Improving Key Pre-Distribution with Deployment Knowledge in Static Sensor Networks

DONGGANG LIU
University of Texas at Arlington
and
PENG NING
North Carolina State University

Pairwise key establishment is a fundamental security service for sensor networks. However, establishing pairwise keys in sensor networks is a challenging problem, particularly due to the resource constraints on sensor nodes and the threat of node compromises. This paper proposes to use both *pre-deployment and post-deployment knowledge* to improve pairwise key pre-distribution in static sensor networks. By exploiting the pre-deployment knowledge, this paper first develops two key pre-distribution schemes, a *closest pairwise keys scheme* and a *closest polynomials scheme*. The analysis shows that these schemes can achieve better performance if the expected location information is available and that the smaller the deployment error is, the better performance they can achieve. The paper then investigates how to use *post-deployment knowledge* to improve pairwise key pre-distribution in static sensor networks. The idea is to load an excessive amount of pre-distributed keys on sensor nodes, prioritize these keys based on sensors' actual locations discovered after deployment, and discard low priority keys to thwart node compromise attacks. This approach is then used to improve the random subset assignment scheme proposed recently to demonstrate its practicality and effectiveness. The analysis indicates that the post-deployment knowledge can also greatly improve the performance and security of key pre-distribution.

Categories and Subject Descriptors: C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; D.4.6 [**Operating Systems**]: Security and Protection—*Cryptographic controls*

General Terms: Security, Design, Algorithms

Additional Key Words and Phrases: Sensor Networks, Key Management, Key Pre-Distribution

1. INTRODUCTION

Sensor networks are ideal candidates for applications such as target tracking, battlefield surveillance, and scientific exploration in hazardous environments. Typically, a sensor net-

Liu's address: Department of Computer Science and Engineering, University of Texas at Arlington, Box 19015, Arlington, TX 76019-0015; email: dliu@cse.uta.edu. Ning's address: Department of Computer Science, North Carolina State University, Raleigh, NC 27695-8207; email: pning@ncsu.edu. A preliminary version of this paper appeared in the *Proceedings of the 1st ACM Workshop on Security of Sensor and Ad Hoc Networks*, pages 72–82, October 2003. Liu's work was done when he was a PhD student in Department of Computer Science at North Carolina State University.

work consists of a potentially large number of resource constrained sensor nodes, which are mainly used to sense physical phenomena (e.g. temperature, humidity) from its immediate surroundings, process, and communicate the sensed data locally, and a few control nodes, which may have more resources and may be used to control the sensor nodes and/or connect the network to the outside world (e.g. a central data processing server). Sensor nodes usually communicate with each other through wireless channels in short distances.

Sensor networks may be deployed in hostile environments, especially in military applications. In such situations, an adversary may physically capture sensor nodes, and intercept and/or modify data/control packets. Therefore, security services such as authentication and encryption are essential to maintain the normal network operations. However, due to the resource constraints on sensor nodes, many security mechanisms such as public key cryptography are not desirable, and sometimes infeasible in sensor networks. For example, though some efficient public key cryptosystems such as Elliptic Curve Cryptography (ECC) have been shown to be feasible on sensor platforms such as MICA2 motes [Gura et al. 2004], it still requires substantial computation (e.g., about 0.81 second for 160-bit ECC point multiplication [Gura et al. 2004]). As a result, simply using public key cryptosystems (such as 160-bit ECC) for authentication will make the verifier vulnerable to Denial of Service (DoS) attacks, since an attacker may flood the verifier with bogus signatures and force it to perform useless but expensive operations. Indeed, providing efficient security services in sensor networks is by no means a trivial task, and it has received a lot of attention recently [Perrig et al. 2001; Eschenauer and Gligor 2002; Chan et al. 2003; Karlof and Wagner 2003; Wood and Stankovic 2002; Liu and Ning 2003a; Du et al. 2003; Liu and Ning 2003b; Liu et al. 2004; Chan and Perrig 2005; Liu et al. 2005c].

In typical sensor networks, a sensor node needs to exchange messages with its neighbor nodes. This forms the basis of other communication patterns such as source-to-sink communications. To ensure the security of such message exchanges, we need to establish a symmetric, pairwise key between two sensor nodes, which is the basis of many other security services such as encryption and authentication. Several key pre-distribution techniques have been developed recently to address this problem. Eschenauer and Gligor [2002] proposed the basic probabilistic key pre-distribution, where each sensor node is assigned a random subset of keys from a key pool before deployment. As a result, two sensor nodes have a certain probability to share at least one key after deployment. By requiring two sensor nodes share at least q pre-distributed keys to establish a pairwise key, Chan et al. [2003] developed the q-composite scheme, which improves the resilience of the basic probabilistic scheme against node compromises. Chan et al. [2003] also developed the random pairwise keys scheme, which pre-distributes random pairwise keys between a particular sensor node and a random subset of other sensor nodes. The random pairwise keys scheme has the property that the compromise of sensor nodes does not lead to the compromise of any pairwise key shared directly between two non-compromised sensor nodes. Du et al. [2003] and Liu and Ning [2003b] independently developed key pre-distribution techniques that significantly enhance the resilience of random key pre-distribution against node compromises. Chan and Perrig [2005] recently developed PIKE, which facilitates pairwise key establishment using peer sensor nodes as trusted intermediaries.(Additional related work is discussed in Section 4.)

Despite the recent advances, key pre-distribution for sensor networks is still not an entirely solved problem. This is particularly because the performance of existing key pre-distribution schemes, especially the probability to establish a common key between communicating nodes and the ability to tolerate compromised nodes, are highly dependent on the memory available on sensor nodes. Due to the need to lower the cost of sensor networks, it is always desirable to seek additional techniques that can further improve the security and performance of key pre-distribution in sensor networks. Moreover, in many sensor network applications, long distance peer-to-peer secure communication between sensor nodes is rare and unnecessary. Thus, the primary goal of secure communication is to provide authentication and/or encryption between neighbor sensor nodes that can directly communicate with each other. This observation offers an opportunity to improve the performance of key pre-distribution.

In this paper, we show that the performance of key pre-distribution can be improved significantly in static sensor networks, where node movements after deployment are restricted to a limited scope, by using pre-deployment and/or post-deployment knowledge.

1.1 Overview of Proposed Approaches

In this paper, we first exploit the *pre-deployment knowledge* of sensor nodes to improve key pre-distribution in static sensor networks. The techniques are based on the observation that in static sensor networks, *although it is difficult to precisely pinpoint sensor nodes' positions, it is often possible to approximately determine their locations.* For example, when we use trucks to deploy static sensor nodes, we can usually keep sensor nodes within a certain distance (e.g., 100 yards) from their target locations, though it is difficult to place the sensor nodes in their expected locations precisely. By taking advantage of this observation, our techniques provide better security and performance than the previous techniques.

We also propose to take advantage of *post-deployment knowledge*, and investigate a new approach, which we refer to as *key prioritization*, to improve the performance of key pre-distribution schemes in static sensor networks. The main idea is to use the memory for applications (e.g., EEPROM on MICA2 motes [Crossbow Technology Inc.]) to store an excessive amount of keying materials, prioritize the keying materials based on sensor nodes' post-deployment information, and discard low priority keying materials to thwart node compromise attacks as well as return memory to the applications. For example, a sensor application may be designated to collect temperature, humidity, etc. at a certain frequency, and buffer the data before transmitting back to the central processing system. During the key pre-distribution phase (before the deployment of the sensor nodes), we may use a large amount of available memory (e.g., in EEPROM) to store pre-distributed keying materials. After a sensor node is deployed, it may first examine its environment to assess the likelihood of using each keying material, prioritize the keying materials accordingly, and then discard the low priority ones.

Note that accessing EEPROM is more expensive than accessing RAM in a typical sensor node. It takes more energy to delete cryptographic keys from the EEPROM than to deleting keys in RAM. Based on the results in [Shnayder et al. 2004], the energy consumed by writing 16 bytes to EEPROM is close to the energy consumed by computing for 237,360 clock cycles (about 29.67 ms). Nevertheless, the key prioritization technique only requires such deletion operations once for each node during the entire lifetime. Therefore, we believe that such deletion operations are feasible in the current generation of sensor networks.

1.2 Main Contributions

The contribution of this paper is three-fold. First, we develop a location-aware deployment model for static sensor networks, and integrate the expected location information with the random pairwise keys scheme [Chan et al. 2003]. The resulting scheme keeps the nice property of the original scheme, i.e., the compromise of sensor nodes does not lead to the compromise of pairwise keys shared directly between non-compromised nodes. However, unlike the original scheme, to achieve a desired performance, our scheme only requires a certain network density, but does not impose any restriction on the network size. Moreover, with the same storage capacity in sensor nodes, our scheme achieves a higher probability to establish pairwise keys than the random pairwise keys scheme, especially when the deployment error is small. Our extension to this basic scheme allows smaller storage overhead and easier deployment of dynamically added sensor nodes.

Second, we develop another key pre-distribution scheme by combining the random subset assignment [Liu and Ning 2003b] with the expected location information. This scheme offers further trade-offs between the security against node captures and the probability of establishing pairwise keys directly between neighbor nodes given certain memory constraint. Our analysis also indicates that this scheme provides a higher probability to establish pairwise keys directly between neighbor sensor nodes and better resistance against node captures than the original scheme in [Liu and Ning 2003b].

Third, we identify a new technique, key prioritization, to improve key pre-distribution using post-deployment knowledge. This technique offers an opportunity to exceed the limit imposed by resource constraints on sensor nodes, and as a result, improves the performance and security of the random subset assignment scheme developed in [Liu and Ning 2003b].

1.3 Organization

The rest of the paper is organized as follows. Sections 2 and 3 describe the techniques to improve key pre-distribution by harnessing sensor nodes' pre-deployment and post-deployment knowledge, respectively. Section 4 reviews the related work on sensor network security. Section 5 concludes this paper and points out several future research directions.

2. KEY PRE-DISTRIBUTION USING PRE-DEPLOYMENT KNOWLEDGE

In static sensor networks, it may be possible to predetermine the locations of sensor nodes to a certain extent. These predetermined locations can be used to improve the performance of pairwise key pre-distribution. In this section, we first introduce a simple location-aware deployment model for this purpose, and then develop two pairwise key pre-distribution schemes that can take advantage of the predetermined location information.

2.1 A Location-Aware Deployment Model

We assume that sensor nodes are deployed in a two dimensional area called *target field*, and two sensor nodes can communicate with each other if they are within each other's *signal range*. The location of a sensor node can be represented by a coordinate in the target field. Each sensor node has an *expected location* that can be predicted or predetermined. After the deployment, a sensor node is placed at a *deployment location* that may be different from its expected location. We call the difference between the expected location and the deployment location of a sensor node the *deployment error* for this sensor node. This deployment model can be characterized by the following three parameters.

(1) **Signal range d_r:** A sensor node can receive messages from another sensor node if the former is located within the signal range of the latter. We model the signal range of a sensor node as a circle centered at its deployment location with the radius d_r. For simplicity, we assume the radius d_r defining the signal range is a network-wide parameter, and denote the signal range with d_r. We say two sensor nodes are *neighbors* if they are physically located within each other's signal range.

(2) **Expected location (L_x, L_y):** The expected location (L_x, L_y) of a sensor node is a coordinate in the two dimensional target field; it specifies where the sensor node is expected to be deployed. Sometimes, a sensor node may be expected to be deployed within an area instead of a particular location. In this case, we assume the sensor node is expected to be deployed at any location in that area with equal probability.

(3) **Deployment pdf ϵ:** We model the actual deployment location of a sensor node with a *probability density function* ϵ. The sensor node expected to be deployed at (L_x, L_y) may appear at a particular area with a certain probability, which is calculated by the integration of probability density function ϵ over this area. In some cases, the sensor node may have certain mobility, and appear somewhere near its expected location with a certain probability. The deployment location of this sensor node at any point in time may also be modeled by the probability density function.

