, which is independent of the number of groups $n + m$ in the network. Clearly this allows further tradeoffs between the storage space and the security performance.

Note that the polynomial-based instantiation is similar to the grid-based scheme [Liu and Ning 2003a] or PIKE [Chan and Perrig 2005], especially when every row (or column) of the sensor nodes in the grid structure are deployed in the same group. However, the main contribution of this article is that *we exploit the possibility of using the deployment knowledge other than the expected or discovered locations to further improve key predistribution*. As we will see, our polynomial-based scheme can achieve much better performance than the grid-based scheme when the group-based deployment knowledge is available. On the other hand, our group construction generates small groups and thus allows us to build an efficient key predistribution for a large-scale network from a

Table I. Notations

n	Number of deployment groups
m	Number of the sensor nodes in the same deployment group
c	Number of compromised sensor nodes
p_d	Probability of having a direct key between two neighbors
p_i	Probability of having an indirect key between two neighbors
p_{cd}	Probability of a direct key being compromised
p_{ci}	Probability of an indirect key being compromised

simple and efficient key predistribution scheme that may only work well for a small network if used directly.

5. EVALUATION

During the evaluation, we assume the same number of deployment groups and cross groups in the network ($m = n$) for simplicity. The analysis will be focused on the probability of establishing keys between sensor nodes and the security of the direct and indirect keys in the presence of compromised sensor nodes. For convenience, we list those frequently used notations in our later analysis in Table I.

5.1 Overheads

This article provides a framework and two efficient instantiations to establish pairwise keys between sensor nodes. The overhead of using such keys in security protocols (e.g., encryption or authentication) depends on the real applications. Thus, in this article, we only focus on the overhead involved in establishing these keys. In particular, we focus on the overheads of the two instantiations.

As we discussed earlier, the storage overhead for the hash key-based instantiation is approximately n (since $m = n$), while the storage overhead for the polynomial-based instantiation is $2t$. In both instantiations, any two sensor nodes in the same deployment group or the same cross group can establish a direct key between each other, and there is no communication overhead involved in the establishment of the direct keys between sensor nodes.

When two sensor nodes are in different deployment groups and different cross groups, they need to establish an indirect key to protect the communication between them. According to the protocol description, the path key establishment may involve one intermediate node, which is either in the same deployment group as the source node or in the same deployment group as the destination node, or two intermediate nodes, where one is in the same deployment group as the source node and the other is in the same deployment group as the destination node. Hence, in practice, the communication overhead for path key establishment is limited in two deployment groups. In addition, every sensor node only needs to discover the list of the nearby sensor nodes in its local area for the path key establishment. This only incurs a small amount of communication overhead.

5.2 Establishing Direct Keys

Consider a particular sensor node u in the deployment group G_i at position (x', y'). Let A denote its *communication area* in which any other sensor node can directly communicate with node u. In this paper, we assume A is a circle centered at (x', y') with radius R, where R is the signal range of a sensor node. Thus the average number of the sensor nodes in any deployment group G_j, including G_i, that finally reside in A can be estimated as

$$n_{i,j}(x', y') = n \iint_A f(x - x_j, y - y_j)\, \mathrm{d}x\mathrm{d}y.$$

For any deployment group G_j other than G_i, we know that there is only one sensor node u' in G_j that shares the same cross group G'_k with node u. Thus the probability of this node u' being deployed in A can be estimated as $\frac{n_{i,j}(x',y')}{n}$. This indicates that, among all those sensor nodes deployed in A, the average number of sensor nodes that belong to the deployment groups other than G_i but share the same cross group G'_k with node u can be estimated as

$$n'_i(x', y') = \frac{\sum_{j=1, j\neq i}^{n} n_{i,j}(x', y')}{n}.$$

When sensor nodes are evenly distributed in the deployment field, it is possible to further simplify the above equation. Suppose the average number of sensor nodes in the communication range of a sensor node is n_A. We have $\sum_{j=1,j\neq i}^{n} n_{i,j}(x', y') = n_A - n_{i,i}(x', y')$. Thus

$$n'_i(x', y') = \frac{n_A - n_{i,i}(x', y')}{n}.$$

In the proposed two instantiations, two sensor nodes in the same group can always establish a direct key. Thus, the average number of the sensor nodes in A that can establish direct keys with node u can be estimated as $(n_{i,i}(x', y') + n'_i(x', y'))$. This means that the probability of u having direct keys with its neighbor nodes can be estimated as

$$p_i(x', y') = \frac{(n_{i,i}(x', y') + n'_i(x', y'))}{n_A}.$$

Hence, for any node in group G_i, the probability of having direct keys with its neighbor nodes can be estimated as

$$p_d = \iint_S f(x - x_i, y - x_i) p_i(x, y)\, \mathrm{d}x\mathrm{d}y,$$

where S denotes the entire deployment field.

p_d can also be used to estimate the probability of any node in any deployment group having a direct key with its neighbor node when S is an infinite field. For a given deployment field S, we simply configure the deployment point of G_i as its geometric centroid, and use the probability of a node in G_i having a direct key with its neighbor node to represent the probability of having a direct key between any two neighbor nodes.

