Revisiting Random Key Pre-distribution Schemes for Wireless Sensor Networks

Joengmin Hwang
Computer Science and Engineering
University of Minnesota
Minneapolis MN 55455
jhwang@cs.umn.edu

Yongdae Kim
Computer Science and Engineering
University of Minnesota
Minneapolis MN 55455
kyd@cs.umn.edu

ABSTRACT

Key management is one of the fundamental building blocks of security services. In a network with resource constrained nodes like sensor networks, traditional key management techniques, such as public key cryptography or key distribution center (e.g., Kerberos), are often not effective. To solve this problem, several key pre-distribution schemes have been proposed for sensor networks based on random graph theory. In these schemes, a set of randomly chosen keys or secret information is pre-distributed to each sensor node and a network is securely formed based on this information. Most of the schemes assumed that the underlying physical network is dense enough, that is, the degree of each node is high.

In this paper, we revisit the random graph theory and use giant component theory by Erdös and Rényi to show that even if the node degree is small, most of the nodes in the network can be connected. Further, we use this fact to analyze the Eschenhauer et. al's, Du et. al's, and Chan et. al's key pre-distribution schemes and evaluate the relation between connectivity, memory size, and security. We show that we can reduce the amount of memory required or improve security by trading-off a very small number of isolated nodes. Our simulation results show that the communication overhead does not increase significantly even after reducing the node degree. In addition, we present an approach by which nodes can dynamically adjust their transmission power to establish secure links with the targeted networked neighbors. Finally, we propose an efficient path-key identification algorithm and compare it with the existing schemes.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*security and protection*

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SASN'04, October 25, 2004, Washington, DC, USA.
Copyright 2004 ACM 1-58113-972-1/04/0010 ...$5.00.

General Terms

Design, Security

Keywords

wireless sensor networks, key pre-distribution

1. INTRODUCTION

Distributed sensor networks have received a lot of attention recently due to their wide range of applications in military as well as civilian operations. Examples include target tracking, scientific exploration, and monitoring of nuclear power plants. Sensor nodes are typically low-cost, battery powered, highly resource constrained, and usually collaborate with each other to accomplish their tasks.

Security services, such as authentication and confidentiality, are critical to secure the communication between sensors in hostile environments. For these security services, key management is a fundamental building block. Since each node has constrained resources and can be captured, traditional key management techniques using public key infrastructure or centralized key management techniques may not be appropriate for sensor networks.

To solve this problem Eschenhauer and Gligor [10] first proposed a random key pre-distribution scheme, which let each sensor node randomly pick a set of keys from a key pool before deployment such that two sensor nodes share a key with a certain probability after deployment. Chan et. al. [6], Du et al. [8], Liu et. al. [15], and Zhu et. al. [21] extended this scheme to further strengthen the security or improve the efficiency. Du et. al. [7] and Liu et. al. [15] provided a random key pre-distribution scheme using deployment knowledge which reduces memory size significantly. Most of the works are based on Erdös and Rényi's random graph theory. The theory explains the relation between the local connectivity (i.e., the probability that two nodes are connected) and the global connectivity (i.e., the probability that the whole network is connected) [8]. More precisely, for a given global connectivity, they could compute the required local connectivity. Given the average number of physical neighbors (we call this number the network node degree), this probability gives us the expected number of secure links per node (we call this number the secure node degree).

All of the above papers provide reasonably strong security. However, they assumed that the underlying physical network is dense enough to enable their key pre-distribution to be effective. For example, in the network with 10,000 nodes, the node degree required by [17] for the network to be connected with 99% certainty is around 14. In [10, 6] they assumed each node has at least 20 (even 40 or 50) physical neighbors among which 14 neighbors should be securely connected. This highly dense connection, we believe, may not be practical in certain sensor networks. In theory, the recommended optimal node degree for best network capacity is between 5 and 8 [11, 14, 18]. Or it may be dynamically controlled so that the power consumption for transmission is minimized. The exact optimal node degree for the whole graph to be connected, however, is known to be an open problem [20]. From this we can notice some gap between the physical node degree which is suggested as optimal in networking and which is required in the existing key pre-distribution scheme. Overall, the focus of our work is to reduce the gap.

This paper deals with the relation among connectivity, security and memory size in key pre-distribution scheme. We first re-establish the required secure node degree using giant component theory of Erdös and Rényi, by which the tighter lower bound on the required node degree will be computed. In non-uniform node distribution Du et al. [7] obtained the largest isolated components by simulation and used it as global connectivity. However, to the best of our knowledge the theoretical analysis on the isolated component based on random graph theory has not been studied. By providing this analysis, we can control the parameters of key pre-distribution scheme, for example, we can increase security, reduce memory size, or increase the connectivity. The analysis also shows that the network can be connected with a reasonable memory size even in a very sparse network, the evaluation of which was impossible previously. Second, we provide an effective way to utilize the current sensor hardware capability to control transmission power in order to obtain both the desirable number of neighbors and the required number of secure links. Given a desired network transmission range, to securely connect to the targeted neighbor nodes in that range, it temporarily adjusts its transmission range. We provide two ways to control transmission ranges, the computation overhead and communication overhead of which will be compared.