Although our techniques can be applied to any deployment model, in this paper, we always evaluate the performance of our techniques with a simple one, where each sensor node randomly appears anywhere at a distance of no more than e away from the expected location. We call e the *maximum deployment error*. Thus, the deployment pdf ϵ for a sensor node u with expected location (L_x, L_y) can be expressed as

$$\epsilon_{(L_x,L_y)}(x,y) = \begin{cases} \frac{1}{\pi e^2}, & ||(L_x,L_y),(x,y)|| \le e \\ 0, & otherwise. \end{cases}$$

where $||\cdot||$ denotes the distance between two locations.

Obviously, this model can be easily extended to a three dimensional space. However, in this paper, we focus on pairwise key establishments in the two dimensional case. Extending our results to the three dimensional model would be straightforward.

2.2 Closest Pairwise Keys Scheme

In this subsection, we develop a pairwise key establishment scheme named *closest pairwise keys scheme* to take advantage of the expected location information. The basic idea is to have each sensor node share pairwise keys with a number of other sensor nodes whose expected locations are closest to the expected location of this sensor node. The following discussion starts with a basic version, which can be considered as the combination of the random pairwise keys scheme [Chan et al. 2003] and the expected location information, and then gives an extended version to further reduce the storage overhead and facilitate dynamic deployment of new sensor nodes.

We assume a setup server is responsible for key pre-distribution. This setup server is aware of the expected location of each sensor node. However, it does not require the network-wide signal range d_r and the deployment pdf ϵ, since these two pieces of information are not used in our technique. We assume each sensor node has a unique, integer-valued ID. We also use this node ID to refer to the sensor node. For convenience, we call a pairwise key shared directly between two neighbor nodes as a *direct key*, and a pairwise key established through other intermediate nodes as an *indirect key*.

2.2.1 *The Basic Version.* The basic idea of the closest pairwise keys scheme is to *pre-distribute pairwise keys between pairs of sensor nodes that have high probabilities to be neighbors.* Though reasonable, this idea is difficult to implement, since it is non-trivial to get the probability that two sensor nodes are neighbors. Indeed, this probability depends on the deployment pdf ϵ, which is generally not available and may vary in different applications. To simplify the situation, we pre-distribute pairwise keys between pairs of sensor nodes whose expected locations are close to each other, hoping that the closer the expected locations of two sensor nodes, the more likely that they are physically located in each other's signal range.

(1) **Pre-Distribution.** Based on the expected locations of sensor nodes, the setup server pre-distributes pairwise keys for each sensor node to facilitate the pairwise key establishment during the normal operation. Specifically, for each sensor node u, the setup server first discovers a set S of c other sensor nodes whose expected locations are closest to the expected location of u, where c is a system parameter determined by the memory constraint. For each node v in S, the setup server randomly generates a unique pairwise key $K_{u,v}$ if no pairwise key between u and v has been assigned. The setup server then assigns $(v, K_{u,v})$ and $(u, K_{u,v})$ to sensor nodes u and v, respectively.

(2) **Direct Key Establishment.** After the deployment of the sensor network, if two sensor nodes u and v want to establish a pairwise key to secure the communication between them, they only need to check whether there is a pre-distributed pairwise key between them. This information is obtained from the setup server at the pre-distribution phase. The algorithm to identify such a common key is trivial, because each pairwise key in a particular sensor node is associated with a node ID.

(3) **Indirect Key Establishment.** When two neighbor nodes cannot establish a direct key, they need to find one or more intermediate nodes to help them setup an indirect session key. A simple way is to have one node (*called source node*) send a request to a number of nodes that share direct keys with it. If one of those contacted nodes also shares a direct key with the other node (*called destination node*), this contacted node can be used as an intermediate node to help establish a common session key. In our later schemes, we will omit the indirect key establishment phase, since it is not directly related to our techniques. Indeed, indirect key establishment can be done with any of the previous schemes (e.g., [Liu and Ning 2003b; Chan et al. 2003; Eschenauer and Gligor 2002]).

(4) **Sensor Addition and Revocation.** During the lifetime of a sensor network, new sensor nodes may be added to replace damaged or compromised sensor nodes. To add a new node, the setup server performs the above pre-distribution process for the new sensor node, and then informs the deployed sensor nodes chosen for the new sensor node the corresponding pairwise keys through secure channels. (Here we assume the communication between each sensor node and the setup server is secured with a unique pairwise key shared between the node and the setup server.) The setup server may know the deployment locations of the deployed sensor nodes. In this case, the setup server may use these deployment locations (instead of their expected locations) to select neighbors for the new sensor node.

Table I. Notations

N	network size
m	average number of neighbor sensor nodes
c	number of keys pre-distributed to sensor nodes
p	probability of sharing a direct key between neighbors
P_c	fraction of compromised direct keys between non-compromised nodes

The detection of compromised nodes is generally a difficult problem, which is beyond the scope of this paper. However, there are several methods that could be used to identify compromised sensor nodes to revoke (e.g., [Marti et al. 2000; Buchegger and Boudec 2002; Du et al. 2005; Liu et al. 2005b]). Once the compromised nodes are detected, it is usually necessary to revoke them from the network. To revoke a sensor node, each sensor node that shares a pairwise key with the revoked node simply deletes the corresponding key from its memory.

Though the above scheme looks similar to the previous methods [Eschenauer and Gligor 2002; Chan et al. 2003], it can achieve better performance if the predetermined location information is available. In the following, we show the improvement over the previous methods through analysis. Table I lists several notations that are often used in our analysis.

2.2.2 *Probability of Establishing Direct Keys.* For simplicity, we assume that the sensor nodes in the network are expected to be evenly distributed in the target field. Thus, if u is one of v's closest c sensor nodes, v is very likely to be one of u's closest c sensor nodes. We use $\dot{u}$ and $\bar{u}$ to represent the deployment location and the expected location of node u, respectively. As discussed in Section 2.1, we model the deployment location of node u as a probability density function $\epsilon_{\bar{u}}(x, y)$.

Consider two sensor nodes u and v. Since they are deployed independently, given the expected locations of u and v, the conditional probability that they are neighbors can be calculated by

$$p(||\dot{u}, \dot{v}|| \leq d_r | \bar{u}, \bar{v}) = \int\int\int\int_{||\dot{u},\dot{v}|| \leq d_r} \epsilon_{\bar{u}}(x_1, y_1)\epsilon_{\bar{v}}(x_2, y_2)\mathrm{d}x_1\mathrm{d}y_1\mathrm{d}x_2\mathrm{d}y_2,$$

where $||\cdot||$ denotes the distance between two locations.

Since sensor nodes are evenly distributed in the target field, the densities of sensor nodes in different small areas are approximately equal. Assume there are on average m nodes in each sensor node's signal range. The density of the network can be estimated by $D = \frac{m+1}{\pi d_r^2}$, where d_r is the radius of the signal range. Thus, on average, each node will get pre-distributed pairwise keys with the sensor nodes whose expected locations are no more than d' away from it, where $d' = \sqrt{\gamma} \times d_r$, and $\gamma = \frac{c}{m+1}$. We call $\gamma = \frac{c}{m+1}$ the *capacity density ratio*. For any v having a pre-distributed pairwise key with u, the probability that v falls into u's signal range can be calculated by

$$p(||\dot{u}, \dot{v}|| \leq d_r | \bar{u}) = \int\int_{||\bar{u},\bar{v}|| \leq d'^2} \frac{p(||\dot{u}, \dot{v}|| \leq d_r | \bar{u}, (x, y))}{\pi d'^2}\mathrm{d}x\mathrm{d}y.$$

Among the sensor nodes that have pre-distributed pairwise keys with sensor node u, the average number of sensor nodes that fall into its signal range can be estimated by $c \times p(||\dot{u}, \dot{v}|| \leq d_r | \bar{u})$. Thus, the probability of establishing a common key between node

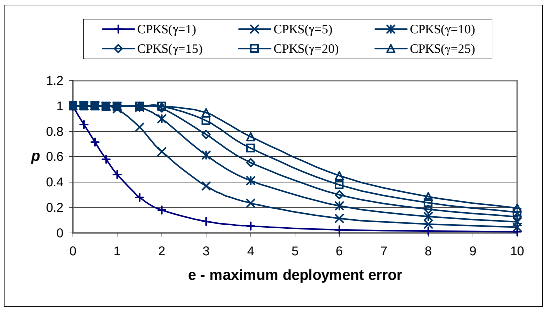


Fig. 1. Probability of establishing direct keys between two neighbor nodes given different values of e and γ. CPKS denotes the closest pairwise keys pre-distribution scheme.

u and its neighbor sensor node can be estimated by

$$p = \frac{c \times p(||\dot{u}, \dot{v}|| \leq d_r|\bar{u})}{m} \approx \gamma \times p(||\dot{u}, \dot{v}|| \leq d_r|\bar{u}).$$

The above p can usually be used to estimate the probability of any node having a direct key with its neighbor node when the target field is infinite. For a limited field in our simulation, we simply use the probability p of the node expected to be deploy at the center of this field having a direct key with its neighbor node to estimate the probability of having a direct key between any two neighbor nodes.

In the following analysis, we always use the radius of signal range, d_r, as the basic unit of distance measurement ($d_r = 1$). For example, a distance 2 implies that the distance is twice as far as d_r. Thus, the deployment error in our discussion also represents the ratio of the deployment error to the signal range. Figure 1 shows the probability of establishing direct keys between neighbor sensor nodes for different values of e and γ. We can see that this probability is not only affected by the maximum deployment error, but also by the capacity density ratio γ. In general, the increase of γ will increase the probability p given certain deployment pdf. However, this probability decreases when the maximum deployment error increases. In practice, we expect to see better performance than that in Figure 1, since the probability of having a smaller deployment error is typically higher than the probability of having a larger one.

2.2.3 *Security Against Node Captures.* There are several attacks against sensor networks, such as DoS attacks [Wood and Stankovic 2002], Sybil attacks [Newsome et al. 2004], and Wormhole attacks [Hu et al. 2003]. These attacks may affect the security of pairwise key establishment in sensor networks. For example, an attacker may create wormholes between different areas so that a node establishes unnecessary keys with other nodes that will disappear once the wormholes are gone. However, these attacks are not unique to the pairwise key establishment in sensor networks. In addition, using expected location information does not reduce the security of any existing key pre-distribution scheme. For simplicity, we focus on the node compromise attacks in this paper.

In node compromise attacks, an adversary may physically capture one or more sensor nodes, and learn all the secrets stored on these nodes. We assume the compromised sensor nodes may collude together to attack the communication between non-compromised sensor nodes. That is, the adversary may try to figure out the pairwise keys established between

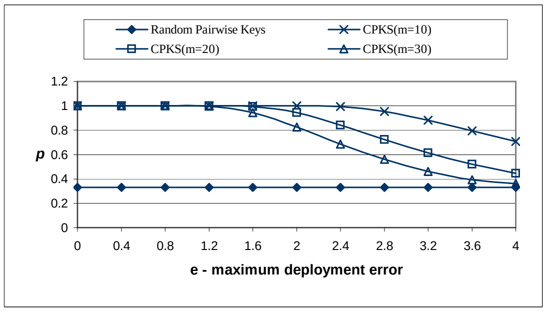


Fig. 2. Probability of establishing direct keys in random pairwise keys scheme and the closest pairwise keys scheme for different m and e given $c = 200$ and $N = 600$.

non-compromised nodes based on the secrets learned from the compromised ones.