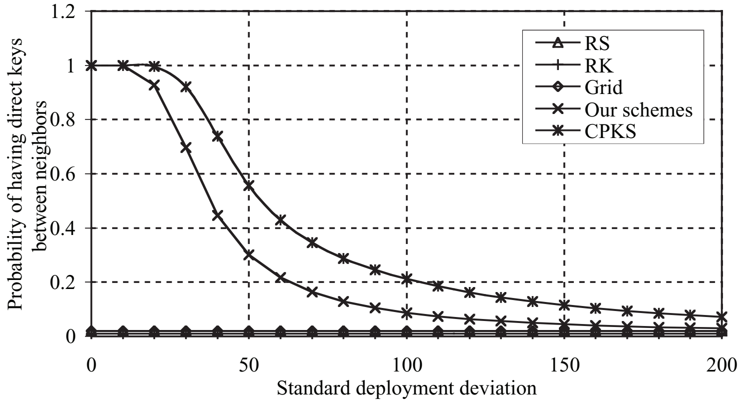

Fig. 3. Probability of having direct keys between neighbor nodes. “RS” represents the random subset assignment scheme. “RK” represents the random pairwise keys scheme. “Grid” represents the grid-based scheme. “CPKS” represents the closest pairwise keys scheme.

According to the above analysis, we can clearly see that both of our instantiations result in the same probability of establishing a direct key between two neighbor nodes in the network since the probability of having a direct key between two sensor nodes in the same group (either deployment group or cross group) for both instantiations is always 1.

During the evaluation, we use the following configuration throughout this article: we assume there are totally 10,000 sensor nodes deployed in a 1000 × 1000 square meters area. These sensor nodes are divided into 100 deployment groups with 100 sensor nodes in each group ($n = m = 100$). We assume sensor nodes are evenly distributed in the deployment field so that the probability of finding a node in each equal size region can be made approximately equal. In other words, the density of the sensor nodes is approximately one sensor node per 100 square meter. We always assume the signal range is $R = 40m$. Thus, there are $\frac{\pi 40^2}{100} \approx 50.27$ neighbors on average for any given sensor node.

Figure 3 illustrates the probability of establishing a direct key between two neighbor sensor nodes for different standard deviation σ of the deployment of sensor nodes. We can clearly see the high probability of establishing a direct key between two neighbor sensor nodes for a small value of σ. This indicates that our framework greatly improves the key predistribution as long as the sensor nodes in the same group are indeed close to each other after deployment.

However, the result also shows that the probability of establishing direct keys for our schemes decreases when the sensor deployment deviation σ increases. In particular, the benefits introduced by our group-based approaches as compared with existing solutions will diminish when the sensor nodes in the same group are not close to each other. This is also confirmed by our later study on the probability of establishing indirect keys and the security in hostile environments. This problem, however, it not surprising since our study is motivated by the location-based grouping of sensor nodes. When the group-based deployment model turns out to be ineffective, our approaches cannot do much to improve the performance. Nevertheless, our study indicates an interesting research direction where the performance of key management is improved by investing more efforts on the sensor deployment.

To better show the effectiveness of our framework, we also compare them with the existing key predistribution techniques such as the random pairwise keys scheme [Chan et al. 2003], the random subset assignment scheme [Liu and Ning 2003a], and the grid-based scheme [Liu and Ning 2003a]. In addition, we will also compare our schemes with the closest pairwise keys scheme (the extended version) [Liu and Ning 2003b], which exploits the knowledge of the expected locations of sensor nodes to facilitate key predistribution.

For the random subset assignment scheme and the grid-based scheme, we assume the same number of bivariate polynomials in the system and the same number of polynomial shares stored on each sensor node as the polynomial-based instantiation. Thus there are $m + n = 200$ bivariate polynomials in the polynomial pools for the random subset assignment scheme and the grid-based scheme. The random subset assignment scheme assigns the polynomial shares of two randomly selected polynomials from the pool to each sensor node. The grid-based scheme arranges 200 polynomials on a 100×100 grid, and each sensor node also gets two polynomial shares.

For the closest pairwise keys scheme, we assume the same storage overhead as our hash key-based instantiation. Hence, every sensor node can store 100 keys, which means that it shares keys with 200 sensor nodes that are closest to itself in terms of the expected locations. In addition, we also assume the sensor nodes are deployed in the same way as our group-based model. The deployment field is equally partitioned into 10×10 cells with one group for each cell. The expected location of every deployment group is the center of the corresponding cell, which is known by every sensor node.

Figure 3 shows the results of comparison. We can see that our schemes have a significantly higher probability of establishing a direct key between two neighbor sensor nodes than the random subset assignment scheme and the pairwise keys scheme. This indicates that our schemes can support larger sensor networks under the same network settings. Therefore, we believe that our framework can substantially improve the performance of existing key predistribution techniques.

Figure 3 also shows that the closest pairwise keys scheme can achieve better performance than the proposed schemes. The reason is that this scheme assume the knowledge of sensors' expected locations, which is not required by our schemes. As discussed before, the expected locations could be difficult to obtain in real-world scenarios. We therefore strongly believe that the group-based model is a more realistic deployment model. On the other hand, the performance of our schemes are actually not significantly worse than the closest pairwise keys scheme.

5.3 Establishing Indirect Keys

In the following, we estimate the probability of having an indirect key between two neighbor sensor nodes when they cannot establish a direct key. Note that the key predistribution instances in both instantiations guarantee that any two nodes in the same group can establish a direct key. Thus the need for establishing indirect keys only happens when two sensor nodes are in different deployment group and different cross group.

In this case, they have to find a valid bridge between these two deployment groups to establish an indirect key. Since there are m cross groups and the nodes in the same group can always establish a direct key in our proposed instantiations, there are totally m valid bridges. Any one of these bridges can help the source node to setup an indirect key with the destination node. On the other hand, the two sensor nodes involved in each of these bridges can be easily computed based on the IDs of the source and the destination node. Therefore, as long as the source or the destination node can communicate with both of the two nodes involved in any of these bridges, they can set up an indirect key. For example, as shown in Figure 2, assume node 1 wants to setup an indirect key with node 12. Suppose node 1 can communicate with node 2, and node 12 can communicate with node 11. In this case, the bridge $\langle 2, 11\rangle$ will be used to setup an indirect key between them since they can communicate with both nodes 2 and 11.