The remainder of the paper is organized as follows. We introduce the set of notations in Section 2. We give an overview of the existing key pre-distribution schemes based on random graph theory in Section 3. In Section 4, we present new analysis using giant component theory and re-evaluate the scheme using theoretical analysis and simulation of giant component size. In Section 5, we examine the way to utilize the sensor hardware capability to control transmission range. Finally, we conclude in Section 6.

2. NOTATION

The following notations are used throughout the rest of this paper.

– d: the expected degree of a node-i.e., the expected number of secure links a node can establish during key-setup
– k: number of keys in a node's key ring
– n: network size, in nodes
– r: communication radius
– n': the expected number of neighbor nodes within communication radius of a given node
– p: probability that two nodes share a key
– P_c: probability that graph is connected
– β: ratio of largest component size to network size
– P: size of the key pool
– A: area of the field
– ω: number of key spaces constructed in networks
– τ: number of key spaces carried by each node
– x: number of nodes captured

3. BACKGROUND

In this section, we introduce some background for this paper. We first review the three key pre-distribution schemes for sensor networks [10, 8, 6] that we are focusing on throughout this paper. Second, we overview random graph theory and its relation to each scheme. Next, we introduce previous results on desired node degree for wireless networks. We finish this section with an example of a sensor with the hardware capability of dynamically controlling its transmission power.

3.1 Key pre-distribution in wireless sensor network

Eschenauer and Gligor. (called basic scheme or EG for simplicity) first introduced a key pre-distribution scheme to the area of wireless sensor networks [10]. In EG, each node randomly picks a subset (called key ring) of keys from a large key pool and any pair of nodes can establish a secure connection if they share at least one common key. More specifically, EG consists of three different phases:

- initialization: Before the sensor nodes are deployed, a set of keys randomly selected from a pool of keys are inserted to the sensor nodes.
- key set-up: After the sensor nodes are deployed, each node first performs a shared key discovery to find out which of its neighbors shares a key with it. When one finds another node, they mutually authenticate to verify that the other party actually owns the key.
- path key identification: After the key set-up is complete, a securely connected graph is formed. A node tries to securely connect all of its neighbors using path key identification methods. In other words, a node can find out the path to all of its neighbors (not securely connected) by means of its securely connected neighbors, and a key is securely delivered from the source node to its targeted neighbor via an indirect path.

Chan, Perrig and Song. (called CPS for simplicity) Chan et. al. further extended this idea and developed two key pre-distribution techniques [6]: q-composite key pre-distribution and a random pair-wise keys scheme. The difference of the q-composite scheme from [10] is that it requires any two nodes to share at least q common keys to establish a secure link.

The random pair-wise keys scheme is a modification of the traditional pair-wise keys scheme based on the observation that not all $n-1$ keys need to be stored in the node's key ring to have a connected random graph with high probability. Each node identity is paired with m other randomly selected distinct node id and a pair-wise key is pre-generated for each pair of nodes. The key is stored in both nodes' key ring along with the id of the other node that also knows the key. This provides much improved security, since any captured node reveals no information about links in which it is not directly involved. The main drawback of the random pair-wise key pre-distribution scheme is its scalability.

Du, Deng, Han and Varshney. (called DDHV for simplicity) Du et. al. combined the basic scheme [8] with Blom's key management scheme [4]. Blom's scheme allows that any pair of $(n-1)$ nodes to find a secret pair-wise key between them with much smaller number (they use λ, $\lambda << n$) of keys than the actual number of nodes. The tradeoff is that, unlike the $(n-1)$-pairwise key scheme, Blom's scheme is not perfectly resilient against node capture. As long as an adversary compromises at most λ nodes, uncompromised nodes are perfectly secure. When an adversary compromises more than λ nodes, all pairwise keys in the entire network are compromised. While Blom's scheme uses a single key space to ensure that any pair of nodes can compute a shared key, Du et. al. construct ω spaces, and each sensor node carries key information from $(2 \leq \tau < \omega)$ randomly selected key spaces. If two nodes carry key information from a common key space, they can compute a pairwise key. A similar method is also developed by Liu and Ning [15] based on polynomial-based key pre-distribution [5].

In all of the above schemes it is not certain that two nodes can generate a pairwise key. Instead, they have only a guarantee with probability p that this will be possible. To find p so that n nodes in the sensor network are connected, they use a random graph theory.

3.2 Random Graph Theory and Key Pre-distribution Scheme

A random graph $G(n,p)$ is a graph of n nodes for which the probability that a link exists between two nodes is p. In a large sensor network with size n, p denotes the probability that two neighboring nodes share a common key or key information, which we call local connectivity. Let P_c be the probability that the graph is connected, which we call global connectivity. Erdös and Rényi [9, 17] provided a theory how to determine p so that P_c is almost 1 (i.e. the graph is almost surely connected). In wireless sensor networks, p changes according to the key ring size or key pool size. Thus, for a network to be connected with probability P_c, p, as determined by the key information (key ring or pool size), should be greater than p as obtained from Erdös and Rényi's Theory. We denote the former by p_{actual} and the latter by $p_{required}$. In this section we show how to compute $p_{required}$ and introduce p_{actual} in EG, DDHV, CPS.

