Short Papers

Cluster-Based Key Predistribution Using Deployment Knowledge

Neeraj Mittal, *Member*, *IEEE Computer Society*, and Ramon Novales, *Student Member*, *IEEE*

Abstract—We present a novel key predistribution scheme that uses deployment knowledge to divide deployment regions into overlapping clusters, each of which has its own distinct key space. Through careful construction of these clusters, network resilience is improved, without compromising connectivity or communications overhead. Experimental results show significant improvement in performance over existing schemes based on deployment knowledge.

Index Terms—Wireless sensor networks, key predistribution, deployment knowledge, key agreement.

1 Introduction

Wireless sensor networks typically consist of a large collection of small, low-cost, battery-powered devices; each sensor node is equipped with integrated sensors, limited data processing capabilities, and a short-range radio. Distributed over an area of interest, they are responsible for collecting and reporting sensor data (e.g., temperature and humidity) [1], [2]. After deployment, communication channels between sensor nodes must be protected against both eavesdropping and tampering. Furthermore, the channels in the network should possess some degree of robustness against the capture of one or more nodes.

Eschenauer and Gligor proposed a key predistribution scheme [3] that assigns a random subset of keys selected from a key pool to each sensor node prior to deployment. After deployment, if two sensor nodes are within communication range of each other, they can establish a secure channel between them if they share at least one key. Otherwise, they must exchange keys using secure links between intermediary nodes.

Schemes extending Eschenauer and Gligor's basic approach have been proposed [4], [5]. Chan et al. [4] extend Eschenauer and Gligor's basic approach by proposing a q-composite scheme in which two neighboring sensor nodes have to share at least q keys, where $q \geq 1$, in order to establish a secure channel. Schemes that distribute keys in a deterministic manner have also been proposed [6], [7], [8]. For a given amount of key storage space and overlap probability, deterministic schemes tend to have better resilience than random schemes because they are able to use a larger key pool when selecting and assigning keys. Du et al. [9] and Liu et al. [10] independently proposed key predistribution schemes in which all channels are completely secure as long as the number of compromised sensor nodes does not exceed a threshold value. This property is achieved by using a pool of matrices [9], [11] or a pool of polynomials [10], [12] instead of a pool of keys. Two neighboring nodes can establish a secure channel between them if they carry information about a common matrix or a common polynomial.

• *The authors are with the Department of Computer Science, The University of Texas at Dallas, Mail Station EC31, PO Box 830688, Richardson, TX 75083. E-mail: {neerajm, rnovales}@utdallas.edu.*

Manuscript received 7 Dec. 2007; revised 11 Sept. 2008; accepted 6 July 2009; published online 4 Aug. 2009.
For information on obtaining reprints of this article, please send e-mail to: tdsc@computer.org, and reference IEEECS Log Number TDSC-2007-12-0188. Digital Object Identifier no. 10.1109/TDSC.2009.34.

Several proposed schemes improve the resilience of wireless sensor networks by taking advantage of deployment knowledge during key assignment. In region-based deployment knowledge [13], [14], [15], the deployment area is divided into a set of deployment regions, and individual sensor nodes are assigned keys based upon the deployment region to which they are assigned. In group-based deployment knowledge [16], [17], the set of sensor nodes is partitioned into a set of groups such that sensor nodes belonging to the same group are expected to be deployed together. However, knowledge regarding *relative* deployment of groups (i.e., which groups of sensor nodes are expected to reside close to each other after deployment) is not available.

In this paper, we propose a novel key predistribution scheme that makes use of region-based deployment knowledge. Our scheme constructs a set of clusters such that each cluster contains a small number of deployment regions, all of which are neighbors of each other. Furthermore, every pair of neighboring deployment regions belongs to at least one cluster. Each cluster has its own distinct key space, and it is from these cluster key spaces that nodes are assigned their keys. In this manner, we guarantee that nodes in neighboring regions share a key with a given overlap probability, while nodes in nonneighboring regions do not share any keys. We make use of the result developed in [18], which states that under certain conditions, maximizing the key pool size used by the scheme also maximizes its resilience. Our clustering scheme is designed to maximize the overall key pool size, which results in greatly improved network resilience without compromising network connectivity or communications overhead. Our experimental results indicate that our scheme has significantly higher resilience than existing schemes using region-based deployment knowledge (e.g., [14], [15]), even in the presence of deployment error. We also compare the performance of our scheme with one that uses group-based deployment knowledge to provide perfect resilience to node compromise [17]. Experimental results show that our scheme provides substantially better performance in terms of connectivity and communications overhead, especially when the deployed sensor network is sparse or key ring space is small.

The rest of the paper is organized as follows: in Section 2, we present our basic predistribution scheme. We discuss enhancements to our basic key scheme to improve its performance in Section 3. Theoretical analysis of the scheme is presented in Section 4 and experimental results are described in Section 5. We present our conclusion and directions for future research in Section 6.

2 A Novel Key Predistribution Scheme Using Deployment Knowledge

Our scheme follows the three-phase model (key predistribution, shared-key discovery, and path-key establishment) proposed by Eschenauer and Gligor [3]. We describe only the key predistribution phase, as the other two phases are unchanged.

Our model for deployment knowledge is similar to the one used by Du et al. [15], Liu and Ning [14], and Huang et al. [13]. The deployment area (or field) is partitioned into a grid of $\alpha \times \beta$ rectangular regions. Each region in the deployment area $R_{i,j}$, where $1 \leq i \leq \alpha$ and $1 \leq j \leq \beta$, is associated with a group of sensor nodes $S_{i,j}$ such that the nodes in the subset $S_{i,j}$ are expected to reside in region $R_{i,j}$ after deployment.