Therefore, an interesting question is to estimate the probability of establishing an indirect key if a sensor node only communicates with the sensor nodes that are not far away. For simplicity, we assume that a sensor node will only contact the sensor nodes that are no more than d hops away from itself for path key establishment, where d is a system parameter. For the ease of analysis, we simply consider the sensor nodes that are within $d \times R$ meters of a given sensor node u as the sensor nodes that are within d hops of node u, where R is the signal range of sensor nodes.

Consider a particular sensor node u in the deployment group G_i. Let us first estimate the average number of sensor nodes in G_i that are no more than $d \times R$ meters away from u. Let (x', y') be the final resident point of node u. Let B denote the area that are no more than $d \times R$ meters away from (x', y'). Thus, the average number of sensor nodes in group G_i that finally reside in B can be estimated as

$$n_d(x', y') = n \iint_B f(x - x_i, y - y_i)\,\mathrm{d}x\mathrm{d}y.$$

Therefore, on average, the number of sensor nodes in G_i that are no more than $d \times R$ meters away from node u can be estimated as

$$n_d = \iint_S f(x - x_i, y - x_i) n_d(x, y)\,\mathrm{d}x\mathrm{d}y,$$

where S denotes the entire deployment field.

Consider two neighbor nodes u and v in different deployment group G_i and G_j, respectively. The average number of other sensor nodes in group G_i that can be reached by either u or v is at least n_d, and the average number of other sensor nodes in group G_j that can be reached by either u or v is at least n_d. As long as these n_d nodes in G_i and n_d nodes in G_j form at least one bridge, the indirect key can be established.

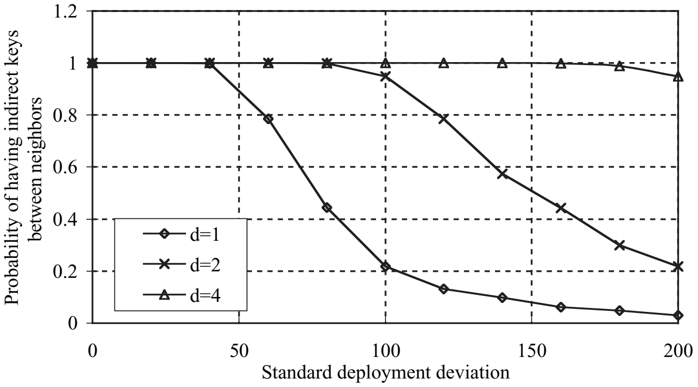

Fig. 4. Probability of having indirect keys between sensor nodes in different deployment groups.

Since node u and v are in different cross group, there are $m - 2$ bridges that involve two nodes and 2 bridges that only involve one node. Therefore, the probability of successfully finding one bridge can be estimated by

$$p_i = 1 - \left(1 - \frac{n_d}{n}\right)^2 \left(\frac{(n-2-n_d)!(n-2-n_d)!}{(n-2)!(n-2-2n_d)!}\right).$$

Figure 4 illustrates the probability of having an indirect key between two neighbor sensor nodes when they cannot set up a direct key. We can clearly see that a sensor node can easily set up an indirect key with its neighbors by only communicating with the sensor nodes in its local area. In addition, when the sensor nodes in the same deployment group are close to each other (leading to a small value of σ), a sensor node only needs to communicate with its neighbors to setup an indirect key with any other neighbor node with a very high probability.

Similarly to the analysis in the previous subsection, we also compare the performance of our schemes (the two instantiations) with the random pairwise keys scheme [Chan et al. 2003], the random subset assignment scheme [Liu and Ning 2003a] and the grid-based scheme [Liu and Ning 2003a]. We configure these schemes in the same way as we did before. Specifically, every sensor node in these schemes store the keying materials that is equivalent to 100 cryptographic keys. There are $m + n = 200$ bivariate polynomials in the polynomial pools for the random subset assignment scheme and the grid-based scheme. The random subset assignment scheme assigns the polynomial shares of two randomly selected polynomials from the pool to each sensor node. The grid-based scheme arranges 200 polynomials on a 100×100 grid, and each sensor node will also get two polynomial shares.

For all these schemes, we assume that a sensor node will only contact other sensor nodes that are no more than $2R$ meters away from itself for path key establishment. For the random subset assignment scheme [Liu and Ning 2003a], the random pairwise keys scheme [Chan et al. 2003], and the closest pairwise keys scheme, two sensor nodes always try to find an intermediate sensor node that can establish direct keys with both the source and the destination node. Hence the probability of having an indirect key can be estimated by $1-(1-p^2)^{n'}$,

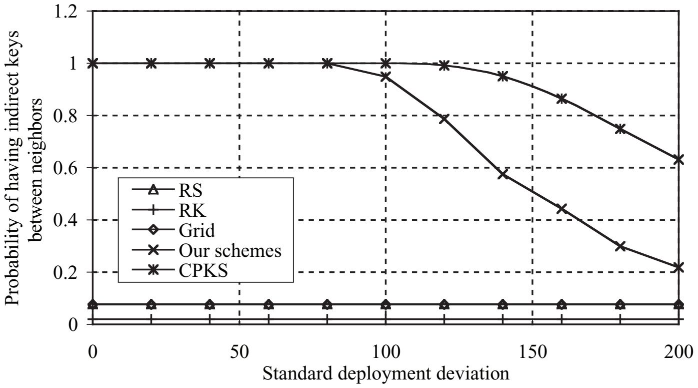

Fig. 5. Probability of having indirect keys between sensor nodes for different schemes.

where p is the probability of having a direct key between neighbor nodes and n' is the total number of sensor nodes contacted in path key establishment. For these three schemes, we always assume that only the source node is responsible for contacting other sensor nodes during the discovery of intermediate nodes.