Computing required local connectivity

Erdös and Rényi [9, 17] showed that, for monotone properties, there exists a value of p such that the property moves from"nonexistent" to "certainly true" in a very large random graph $G(n,p)$. We find p such that it is "almost certainly true" that the graph is connected. The function defining p is called the threshold function of a property. Given a desired probability P_c for graph connectivity, the threshold function p is defined by:

Theorem 1.

$$P_c = \lim_{n\to\infty} P_r[G(n,p) \text{ is connected }] = e^{-e^{-c}}$$

$$\text{where } p = \frac{\ln(n)}{n} + \frac{c}{n} \text{ and } c \text{ is any real constant.}$$

We define the average node degree $d = p(n-1)$ as the average number of edges connected to each node. For a given density of sensor network deployment, let n' be the expected number of neighbors within the wireless communication range of a node. Since the expected node degree should be at least d as calculated above, the required local connectivity $p_{required}$ can be estimated as $p_{required} = \frac{d}{n'}$

Computing actual local connectivity

The actual local connectivity is determined by key ring size and key pool size in EG, by the key space in DDHV and by the number of pairwise keys stored in each node in CPS. Following is the detailed description.

EG: We set the probability that two nodes share at least one key in their key rings of size k chosen from a given pool of P keys to p_{actual}. Since $p_{actual} = 1 - P_r[\text{two nodes do not share any key}]$,

$$p_{actual} = 1 - \frac{((P-k)!)^2}{(P-2k)!P!}. \tag{1}$$

DDHV: p_{actual} is the actual probability of any two neighboring nodes sharing at least one space (i.e., the probability that they can establish a common key). Since $p_{actual} = 1 - P_r[\text{two nodes do not share any space}]$,

$$p_{actual} = 1 - \frac{((\omega-\tau)!)^2}{(\omega-2\tau)!\omega!}. \tag{2}$$

CPS: p_{actual} is the actual probability of any two neighboring nodes sharing a pairwise key. If a node can store m keys in the network with size n, then the actual probability is

$$p_{actual} = \frac{m}{n}. \tag{3}$$

3.3 Desirable node degree in the network

The desirable node degree is discussed on the context of network connectivity, network capacity and energy consumption by controlling transmission power. With a high node degree, the network connectivity increases, but the interference between the neighboring nodes increases, and, therefore, the network capacity decreases. On the other hand, if we decrease the node degree the connectivity decreases, which in the extreme case results in the network being disconnected. The low degree reduces the communication interference, but since the connectivity is poor the number of hops will be increased, which will decrease the network capacity. The node degree is controlled by adjusting transmission power. Higher node degree requires higher transmission power, which increases the energy consumption. The optimal node degree to maximize the network capacity was considered in the papers [14, 18, 11]. Kleinrock et. al. [14] proposed that six is the magic number for network in the slotted ALOHA protocol. Takagi et. al. [18] revised this number to 8 and suggested that 5 and 7 is optimal in the other protocol. Hou et. al. [11] considered the magic number when the transmission range can be adjusted and suggested 6 and 8 is optimal. Xue et. al. [20] investigate the node degree for the network to be connected. According to their study, if each node is connected to less than

$0.074 \log n$ nearest neighbors then the network is asymptotically disconnected with probability 1 as n increases, while if each node is connected to more than $5.1774 \log n$ nearest neighbors then the network is asymptotically connected with probability approaching 1 as n increases. The exact constant remains as an open problem.

3.4 Sensor Hardware

Examples of sensor network projects include Smart Dust, WINS and μAMPS [19, 2, 13, 1]. Some of sensor node platforms have a fixed radio range but some others have the capability to adjust radio range. For example, the second prototype of a power-aware microsensor developed for the μAMPS project set the transmission power to one of six different levels by the processor. The maximum bit-rate of a point-to-point wireless link is 1Mbps. The power aware features of the μAMPS board allow for thirteen relevant power consumption stages. Among those thirteen states are six states of different power amplifier gains to support transmission distances from 10 meters to 100 meters. In the off state, the radio consumes no power. In the idle state, the radio consumes 60 mW. In the receive state, the radio consumes 280 mW. In the lowest transmission state, the radio consumes 330mW. In the highest transmission state, the radio consumes 1.1W. There are other minor intermediate stages. For example, the MAC protocol can change the radio range to include only the optimal number of nodes. In our work, we utilize this transmission power control feature to bridge the gap between network transmission range and secure transmission range.

4. REVISITING RANDOM GRAPH THEORY AND ITS APPLICATION

In this section, we investigate Erdös-Rényi's random graph theory in detail and re-evaluate the key pre-distribution schemes.

4.1 Calculation of a giant component

Erdös and Rényi [9] discovered that the random graph undergoes four phase transitions with changes of p. The theory in the previous section corresponds to the third phase transition, where the probability that the graph is connected approaches 1. We need to note that this is the probability that the whole graph is connected, not the ratio of the giant component to the network size. In this section we shift our main focus to compute the required local connectivity to obtain a sufficiently large giant component. For this purpose, the second phase transition of Erdös and Rényi's theory is applicable [3]. The theory is as follows: Let $p = a/n$ with $a < 1$. Then $G(n, p)$ will consist of small components, the largest of which is of size $\Theta(\ln n)$. Now, suppose $p = a/n$ with $a > 1$. With this small change, many components will join together to form a giant component of size $\Theta(n)$. Except for the giant component, the other components are relatively small, the largest of which has size $\Theta(\ln n)$. The reason for this sudden change is because bigger giant component tend to absorb larger number of small components. The size of a giant component can be computed by the following Theorem 2 [12].