Neighboring regions are combined into clusters (regions may belong to more than one cluster), and each cluster has an associated key space. This key space provides key overlap between the regions within the cluster. The key pool in our scheme is thus composed of the *mutually exclusive* key spaces of all the clusters. If p

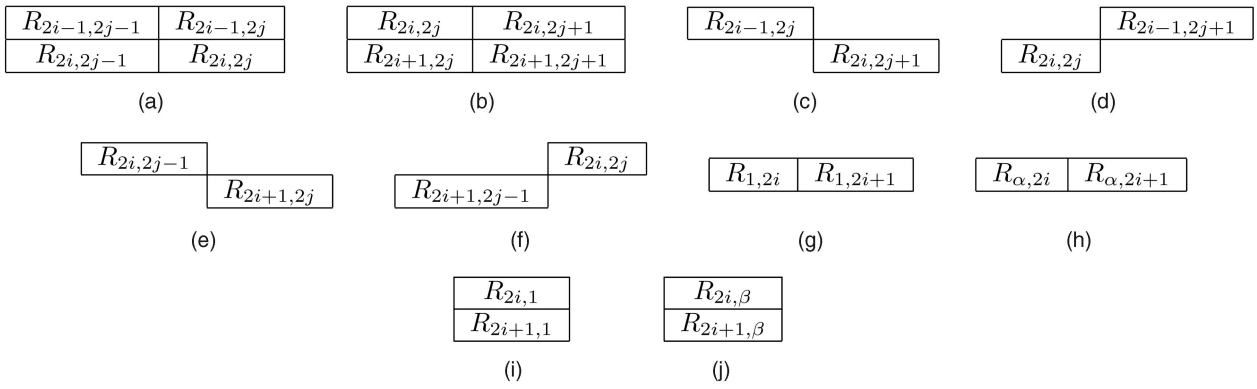


Fig. 1. Cluster types. (a) Type I. (b) Type II. (c) Type III-A. (d) Type III-B. (e) Type IV-A. (f) Type IV-B. (g) Type V-A. (h) Type V-B. (i) Type V-C. (j) Type V-D.

denotes some overlap probability, our scheme ensures that *a sensor node shares a key with a sensor node in a neighboring region with a probability of at least p*. Further, *sensor nodes belonging to nonneighboring regions do not share any keys*.

2.1 Cluster Types

Our scheme uses five types of clusters, described below. We assume that both α and β are even numbers greater than or equal to 4. We believe that this assumption is not too restrictive and can easily be ensured in practice.

2.1.1 Clusters of Type I

Each cluster of type I consists of four regions, and there are $\frac{\alpha}{2} \times \frac{\beta}{2}$ clusters of this type. The (i,j)th cluster of type I, where $1 \leq i \leq \frac{\alpha}{2}$ and $1 \leq j \leq \frac{\beta}{2}$, consists of the regions shown in Fig. 1a.

2.1.2 Clusters of Type II

Each cluster of type II consists of four regions, and there are $\frac{\alpha-2}{2} \times \frac{\beta-2}{2}$ clusters of this type. The (i,j)th cluster of type II, where $1 \leq i \leq \frac{\alpha-2}{2}$ and $1 \leq j \leq \frac{\beta-2}{2}$, consists of the regions in Fig. 1b.

2.1.3 Clusters of Type III

Each cluster of type III consists of two diagonal regions, and there are $\frac{\alpha(\beta-2)}{2}$ clusters of this type. The clusters of type III can be further divided into two subtypes, namely type III-A and type III-B. When $1 \leq i \leq \frac{\alpha}{2}$ and $1 \leq j \leq \frac{\beta-2}{2}$, the (i,j)th cluster of type III-A consists of the regions shown in Fig. 1c, and the (i,j)th cluster of type III-B consists of the regions shown in Fig. 1d.

2.1.4 Clusters of Type IV

Each cluster of type IV consists of two diagonal regions, and there are $\frac{(\alpha-2)\beta}{2}$ clusters of this type. The clusters of type IV can be further divided into two subtypes, namely, type IV-A and type IV-B. When $1 \leq i \leq \frac{\alpha-2}{2}$ and $1 \leq j \leq \frac{\beta}{2}$, the (i,j)th cluster of type IV-A consists of the regions shown in Fig. 1e, and the (i,j)th cluster of type IV-B consists of the regions shown in Fig. 1f.

2.1.5 Clusters of Type V

Each cluster of type V consists of two noncorner boundary regions, and there are $\alpha + \beta - 4$ clusters of this type. The clusters of type V can be further divided into four subtypes depending on the boundary regions contained in the cluster, namely type V-A, type V-B, type V-C, and type V-D. When $1 \leq i \leq \frac{\alpha-2}{2}$, the ith cluster of type V-A consists of the two top boundary regions shown in Fig. 1g, and the ith cluster of type V-B consists of the two bottom boundary regions shown in Fig. 1h. When $1 \leq i \leq \frac{\beta-2}{2}$, the ith cluster of type V-C consists of the two left boundary regions shown in Fig. 1i, and the ith cluster of type V-D consists of the two right boundary regions shown in Fig. 1j.

2.2 Cluster Construction

It can be shown using induction on α and β that the set of clusters we construct is *complete and nonredundant* in the sense that if regions R and S are neighbors of each other, then there is exactly one cluster C that provides key overlap among sensor nodes belonging to regions R and S; that is, $\{R, S\} \subseteq C$. The clustering scheme is designed to ensure uniform key distribution (i.e., each key has an equal probability of being assigned to a node) and to *maximize* the size of the key pool that is generated, which is given by the sum of the size of all key spaces. The results from [18] show that when key distribution is uniform, maximizing the key pool size will maximize network resilience.