For the grid-based scheme [Liu and Ning 2003a], since two sensor nodes can always immediately compute the paths for the path key establishment, we assume that they always try to find a path that only involves up to two nodes. There are $n + m - 2$ potential paths that can help establish an indirect key between two sensor nodes. Two of them involve only one intermediate node, and $n + m - 4$ of them involve two intermediate nodes. The probability that none of them works can be estimated by

$$\left(1 - \frac{n'}{nm-2}\right)^2 \left(1 - \left(\frac{n'(n'-1)}{(nm-2)(nm-3)}\right)\right)^{n+m-4}.$$

Hence the probability of having an indirect key can be estimated by

$$1 - \left(1 - \frac{n'}{nm-2}\right)^2 \left(1 - \left(\frac{n'(n'-1)}{(nm-2)(nm-3)}\right)\right)^{n+m-4}.$$

Figure 5 compares the probability of establishing indirect keys between neighbor sensor nodes for different schemes. We can clearly see that our proposed instantiations outperform all the previous location-unaware schemes significantly for a reasonable value of σ. In other words, as long as the sensor nodes are deployed in groups, our framework can be used to obtain high performance key predistribution schemes for sensor networks. Similarly, we also note that the closest pairwise keys scheme can achieve better performance than our schemes. The reason has been explained before.

Figure 5 also shows that it is not guaranteed that every two neighbor nodes can find a valid bridge for use, especially for a large deployment deviation σ. However, we do not suggest the development of any additional protocol for such problem due to the following reasons. First, compared with developing a new complicated protocol, investing more efforts on sensor deployment and allocating more storage space for keys seem far more reasonable and practical to improve the probability of establishing keys between sensor nodes. Second, it is usually not necessary to guarantee that every two neighbor nodes can establish a pairwise key since node redundancy is often used for fault tolerance in sensor networks. In most cases, the network will function properly as long as there are sufficient number of "working" sensor nodes.

5.4 Security Analysis

The main threat we consider in the security analysis is the compromise of sensor nodes since it is quite clear that an attacker has no way to infer the shared key established between any two sensor nodes without compromising sensor nodes. We assume the attacker compromises a random set of c sensor nodes in the network. We assume that the secrets in a sensor node will be disclosed to the attacker once this sensor node is compromised. This subsection focuses on the impact of compromised sensor nodes on direct key establishment and path key establishment.

Similar to the analysis in the previous subsection, we investigate the security of the proposed two instantiations and compare them with the random subset assignment scheme [Liu and Ning 2003a], the random pairwise keys scheme [Chan et al. 2003], the grid-based scheme [Liu and Ning 2003a], and the closest pairwise keys scheme [Liu and Ning 2003b].

During the evaluation, we always assume that the memory usage at each sensor node is equivalent to store 100 cryptographic keys. According to the previous configuration, there are 10, 000 sensor nodes in the network, and $n = m = 100$. For our polynomial-based instantiation, the random subset assignment [Liu and Ning 2003a] and the grid-based scheme [Liu and Ning 2003a], the bivariate polynomials have the degree of $t = 49$.

5.4.1 *Impact on Direct Key Establishment.* For the hash key-based instantiation, every predistributed pairwise key is only known by the related two sensor nodes. Certainly, once a few sensor nodes are compromised, the attacker will know the master keys of these compromised nodes and also get a number of hash images of the master keys at some other noncompromised sensor node. However, due to the one-way hash function, no matter how many hash images of a master key is disclosed, it is still computationally infeasible to recover this master key. This indicates that, no matter how many sensor nodes are compromised, the direct key shared between noncompromised sensor nodes are still secure. In other words, the hash key-based instantiation provides the perfect security guarantee against node compromise.

When compared with the random pairwise keys scheme [Chan et al. 2003], we note that the random pairwise keys scheme can also provide the perfect security guarantee against node compromise. However, as shown in the previous analysis, the probability of having a direct key between two neighbor nodes in the random pairwise keys scheme is much lower than that in the hash key-based instantiation. The reason is that the proposed group-based deployment model allows us to predict a sensor's neighbor nodes more accurately. For example, when a sensor node can only store 100 keys, the probability of having a direct key between neighbors is only 0.01 for the random pairwise keys scheme. In contrast, for the hash key-based instantiation, we can achieve a much higher probability of having a direct key between neighbors, especially when the sensor nodes in the same group are close to each other after deployment, as shown in Figure 3. As a result, given the same storage overhead and security performance, our hash key-based instantiation can achieve a much better performance in terms of the probability of establishing a direct key between neighbor sensor nodes.

When compared with the closest pairwise keys scheme [Liu and Ning 2003b], we note that this scheme can also achieve perfect security guarantee against node compromise. Additionally, as we discussed before, this scheme can also achieve a higher probability of establishing keys for sensor networks. However, we still believe that our hash key-based scheme is more practical in most applications due to the following two reasons. First, the closest pairwise keys scheme uses the expected locations of sensor nodes, which may not be available. Second, the performance of our scheme is not significantly worse than the closest pairwise keys scheme.

As mentioned, the grid-based scheme [Liu and Ning 2003a] can be considered as our polynomial-based instantiation if a row or a column of sensor nodes in the grid are deployed in the same group. This means that the grid-based scheme and our polynomial-based instantiation have the same security performance against the node compromise attacks given the same settings (e.g., storage overhead, network size). On the other hand, Figure 3 tells us that our polynomial-based instantiation can achieve much higher probability of establishing direct keys between neighbor nodes than the grid-based scheme. This implies that our polynomial-based instantiation is more desirable than the grid-based scheme when the group-based deployment model is made possible. Thus, in our later security analysis, we will simply skip the security comparison between the grid-based scheme [Liu and Ning 2003a] and the polynomial-based instantiation.