From the scheme it is easy to see that each pre-distributed pairwise key between two sensor nodes is randomly generated. Thus, no matter how many sensor nodes are compromised, the direct keys between non-compromised sensor nodes are still secure. We call this property as *perfect security against node captures*. However, once a sensor node is compromised, the session keys this sensor node helps establish may be compromised. For example, an attacker may have saved a copy of an indirect session key encrypted by a direct key, and thus will be able to decrypt the session key once she gets the corresponding direct key from a compromised node. Thus, if either of the source or destination sensor nodes notices that the intermediate sensor node is compromised, it should remove the corresponding pre-distributed pairwise key, and initiate a request to establish a new session key. The delay in detecting compromised sensor nodes still poses a threat. One way to mitigate this threat is to derive the session key by combining (e.g., XOR) the keys generated from multiple paths, as suggested in [Chan et al. 2003].

2.2.4 *Overhead.* Ideally, each sensor node stores c pairwise keys. However, this does not necessarily happen because of the asymmetry in sensor nodes' locations. Consider a pair of sensor nodes u and v. In the pre-distribution step, v is one of u's closest c sensor nodes; however, u is not necessarily one of v's closest c sensor nodes. In this case, v has to store the pairwise key between u and v in addition to its own pre-distributed c pairwise keys. Thus, the storage overhead in each sensor node comes from two parts: One consists of the pairwise keys generated for itself, and the other consists of the pairwise keys generated for other sensor nodes. Hence, each sensor node has to store at least c keys and c sensor IDs. The actual number of pairwise keys stored in a particular sensor node may be much larger than c. Nevertheless, if the sensor nodes are approximately evenly distributed in the target field, it is very likely that if sensor node u is among sensor node v's closest c sensor nodes, then v is among u's closest c sensor nodes.

To establish a common key with a given neighbor node, a sensor node only needs to check if it has a pre-distributed pairwise key with the given node (because each pairwise key is associated with a node ID). Thus, there is no communication and computation overhead during direct key establishment. The establishment of an indirect session key requires one broadcast request message, and potentially a number of unicast reply messages with the techniques in [Eschenauer and Gligor 2002; Chan et al. 2003; Liu and Ning 2003b].

2.2.5 *Improvements.* Our basic scheme can be considered as an extension to the random pairwise keys scheme. These two schemes have some common properties. In both schemes, the compromise of sensor nodes does not lead to the compromise of direct keys shared between non-compromised sensor nodes. However, our scheme further takes advantage of expected location information, and thus is able to achieve better performance than the random pairwise keys scheme. First, the random pairwise keys scheme has a restriction on the network size, while our scheme has no direct restriction on the network size. Second, given the same storage capacity c for pairwise keys and the total number N of sensor nodes, the probability of establishing direct keys in our scheme is always better than the random pairwise keys scheme. This is illustrated in Figure 2, which compares the probability of establishing direct keys in both schemes for different m and e given that $c = 200$ and $N = 600$. It shows that the probability p of establishing direct keys is improved significantly in our scheme, especially when e is less than two times of the signal range. When the maximum deployment error e increases, this probability gradually decreases. In the extreme case, when there is no knowledge about where the nodes may reside, the technique degenerates into the original random pairwise keys scheme.

2.2.6 *Comparison with Previous Methods.* Now let us compare our scheme with the basic probabilistic scheme [Eschenauer and Gligor 2002], the q-composite scheme [Chan et al. 2003], and the random subset assignment scheme [Liu and Ning 2003b]. As discussed earlier, our proposed scheme has a high probability to establish direct keys between neighbor sensor nodes given reasonable capacity density ratio γ and maximum deployment errors. At the same time, our scheme does not put any limitation on the network size.

Note that the closest pairwise keys scheme provides perfect security against node capture attacks, while the basic probabilistic scheme and the q-composite scheme cannot achieve perfect security guarantee. Although the random subset assignment scheme can be configured to achieve perfect security guarantee, it can only support a limited number of sensor nodes to ensure a certain probability of having direct keys between sensor nodes. Thus, a more direct and reasonable way to make comparisons between different schemes is to show the security against node compromise attacks given the same probability of establishing direct keys between sensor nodes.

Figure 3 shows that given the same storage overhead and the same probability of establishing direct keys between sensor nodes, our scheme does not lead to compromise of direct keys belonging to non-compromised sensor nodes, while in the other three schemes [Eschenauer and Gligor 2002; Chan et al. 2003; Liu and Ning 2003b], the direct keys shared between non-compromised sensor nodes are compromised quickly when the number of compromised sensor nodes increases.

Compared with the grid-based scheme [Liu and Ning 2003b], the closest pairwise keys scheme has some advantages. First, the closest pairwise keys scheme provides perfect security guarantee. Though the grid-based scheme can also provide perfect security guarantee, it has limitations on the maximum supported network size given certain storage constraint [Liu and Ning 2003b]. In contrast, the closest pairwise keys scheme has no limitations on the maximum supported network size. Second, the closest pairwise keys scheme can achieve a higher probability of establishing direct keys between two neighbor nodes than the grid-based scheme. Though the grid-based scheme can guarantee to establish a direct or indirect key between any two sensor nodes, it requires that any two sensor nodes can communicate with each other, which may not be true in real applications.

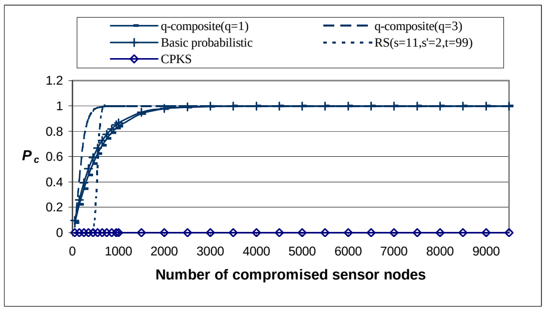


Fig. 3. Fraction of compromised pairwise keys between non-compromised sensor nodes v.s. number of compromised sensor nodes. RS denotes the random subset assignment scheme

2.2.7 *The Extended Version.* The basic scheme described above has two limitations. First, if the sensor nodes are not evenly distributed in the target field, it is possible for a sensor node to have a large number of neighbor nodes that are not among the closest c sensor nodes of u, but consider u as one of their closest c sensor nodes. As a result, node u has to store a lot of pairwise keys generated by the setup server. Second, to add a new sensor node after deploying the sensor network, the setup server has to inform a number of existing sensor nodes in the network about the addition of the new sensor node, which may introduces a lot of communication overhead.

We propose an alternative way to pre-distribute the secret information so that (1) the storage overhead in each node is small and fixed no matter how the sensor nodes are deployed, and (2) no communication overhead is introduced during the addition of new sensor nodes. The technique is based on a pseudo random function (PRF) [Goldreich et al. 1986] and a master key shared between each sensor node and the setup server.

(1) **Pre-Distribution:** For each sensor u, the setup server first randomly generates a master key K_u, and determines a set S of c other sensor nodes whose expected locations are closest to that of u. The setup server then distributes to sensor u a set of pairwise keys (together with the IDs) with those selected sensor nodes. These keys are generated by the setup server in the following way: For each $v \in S$, the setup server generates a pseudo random number $k_{u,v} = PRF_{K_v}(u)$ as the pairwise key shared between u and v, where K_v is the master key for v. As a result, for each $v \in S$, node u stores the pairwise key $k_{u,v}$, while node v can compute the same key with its master key and the ID of node u. We call v a *master sensor node* of u if the direct key $k_{u,v}$ shared between them is derived by $k_{u,v} = PRF_{K_v}(u)$. Accordingly, we call u a *slave sensor node* of v if v is a master sensor node of u.

(2) **Direct Key Establishment:** The direct key establishment stage is similar to the basic scheme. The only difference is that one of two sensor nodes has a pre-distributed pairwise key and the other only needs to compute the key using its master key and the ID of the other party. For example, if u finds that it has the pre-distributed pairwise key $PRF_{K_v}(u)$ with v, it then notifies sensor v that it has such a key. Sensor node v only needs to compute $PRF_{K_v}(u)$ by performing a single pseudo random function.

(3) **Sensor Addition and Revocation:** To add a new sensor node u, the setup server selects c sensor nodes closest to the expected location of u. For each of these c sensor

nodes, the setup server retrieves v's master key K_v and computes $k_{u,v} = PRF_{K_v}(u)$, and then distributes v and $k_{u,v}$ to u. Revoking a sensor is a little more complex than in the basic scheme. To revoke sensor v, all its slave sensor nodes need to remove the corresponding keys from their memory. Moreover, v's master sensor nodes have to remember v's ID in order to avoid establishing a direct key with v later.

In this extension, each sensor node needs to store a master key which is shared with the setup server and c pre-distributed pairwise keys. Thus, the storage overhead for keys in each sensor node is at most $c + 1$. To establish a pairwise key, one of them can initiate a request by informing the other party that it has the pre-distributed pairwise key. (Note that this message only indicates the existence of such a key and the exact value is never disclosed in the communication channel.) Once the other party receives such a message, it can immediately compute the pairwise key by performing one PRF operation. Thus, the communication overhead in the above scheme involves only one short request message and the computation overhead only involves one efficient PRF operation.

Based on the security of PRF [Goldreich et al. 1986], if a node's master key is not disclosed, no matter how many pairwise keys generated with this master key are disclosed, it is still computationally infeasible for an attacker to recover the master key and the non-disclosed pairwise keys generated with different IDs. Thus, node compromise does not lead to the compromise of the direct keys shared between non-compromised nodes.

The extended scheme introduces some additional overhead by requiring master sensor nodes to remember the IDs of their revoked slaves. We consider this an acceptable overhead due to the following reasons. First, the storage overhead for a node ID is much smaller than that for one cryptographic key. Second, in normal situations when authentication of the revocation information is ensured, the number of revoked slave nodes is usually less than m, the average number of sensor nodes in each sensor's signal range. One may argue that if the authentication of revocation information can be bypassed, an attacker may convince a sensor node to store many node IDs to exhaust its memory. However, in this case, the node can be convinced to do anything, and should be considered compromised.

2.3 Closest Polynomials Pre-Distribution Scheme

The scheme presented earlier still has some limitations. In particular, given the constraints on the storage capacity, node density, signal range and deployment pdf, the probability of establishing direct keys is fixed. For a particular sensor network, it is not convenient to adjust the last three parameters. Thus, one has to increase the storage capacity for pairwise keys to increase the probability of establishing direct keys. This may not be a feasible solution in certain sensor networks given the memory constraints on sensor nodes.

This subsection presents a key pre-distribution scheme, called *closest polynomials pre-distribution scheme*, by combining the expected locations of sensor nodes with the random subset assignment scheme in [Liu and Ning 2003b]. The resulting technique allows trade-offs between the security against node captures and the probability of establishing direct keys with a given memory constraint. Moreover, it does not require the setup server be aware of the global network topology, making the deployment much easier.

In the following, we first review the random subset assignment scheme in [Liu and Ning 2003b], then present an improved key pre-distribution scheme using expected locations, and finally analyze the security and performance of this scheme.

2.3.1 *Overview of The Random Subset Assignment Scheme.* We choose the random subset assignment scheme because it can be considered as a generalization of several key pre-distribution schemes. Indeed, the basic probabilistic key pre-distribution scheme [Eschenauer and Gligor 2002] is a special case of the random subset assignment scheme when each key is considered as a 0-degree polynomial [Liu and Ning 2003b]. Moreover, the key pre-distribution scheme in [Du et al. 2003] is essentially equivalent to the random subset assignment scheme in [Liu and Ning 2003b]. Since most key pre-distribution schemes (except for the random pairwise keys scheme [Chan et al. 2003]) are based on the random and independent distribution of key units to sensor nodes, the results obtained through improving the random subset assignment scheme can be easily generalized to those schemes.