It can be verified that each region belongs to at most two clusters that span four regions and at most two clusters that span two regions. Specifically, each corner region belongs to one cluster of type I. Each noncorner boundary region belongs to three clusters: one of type I, one of type III (top and bottom boundary regions) or type IV (left and right boundary regions), and one of type V. Finally, an interior (nonboundary) region belongs to four clusters: one of type I, one of type II, one of type III, and one of type IV. For example, a corner region $R_{1,1}$ belongs to the cluster $\{R_{1,1}, R_{1,2}, R_{2,1}, R_{2,2}\}$ (type I). A top boundary region $R_{1,2}$ belongs to three clusters: $\{R_{1,1}, R_{1,2}, R_{2,1}, R_{2,2}\}$ (type I), $\{R_{1,2}, R_{2,3}\}$ (type III-A), and $\{R_{1,2}, R_{1,3}\}$ (type V-A). An interior region $R_{2,2}$ belongs to four clusters: $\{R_{1,1}, R_{1,2}, R_{2,1}, R_{2,2}\}$ (type I), $\{R_{2,2}, R_{2,3}, R_{3,2}, R_{3,3}\}$ (type II), $\{R_{1,3}, R_{2,2}\}$ (type III-B), and $\{R_{2,2}, R_{3,1}\}$ (type IV-B). Fig. 2 depicts the clusters to which region $R_{2,2}$ belongs.

Cluster construction has a large impact on scheme performance. Simulation results using a naive clustering scheme (the sole cluster type is a 2×2 block of regions) show that our scheme has 24.2 percent fewer compromised channels when $c_{local} = 0.25$ and 1,000 out of 10,000 nodes are captured.

2.3 Assigning Key Rings to Sensor Nodes

We use Lee and Stinson's scheme [6] to generate the key space and corresponding collection of key sets for each cluster. A key space can be thought of as the key pool for a particular cluster; the overall key pool of the scheme is then composed of the union of all key spaces. Lee and Stinson's scheme was chosen because the results from [18] show that it is near-optimal in terms of key pool size, and therefore, near-optimal in terms of network resilience. We use the term "key set" to describe the set of keys selected from a key space and assigned to a particular node. A node's key ring is the set of all keys assigned to that node; it is composed of the union

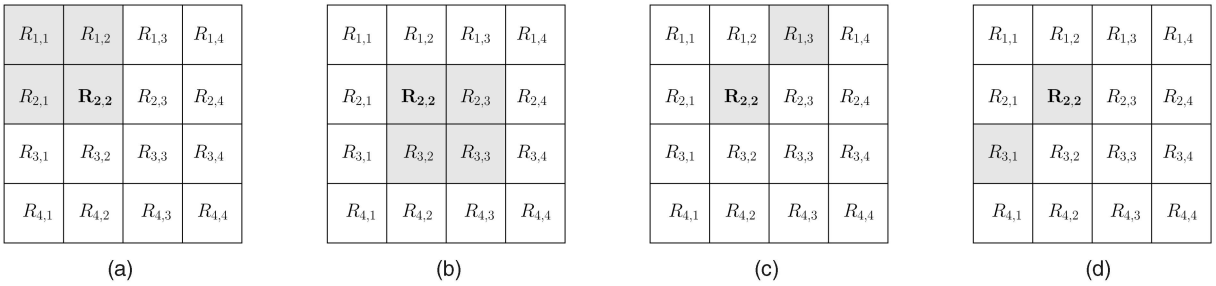


Fig. 2. Clusters to which region $R_{2,2}$ belongs: (a) Type I. (b) Type II. (c) Type III-B. (d) Type IV-B.

of key sets drawn from the key spaces of the clusters in which the node's region is a member.

For a region $R_{i,j}$, let $CS_{i,j}$ denote the set of clusters to which $R_{i,j}$ belongs. Each sensor node belonging to the region $R_{i,j}$ is assigned one key set from the key space of each cluster in $CS_{i,j}$. The key set is selected uniformly at random from the set of all key sets of a key space.

Before we can construct the key space for a cluster, we need to determine the size of a key set of the key space, i.e., the number of keys a node is assigned from the key space. An obvious initial approach is to size each key set equally. With this approach, however, keys in the key spaces for larger clusters are assigned to more nodes as compared to those for smaller clusters. Intuitively, we desire a uniform distribution of keys across nodes to avoid the creation of "high-value targets"—keys which occur much more frequently have a greater adverse effect on resilience should they be compromised.

Let K_x, for $x \in \{2,4\}$, denote the size of the key space of a cluster containing x regions. Let s_x, for $x \in \{2,4\}$, denote the size of a key set of a key space of size K_x. Since Lee and Stinson's approach is close to optimal, we can use results from [18] to obtain $K_x \approx \frac{s_x^2}{p}$ for $x \in \{2,4\}$. A sensor node in an interior region gets four key sets, two of size s_2 and two of size s_4. Since a sensor node can store at most s keys, we have $2s_2 + 2s_4 \approx s$.

A key space for a cluster containing four regions (type I or II) provides key overlap among twice as many sensor nodes as a key space for a cluster containing only two regions (type III, IV, or V). Each region contains approximately $\frac{N}{\alpha\beta}$ sensor nodes. To ensure uniform key distribution, the number of secure channels per key for a key space of size K_4 should be the same as the number of secure channels per key for a key space of size K_2. We have:

$$\frac{\frac{4N}{\alpha\beta}\left(\frac{4N}{\alpha\beta}-1\right)p}{K_4} \approx \frac{\frac{2N}{\alpha\beta}\left(\frac{2N}{\alpha\beta}-1\right)p}{K_2}.$$

Assuming that N is sufficiently large such that $(\frac{2N}{\alpha\beta}-1) \approx \frac{2N}{\alpha\beta}$, we obtain $\frac{K_4}{K_2} \approx 4$. Using the results above, we obtain $\frac{K_4}{K_2} \approx 4$, and use it, in turn, to obtain that $s_2 \approx \frac{s}{6}$ and $s_4 \approx \frac{s}{3}$.