Next we will study the security of the direct keys established in the polynomial-based instantiation and compare it with the random subset assignment scheme [Liu and Ning 2003a]. Since every sensor node can store 100 keys, the degree t of polynomials in our instantiation is set to 49. For the random subset assignment scheme, we also set the degree of the bivariate polynomial to 49 and assign every sensor node two shares on two randomly selected polynomials from a polynomial pool. The size of the polynomial pool in the random subset assignment scheme is configured in such a way that the probability of having a direct key between two neighbor sensor nodes is the same as that in our proposed polynomial-based instantiation. In this way, we actually compare the security of two schemes under the same storage overhead and the same probability of establishing direct keys between neighbor sensor nodes.

In the following, we first look at the impact of the compromised sensor nodes on the polynomial-based instantiation. Consider a direct key between any two noncompromised sensor nodes in the same group G_i (either a deployment group or a cross group). Since there are totally c compromised sensor nodes, the probability of j sensor nodes in group G_i being compromised can be estimated as

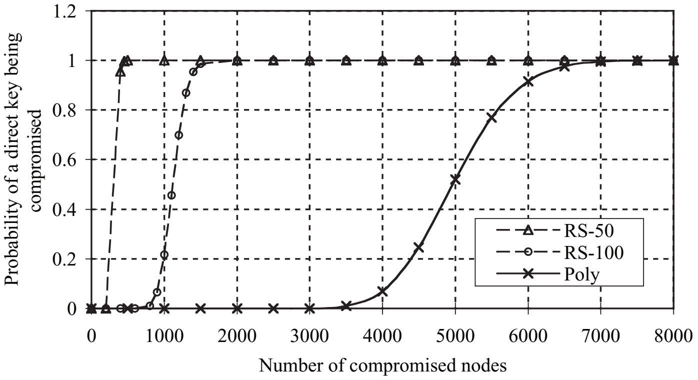

Fig. 6. Probability of a direct key between two noncompromised nodes being compromised. "RS-50" represents the random subset assignment scheme when $\sigma = 50$. "RS-100" represents the random subset assignment scheme when $\sigma = 100$. "Poly" represents our polynomial-based instantiation.

$\frac{c!}{(c-j)!j!}\frac{(n-1)^{c-j}}{n^c}$ for $j \leq m-1$. Based on the result in Blundo et al. [1993], as long as there is no more than t compromised nodes, the direct key between two noncompromised nodes in G_i is still secure. Hence the probability of any direct key between two non-compromised sensor nodes in G_i being compromised can be estimated as

$$p_{cd} = 1 - \sum_{j=0}^{t} \frac{c!}{(c-j)!j!}\frac{(n-1)^{c-j}}{n^c}.$$

Figure 6 compares the probability of a direct key between two noncompromised sensor nodes being compromised for the polynomial-based instantiation and the random subset assignment scheme [Liu and Ning 2003a]. In this figure, we consider the case when σ is set to 50 and the case when σ is set to 100. When $\sigma = 50$, the probability of having direct keys between neighbor sensor nodes in our polynomial-based instantiation is 0.3. Thus the size of the polynomial pool in the random subset assignment scheme is set to 13 to achieve the same probability of having direct keys between neighbor sensor nodes. Similarly, when $\sigma = 100$, we will configure the polynomial size in the random subset assignment scheme as 45. From the figure, we can clearly see that the our polynomial-based instantiation can achieve much better security performance given the same settings, especially when the sensor nodes in the same deployment group are close to each other after deployment.

5.4.2 *Impact on Path Key Establishment.* In the following, we study the impact of compromised sensor nodes on the indirect keys established between neighbor sensor nodes. Note that when the compromised sensor nodes can be detected, two noncompromised nodes can always reestablish an indirect key through the path key establishment and avoid those compromised sensor nodes or compromised key predistribution instances. However, it is usually very difficult to detect compromised sensor nodes. When the compromised nodes cannot be detected, the indirect key between two noncompromised nodes may be disclosed to the attacker without being noticed. Thus, in the following analysis, we focus on the probability of a given indirect key between two noncompromised sensor nodes being compromised when we cannot detect the compromised sensor nodes.

The establishment of an indirect key between sensor nodes involves up to two intermediate nodes. Without loss of generality, we assume the source node u in group G_i wants to setup an indirect key with the destination node v in group G_j. We also assume the indirect key is established through a bridge $\langle u', v' \rangle$, where $u' \in G_i$ and $v' \in G_j$. Since the key established between node u and node v is an indirect key, we have either $u \neq u'$ or $v \neq v'$. Note that u and v are also in different cross groups. We need to consider the following two cases:

(1) $u = u'$ *or* $v = v'$. The probability that this case happens can be estimated as $\frac{2}{m}$. Clearly the path key establishment only involves one intermediate node. The indirect key will be still secure if this node is not compromised and the two related direct keys are not compromised. Hence the probability of the indirect key being compromised can be estimated by $p_1 = 1 - (1 - p_{cd})^2(1 - \frac{c}{nm-2})$

(2) $u' \neq u$ *and* $v' \neq v$. The probability that this case happens can be estimated as $\frac{m-2}{m}$. Clearly the path key establishment only involves two intermediate nodes. Similarly, the probability of the indirect key being compromised can be estimated by $p_2 = 1 - (1 - p_{cd})^3(1 - \frac{c}{nm-2})^2$.