THEOREM 2. *Consider an Erdös-Rényi random graph $G(n, p)$. For fixed $a > 0$, let $p = a/n$. The following Assertions hold with probability $\rightarrow 1$ as $n \rightarrow \infty$*

1. *If $a < 1$, the largest component size is at most $\frac{3}{(1-a)^2} \log n$.*
2. *If $a > 1$, there is a unique giant component with $(1 + o(1))\beta n$ nodes, where $\beta = \beta(a) > 0$ solves $\beta + e^{-\beta a} = 1$. The second largest component size is at most $\frac{16a}{a-1^2} \log n$.*

In the above theorem, β is the number of nodes belonging to the giant component divided by the number of nodes provided. Since $p = \frac{a}{n}$, then $a = pn \simeq d$, and we can say that a represents node degree d. Given a desired ratio β for a giant component size, the required local connectivity can be estimated as

$$p_{required} = \frac{a}{n'}$$

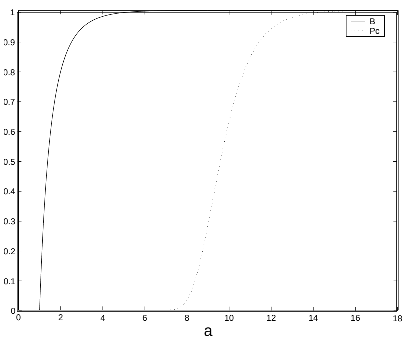


Figure 1: $a - \beta, a - P_c$ relation, where a is node degree, β is the ratio of largest component size to network size and P_c is the probability that graph is connected.

Figure 1 shows the relation between a and β and a and P_c with network size 10000. When node degree is around 6, β is close to 1. When the node degree is increased to around 14 or more, P_c approaches 1. This means that even with very small node degree most of nodes are connected to each other. Increasing the node degree to make P_c very close to 1 causes only a small number of isolated nodes to be connected.

4.2 Re-evaluation of EG

Given k and P we can compute β and P_c when $p_{actual} = p_{required}$ from Theorem 1, 2 and equation (1). Figure 2 shows the relation between k and β and the relation between k and P_c for $P = 100,000$ and $n = 10,000$. Theorem 1 determines k where P_c is very near probability 1 and Theorem 2 determines k where β is very near 1. Following are specific examples in which we can compare the basic scheme when Theorem 1 is applied with the one when Theorem 2 is applied. Less than 2% isolated nodes is assumed as a reasonable trade-off for reducing memory size. To create a giant component with size more than 98%, we choose constrained parameters $a = 3.9918$ where $\beta = 0.98$. Given $n = 10,000$, $A = 1000 \times 1000$ and $r = 40$, $p_{required} = 0.3664$ is computed by Theorem 1 when $P_c = 0.9999$ and $p_{required} = 0.0794$ by Theorem 2 when $\beta = 0.98$ respectively. (n' is computed by $\frac{r^2\pi}{A} \times n$). With $P = 100,000$, k is determined by equation (1) such that $p_{actual} \geq p_{required}$. To make p_{actual} larger than $p_{required} = 0.3664$, $k = 214$ is required, while to make p_{actual} larger than $p_{required} = 0.0794$,

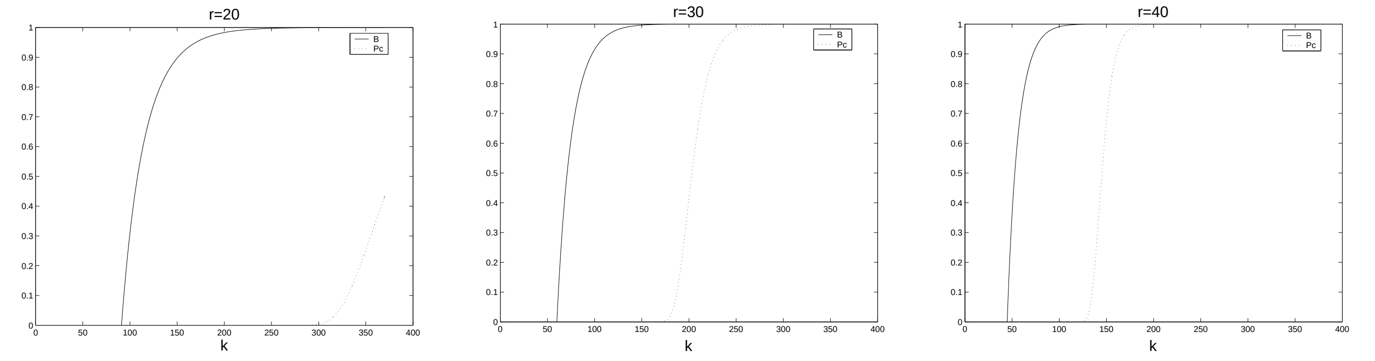


Figure 2: $k - \beta$, $k - P_c$ relation, where k is key ring size, β is the ratio of largest component size to network size, P_c is the probability that graph is connected and r is communication radius.