Before reviewing the random subset assignment scheme developed in [Liu and Ning 2003b], we first review a polynomial-based key pre-distribution scheme in [Blundo et al. 1993], which was developed for group key pre-distribution. Though using it for group key pre-distribution is generally not practical because of its overhead, its special case for pairwise keys is feasible in sensor networks. For simplicity, we only discuss the special case of pairwise key establishment.

To pre-distribute pairwise keys, the setup server randomly generates a bivariate t-degree polynomial $f(x, y)$ over a finite field F_q, where q is a prime number that is large enough to accommodate a cryptographic key, such that it has the property of $f(x, y) = f(y, x)$. (In the following, we assume all the bivariate polynomials have this property without explicit statement.) It is assumed that each sensor node has a unique ID. For each node i, the setup server computes a *polynomial share* of $f(x, y)$, $f(i, y)$. Thus, for any two sensor nodes i and j, node i can compute the common key $f(i, j)$ by evaluating $f(i, y)$ at point j, and node j can compute the same key ($f(j, i) = f(i, j)$) by evaluating $f(j, y)$ at point i.

In this approach, each sensor node needs to store a t-degree polynomial, which occupies $(t + 1) \log q$ storage space. To establish a pairwise key, both sensor nodes need to evaluate the polynomial at the ID of the other sensor node. There is no communication overhead during the pairwise key establishment process. The security proof in [Blundo et al. 1993] ensures that this scheme is unconditionally secure and t-collusion resistant. That is, the collusion of no more than t compromised sensor nodes knows nothing about the direct key between any two non-compromised nodes. However, the polynomial-based key pre-distribution scheme can only tolerate no more than t compromised nodes, where the value of t is limited by the storage capacity for pairwise keys in a sensor node. Indeed, the larger a sensor network is, the more likely that an adversary compromises more than t sensor nodes and then the entire network.

The random subset assignment scheme combines the idea of key pool in [Eschenauer and Gligor 2002] with the polynomial-based key pre-distribution scheme in [Blundo et al. 1993]. Specifically, a setup server first randomly generates a pool of bivariate polynomials, each of which is uniquely identified by a polynomial ID. The setup server then chooses a random subset of polynomials and distributes the polynomial shares and the polynomial IDs to each node. To establish pairwise keys after the deployment, two sensor nodes need to identify a common polynomial they share by exchanging their polynomial IDs, and use the polynomial-based scheme to compute the pairwise key if such a common polynomial is identified. The indirect key establishment can be achieved in the same way as in [Eschenauer and Gligor 2002; Chan et al. 2003], where two sensor nodes try to find a number of intermediate nodes to help them setup a temporary session key. The analysis in [Liu

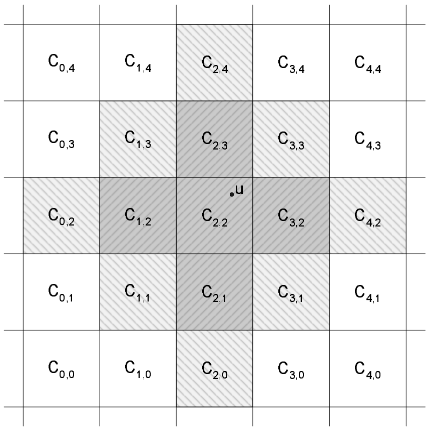


Fig. 4. Partition of a target field

and Ning 2003b] shows the random subset assignment scheme has better performance and security than the techniques in [Eschenauer and Gligor 2002; Chan et al. 2003].

2.3.2 *The Closest Polynomials Pre-Distribution Scheme.* Instead of randomly selecting polynomials for each sensor node as in the original random subset assignment scheme, the main idea of the proposed technique is to select polynomials for each sensor node based on its expected location. Specifically, we partition the target field into small areas called *cells*, each of which is associated with a unique random bivariate polynomial. Then, we distribute to each sensor node a set of polynomial shares that belong to the cells closest to the one that this sensor node is expected to locate in. For simplicity, we assume the target field is a rectangle area that can be partitioned into equal-sized squares.

(1) **Pre-Distribution:** The target field is first partitioned into a number of equal sized squares $\{C_{i_c,i_r}\}_{i_c=0,1,\ldots,C-1,i_r=0,1,\ldots,R-1}$, each of which is a *cell* with the coordinate (i_c, i_r) denoting row i_r and column i_c. For convenience, we use $s = R \times C$ to denote the total number of cells. The setup server randomly generates s bivariate t-degree polynomials $\{f_{i_c,i_r}(x, y)\}_{i_c=0,1,\ldots,C-1,i_r=0,1,\ldots,R-1}$, and assigns $f_{i_c,i_r}(x, y)$ to cell C_{i_c,i_r}. Figure 4 shows an example partition of a target field.
For each sensor node, the setup server first determines its *home cell*, in which this node is expected to locate. The setup server then discovers four cells adjacent to this node's home cell. Finally, the setup server distributes to the sensor node its home cell coordinate and the shares of the polynomials for its home cell and the four selected cells. For example, in Figure 4, node u is expected to be deployed in cell $C_{2,2}$. Obviously, cell $C_{2,2}$ is its home cell, and cells $C_{2,1}$, $C_{1,2}$, $C_{2,3}$ and $C_{3,2}$ are the four cells adjacent to its home cell. Thus, the setup server gives this node the coordinate $(2, 2)$ and the polynomial shares $f_{2,2}(u, y)$, $f_{2,1}(u, y)$, $f_{1,2}(u, y)$, $f_{2,3}(u, y)$, and $f_{3,2}(u, y)$.

(2) **Direct Key Establishment:** After deployment, if two sensor nodes want to setup a pairwise key, they first need to identify a shared bivariate polynomial. If they can find at least one such polynomial, a common pairwise key can be established directly using the basic polynomial-based key pre-distribution presented in Section 2.3.1. A simple way is to let the source node disclose its home cell coordinate to the destination node.

From the coordinate of the home cell of the source node, the destination node can immediately determine the IDs of polynomial shares the source node has.

(3) **Sensor Addition and Revocation:** To add a new sensor node, the setup server only needs to pre-distribute the related polynomial shares and the home cell coordinate to the new node, in the same way as in the pre-distribution phase. The revocation method is also straightforward. Each node only needs to remember the IDs of the compromised sensor nodes that share at least one common bivariate polynomial with itself. Thus, in addition to the polynomial shares, the sensor node also needs to store a number of compromised sensor node IDs. If more than t nodes that share the same bivariate polynomial are compromised, a non-compromised sensor node that has a share of this polynomial simply removes the corresponding share and all the related compromised sensor node IDs from its memory.

2.3.3 *Probability of Establishing Direct Keys.* Similar to the analysis for the closest pairwise keys scheme, we also use $\dot{u}$ and $\bar{u}$ to represent the actual deployment location and the expected location of node u, respectively.

Consider two sensor nodes u and v. Since they are deployed independently, given the expected location of u and v, the conditional probability that they are neighbors can be calculated by

$$p(||\dot{u},\dot{v}|| \le d_r|\bar{u},\bar{v}) = \int\int\int\int_{||\dot{u},\dot{v}||\le d_r} \epsilon_{\bar{u}}(x_1,y_1)\epsilon_{\bar{v}}(x_2,y_2)\mathrm{d}x_1\mathrm{d}y_1\mathrm{d}x_2\mathrm{d}y_2,$$

where $||\cdot||$ denotes the distance between two locations.

Assume these two sensor nodes u and v are expected to be deployed in cell C_{i_c,i_r} and C_{j_c,j_r}, respectively. To simplify our analysis, we assume that a sensor node is expected to locate randomly in its home cell. In other words, if sensor v is expected to be in cell C_{j_c,j_r}, then the probability density function for the expected location of v is $\frac{1}{L^2}$ for any location in the cell, and 0 otherwise. Therefore, the conditional probability that u and v are in each other's signal range given that u is expected to be deployed at location $\bar{u}$ and v is expected to be deployed in cell C_{j_c,j_r} can be calculated by

$$p(||\dot{u},\dot{v}|| \le d_r|\bar{u},C_{j_c,j_r}) = \int\int_{C_{j_c,j_r}} \frac{p(||\dot{u},\dot{v}|| \le d_r|\bar{u},(x,y))}{L^2}\mathrm{d}x\mathrm{d}y.$$

Hence, given u and v's home cells C_{i_c,i_r} and C_{j_c,j_r}, the probability of nodes u and v being able to directly communicate with each other can be estimated by

$$p(||\dot{u},\dot{v}|| \le d_r|C_{i_c,i_r},C_{j_c,j_r}) = \int\int_{C_{i_c,i_r}} \frac{p(||\dot{u},\dot{v}|| \le d_r|(x,y),C_{j_c,j_r})}{L^2}\mathrm{d}x\mathrm{d}y.$$

Assume that on average, N_{cell} sensor nodes are expected to be deployed in each cell. Thus, among all the sensor nodes with home cell C_{j_c,j_r}, the average number of sensor nodes that the sensor node u with home cell C_{i_c,i_r} can directly communicate with can be estimated by $N_{cell} \times p(||\dot{u},\dot{v}|| \le d_r|C_{i_c,i_r},C_{j_c,j_r})$. Therefore, overall, the average number of sensor nodes that u can directly communicate with can be estimated by

$$n_u = N_{cell}\cdot\sum_{\forall C_{j_c,j_r}} p(||\dot{u},\dot{v}|| \le d_r|C_{i_c,i_r},C_{j_c,j_r}).$$

Let $\mathcal{S}_{i_c,i_r}$ denotes the set of the home cells of the sensor nodes that share at least one common polynomial with the node whose home cell is C_{i_c,i_r}. According to the pre-distribution procedure, there are 13 such cells in each $\mathcal{S}_{i_c,i_r}$. For example, Figure 4 shows $\mathcal{S}_{2,2}$, which consists of all the shaded cells. Thus, the average number of neighbor sensor nodes that can establish a common key with u directly can be estimated by

$$n'_u = N_{cell} \cdot \sum_{C_{j_c,j_r} \in \mathcal{S}_{i_c,i_r}} p(||\dot{u}, \dot{v}|| \le d_r | C_{i_c,i_r}, C_{j_c,j_r}).$$

Hence, the probability of establishing a common key directly between u and a neighbor node of u can be estimated by

$$p = \frac{n'_u}{n_u} = \frac{\sum_{C_{j_c,j_r} \in \mathcal{S}_{i_c,i_r}} p(||\dot{u}, \dot{v}|| \le d_r | C_{i_c,i_r}, C_{j_c,j_r})}{\sum_{\forall C_{j_c,j_r}} p(||\dot{u}, \dot{v}|| \le d_r | C_{i_c,i_r}, C_{j_c,j_r})}.$$

Similar to the analysis for the closest pairwise keys scheme, the above p can be used to estimate the probability of any node having a direct key with its neighbor node when the target field is an infinite field. For a limited field in our simulation, we simply use the probability p of the node with home cell $C_{C/2,R/2}$ having a direct key with its neighbor node to estimate the probability of having a direct key between any two neighbor nodes.

We use the simple deployment model described before to evaluate the performance, with signal range d_r as the basic unit for distance measurement ($d_r = 1$). Figure 5 shows the probability of establishing direct keys for different cell side length L and maximum deployment error e. Obviously, the probability of establishing direct keys increases as the cell side length L grows and decreases as the maximum deployment error e grows.