It can be verified that a sensor node belonging to 1) a corner region stores approximately $\frac{s}{3}$ keys, 2) a noncorner boundary region stores approximately $\frac{s}{3} + \frac{2\times s}{6} = \frac{2s}{3}$ keys, and 3) an interior region stores approximately $\frac{2\times s}{3} + \frac{2\times s}{6} = s$ keys. Note that sensor nodes belonging to boundary regions do not fully utilize their space for storing keys. This is because a boundary region has fewer neighboring regions (at most five) than an interior region (exactly eight).

3 Enhancements to the Basic Scheme

3.1 Controlling Intraregion Overlap Probability

One of the parameters in our basic key predistribution scheme, p, primarily determines the overlap probability between sensor nodes belonging to neighboring regions. It also *indirectly* governs the overlap probability between sensor nodes belonging to the same region. Many times, it is desirable to have a greater degree of control over the overlap probability between sensor nodes belonging to the same region. To that end, we introduce another parameter in our scheme, denoted by q. For each region $R_{i,j}$, we construct a cluster that contains only the region $R_{i,j}$. However, the key space for such a cluster ensures that two sensor nodes share a key with probability q (not p as with earlier key spaces).

Let K_1 denote the size of such a key space and let s_1 denote the size of a key set of such a key space. Thus, $K_1 \approx \frac{s_1^2}{q}$. To ensure uniform key distribution:

$$\frac{\frac{N}{\alpha\beta}\left(\frac{N}{\alpha\beta}-1\right)q}{K_1} \approx \frac{\frac{2N}{\alpha\beta}\left(\frac{2N}{\alpha\beta}-1\right)p}{K_2}$$

implying that $\frac{K_2}{K_1} \approx 4\,\frac{p}{q}$. By combining previous results, we obtain $\frac{s_2}{s_1} \approx 2\,\frac{p}{q}$. Also, observe that $s_1 + 2s_2 + 2s_4 \approx s$. Using the previous result, $\frac{s_4}{s_2} \approx 2$, and solving for s_1, we obtain $s_1 \approx (\frac{q}{12\,p+q})s$. This, in turn, implies that $s_2 \approx (\frac{2\,p}{12\,p+q})s$ and $s_4 \approx (\frac{4\,p}{12\,p+q})s$.

3.2 Utilizing Unused Boundary Region Space

As discussed earlier, sensor nodes belonging to boundary regions do not utilize their space completely. We can make use of this unused space to store more keys in these sensor nodes. For each noncorner boundary region (i.e., top, bottom, left, and right), we construct a new cluster which spans two regions—the boundary region and its nonboundary neighbor. For corner regions, we construct 16 clusters, four for each corner region, spanning only that corner region. By making use of this unused space, the key pool size increases significantly, which increases resilience of the network without sacrificing the overlap probability. We refer to the key predistribution obtained after the two enhancements described in Sections 3.1 and 3.2 as the *enhanced scheme.*

3.3 From Key Distribution to Instance Distribution

We have so far assumed that a key corresponds to a simple symmetric cryptographic key. Therefore, once even a *single* node containing a given key is compromised, all channels encrypted using that key are compromised as well. However, there are other more sophisticated schemes which use either a matrix of size $(\lambda+1)\times(\lambda+1)$ [11] or a bivariate polynomial of degree λ [12] to compute a pairwise key between two sensor nodes. In these schemes, all channels are completely secure as long as λ or fewer nodes have been compromised. Once $\lambda+1$ nodes have been compromised, the scheme is completely broken—all channels are compromised.

Du et al. [9], [15], Liu et al. [10], [16], and Yu and Guan [19] describe key predistribution schemes in which the key pool consists of multiple instances of these cryptographic schemes (matrix or polynomial-based). Each sensor node carries a subset of the instances selected from the instance pool. If two sensor nodes share at least one instance, then the common instance can be used to

establish a pairwise key among the nodes. The two schemes can be simply seen as a kind of key predistribution scheme in which *a key corresponds to an instance of a matrix- or polynomial-based scheme* [14].

4 Theoretical Analysis of the Scheme

4.1 Shared Key Discovery

To determine whether two neighboring sensor nodes share a common key, they only need to exchange information about the key spaces from which their keys were drawn. Every key space can be uniquely described by the following: 1) The type of the cluster corresponding to the key space, 2) the index of the cluster within the clusters of that type, and 3) the index of the key set within the key space; for Lee and Stinson's scheme, this is the (i, j) tuple. Therefore, sensor nodes only need to exchange $O(1)$ amount of information to support shared key discovery—each node possesses at most seven key sets, each of which can be described by a four-tuple of integers.

Shared keys are possible only if both nodes possess clusters of the same type, and those clusters have the same index within the cluster type. Since each node receives at most seven key sets under our enhanced scheme, these two conditions can be checked in $O(1)$ time. For Lee and Stinson's scheme, the keys belonging to a key set are described by a tuple of the form $\{(x, (ix + j) \bmod n) \mid 0 \leq x \leq s - 1\}$ [6]. If two nodes have key sets (i, j) and (i', j') which satisfy conditions 1 and 2 above, then they can determine the shared key (should it exist) by solving at most three equations: $ix + j = i'x + j'$, $ix + j + n = i'x + j'$, and $ix + j = i'x + j' + n$. Should a valid solution for x exist, then a shared key exists, and x can be used in the above tuple to describe it. If a valid value for x does not exist, then there is no shared key. Clearly, these calculations can also be completed in $O(1)$ time; therefore, the presence of shared keys can also be determined in $O(1)$ time.

4.2 Resilience

Since we ensure uniform key distribution, the resilience analysis of our scheme is quite straightforward. Assume that w nodes have been compromised and consider a channel c between two sensor nodes that have not been compromised. Lee and Stinson show in [6] that the probability that channel c has been compromised given that w nodes have been compromised is given by $1 - (1 - \frac{s}{K})^w$, where K is the size of the key pool used by the scheme. Simluation results were found to be within five percent of the theoretical values.