$k = 91$ is enough. Theorem 2 gives a reduction of half of the original memory size with the trade-off of 2% isolated nodes. The reduction of memory size is related to the resilience against node capture. If a node v is captured, k keys are compromised. The probability that a random link between two random nodes $i, j, (v \neq i, j)$ is not compromised is $(1 - \frac{k}{P})$. When x number of nodes are captured the expected fraction of the links compromised is $1 - (1 - \frac{k}{P})^x$. Figure 3 shows the comparison of this estimation for the different key ring size. Since we can reduce k by Theorem 2, the number of links an adversary could attack decreases and we can say that a reduction of k leads to a higher resilience against node capture.

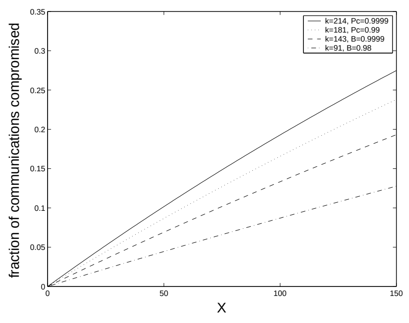


Figure 3: Probability that a random link between two random nodes i, j can be decrypted by the adversary when the adversary has captured some set of x nodes except i or j, where k is key ring size and r is communication radius.

4.3 Re-evaluation of DDHV

Given τ, ω and n' we can compute β and P_c when $p_{actual} = p_{required}$ from Theorem 1, 2 and equation (2). Figure 4 shows the relation between ω and β, and relation between ω and P_c for the fixed $\tau = 2$. Theorem 1 computes ω where P_c is very close to probability 1 and Theorem 2 computes ω where β almost m reaches 1. As we can see in the figure, by applying Theorem 2 we can increase the range of ω, the increase of which is more significant as r is bigger. In the basic scheme, $p_{required}$ decides the minimum p_{actual}, which decides k and P. The key ring size k determines memory size and security about resilience against node capture. However, in the Du's pairwise scheme, p_{actual} is controlled by ω and τ and memory size k is parameterized by the deployer. The parameters τ, ω, k and security are related to each other. According to [8], the security level is linearly related to $k\frac{\omega}{\tau^2}$. Thus, among pairs of ω and τ which satisfy p_{actual}, the deployer selects the one which maximizes $\frac{\omega}{\tau^2}$. Apart from the selected ω and τ, the memory size k is selected, where larger k provides better resilience against node captures. By increasing ω by Theorem 2, we can either reduce the memory size or increase the security level or both. For example, with $r = 40$ for fixed $\tau = 2$, to satisfy $P_c = 0.9999$ by Theorem 1, at most $\omega = 10$ can be selected. But, to create a giant component with size more than 98% by Theorem 2, $\omega = 49$ can be selected. If the memory size k is fixed, the security level computed by $\frac{\omega}{\tau^2} = \frac{49}{2^2}$ is 4.9 times higher than $\frac{\omega}{\tau^2} = \frac{10}{2^2}$. Conversely, if we fix the security level $k\frac{\omega}{\tau^2}$, we can reduce the memory size 4.9 times.

4.4 Re-evaluation of CPS

A drawback of CPS is that the supportable network size is smaller than in other schemes. From equation (3), the maximum supportable network size is determined by

$$n = \frac{m}{p}.$$

Figure 5 shows the supportable network size for different local connectivity. By applying Theorem 2, instead of Theorem 1, we can increase the supportable network size. For example, for fixed $m = 200$, to satisfy $P_c = 0.9999$ by Theorem 1, $p = 0.3664$ and $n = 545$. But to create a giant component with size more than 98%, $p = 0.0794$ is enough and this increases the supportable network size to $n = 2518$.

4.5 Communication Overhead

In the previous section, by reducing the shared key information we could save the memory size. However after the key set-up phase, the graph connected via secure links must have a very sparse node degree. The two neighboring nodes which *a priori* did not establish a secure link should find a path to each other over the graph, in order to do path key identification. However, the sparse node degree will increase the path length which will affect the communication overhead. In this section we discuss the communication overhead in the path key identification phase, by the number of hops increased when we apply Theorem 2. We first describe two

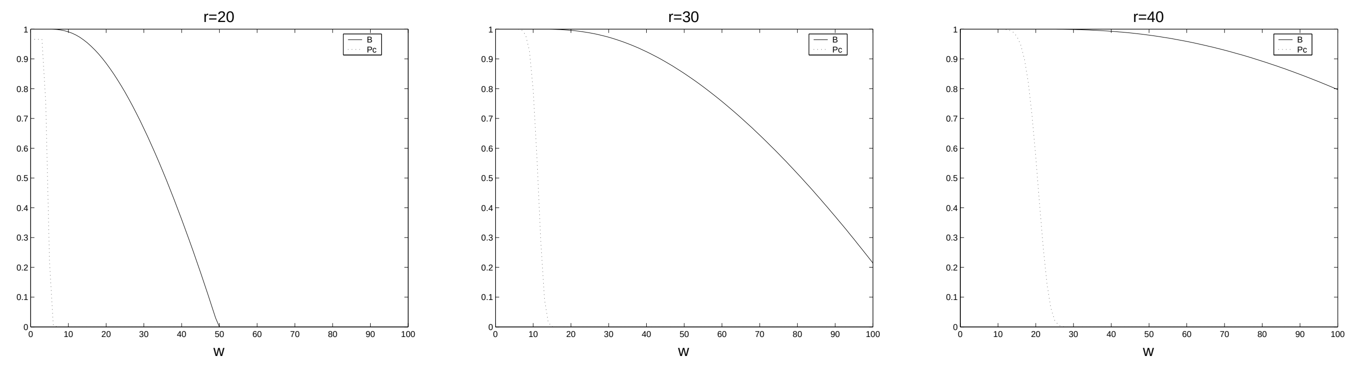


Figure 4: $\omega - \beta$, $\omega - P_c$ relation, where ω is number of key spaces constructed in networks, τ is number of key spaces carried by each node, β is the ratio of largest component size to network size, P_c is the probability that graph is connected and r is communication radius.