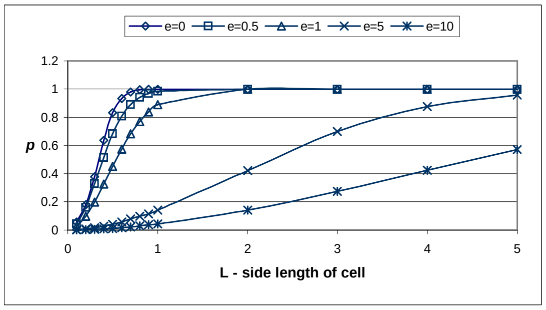
In general, the larger L is, the higher the probability of establishing a direct key between two neighbor nodes. However, the larger cell side length also leads to a larger number of sensor nodes sharing the same bivariate polynomial, which in turn degrades the security performance. Thus, we have to find the minimum value of L to meet the other constraints so that we can maximize the security performance. Figure 5 provides a guideline to determine the minimum value of L given the other constraints.

2.3.4 *Security against Node Captures.*

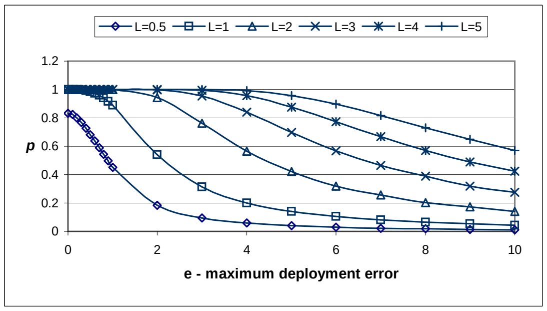
According to the result of the polynomial-based key pre-distribution in [Blundo et al. 1993], as long as no more than t polynomial shares of a bivariate polynomial are disclosed, an attacker knows nothing about the pairwise keys established through this polynomial between non-compromised nodes. Thus, the security of our scheme depends on the average number of sensor nodes sharing the same polynomial, which is equivalent to the number of sensor nodes that are expected to be located in a cell and its four adjacent cells.

As discussed in Section 2.2, the density of the sensor nodes in the network can be estimated by $D = \frac{m+1}{\pi d_r^2}$. The average number of sensor nodes that are expected to be located in a cell is $\frac{(m+1)L^2}{\pi d_r^2}$. Thus, the average number of sensor nodes that share the polynomial of a particular cell can be estimated by $N_s = \frac{5(m+1)L^2}{\pi d_r^2}$. Using the signal range as the basic unit of distance measurement ($d_r = 1$), we have $N_s = \frac{5(m+1)L^2}{\pi}$.

We consider two types of attacks against the closest polynomials pre-distribution scheme. One is the *localized attack*, which targets at the sensor nodes in a particular area in order to compromise the communication security in this area. The other is the *random attack*, which randomly selects sensor nodes to compromise.



(a)



(b)

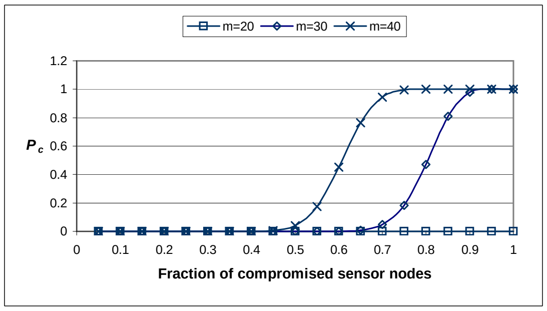
Fig. 5. Probability of establishing direct keys between two neighbor nodes given different cell side length L and maximum deployment error e

In a localized attack, the attacker must compromise more than t out of N_s sensor nodes in order to compromise the direct keys between non-compromised sensor nodes in that area. In addition, the compromise of a particular area does not affect the direct keys in any other area because all bivariate polynomials are chosen randomly and independently.

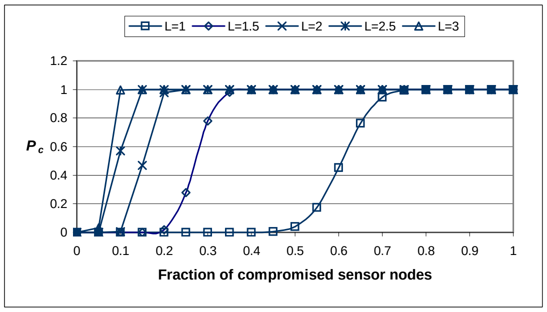
Consider a random attack. We assume a fraction p_c of sensor nodes in the network have been compromised by an attacker. This means that each sensor node has the probability of p_c being compromised. Thus, among N_s sensor nodes that have polynomial shares of a particular cell, the probability that exactly i sensor nodes have been compromised can be estimated by

$$P_c(i) = \frac{N_s!}{(N_s - i)!i!} p_c^i (1 - p_c)^{N_s - i}.$$

Therefore, the probability that the bivariate polynomial assigned to this cell is compromised, which is equivalent to the probability that a direct key between two non-compromised nodes being compromised, can be estimated by $P_c = 1 - \sum_{i=0}^{t} P_c(i)$. Figure 6 includes the relationship between the fraction of compromised direct keys for non-compromised sensor nodes and the fraction of compromised nodes under different combination of m and L given the storage capacity that is equivalent to 200 cryptographic keys ($t = 39$). An interesting result is that regardless of the total number of sensor nodes in the network, the less the density of the sensor network, the higher the security guarantee it can provide.



(a) L=1



(b) m=40

Fig. 6. Fraction of compromised direct keys between non-compromised sensor nodes v.s. fraction of compromised sensor nodes. Assume each node has available storage equivalent to 200 cryptographic keys.

2.3.5 *Overhead.* In this scheme, each sensor node needs to store the coordinate of its home cell and the polynomial shares of five cells. The storage overhead for the coordinate of its home cell is negligible. Thus, each sensor node needs to allocate $5(t+1)\log q$ memory space to store the secret. When there are compromised sensor nodes, each non-compromised sensor node also needs to store the IDs of the compromised sensor nodes with which it shares at least one common polynomial. However, for each of the 5 polynomials, a non-compromised sensor node only needs to store up to t IDs; it can remove the corresponding polynomial share and all the related IDs if the number of compromised sensor nodes sharing the polynomial exceeds t.

To establish a common key between two neighbor nodes, one of them initiates a request by disclosing its home cell coordinate. Once the other party receives such a message, it can immediately determine the common pairwise key and reply a message to identify the corresponding key. Thus, the communication overhead includes two short messages.

To compute the common key with a given sensor node, each sensor node needs to evaluate a t-degree polynomial. Thus, the computation overhead in each sensor node mainly comes from the evaluation of this polynomial, which can be done efficiently by using the optimization technique in [Liu and Ning 2003b].

2.3.6 *Improvements.* Compared with the original random subset assignment scheme in [Liu and Ning 2003b], the closest polynomials pre-distribution scheme can achieve bet-

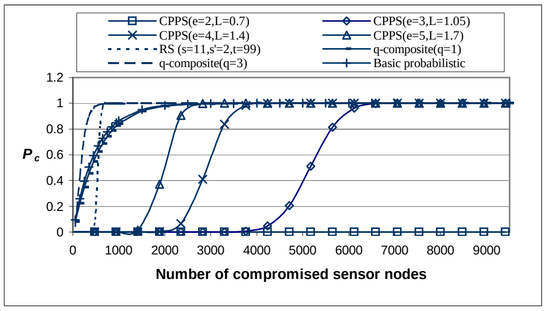


Fig. 7. Fraction of compromised direct keys between non-compromised sensor nodes v.s. number of compromised sensor nodes. Assume each node has available storage equivalent to 200 cryptographic keys. Assume $p = 0.33$ and $m = 40$. CPPS denotes the closest polynomials pre-distribution random scheme.

ter performance due to the explicit usage of expected locations. First, given certain storage constraint and the required probability of sharing direct keys between sensor nodes, the random subset assignment can only tolerate a small number of compromised sensor nodes, while the closest polynomials pre-distribution scheme can tolerate a large fraction of compromised nodes. Figure 7 shows that the security can be improved significantly by using prior deployment knowledge of sensor nodes. (To save space, this figure also includes the security performance of other techniques, which will be discussed in the later comparison.) Second, the probability of sharing direct keys between sensor nodes in the random subset assignment scheme is fixed given certain polynomial pool size and the storage constraint on sensor nodes, while for the closest polynomials pre-distribution scheme, this probability is independent from the total number of polynomials in the pool. Indeed, it can be further improved by increasing cell side length L for a given maximum deployment error e as shown in Figure 5.

2.3.7 *Comparison.* Now let us compare our scheme in this subsection with the previous methods (the basic probabilistic scheme [Eschenauer and Gligor 2002], the q-composite scheme [Chan et al. 2003], the random pairwise keys scheme [Chan et al. 2003], the grid-based scheme [Liu and Ning 2003b], and the closest pairwise keys scheme). Evaluation of those schemes requires the network size. To be fair, we use the following method to estimate the network size. Assume on average, there are m sensor nodes that fall into each sensor's signal range. Based on the analysis in [Chan et al. 2003], we estimate the total number of sensor nodes in the network is $N = 2^{mp}$ to make sure the network is fully connected with a high probability if the node only contacts its neighbor nodes, where p is the probability of establishing a direct key between two neighbor sensor nodes.

Let us first compare our new scheme with the basic probabilistic scheme [Eschenauer and Gligor 2002] and the q-composite scheme [Chan et al. 2003]. Figure 7 compares the fraction of compromised direct keys shared between non-compromised sensor nodes given the same p, m, and storage overhead. We can see that our scheme performs significantly better than the other two schemes. It also shows that the more precise the sensor deployment is, the higher security it can guarantee.

We then compare our new scheme in this subsection with the random pairwise keys scheme [Chan et al. 2003]. By limiting the number of sensor nodes sharing the same

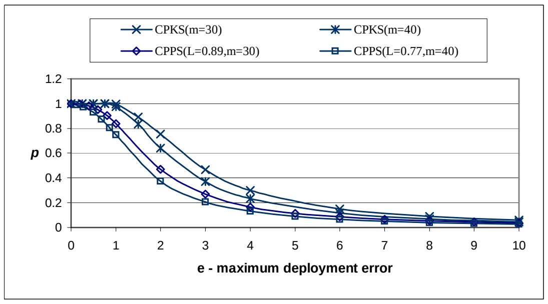


Fig. 8. Probability of establishing pairwise key directly between two neighbor nodes given different e and m. The length of cell side in CPPS is configured so that it is perfectly resistant to the node captures. Assume each node has available storage equivalent to 200 cryptographic keys.

bivariate polynomial, our proposed scheme can be modified to provide perfect security against node captures. In this case, we have $N_s = \frac{5(m+1)L^2}{\pi} \leq (t+1)$. From the previous result, we know that the value of p only depends on L and e. Thus, given the same probability of establishing direct keys between sensor nodes, our proposed scheme has no limit on the total number of sensor nodes it can support. However, the random pairwise key scheme can only support at most $\frac{c}{p}$ sensor nodes, where c is the number of cryptographic keys a sensor node stores [Chan et al. 2003]. Thus, our new scheme can achieve better performance when the expected location information is available.

Similar to the closest pairwise keys scheme, the closest polynomials pre-distribution scheme has some advantages over the grid-based scheme in [Liu and Ning 2003b]. First, in order to provide certain security guarantee against node compromise attacks, the grid-based scheme has limitations on the maximum supported network size give certain storage constraint as pointed out in [Liu and Ning 2003b]. However, the closest polynomials pre-distribution scheme has no limitations on the maximum supported network size given reasonable maximum deployment errors. Second, the closest polynomials pre-distribution scheme provides higher probability of establishing direct keys between sensor nodes than the grid-based scheme. Though the grid-based scheme can guarantee to establish a direct or indirect key between any two sensor nodes, it requires any two sensor nodes can communicate with each other, which may not be true in real applications.