5 Experimental Analysis of the Scheme

In this section, we present simulation results comparing the performance of our enhanced scheme with the schemes proposed by Du et al. in [15], Liu and Ning in [14], and Zhou et al. in [17]. We refer to the three schemes as the DDHV scheme, the LN scheme, and the ZNR scheme, respectively.

5.1 Scheme Selection

The DDHV scheme is selected due to similarities with our proposed scheme. Both schemes use region-based deployment knowledge, and both schemes are *configurable*, in that the local connectivity can be adjusted using scheme parameters prior to deployment. Although the LN scheme is not configurable in the above sense, we compare its resilience with our scheme due to its use of region-based deployment knowledge. The ZNR scheme uses group-based (instead of region-based) deployment knowledge, and is selected for comparison because it features perfect resilience against node capture. In this case, it is informative to examine the tradeoff between resilience, global connectivity, and flooding overhead.

5.2 System Configuration

In our simulations, we use the same setup used in [15] unless otherwise indicated. The deployment area is $1{,}000\ m \times 1{,}000\ m$ and divided into a 10×10 grid. Each grid region is $100\ m \times 100\ m$. There are 10,000 nodes in the sensor network, and nodes are divided into groups of equal size, one group per grid region. Nodes are deployed from the center of each grid region using a 2D Gaussian distribution with standard deviation of $50\ m$. The wireless communication range for each node is $40\ m$.

5.3 Evaluation Metrics

5.3.1 Global Connectivity

It is highly desirable to have a high global connectivity in the deployed network—nodes which are unreachable via secure channels can be considered lost and unable to perform their mission. Global connectivity is measured as the ratio of the size of the largest securely connected component and the size of the entire network.

5.3.2 Flooding Overhead

If two neighboring nodes wish to communicate, but do not share a common key, it is necessary to use a path-based mechanism to obtain a common key. Two such nodes can use flooding to determine a path between them. We measure flooding overhead as the fraction of neighboring sensor nodes that can establish a secure channel when path length is restricted to a certain number of hops.

5.3.3 Resilience

If an adversary captures a sensor node and extracts its keys, it is desirable that the damage to the network be limited. Clearly, any channels between the captured node and other nodes in the network are compromised. If the captured keys are used elsewhere in the network, however, then the adversary will also be able to eavesdrop on channels between uncompromised nodes. Resilience is measured as the fraction of compromised channels between uncaptured nodes when a certain number of nodes have been captured.

5.4 Simulation Conventions

In simulations where the size of the deployment area is changed, the standard deviation of the deployment distribution is half of the grid region dimension (e.g., if the deployment area is $1{,}600\ m \times 1{,}600\ m$, each grid region is $160\ m \times 160\ m$, and the standard deviation of the deployment distribution is 80 m). Unless otherwise indicated, comparisons between schemes are made at points of equal (within five percent) local connectivity. When comparisons are made between a configurable and nonconfigurable scheme (e.g., DDHV and ZNR), the parameters of the configurable scheme are adjusted so that the local connectivity matches that of the nonconfigurable scheme. When two configurable schemes are compared (e.g., our scheme and DDHV), the parameters of both schemes are adjusted so that the local connectivities match a predetermined value. For the DDHV scheme, the key pool size K and the overlapping factors a and b are adjusted to maximize local connectivity [15], thus maximizing the key pool size for the target local connectivity. For our scheme, the overlap probabilities p and q are adjusted to control local connectivity.

As there are 100 groups, and 100 sensor nodes per group, each sensor node using the ZNR scheme carries 99 intragroup keys, with the remainder of its key storage space used for intergroup keys. Note that due to the limitations of the function $\mathcal{F}_i$ used in the ZNR scheme, the largest key ring size for the ZNR scheme in these experiments is 198. For $s > 198$, the pairs of nodes between groups generated by $\mathcal{F}_i$ begin to repeat, essentially wasting key ring space since the two nodes already share a key.

5.5 Global Connectivity

Simulation results show that global connectivity for our scheme, the DDHV scheme, and the ZNR scheme are roughly equivalent until the key ring size approaches 100. While our scheme and the DDHV scheme exhibit similar performance at $s = 100$ (global connectivities

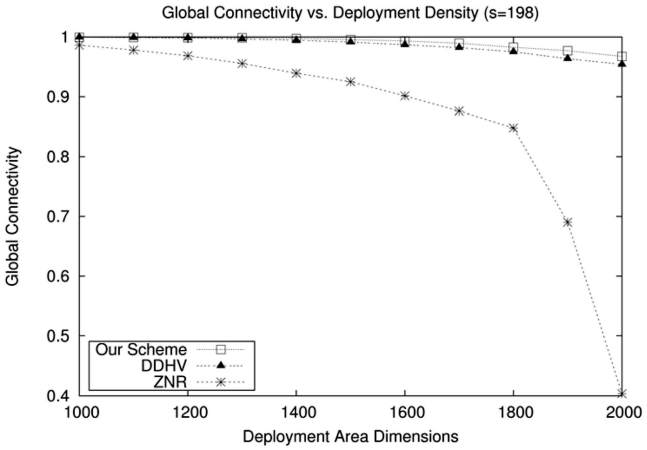


Fig. 3. Simulation results: global connectivity versus deployment density.

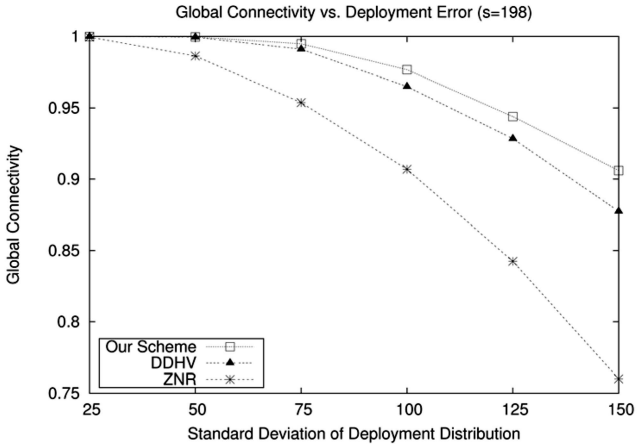


Fig. 4. Simulation results: global connectivity versus deployment error.

of 0.9995 and 0.99888, respectively), the global connectivity for the ZNR scheme decreases to 0.0308 when $s = 100$. This effect is due to the decreasing number of keys allocated for intergroup channels—at $s = 100$, there is only one key per node devoted to intergroup channels.