Overall, the probability of an indirect key between noncompromised sensor nodes being compromised can be estimated as

$$p_{ci}(c) = \frac{2 \times p_1 + (m-2) \times p_2}{m}.$$

For the purpose of comparison, in the random subset assignment scheme [Liu and Ning 2003a] and the random pairwise keys scheme [Chan et al. 2003], two sensor nodes always try to find an intermediate sensor node that can establish direct keys with both the source and the destination node. Clearly an indirect key will be still secure if (1) the intermediate node has not been compromised and (2) the two related direct keys used for the path key establishment have not been compromised. Thus the probability of an indirect key between two noncompromised sensor nodes being compromised is $1 - (1 - p)^2(1 - \frac{c}{nm-2})$, where p is the probability of a direct key between two noncompromised sensor nodes being compromised for the random subset assignment scheme [Liu and Ning 2003a] or the random pairwise keys scheme [Chan et al. 2003].

For the random subset assignment scheme, the size of the polynomial pool in the random subset assignment scheme is configured in such a way that the probability of having a direct key between two neighbor sensor nodes is the same as that in our proposed polynomial-based instantiation. For simplicity, we assume the standard deviation of the sensor deployment is 50 m, leading to a probability of 0.3 for establishing direct keys between neighbor sensor nodes. In this case, we set the polynomial pool size to 13 such that the probability of having a direct key between neighbor sensor nodes is also 0.3.

For the grid-based scheme [Liu and Ning 2003a], since two sensor nodes can always immediately compute the paths for the path key establishment, we

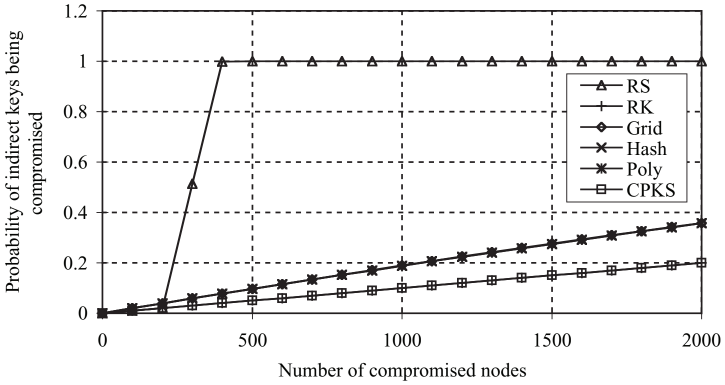

Fig. 7. The probability of an indirect key being compromised for different schemes. "Hash" represents the hash key-based instantiation.

assume that they always try to find a path that only involves up to two nodes. The probability of only involving one intermediate node is $\frac{2}{n+m-2}$. In this case, this indirect key will be compromised at a probability of $1-(1-p)^2(1-\frac{c}{nm-2})$, where p is the probability of a direct key between two noncompromised sensor nodes being compromised. The probability of involving two intermediate nodes in the path key establishment is $\frac{n+m-4}{n+m-2}$. In this case, the probability of the indirect key being compromised can be estimated by $1-(1-p)^3(1-\frac{c}{nm-2})^2$. Hence, overall, the probability of an indirect key in the grid-based scheme being compromised can be estimated by

$$\frac{2\big(1-(1-p)^2\big(1-\frac{c}{nm-2}\big)\big)+(n+m-4)\big(1-(1-p)^3\big(1-\frac{c}{nm-2}\big)^2\big)}{n+m-2}.$$

Figure 7 shows the probability of an indirect key between two noncompromised sensor nodes being compromised for these different schemes. We can see that our two instantiations has better security performance than the random subset assignment scheme given a reasonable σ in terms of the fraction of compromised indirect keys given a certain number of compromised sensor nodes. The random pairwise keys scheme and the closest pairwise keys scheme can have slightly better security than our schemes. However, as we mentioned before, our instantiations can achieve a high probability of having a direct key between neighbor sensor nodes and do not assume the knowledge of the expected locations, while the random pairwise keys scheme can hardly establish a direct key between two neighbor sensor nodes and the closest pairwise keys scheme assumes the knowledge of the expected locations. The grid-based scheme has a similar security performance when compared with the proposed two instantiations. However, our proposed schemes achieve much better performance in terms of the probability of establishing direct keys between neighbors when the sensor nodes are deployed in groups as shown in Figure 3. Therefore, we strongly believe that the proposed two instantiations are usually more practical than the existing key predistribution schemes when the group-based deployment model is made possible.

In Figure 7, we also note that the probability of an indirect key between two noncompromised nodes being compromised increases at a nonnegligible rate when there are more and more compromised sensor nodes. To further enhance the security of our techniques under compromised sensor nodes, we can certainly apply the *multipath reinforcement* idea from Chan et al. [2003]. Basically, we have every two sensor nodes establish multiple indirect keys from a number of independent paths and then XOR these keys to generate the actual indirect key used for secure communication.

The multipath reinforcement requires a number of independent paths to setup an actual indirect key. This is actually nontrivial for many existing key predistribution techniques. However, in our proposed instantiations, every sensor node can easily compute m paths to any other sensor node that cannot establish a direct key with itself. Due to the group-based deployment model, it is very likely that this sensor node will find most of these paths in its local area. As a result, the multipath reinforcement can be efficiently implemented in our framework to further improve key predistribution. Once this idea is applied in our framework, the attacker has to break all paths to compromise the actual indirect keys. The security of our schemes can thus be enhanced.

5.5 Discussion

In this article, we present a novel group-based framework for improving key predistribution and two efficient instantiations under this new framework. The analysis in this section demonstrates the significant improvements achieved by applying our framework when the sensor nodes are deployed in groups.