Now we compare our new scheme proposed in this subsection with the closest pairwise keys scheme in Section 2.2. For the closest pairwise key pre-distribution, given a fixed storage capacity c, signal range d_r, node density D, and the maximum deployment error e, the probability of establishing direct keys between sensor nodes is fixed. However, for our new scheme proposed in this subsection, given the above constraints, it can still achieve arbitrary high probability to establish direct keys between sensor nodes by increasing the cell side length L as shown in Figure 5. For example, in the closest pairwise keys scheme, if $\gamma = 5$, $e = 3$, the probability of having a common pairwise key between two neighbor nodes is 0.4. In contrast, our new scheme allows us to increase cell side length to achieve a higher probability of establishing direct keys between neighbor sensor nodes and still provide certain degree of security.

An advantage of the closest pairwise key scheme is that the compromise of sensor nodes does not lead to the compromise of direct keys shared between non-compromised sensor nodes. By having $N_s \leq (t+1)$, the closest polynomials pre-distribution scheme can also provide this security property. To further compare these two schemes under this condition, Figure 8 shows the probabilities of establishing direct keys under different node densities and maximum deployment errors, assuming the storage capacity is equivalent to 200 cryptographic keys. We can see that although the closest pairwise keys scheme has a higher probability to establish direct keys between neighbor sensor nodes, our new scheme is not significantly worse. Considering the flexibility to trade-off security and performance in the closest polynomials pre-distribution scheme, we conclude that this scheme is more desirable than the closest pairwise keys scheme in certain applications.

3. KEY PRE-DISTRIBUTION USING POST-DEPLOYMENT KNOWLEDGE

In this section, we propose to take advantage of the post-deployment knowledge of sensor nodes to improve the pairwise key pre-distribution in static sensor networks. The main idea is to assign each sensor node an excessive amount of pre-distributed keys by using the memory for sensing applications, prioritize the pre-distributed keys based on post-deployment knowledge, and discard low priority keys to thwart node compromise attacks and return memory to the applications. We call this process *key prioritization*.

We do not assume any prior knowledge of sensors' locations. However, we assume that every sensor node can discover its real deployment location securely after the deployment of sensor networks. This assumption is practical. As pointed out in [Akyildiz et al. 2002], "most of the sensing tasks require the knowledge of positions," and "location finding systems are also required by many of the proposed sensor network routing protocols." Indeed, there have been a series of recent advances in determining individual sensor nodes' positions (with a global positioning system (GPS) or local references) [Li and Halpern 2001; Niculescu and Nath 2001] as well as securing location discovery [Sastry et al. 2003; Lazos and Poovendran 2004; Du et al. 2005; Liu et al. 2005a; 2005b]. Thus, we believe that in many sensor network applications, it is possible for the sensor nodes to determine their deployment locations securely.

Using memory for applications to store an excessive amount of pre-distributed keys is practical in sensor networks. Though sensor nodes are memory constrained, EEPROM, which is usually used to save sensed data, is much more plentiful than RAM on a sensor node. For example, a typical MICA2 mote [Crossbow Technology Inc.] comes with 512KB EEPROM, but only 4KB RAM. Thus, we may store an excessive amount of keying materials. However, in this situation, the compromise of a sensor node reveals more secrets in the network. To deal with this problem, we propose to remove the keying materials that are less likely to be used (based on the post-deployment knowledge). We further assume that an attacker cannot recover the removed keys at sensor nodes even if these nodes are compromised later. Moreover, the removal of low priority keys also returns memory to sensing applications, which may be desirable in certain scenarios.

For the sake of presentation, we refer to the pre-distributed keying materials used in a key pre-distribution scheme (e.g., [Eschenauer and Gligor 2002; Chan et al. 2003; Du et al. 2003; Liu and Ning 2003b; Zhu et al. 2003; Liu and Ning 2003c]) as key units. More specifically, a *key unit* is a minimal piece of keying material from which a valid key can be derived. A key unit in the probabilistic key pre-distribution scheme [Eschenauer and Gligor 2002], the q-composite scheme [Chan et al. 2003], or the random pairwise keys scheme [Chan et al. 2003] is simply a pre-distributed key. In the polynomial pool-based key pre-distribution schemes [Liu and Ning 2003b], a key unit is a t-degree polynomial from which a node can compute keys shared with others. In the pairwise key pre-distribution scheme presented in [Du et al. 2003], a key unit is a row of the secret matrix A_i in a key space S_i. A common property of the key units in all these schemes is that *two sensor nodes sharing the same or relevant key units can derive a common key*.

In the following, we first present an approach of key prioritization in static sensor networks, and then show how to improve the random subset assignment scheme [Liu and Ning 2003b] using this approach.

3.1 Key Prioritization Using Post-Deployment Knowledge

By using memory for sensing applications, a sensor node can keep a large number of key units during key pre-distribution. By prioritizing these key units based on post-deployment knowledge, a sensor node can give up the key units that are less likely to be used for pairwise key establishment to thwart node compromise attacks and return the memory to the sensing applications. As a result, it has a higher probability to keep those key units that may be required for secure communications with its neighbor nodes.

Specifically, we prioritize pre-distributed key units based on the deployment locations of sensor nodes. In order to do so, we map each key unit to a location in the sensor network field before deployment. After the sensor network is deployed, if a sensor node can discover its location, it can prioritize the pre-distributed key units based on this location. The node may rank all the key units according to the distances between its location and the locations of key units so that the closer a key unit is from the sensor node, the higher priority it has. As a result, sensor nodes close to each other are more likely to keep a common key unit than those that are far away from each other, and thus have a higher probability to establish a common key.

An attractive feature of using deployment locations is that there is almost no overhead. Once determining its location, a node only needs to perform simple computation to rank the pre-distributed key units, and no communication with other sensor nodes is required. Moreover, this approach allows incremental deployment of sensor nodes, since the only information a sensor node needs to prioritize its key units is its own location.

The approach described above can be used to improve many key pre-distribution techniques (e.g. the basic probabilistic scheme [Eschenauer and Gligor 2002], the q-composite scheme [Chan et al. 2003], the random subset assignment scheme in [Liu and Ning 2003b]). However, the random pairwise keys scheme is based on a different approach [Chan et al. 2003], where each key is related to two particular sensor nodes. This makes it useless to apply the above approach in such a scheme. Nevertheless, the random pairwise keys scheme can still have a better performance by loading an excessive amount of keys in the memory for sensing applications. Since it provides perfect security guarantee, it is unnecessary to thwart node compromise attacks by removing low priority keys from memory when the applications have enough memory.

3.2 Improving Random Subset Assignment Scheme with Deployment Locations

In the random subset assignment scheme, the more polynomial shares a node has, the more likely it can establish a common key with other sensor nodes. The improved scheme reuses the memory for sensing applications to keep more polynomial shares during key pre-distribution, gives higher priority to the polynomial shares that are most likely to be used after the deployment location is known, and discards low-priority polynomial shares to thwart node compromise attacks and returns memory to the applications.

3.2.1 *The Improved Scheme.* The details of the improved scheme are described below. The improvements are mainly in the key pre-distribution and key prioritization phases.

(1) **Key Pre-Distribution:** The key pre-distribution phase consists of two stages. In the first stage, the setup server randomly generates a set $\mathcal{F}$ of bivariate t-degree polynomials, and associates each polynomial with a unique location in the target field. These locations are evenly distributed over the entire target field. For the sake of presentation, we use $f_{x,y}$ to denote the bivariate polynomial in $\mathcal{F}$ associated with the location coordinate (x, y). For convenience, we also use the location coordinates as the IDs of the corresponding bivariate polynomials. In the second stage, for each sensor node, the setup server randomly picks a set of c bivariate polynomials from the polynomial pool, and distributes the corresponding polynomial shares as well as their locations to the sensor node.

(2) **Key Prioritization:** After deployment, each sensor node first determines its location. Then based on this location and the locations associated with the pre-distributed polynomial shares, the sensor node ranks the polynomial shares in terms of their distances to its deployment location. The closer a polynomial share to this sensor node, the higher priority it has. For simplicity, we assume the sensor node chooses the highest c' polynomial shares to save, where c' is the number of shares it keeps in the memory reserved for keys. (A sensor node may keep more polynomial shares until the sensing applications require the corresponding memory. However, as we will show in our analysis, this makes the sensor network more vulnerable to node compromises.)

(3) **Direct Key Establishment:** This phase can be performed in the same way as in the original random subset assignment scheme [Liu and Ning 2003b]. To establish a direct key between two sensor nodes, they only need to identify a common bivariate polynomial shared between them, which can be achieved by exchanging the IDs of polynomials that they have shares of.

(4) **Addition and Revocation:** To add a new sensor node, the setup server only needs to perform the above key pre-distribution and key prioritization in the same way. The revocation method is also straightforward. Each node only needs to remember the IDs of the compromised sensor nodes that shares at least one common bivariate polynomial with itself. If more than t nodes that share the same bivariate polynomial are compromised, a non-compromised sensor node that has a share of this polynomial simply removes the corresponding share and all the related compromised sensor node IDs from its memory.

3.2.2 *Probability of Establishing Direct Keys between Neighbor Nodes.* The ideas of using pre-deployment and post-deployment knowledge to improve the performance of key pre-distribution are two independent techniques. They can combine together easily to achieve better performance and security. In the rest of this subsection, we only analyze the improvements on the performance and security introduced by using key prioritization based on post-deployment knowledge.

Similarly, we use the signal range d_r as the basic unit to measure distances ($d_r = 1$). Assume each sensor node has m neighbor sensor nodes on average. In other words, there are $(m+1)$ sensor nodes on average in an area of $\pi \cdot d_r^2 = \pi$. We further assume the size of

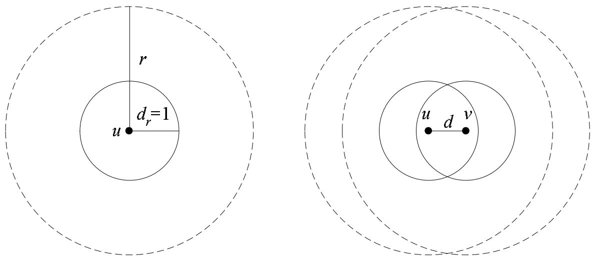


Fig. 9. Shared circles of neighbor sensor nodes

the target field is S. Then the total number of sensor nodes in the network can be estimated as $\frac{S\times(m+1)}{\pi}$.

Consider a sensor node u that gets a set of c polynomial shares during the pre-distribution phase and keeps c' of them after key prioritization. Based on the key prioritization process, node u keeps c' polynomial shares whose associated locations are no more than r away from u's location, where $r = \sqrt{\frac{S\times c'}{\pi c}} = \sqrt{\frac{c'\times N}{c\times(m+1)}}$. In other words, all the bivariate polynomial shares that fall into the circle centered at this node's location with radius r are still in the sensor node's memory. For convenience, we call such a circle the *share circle*. The left part of Figure 9 illustrates the share circle and the signal range of node u. The inner solid circle represents the area in which u can communicate with other sensor nodes directly, while the outer dashed circle represents the area in which the c' polynomial shares that u keeps fall.

Consider a pair of neighbor sensor nodes u and v. As illustrated in the right part of Figure 9, the polynomials that u and v share fall into the outer circles of both u and v. In the following, we first estimate how many polynomial shares u or v has in this area, and then estimate the probability that u and v have shares of a common polynomial (i.e., the probability that they can establish a direct key).