Fig. 3 shows the effects of deployment density on global connectivity. In Fig. 4, we demonstrate the effect of varying the standard deviation of the deployment distribution. In both cases, our scheme shows slightly improved performance over the DDHV scheme. The performance of the ZNR scheme, however, is much worse, especially as the deployment area dimensions increase. This can be attributed to the low percentage of keys

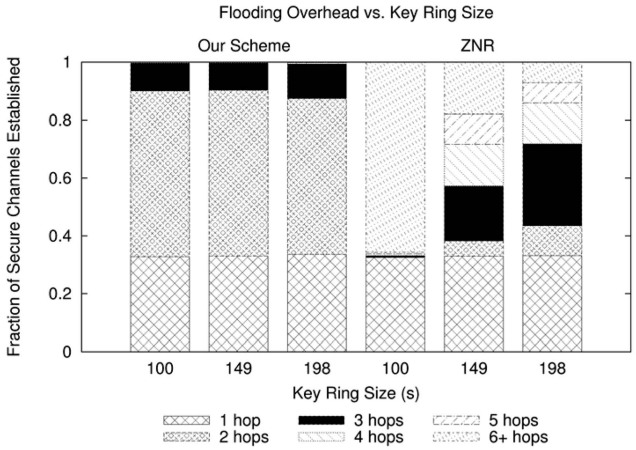


Fig. 5. Simulation results: flooding overhead versus key ring size.

allocated by the ZNR scheme to neighboring groups. Although the ZNR scheme splits its available key space equally between intra- and intergroup keys (99 keys each), the intergroup keys are allocated over all groups instead of just the neighboring ones. Both our scheme and the DDHV scheme, however, use the entire intergroup key allocation for neighboring groups.

5.6 Flooding Overhead

Simulations were performed to compare the flooding overhead of our scheme with that of the DDHV scheme. Local connectivity was varied from 0.25 to 0.75. Performance is nearly identical for both schemes; the vast majority of secure connections are established within three hops.

The simulation results in Fig. 5 compare the flooding overhead of our scheme and the ZNR scheme as the key ring size is varied. When $s = 100$, our scheme establishes almost all secure connections within three hops, whereas the ZNR scheme requires six or more hops for over 60 percent of the connections. In fact, the global connectivity results in Figs. 3 and 4 show that many nodes will *never* be connected using the ZNR scheme, regardless of the number of hops allowed. As with the global connectivity results, this can be explained by the differences in intergroup key allocation between the schemes.

Finally, we compare the flooding overhead of the schemes versus node deployment density in Fig. 6. Our scheme gives better performance than the DDHV scheme as the node density decreases. As the network becomes more sparse, the ZNR scheme appears to converge to two types of connections: those established in one hop, and those established in six or more hops.

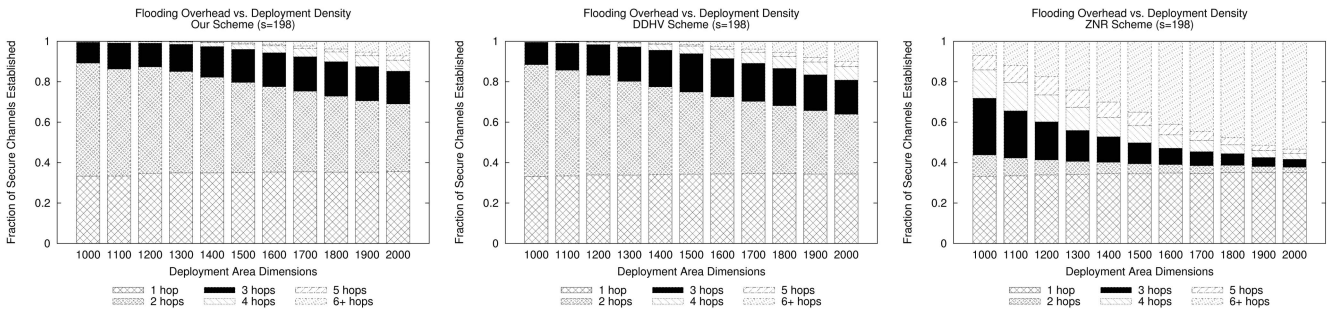
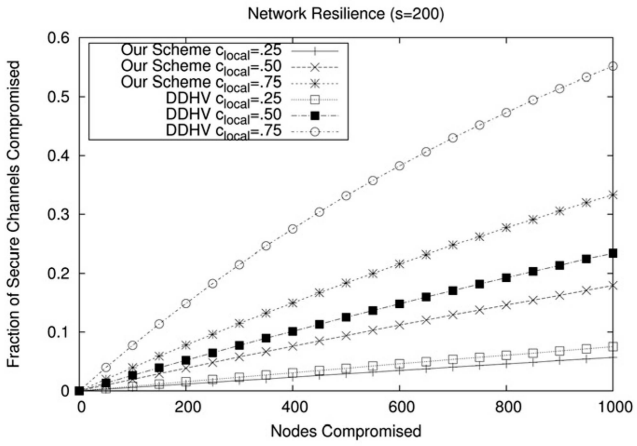
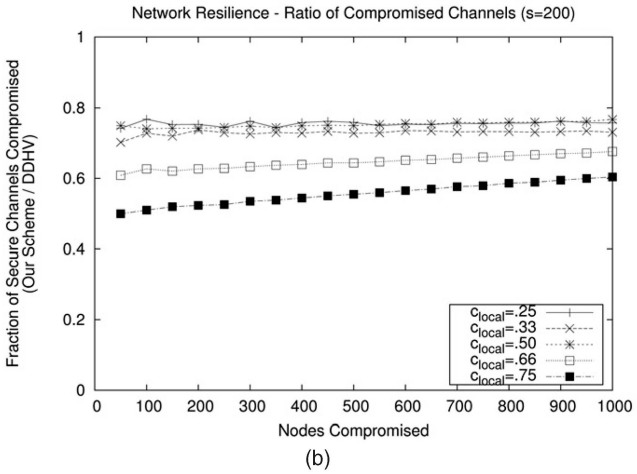