We strongly believe that the proposed two instantiations are practical for the current generation of sensor networks. First, these two schemes allow efficient implementation since a sensor node can either directly set up a key with its neighbor or immediately identify the nodes in its local area to set up an indirect key as long as it has the list of the nearby sensor nodes. Second, the proposed instantiations can achieve much better security and performance than the existing key predistribution schemes when the sensors' expected or discovered locations are not available. Therefore, the two instantiations are more practical than the previous schemes in practice, especially when the sensor nodes can be deployed in groups.

When the proposed two instantiations are compared with each other, we note that both of them have advantages and disadvantages. First, the hash key-based scheme can provide perfect security guarantee, and the key can be derived by performing one efficient hash function. For the polynomial-based scheme, the sensor node has to evaluate a t-degree polynomial to compute the pairwise key. Although the polynomial-based scheme can also provide perfect security guarantee when we configure $t = m-1$, it introduces more (almost double) storage overhead than the hash key-based scheme. Second, the polynomial-based scheme can provide unconditional security when the number of compromised sensor nodes in one group is less than t, the degree of the bivariate polynomial, while the hash key-based scheme only provides computational security guarantee. Third, the storage overhead of the hash key-based scheme increases linearly with the number of groups $(n + m)$ in the network, while the storage overhead in the polynomial-based scheme only depends on the security parameter t. In other words, the polynomial-based scheme allows further trade offs between the storage overhead and the security of key predistribution. Finally, the polynomial-based scheme can be extended whenever we want to deploy new groups. In contrast, for the hash key-based scheme, we have to reserve additional space for the future deployment.

Based on the above discussion, we can conclude that the hash key-based solution is more appropriate when the storage space allows since it provides perfect security with small overhead in computing the shared keys. However, when we have severe storage constraints, we should use the polynomial-based solution since it can trade off the security with the storage space and provides an efficient way to extend the network size.

6. RELATED WORK

A number of techniques have been proposed to establish pairwise keys in resource constrained sensor networks. A basic probabilistic key predistribution scheme was introduced in Eschenauer and Gligor [2002] and improved in Chan et al. [2003]. The limitation of these approaches is that a small number compromised sensor nodes may affect the secure communication between a large number of noncompromised sensor nodes. A random pairwise keys scheme was proposed in Chan et al. [2003]. Although this technique provides perfect security against node capture attacks, it cannot scale to large sensor networks. To improve the resilience of sensor networks against node compromises, two threshold-based key predistribution techniques were developed in Liu and Ning [2003a] and Du et al. [2003]. A cooperative protocol was developed to enhance the security of pairwise key establishments [Pietro et al. 2003]. The giant component theory was used in Hwang and Kim [2004] to further improve the performance and provide a tradeoff between connectivity, memory size, and security. In this article, we demonstrate that the performance of these key predistribution techniques can be further improved significantly by using our framework.

The grid-based idea was first proposed in Liu and Ning [2003a] to arrange the secrets in sensor networks based on a logical grid. A similar idea was later used in PIKE [Chan and Perrig 2005]. However, the grids considered in these two studies are logical grids, while this article investigates the possibility of using the locality of group deployment to improve the performance of the existing key predistribution techniques.

The prior deployment knowledge of sensor nodes has been used to improve the performance of many key predistribution protocols [Du et al. 2004; Liu and Ning 2003b; Huang et al. 2004; Yu and Guan 2005]. The postdeployment knowledge of sensor nodes has also been used to improve key predistribution [Liu and Ning 2005]. The technique in this article differs from these approaches in that it does not require the expected or discovered location information of sensor nodes, and is therefore desirable for the scenarios where it is difficult to deploy the sensor nodes at their expected locations or correctly estimate the sensors' locations after deployment.

There are many other studies on sensor network security, including key management schemes [Carman et al. 2000; Zhu et al. 2003; Anderson et al. 2004], tamper-resistant hardware [Basagni et al. 2001], efficient broadcast authentication [Perrig et al. 2001], secure data aggregation and in-networking processing [Deng et al. 2003; Hu and Evans 2003; Przydatek et al. 2003], and vulnerabilities, attacks, and countermeasures [Wood and Stankovic 2002; Karlof and Wagner 2003]. We consider them complementary to our techniques in this article.

7. CONCLUSION AND FUTURE WORK

In this article, we developed a general framework that can be used to improve the performance of existing key predistribution schemes. This framework does not require the knowledge of sensors' expected locations or actual deployment locations. We also presented two efficient instantiations in this framework. The analysis demonstrated that this framework can significantly improve the security as well as the performance of existing key predistribution protocols.

Several research directions are worth studying further, including detailed performance evaluation through simulation, the implementation of these techniques on real sensor platforms, and the evaluation through field experiments.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their valuable comments.

REFERENCES

AKYILDIZ, I., SU, W., SANKARASUBRAMANIAM, Y., AND CAYIRCI, E. 2002. Wireless sensor networks: A survey. *Comput. Netw. 38,* 4, 393–422.

ANDERSON, R., CHAN, H., AND PERRIG, A. 2004. Key infection: Smart trust for smart dust. In *Proceedings of the IEEE International Conference on Network Protocols* (ICNP 2004).

BASAGNI, S., HERRIN, K., BRUSCHI, D., AND ROSTI, E. 2001. Secure pebblenets. In *Proceedings of the ACM International Symposium on Mobile ad hoc Networking and Computing*. 156–163.

BLUNDO, C., DE SANTIS, A., HERZBERG, A., KUTTEN, S., VACCARO, U., AND YUNG, M. 1993. Perfectly-secure key distribution for dynamic conferences. In *Advances in Cryptology—CRYPTO '92*. Lecture Notes in Computer Science, vol. 740. Springer Berlin, Germany. 471–486.

CAPKUN, S. AND HUBAUX, J. 2005. Secure positioning of wireless devices with application to sensor networks. In *Proceedings of IEEE InfoCom'05*.