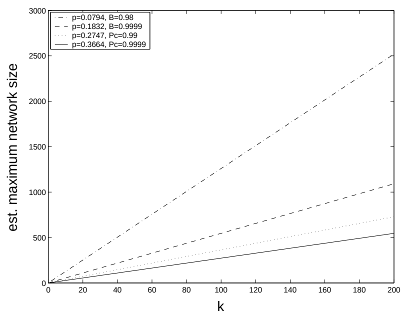


Figure 5: Network size for Chan's random pairwise scheme, where k is key ring size and p is probability that two nodes share a key.

ways to set up a path key identification, called cascade-off counting and cascade-on counting.

Cascade-off Counting After the key set-up phase, some of the neighbors are already connected via a shared key. We denote the graph consisting of the nodes and the links established by the shared key in the key set-up phase by G_{s1}. Then, in the path key identification phase, i discovers the route to j using only G_{s1}. This route is used to establish a shared key.

Cascade-on Counting As in the above counting, after a key set-up phase the G_{s1} is created. During the path key identification phase, some of the neighboring nodes will establish additional secure link via the path on G_{s1}. We add those secure links to G_{s1} and denote it by G_{s2}. Now the path key identification is performed on G_{s2} while new links are continuously added. The process will be repeated until all neighboring nodes are securely connected.

While the Cascade-off Counting gives us a well-defined bound of delay for each pair of nodes to reach each other, there is no reason why we should not use the previously established secure paths. [1]

We use simulation to measure the communication overhead when we apply Theorem 2 to the basic scheme. The number of hops required during path key identification is simulated based on the two counting methods above. We assume a network of 10,000 nodes in the monitored area 1000×1000 with transmission range 40 whose corresponding average number of neighbors is 50.26. [2]

Figure 6 and 7 show the ratio of the number of nodes that are isolated, unreachable in 3 hops, reachable in 3 and 2 hops, and directly connected for each counting method. As observed in the Cascade-off Counting in figure, when the key ring size is very small, the ratio of reachable nodes (ratio of the colored area) is small also. In some case, even if most of the nodes are reachable, many neighbors are not reachable within three hops (e.g. $r = 40, k = 100$). In the cascade-on counting, most of neighbors are reachable in two hops and at most within 3 hops.

The reason why the ratio of two hops increases in Cascade-on Counting is as follows: To be connected within two hops, the neighboring nodes i and j must have at least one common secure neighbor h. In the Cascade-off Counting, h should be the one having the shared key with i and j in their key ring. However, in the Cascade-on Counting, h should be the one either having a shared key or having a newly generated key during the path key identification phase with i and j. Thus, the probability to have at least one common neighbor h between i and j increases. The slight increase of secure links established during the path key identification helps to find common secure neighbors, the effects of which are small initially, but as the secure links increase, the effects increase.

4.6 Isolated Nodes

Theorem 2 provides a tighter lower bound on the connectivity condition. As a result, as we have shown in the previous section, we can reduce the memory size or improve security. While this idea is based on the assumption that a small fraction of isolated nodes is reasonable trade-off, in certain environments we might need to connect these isolated nodes to the network. To connect isolated nodes to the network, the isolated nodes need to detect it is isolated. Existing network partition detection algorithms may be used for this purpose. But according to Theorem 2, when the graph gets into the second phase transition, except for the giant component, the size of the remaining components is very small,

[1] Eschenhauer et. al. [10] seems to use Cascade-off for their simulation, while Du et. al. [8] provided theoretical analysis for Cascade-off.

[2] This setting was used in [7].

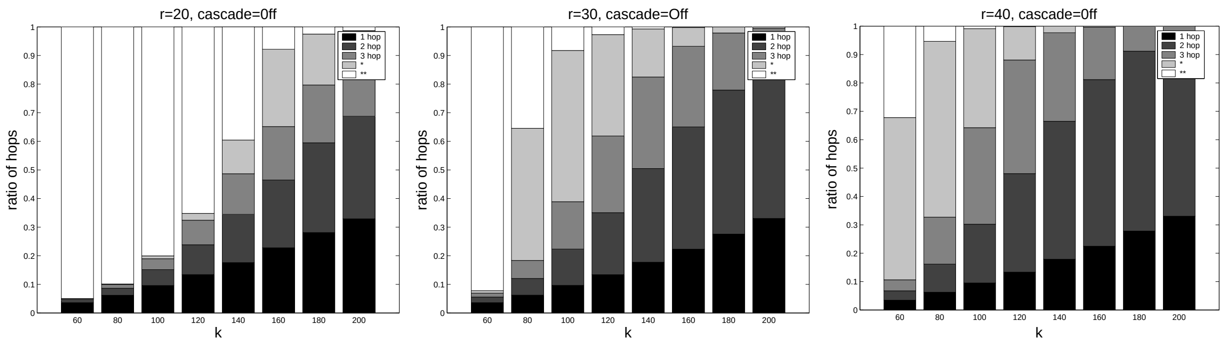


Figure 6: Ratio of hops: cascade-on, * reachable with more than 3 hops, ** Isolat ed, k is key ring size, r is communication radius.