Assume the distance between u and v is d. Since they are neighbors, we have $d < d_r = 1$. It is easy to see that in a large sensor network, the total number of sensor nodes in the network is usually much larger than the number of neighbor nodes for a particular sensor node. Thus, we usually have $c' \times N \geq c \times (m+1)$, which implies $r \geq 1$ (since $r = \sqrt{\frac{c'\times N}{c\times(m+1)}}$). Then we can estimate the size of the overlapped area of the share circles of u and v as

$$S_o(d) = 2 \times \frac{2 \times \arccos(d/2r)}{2\pi} \times \pi \times r^2 - d \times \sqrt{r^2 - d^2/4}.$$

On average, the number of pre-distributed polynomial shares that fall into this overlapped area for both u and v can be estimated by $n(d) = \lfloor c \times \frac{S_o(d)}{S} \rfloor$. In other words, both u and v have shares of about $n(d)$ polynomials that fall into this area. We can further estimate the total number of polynomials that are distributed over this area as $n_t(d) = \lfloor |\mathcal{F}| \times \frac{S_o(d)}{S} \rfloor$. Therefore, the probability that sensor nodes u and v share a common polynomial can be estimated by

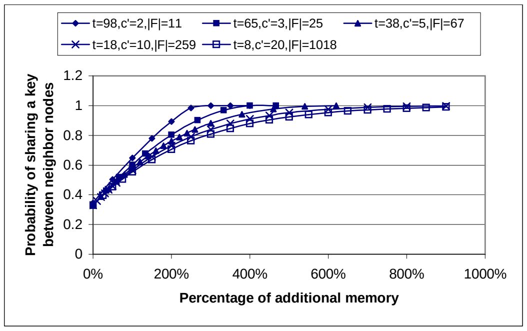


Fig. 10. Probability of sharing a direct key between neighbor sensor nodes.

$$p(d) = 1 - \prod_{i=0}^{n(d)-1} \frac{n_t(d) - n(d) - i}{n_t(d) - i}.$$

Thus, the (average) probability of sharing a polynomial between neighbor sensor nodes, which is equivalent to the probability of establishing a common direct key between two neighbor sensor nodes, can be estimated by

$$p = \int_0^{d_r} \int_0^{2\pi} \frac{p(\rho)\rho}{\pi d_r^2} \mathrm{d}\rho \mathrm{d}\theta = 2 \int_0^1 p(\rho)\rho \mathrm{d}\rho.$$

Figure 10 shows the probabilities of establishing direct keys between neighbor sensor nodes using key prioritization with different parameters. We assume the sensor network has $N = 10,000$ sensor nodes, the average number of neighbor sensor nodes is $m = 30$, and each sensor node's memory for key units after key prioritization (c') is equivalent to 200 cryptographic keys. The number of polynomials in the polynomial pool $\mathcal{F}$ is chosen to make the probability of sharing a direct key between two neighbors be 0.33 if no additional memory is allocated to store polynomial shares in the key pre-distribution phase. We can clearly see that the probability of sharing a direct key between two neighbor sensor nodes is improved significantly as the memory (c) allocated for pre-distributed polynomial shares (before key prioritization) increases.

3.2.3 *Security Analysis.* Note that there are more polynomial shares stored in a sensor node before key prioritization than after it. Thus, compromising sensor nodes before the key prioritization phase reveals more secrets than compromising the same set of sensor nodes after it. An attacker may take advantage of this observation and attack the network between the key pre-distribution and key prioritization phases. However, once a sensor node determines its location, it can finish key prioritization instantly. For convenience, we refer to the time period between pre-distributing key units to a sensor node and completing key prioritization in the sensor node the *window of vulnerability*. Intuitively, the shorter the window of vulnerability is, the less secrets may be disclosed due to compromised sensor nodes.

In the following, we evaluate the ability of the improved scheme to tolerate compromised sensor nodes in two situations.

Situation 1: No node compromises during the window of vulnerability. In this situ-

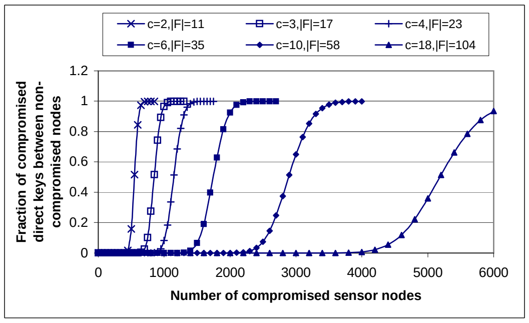


Fig. 11. Security performance of the improved scheme in situation 1.

ation, we assume the sensor network is well protected during the window of vulnerability. That is, the sensor network is assumed to be secure and none of its sensor nodes is compromised between key pre-distribution and key prioritization. After this time period, the network may be exposed to attacks.

Assume an attacker randomly compromises N_c sensor nodes in the network after the window of vulnerability. Consider any polynomial f in $\mathcal{F}$. The probability that a sensor node has a polynomial share of f is $\frac{c'}{|\mathcal{F}|}$. Then we can estimate the probability that exactly i out of N_c compromised sensor nodes have shares of this polynomial by

$$P_c(i) = \frac{N_c!}{(N_c - i)!i!}(\frac{c'}{|\mathcal{F}|})^i(1 - \frac{c'}{|\mathcal{F}|})^{N_c - i}.$$

According to the results in [Blundo et al. 1993], an attacker can compute any key generated using a t-degree polynomial if he/she has at least $t+1$ distinctive shares of this polynomial. Thus, the probability of f being compromised is $P_c = 1 - \sum_{i=0}^{t} P_c(i)$. Since f is any polynomial in $\mathcal{F}$, the fraction of compromised direct keys between non-compromised sensor nodes can be estimated as P_c. The ratio $\frac{c'}{|\mathcal{F}|}$ in this formula also implies that given the same value of t, the security performance against random node compromises will be the same if the ratio between the number of shares stored in one sensor node after key prioritization and the total number of polynomials in the polynomial pool is the same. Let us revisit Figure 10. Every line in this Figure has the same ratio $\frac{c'}{|\mathcal{F}|}$ (and thus the same security performance against random node compromises). The shape of each line indicates that by using more extra memory for key units before key prioritization, we can significantly improve the probability of sharing common keys between neighbor sensor nodes without reducing the security.

Figure 11 shows the security performances of the improved scheme under different conditions. Following [Liu and Ning 2003b], we evaluate the security performance using the fraction of compromised direct keys between non-compromised sensor nodes given a number of compromised sensor nodes. We assume each sensor node keeps key materials equivalent to 200 keys after key prioritization, the probability of sharing a common key between two sensor nodes is 0.33, and each sensor node only keeps $c' = 2$ key units after key prioritization. Parameter c represents the number of key units distributed to each sensor node during key pre-distribution. In the special case of $c = 2$, the improved scheme becomes the original random subset assignment scheme proposed in [Liu and Ning 2003b]. (Note that for different values of c, we need different number of polynomials in the polynomial pool to maintain the same probability of sharing common keys between neighbor sensor nodes.) Figure 11 shows that when c increases by 1, the vertical line shifts to the right significantly. This means the security is improved significantly as the additional memory for polynomial shares at the pre-distribution stage increases.

Situation 2: Limited node compromises during the window of vulnerability. In the second situation, we assume an attacker is able to compromise up to N_t sensor nodes after key pre-distribution but before key prioritization. We consider the worst case in the following analysis, that is, the attacker has compromised N_t sensor nodes. Assume the attacker randomly compromises N_c sensor nodes after the key prioritization phase. Consider any polynomial f in $\mathcal{F}$. The probability that a sensor node compromised before key prioritization has a polynomial share of f is $\frac{c}{|\mathcal{F}|}$. Similarly, the probability that a sensor node compromised after key prioritization has a polynomial share of f is $\frac{c'}{|\mathcal{F}|}$. Thus, the probability that exactly i compromised sensor nodes have polynomial shares of f can be calculated by

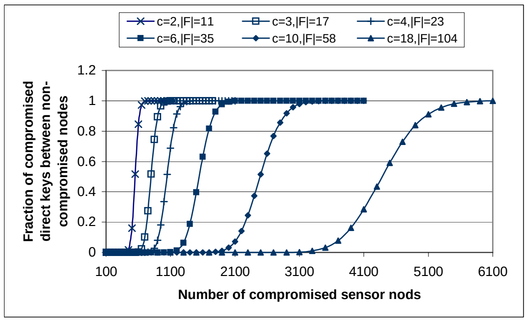
$$P_c(i) = \sum_{j+k=i} \frac{N_t!}{(N_t-j)!j!}(\frac{c}{|\mathcal{F}|})^j(1-\frac{c}{|\mathcal{F}|})^{N_t-j}\frac{N_c!}{(N_c-k)!k!}(\frac{c'}{|\mathcal{F}|})^k(1-\frac{c'}{|\mathcal{F}|})^{N_t-k}.$$

Therefore, the probability of this polynomial being compromised can be estimated as $P_c = 1 - \sum_{i=0}^{t} P_c(i)$. Since f is any polynomial in the polynomial pool, the fraction of compromised direct keys between non-compromised sensor nodes can be also estimated as P_c.

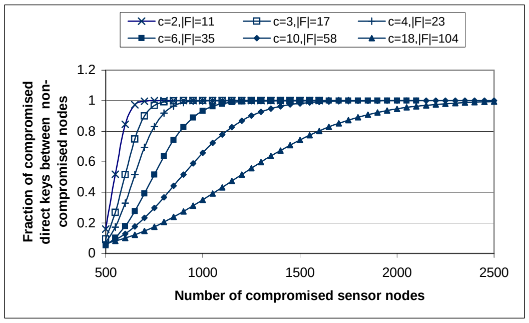
Figure 12 shows the security performance of the improved scheme when an attacker compromises a few sensor nodes before the sensor nodes finish key prioritization. Similar to situation 1, we assume each sensor node keeps key materials equivalent to 200 keys after key prioritization, the probability of sharing a common key between two sensor nodes is 0.33, and each sensor node only keeps $c' = 2$ key units after key prioritization. When there are only a few number of compromised sensor nodes before the key prioritization phase (e.g., 100 nodes in Figure 12(a)), the security performance is enhanced significantly by increasing the number of pre-distributed polynomial shares at the pre-distribution phase. When there are too many compromised sensor nodes before the key prioritization phase (e.g., 500 nodes in Figure 12(b)), the security performance can still be improved by increasing the number of polynomial shares at the pre-distribution phase, but it is not as significant as in the previous case. In general, the fewer sensor nodes compromised before key prioritization, the more improvement we can achieve by increasing the number of pre-distributed polynomial shares in the key pre-distribution phase.

Since a sensor node can finish key prioritization almost instantly after it determines its location, we believe the sensor nodes can be protected fairly well during the window of vulnerability. Indeed, a sensor node is not vulnerable at the same level during the entire window of vulnerability. The most vulnerable period of time is the period after deployment and before key prioritization, which is actually quite short due to the ease of completing key prioritization. Thus, we believe that it is unlikely that an attacker can compromise a large number of sensor nodes before they finish key prioritization.

3.2.4 *Overheads.* Since the improved scheme uses application memory only before key prioritization, it has almost the same overheads as the original scheme. Specifically,



(a) $N_t = 100$



(b) $N_t = 500$

Fig. 12. Security performance of the improved scheme in Situation 2.

each sensor node has to store c polynomial shares and the locations associated with these shares in the key pre-distribution phase, and only keeps c' of them after determining its deployment location. A location coordinate can be represented with integers much smaller than a polynomial share (e.g., 4 bytes to encode x or y coordinate). Each location coordinate takes approximately the same space as one key. Thus, the storage overhead in a sensor node is approximately $c(t+2)\log q$ in the key pre-distribution phase. Similarly, after key prioritization, the storage overhead for key units is about $c'(t+2)\log q$. In addition, each sensor node needs to remember the IDs of compromised sensor nodes with which this node shares at least one common polynomial. This introduces at most $c't\log q'$ storage overhead after key prioritization, where q' is a number that is large enough to accommodate all sensor nodes in the network. To establish a common key with a given neighbor node, two sensor nodes only need to exchange their polynomial IDs. To compute a pairwise key, both sensor nodes need to evaluate a t-degree polynomial, which can be done efficiently with the optimization technique in [Liu and Ning 2003b].