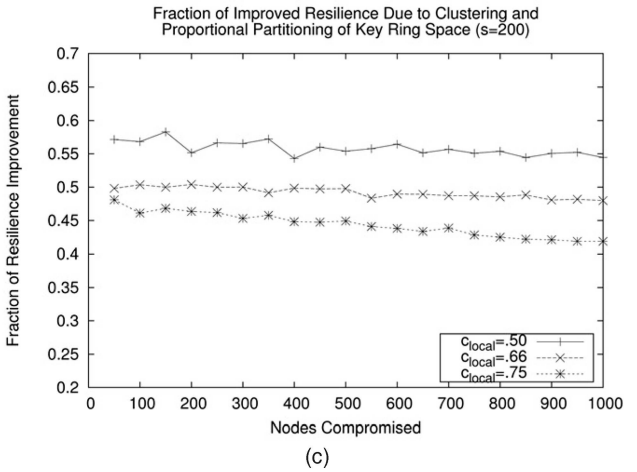
Fig. 6. Simulation results: the effect of deployment density on flooding overhead.



(a)



(b)



(c)

Fig. 7. Simulation results: (a) network resilience, (b) ratio of compromised channels (our scheme/DDHV), and (c) fraction of improved resilience due to clustering and proportional partitioning of key ring space.

5.7 Resilience

We first compare the resilience of our scheme with the DDHV scheme using simple cryptographic keys (i.e., $\lambda = 0$); these results are shown in Fig. 7. The actual fraction of compromised channels is shown in Fig. 7a, while Fig. 7b presents the simulation results as a ratio. Next, we compare the schemes when the key pool consists of instances of matrices ($\lambda = 9$), and we reduce the key ring sizes to $s = 16$ and $s = 20$ to reflect the increased storage requirements for these matrices. Results are shown in Figs. 8a and 8b. For small values of s, the enhanced scheme does not perform as well as the basic scheme due to the extra partitioning of s. Setting $s_1 = 0$ reduces the number of partitions of s from five to four, and effectively returns us to the basic scheme. Performance is similar for both schemes for values of $c_{\text{local}} \leq 0.50$, so only $c_{\text{local}} \geq 0.50$ is shown. In both cases, the performance of our scheme relative to the DDHV scheme improves rapidly as the local connectivity increases.

The improved resilience of our scheme can be attributed to a number of factors. First, we use a novel clustering scheme specifically designed to maximize the size of the key pool. Second, we use proportional partitioning of key ring space based on the number of channels a key space is required to secure. This helps to eliminate "high-value targets." Third, we use Lee and Stinson's scheme [6] for constructing various key spaces, which yields optimally sized key spaces in the absence of deployment knowledge. The DDHV scheme, as presented in [15], uses a probabilistic scheme similar to the one presented in [3] for selecting keys from a key space. To determine the fraction of improvement due to the first two factors (as opposed to the use of Lee and Stinson's scheme), we modify our scheme to construct key spaces using a probabilistic scheme similar to the one used in the DDHV scheme. The simulation results are shown in Fig. 7c. As shown in the figure, the first two factors are responsible for up to 58 percent improvement in resilience when local connectivity is 0.5, up to 50 percent improvement when local connectivity is 0.66, and up to 48 percent improvement in resilience when local connectivity is 0.75. This clearly indicates that a significant fraction of improvement in resilience is due to the clustering technique and proportional partitioning of key ring space.

Finally, we compare the resilience of our scheme and the LN scheme. To obtain accurate results, we ensure that the two schemes each use the same amount of key storage memory on a sensor node. For the LN scheme, s is set to five (a sensor node carries information about at most five polynomials) and $\lambda = 23$, which implies that $m = s(\lambda + 1) = 120$. As discussed above, we use our basic scheme due to the small key ring size. For our scheme, $s_1 = 0$, $s_2 = 1$, and $s_4 = 2$, which implies that $s = s_1 + 2(s_2 + s_4) = 6$. Therefore, $\lambda = 19$ to obtain $m = 6 * (19 + 1) = 120$. Simulation results are shown in Fig. 8c. Both schemes perform similarly until approximately 300 nodes have been compromised; at this point, our scheme provides superior resilience.

6 Conclusion and Future Work

We have presented a new key predistribution scheme that uses region-based deployment knowledge to assign keys to sensor nodes. Our simulation results show a significant improvement in resilience over existing schemes using region-based deployment knowledge. This improvement in resilience grows as the local connectivity of the network increases. Our scheme also provides improved global connectivity and flooding overhead when compared to a perfectly resilient scheme that uses group-based deployment knowledge, especially when the network is sparse or the key ring size is small. Additionally, our scheme does not place upper or lower bounds on the key ring size, as does the ZNR scheme.

Future work can be done to characterize the effects of localized network attacks, as well as the effects of different deployment distributions. We also believe that our scheme can be adapted to use nonrectangular deployment regions, such as the hexagonal scheme in [19].

References

[1] J.M. Kahn, R.H. Katz, and K.S.J. Pister, "Next Century Challenges: Mobile Networking for Smart Dust," *Proc. Ann. ACM/IEEE MobiCom*, pp. 483-492, 1999.