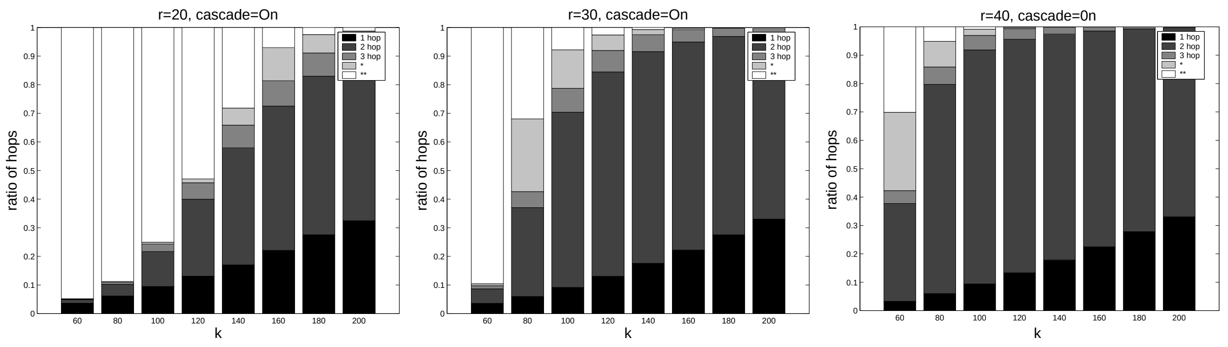


Figure 7: Ratio of hops: cascade-off, * reachable with more than 3 hops, ** Isola ted, k is key ring size, r is communication radius.

usually 1 in our simulation setting. Thus the cost for the partition detection algorithm is not expensive, or when the component size is 1, it can recognize itself as isolated without any partition detection algorithm. The isolated nodes can be securely connected to the giant component by finding one of the nodes in the giant component which shares a common key with it. To do this it should either increase transmission range or broadcast the message to find a pair with 2 or more hops.

5. KEY PRE-DISTRIBUTION USING TRANSMISSION RANGE ADJUSTMENT

In this section, we introduce a modified key pre-distribution scheme, when we can control the transmission range. To save energy consumption, it is desirable to minimize the transmission power such that the interference and energy consumption is minimized while enough links capacity for transmission should be maintained. Previous works [18, 14, 11] indicate that the transmission power to keep 5–8 node degree is optimal. However, for key pre-distribution scheme to be used effectively on the sensor network, this optimal node degree (in terms of networking) is not sufficient. In this section, we introduce a way to provide key pre-distribution under this optimal node degree and analyze its efficiency. We assume the sensor hardware to be capable of controlling its transmission range. (The example sensor hardware is introduced in Section 3.)

We use the following notations: Desired network transmission range is denoted as r_t, and the average number of neighbors from a node within radius r_t is denoted as n_t. We denote **secure transmission range as r_f, secure neighbor as n_f, which are the transmission range and the number of neighbors required in the key pre-distribution scheme**. The network transmission r_t will be provided a priori, and the sensor will increase transmission range temporarily after the deployment so that its range r_f is bigger than r_t.

The protocol description and simulation results introduced in the rest of this section is based upon EG. However, these can be easily applied to DDHV and CPS.

5.1 Protocol

The EG protocol changes slightly as follows. In the key set-up phase, the neighboring nodes in the **secure transmission range** r_f who have a shared key, set up a secure link using the key. In the path key identification phase, if the neighboring nodes i and j in the range r_t do not share a key, node i generates a new key and seds this key to j via the already established secure links. In other words, 1) node i tries to share a common key with all neighbors in the **large disk with radius** r_f in the key set-up phase, and 2) if it could not connect to node j in the **small disk with radius** r_t, it uses the secure channels in the large disk to set up a common key with the node j. For 2), we introduce two possible ways to send the generated key to the neighbor j (see path key identification below), and use them as part of path key identification phase. If there are still remaining neighboring nodes which are not connected after path key set-up, it can select whether to repeat the process again or

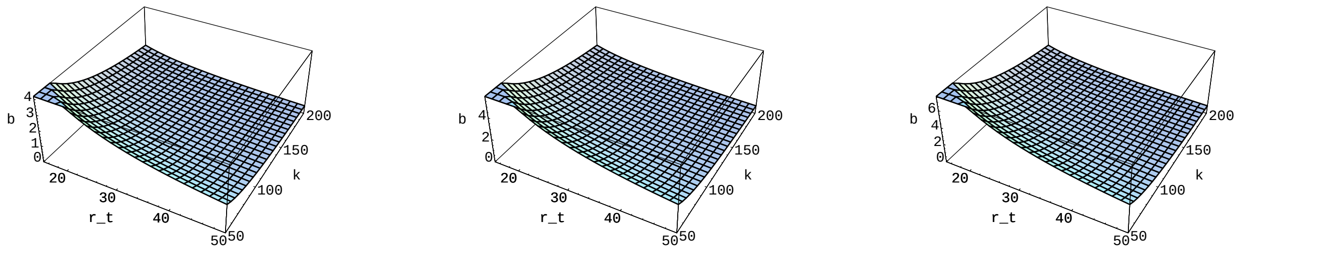


Figure 8: b, r_t, k relation, when desired node degree $d =$ 6,12 and 20 used, r_t is desired network transmission range, k is key ring size, b is the required range extension.