4. RELATED WORK

A number of techniques have been proposed to establish pairwise keys in resource constrained sensor networks. A basic probabilistic key pre-distribution 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 non-compromised 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, Du et al. [2003] extended Blom's scheme [Blom 1985] to provide resilient key pre-distribution through multiple instances of Blom's scheme. Independently, Liu and Ning [2003b] as well as Liu et al. [2004] extended Blundo's scheme [Blundo et al. 1993] into a number of resilient schemes, which achieve resilience against node compromises through either a random selection among multiple instances of Blundo's scheme (random subset assignment scheme), an artificial grid (grid-based scheme), or a multi-dimension hypercube (hypercube-based scheme). All of these techniques provide a nice threshold property, which enables the resilience property against node compromises. In this paper, we demonstrate that the performance and security of these schemes can be further improved significantly by using pre-deployment and post-deployment knowledge.

Du et al. independently developed a scheme using pre-deployment knowledge [Du et al. 2004] when a preliminary version of this paper [Liu and Ning 2003c] was prepared. Du et al. improved the basic probabilistic key pre-distribution using the expected locations of sensor nodes. This paper includes additional results on improving the random pairwise keys scheme [Chan et al. 2003] and the random subset assignment scheme [Liu and Ning 2003b] with pre-deployment knowledge. In addition, this paper develops a novel technique to further improve key pre-distribution with post-deployment knowledge.

Anderson et al. proposed to establish keys based on multiple insecure wireless links [Anderson et al. 2004]. This technique can build up resilient sensor networks as long as no more than a certain fraction of communications are eavesdropped during the window of vulnerability. In contrast, our key prioritization technique is still secure even if an attacker can eavesdrop every communication link. Indeed, the attacker has to physically capture a certain number of sensor nodes during or after the window of vulnerability in order to compromise the communication between sensor nodes. Recently, Chan and Perrig [2005] also used an artificial grid (similar to the grid-based scheme [Liu and Ning 2003b] and the hypercube-based scheme [Liu et al. 2004]) to facilitate key establishment using peer sensor nodes as trusted intermediaries. In addition to the grid structure, they also consider the communication overhead and the routing between nodes during the selection of peer sensor nodes.

There are many other related studies in sensor network security, which are mostly on key management, authentication, and vulnerability analysis. Carman, Kruus, and Matt studied the performance of a number of key management approaches in sensor networks on different hardware platform [Carman et al. 2000]. Stajano and Anderson proposed to bootstrap trust between devices through location limited channels such as physical contact [Stajano and Anderson 1999]. Wong and Chan proposed to reduce the computation overhead for key exchange in low power devices with the help of a more powerful server [Wong and Chan 2001]. Basagni et al. presented a key management scheme to secure the communication by periodically updating the symmetric keys shared by all sensor nodes with the assumption of tamper-resistant hardware to protect the keys [Basagni et al. 2001]. Zhu et al. proposed a protocol suite named LEAP (Localized Encryption and Authentication Protocol) to help establish individual keys between sensors and abase station, pairwise keys between sensors, cluster keys within a local area, and a group key shared by all nodes [Zhu et al. 2003].

Perrig et al. developed a security architecture for sensor networks, which includes SNEP, a security primitive building block, and a broadcast authentication technique μTESLA [Perrig et al. 2001], an adaption of TESLA [Perrig et al. 2000; 2001; 2002] in sensor networks. Liu and Ning extended this technique to a multi-level key chain method to prolong the time period covered by a μTESLA instance [Liu and Ning 2003a]. These techniques address the critical broadcast authentication problem in sensor networks, and are complementary to the techniques in this paper.

Deng, Han and Mishra developed a collection of mechanisms to improve the security for in-networking processing [Deng et al. 2003]. Zhu et al. proposed an interleaved hop-by-hop authentication to defeat malicious data injection in sensor networks [Zhu et al. 2004]. The problem of secure data aggregation in the presence of compromised nodes was studied in [Hu and Evans 2003a] and [Przydatek et al. 2003]. Our key pre-distribution schemes can be used to address the key management issues in these techniques.

Wood and Stankovic identified a number of DoS attacks in sensor networks [Wood and Stankovic 2002]. Karlof and Wagner pointed out security goals for routing in sensor networks and analyzed the vulnerabilities as well as the countermeasures for a number of existing routing protocols [Karlof and Wagner 2003]. Sastry, Shankar and Wagner proposed a location verification technique based on the round trip time (RTT) [Sastry et al. 2003] to detect false location claims. Hu and Evans proposed to use directional antenna to detect wormhole attacks in wireless Ad Hoc networks [Hu and Evans 2003b]. Newsome et al. studied the Sybil attacks in sensor networks where a node illegitimately claims multiple identities, and developed techniques to defend against this attack [Newsome et al. 2004]. Our proposed techniques can be combined with these techniques to further enhance the security in sensor networks.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we presented three schemes to take advantage of sensor nodes' pre-deployment knowledge or post-deployment knowledge, aiming at improving pairwise key establishment in static sensor networks. The analysis shows that when sensor nodes in a network can be deployed to the expected locations with a certain precision or can discover its location after deployment, these schemes provide better security and performance over the previous solutions.

Several directions are worth future research. First, the ideas of using pre-deployment knowledge and post-deployment knowledge are two mutually orthogonal techniques. It may be desirable to combine them together to provide better performance and security. Second, we would like to investigate other pre-deployment or post-deployment knowledge to improve pairwise key pre-distribution in addition to the expected locations or the deployment locations of sensor nodes.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable comments. This work is supported by US Army Research Office (ARO) under grant W911NF-05-1-0247, and by National Science Foundation (NSF) under grant CAREER-0447761.

REFERENCES

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

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

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

BLOM, R. 1985. An optimal class of symmetric key generation systems. In *Advances in Cryptology: Proceedings of EUROCRYPT 84, Lecture Notes in Computer Science Volume 209*. 335–338.

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, LNCS 740*. 471–486.

BUCHEGGER, S. AND BOUDEC, J. L. 2002. Performance analysis of the CONFIDANT protocol (cooperation of nodes: Fairness in dynamic ad-hoc networks). In *Proceedings of The Third ACM International Symposium on Mobile Ad Hoc Networking and Computing*. 226–236.

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

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 *IEEE Symposium on Research in Security and Privacy*. 197–213.

CROSSBOW TECHNOLOGY INC. Wireless sensor networks. `http://www.xbow.com/Products/Wireless_Sensor_Networks.htm`. Accessed in May 2005.

DENG, J., HAN, R., AND MISHRA, S. 2003. Security support for in-network processing in wireless sensor networks. In *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 10th ACM Conference on Computer and Communications Security (CCS'03)*. 42–51.

DU, W., FANG, L., AND NING, P. 2005. Lad: Localization anomaly detection for wireless sensor networks. In *Proceedings of the 19th IEEE International Parallel & Distributed Processing Symposium (IPDPS '05)*.

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.

GOLDREICH, O., GOLDWASSER, S., AND MICALI, S. 1986. How to construct random functions. *Journal of the ACM 33,* 4 (October), 792–807.

GURA, N., PATEL, A., WANDER, A., EBERLE, H., AND SHANTZ, S. 2004. Comparing elliptic curve cryptography and RSA on 8-bit CPUs. In *Proceedings of Workshop on Cryptographic Hardware and Embedded Systems (CHES 2004)*.

HU, L. AND EVANS, D. 2003a. Secure aggregation for wireless networks. In *Workshop on Security and Assurance in Ad Hoc Networks*.

HU, L. AND EVANS, D. 2003b. Using directional antennas to prevent wormhole attacks. In *Proceedings of the 11th Network and Distributed System Security Symposium*. 131–141.

HU, Y., PERRIG, A., AND JOHNSON, D. 2003. Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. In *Proceedings of INFOCOM 2003*.

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

LAZOS, L. AND POOVENDRAN, R. 2004. Serloc: Secure range-independent localization for wireless sensor networks. In *ACM workshop on Wireless security (ACM WiSe 2004)*. Philadelphia, PA.

LI, L. AND HALPERN, J. 2001. Minimum-energy mobile wireless networks revisited. In *Proceedings of IEEE International Conference on Communications (ICC '01)*.

LIU, D. AND NING, P. 2003a. Efficient distribution of key chain commitments for broadcast authentication in distributed sensor networks. In *Proceedings of the 10th Annual Network and Distributed System Security Symposium (NDSS'03)*. 263–276.

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

LIU, D. AND NING, P. 2003c. Location-based pairwise key establishments for static sensor networks. In *2003 ACM Workshop on Security in Ad Hoc and Sensor Networks (SASN '03)*. 72–82.

LIU, D., NING, P., AND DU, W. 2005a. Attack-resistant location estimation in wireless sensor networks. In *Proceedings of the Fourth International Conference on Information Processing in Sensor Networks (IPSN '05)*.

LIU, D., NING, P., AND DU, W. 2005b. Detecting malicious beacon nodes for secure location discovery in wireless sensor networks. In *Proceedings of the 25th International Conference on Distributed Computing Systems (ICDCS '05)*.

LIU, D., NING, P., AND DU, W. 2005c. Group-based key pre-distribution in wireless sensor networks. In *Proceedings of 2005 ACM Workshop on Wireless Security (WiSe 2005)*.

LIU, D., NING, P., AND LI, R. 2004. Establishing pairwise keys in distributed sensor networks. *ACM Transactions on Information and System Security (TISSEC)*. To appear.

MARTI, S., GIULI, T. J., LAI, K., AND BAKER, M. 2000. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of the Sixth annual ACM/IEEE International Conference on Mobile Computing and Networking*. 255–265.

NEWSOME, J., SHI, R., SONG, D., AND PERRIG, A. 2004. The sybil attack in sensor networks: Analysis and defenses. In *Proceedings of IEEE International Conference on Information Processing in Sensor Networks (IPSN 2004)*.

NICULESCU, D. AND NATH, B. 2001. Ad hoc positioning system (APS). In *Proceedings of IEEE GLOBECOM '01*.

PERRIG, A., CANETTI, R., SONG, D., AND TYGAR, D. 2000. Efficient authentication and signing of multicast streams over lossy channels. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy*.

PERRIG, A., CANETTI, R., SONG, D., AND TYGAR, D. 2001. Efficient and secure source authentication for multicast. In *Proceedings of Network and Distributed System Security Symposium*.

PERRIG, A., CANETTI, R., SONG, D., AND TYGAR, D. 2002. The TESLA broadcast authentication protocol. In *RSA Cryptobytes*.

PERRIG, A., SZEWCZYK, R., WEN, V., CULLER, D., AND TYGAR, D. 2001. SPINS: Security protocols for sensor networks. In *Proceedings of Seventh Annual International Conference on Mobile Computing and Networks*.

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)*.

SASTRY, N., SHANKAR, U., AND WAGNER, D. 2003. Secure verification of location claims. In *ACM Workshop on Wireless Security*.

SHNAYDER, V., HEMPSTEAD, M., CHEN, B., WERNER-ALLEN, G., AND WELSH, M. 2004. Simulating the power consumption of large-scale sensor network applications. In *Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys'04)*.

STAJANO, F. AND ANDERSON, R. 1999. The resurrecting duckling: security issues for ad hoc networks. In *Proceedings of the 7th International Workshop on Security Protocols*. 172–194.

WONG, D. AND CHAN, A. 2001. Efficient and mutually authenticated key exchange for low power computing devices. In *Proceedings of ASIA CRYPT*.

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

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

ZHU, S., SETIA, S., JAJODIA, S., AND NING, P. 2004. An interleaved hop-by-hop authentication scheme for filtering false data in sensor networks. In *Proceedings of 2004 IEEE Symposium on Security and Privacy*.