[2] G.J. Pottie and W.J. Kaiser, "Wireless Integrated Network Sensors," *Comm. ACM*, vol. 43, no. 5, pp. 551-558, 2000.

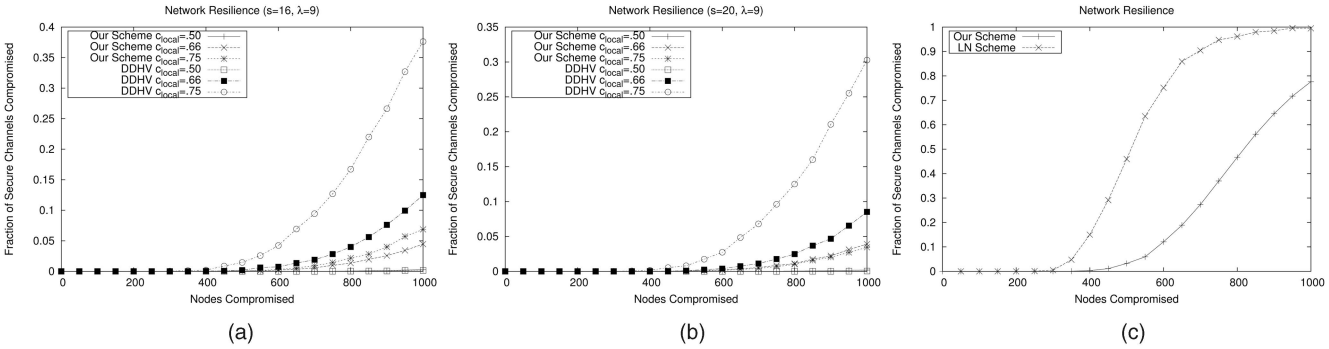


Fig. 8. Simulation results: network resilience using instances of matrices or polynomials.

[3] L. Eschenauer and V. Gligor, "A Key Management Scheme for Distributed Sensor Networks," *Proc. Ninth ACM Conf. Computer and Comm. Security,* pp. 41-47, Nov. 2002.

[4] H. Chan, A. Perrig, and D. Song, "Random Key Predistribution Schemes for Sensor Networks," *Proc. IEEE Symp. Security and Privacy,* pp. 197-213, May 2003.

[5] A.K. Das, "An Identity-Based Random Key Pre-Distribution Scheme for Direct Key Establishment to Prevent Attacks in Wireless Sensor Networks," *Int'l J. Network Security,* vol. 6, no. 2, pp. 134-144, 2008.

[6] J. Lee and D.R. Stinson, "Deterministic Key Predistribution Schemes for Distributed Sensor Networks," *Proc. Ann. Symp. Selected Areas in Cryptography,* pp. 294-307, 2004.

[7] S.A. Çamtepe and B. Yener, "Combinatorial Design of Key Distribution Mechanisms for Wireless Sensor Networks," *Proc. Ninth European Symp. Research Computer Security (ESORICS),* pp. 293-308, 2004.

[8] J. Lee and D.R. Stinson, "A Combinatorial Approach to Key Predistribution for Distributed Sensor Networks," *Proc. IEEE Wireless Comm. and Networking Conf.,* 2005. CD-ROM, paper PHY53-06.

[9] W. Du, J. Deng, Y.S. Han, P.K. Varshney, J. Katz, and A. Khalili, "A Pairwise Key Pre-Distribution Scheme for Wireless Sensor Networks," *ACM Trans. Information and System Security (TISSEC),* vol. 8, no. 2, pp. 228-258, May 2005.

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

[11] R. Blom, "An Optimal Class of Symmetric Key Generation Systems," *Advances in Cryptology—Proc. Workshop Theory and Application of Cryptographic Techniques (EUROCRYPT),* pp. 335-338, 1984.

[12] C. Blundo, A.D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-Secure Key Distribution for Dynamic Conferences," *Advances in Cryptology—Proc. Ann. Int'l Cryptology Conf. (CRYPTO),* pp. 471-486, Aug. 1992.

[13] D. Huang, M. Mehta, D. Medhi, and L. Harn, "Location-Aware Key Management Scheme for Wireless Sensor Networks," *Proc. ACM Workshop Security of Ad Hoc and Sensor Networks (SASN),* pp. 29-42, Oct. 2004.

[14] D. Liu and P. Ning, "Improving Key Pre-Distribution with Deployment Knowledge in Static Sensor Networks," *ACM Trans. Sensor Networks,* vol. 1, no. 2, pp. 204-239, 2005.

[15] W. Du, J. Deng, Y.S. Han, and P.K. Varshney, "A Pairwise Key Pre-Distribution Scheme for Sensor Networks Using Deployment Knowledge," *IEEE Trans. Dependable and Secure Computing,* vol. 3, no. 1, pp. 62-77, Jan./Mar. 2006.

[16] D. Liu, P. Ning, and W. Du, "Group-Based Key Predistribution for Wireless Sensor Networks," *ACM Trans. Sensor Networks,* vol. 4, no. 2, pp. 1-30, 2008.

[17] L. Zhou, J. Ni, and C.V. Ravishankar, "Supporting Secure Communication and Data Collection in Mobile Sensor Networks," *Proc. IEEE INFOCOM,* pp. 1-12, Apr. 2006.

[18] N. Mittal and T. Belagodu, "On Maximum Key Pool Size for a Key Pre-Distribution Scheme in Wireless Sensor Networks," *Int'l J. Computers and Applications,* vol. 31, no. 1, 2009.

[19] Z. Yu and Y. Guan, "A Key Pre-Distribution Scheme Using Deployment Knowledge for Wireless Sensor Networks," *Proc. Fourth Int'l Symp. Information Processing in Sensor Networks (IPSN),* pp. 261-268, Apr. 2005.