CARMAN, D., KRUUS, P., AND MATT, B. J. 2000. Constrains and approaches for distributed sensor network security. Tech. rep. 00-010. NAI Labs, Glenwood, MD.

CHAN, H. AND PERRIG, A. 2005. PIKE: Peer intermediaries for key establishment in sensor networks. In *Proceedings of IEEE Infocom*.

CHAN, H., PERRIG, A., AND SONG, D. 2003. Random key predistribution schemes for sensor networks. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*. 197–213.

DENG, J., HAN, R., AND MISHRA, S. 2003. Security support for in-network processing in wireless sensor networks. In *Proceedings of the 2003 ACM Workshop on Security in ad hoc and Sensor Networks* (SASN '03).

DU, W., DENG, J., HAN, Y. S., CHEN, S., AND VARSHNEY, P. 2004. A key management scheme for wireless sensor networks using deployment knowledge. In *Proceedings of IEEE INFOCOM'04*.

Du, W., Deng, J., Han, Y. S., and Varshney, P. 2003. A pairwise key pre-distribution scheme for wireless sensor networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security* (CCS'03). 42–51.

Eschenauer, L. and Gligor, V. D. 2002. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*. 41–47.

Gay, D., Levis, P., von Behren, R., Welsh, M., Brewer, E., and Culler, D. 2003. The nesC language: A holistic approach to networked embedded systems. In *Proceedings of the Conference on Programming Language Design and Implementation* (PLDI 2003).

Hartung, C., Balasalle, J., and Han, R. 2005. Node compromise in sensor networks: The need for secure systems. Tech. Rep. CU-CS-990-05. University of Colorado at Boulder, Boulder, CO.

Hill, J., Szewczyk, R., Woo, A., Hollar, S., Culler, D., and Pister, K. S. J. 2000. System architecture directions for networked sensors. In *Proceedings of the Conference Architectural Support for Programming Languages and Operating Systems*. 93–104.

Hu, L. and Evans, D. 2003. Secure aggregation for wireless networks. In *Proceedings of the Workshop on Security and Assurance in ad hoc Networks*.

Huang, D., Mehta, M., Medhi, D., and Harn, L. 2004. Location-aware key management scheme for wireless sensor networks. In *Proceedings of the 2nd ACM Workshop on Security of ad hoc and Sensor Networks* (SASN '04). 29–42.

Hwang, J. and Kim, Y. 2004. Revisiting random key pre-distribution schemes for wireless sensor networks. In *Proceedings of the 2nd ACM Workshop on Security of ad hoc and Sensor Networks* (SASN '04). 43–52.

Karlof, C. and Wagner, D. 2003. Secure routing in wireless sensor networks: Attacks and countermeasures. In *Proceedings of the 1st IEEE International Workshop on Sensor Network Protocols and Applications*.

Lazos, L., Capkun, S., and Poovendran, R. 2005. Rope: Robust position estimation in wireless sensor networks. In *Proceedings of the Fourth International Conference on Information Processing in Sensor Networks* (IPSN '05).

Lazos, L. and Poovendran, R. 2004. Serloc: Secure range-independent localization for wireless sensor networks. In *Proceedings of the ACM Workshop on Wireless Security* (WiSe 2004, Philadelphia, PA).

Li, Z., Trappe, W., Zhang, Y., and Nath, B. 2005. Robust statistical methods for securing wireless localization in sensor networks. In *Proceedings of the Fourth International Conference on Information Processing in Sensor Networks* (IPSN '05).

Liu, D. and Ning, P. 2003a. Establishing pairwise keys in distributed sensor networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security* (CCS'03). 52–61.

Liu, D. and Ning, P. 2003b. Location-based pairwise key establishments for static sensor networks. In *Proceedings of the 2003 ACM Workshop on Security in ad hoc and Sensor Networks* (SASN '03). 72–82.

Liu, D. and Ning, P. 2005. Improving key predistribution with deployment knowledge in static sensor networks. *ACM Trans. Sensor Netw. 1,* 2, 204–239.

Liu, D., Ning, P., and Du, W. 2005. Attack-resistant location estimation in wireless sensor networks. In *Proceedings of the Fourth International Conference on Information Processing in Sensor Networks* (IPSN '05).

Niculescu, D. and Nath, B. 2001. Ad hoc positioning system (APS). In *Proceedings of IEEE GLOBECOM '01*.

Perrig, A., Szewczyk, R., Wen, V., Culler, D., and Tygar, D. 2001. SPINS: Security protocols for sensor networks. In *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networks*.

Pietro, R. D., Mancini, L. V., and Mei, A. 2003. Random key assignment for secure wireless sensor networks. In *Proceedings of the 2003 ACM Workshop on Security in ad hoc and Sensor Networks* (SASN '03).

Przydatek, B., Song, D., and Perrig, A. 2003. SIA: Secure information aggregation in sensor networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems* (SenSys '03).

RAY, S., UNGRANGSI, R., PELLEGRINI, F. D., TRACHTENBERG, A., AND STAROBINSKI, D. 2003. Robust location detection in emergency sensor networks. In *Proceedings of IEEE INFOCOM 2003*.

WOOD, A. D. AND STANKOVIC, J. A. 2002. Denial of service in sensor networks. *IEEE Comput. 35,* 10, 54–62.

YU, Z. AND GUAN, Y. 2005. A key predistribution scheme using deployment knowledge for wireless sensor networks. In *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks* (IPSN).

ZHU, S., SETIA, S., AND JAJODIA, S. 2003. LEAP: Efficient security mechanisms for large-scale distributed sensor networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security* (CCS'03). 62–72.

Received August 2006; revised February 2007; accepted September 2007