Table 1: Required range extension b

When $p'' = 0.99$

k \ r_t	15	30	45
200	2.6	1.4	1.2
180	3.0	1.8	1.2
160	3.8	2.0	1.4
140	4.8	2.6	1.8
120	6.2	3.2	2.4
100	8.8	4.4	3.2

When $p'' = 0.90$

k \ r_t	15	30	45
200	1.8	1.2	1.0
180	2.2	1.2	1.0
160	2.6	1.4	1.2
140	3.4	1.6	1.4
120	4.4	2.4	1.8
100	6.2	3.2	2.2

not. The overall protocol is described as below:

Key Set-up

1. Each node broadcasts its id and the list of id of the keys on their key ring.
2. Neighboring nodes which discover a shared key in their rings verify that their neighbor actually holds the key through a challenge response protocol. [3]

Path Key Identification Either of the below can be used (and can be repeated if needed).

Flood with TTL (FTTL in short) Neighboring nodes i and j which do not share a common key establish an indirect route using an algorithm that involves local flooding. Node i broadcasts its id and the neighbor's id not connected with TTL $= t$ via the secure channels that have been established in the key set-up phase. Node i picks an unused key (or generates a random key) K and sends it to j using the route. Node i and j use this secret key K as their pair-wise key. If there is a neighbor not connected, try path key identification again with increased TTL.

Common Secure Neighbor (CSN in short) Neighboring nodes i and j which do not share a key find a common secure neighbor from the message in 1 (in Key set-up phase) without any additional packets. Node i compares the key rings in the broadcast message with its own and j's and find the secret common neighbor h which shares a key with both of i and j. Node i generates a random key K and sends it to j via the common secret neighbor. Nodes i and j use this secret key K as their pair-wise key. If there is a neighbor not connected, run path key identification again.

[3]Different schemes can be used in key set-up phase, i.e., pseudo random key index transformation [16].

5.2 Computing Secure Transmission Range r_f

Given the desired r_t we can compute necessary r_f as follows depending on the path key set-up schemes.

Flood with TTL

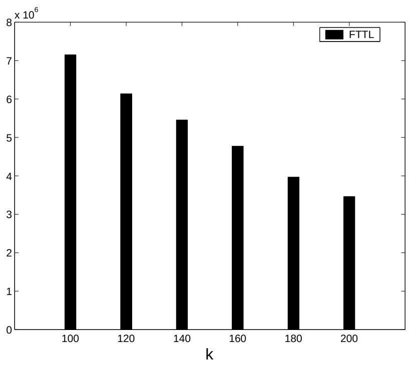
Given a certain network transmission range r_t, we calculate secure transmission range $r_f = b \times r_t$. The simplest way to decide the secure node degree is to use Theorem 1 or 2. Given the desired node degree, we find the ratio of the required r_f to the given r_t. That is,

$$d = pr_f^2\pi\rho = pb^2r_t^2\pi\rho,$$

when ρ is the number of nodes per unit area. Thus,

$$b = \frac{1}{r_t}\sqrt{\frac{d}{\rho p \pi}}.$$

Figure 8 shows the necessary b in EG according to the various r_t and k. For a desired node degree $d = 12$, and network transmission range $r_t = 20$, if we use the same secure transmission range $r_f = r_t, b = 1$, we cannot obtain the desired node degree 12 even with $k = 200$. However, by temporarily increasing secure transmission range ($r_f = 63, b = 3.16$) we can significantly reduce memory size to around $k = 100$. This increases the transmission interference temporarily in the key establishment process. However, this trade-off seems to be reasonable in the case where additional node deployment is considered (since, each node needs to maintain the whole key ring for the additional nodes).



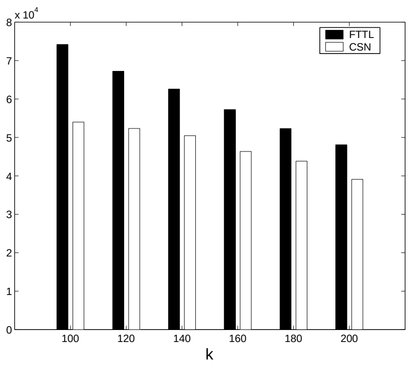


Figure 9: Communication overhead in path key set-up for $d = 12, r_t = 15$ and $p'' = 0.9$ (a) Broadcast cost in FTTL (b) Unicast cost in FTTL and CSN

Common Secure Neighbor

We first find the probability that two neighboring nodes are reachable with two hops. We follow the analysis provided by [8] and apply it to EG. We assume the distance between two nodes i and j is z. To establish a path key in two hops, a pair of nodes must set up a shared key through a common neighbor. The common neighbor must be in the overlapped region of the transmission range of node i and node j. The size of this overlapped region is:

$$A(z) = 2r_f^2 cos^{-1}(z/2r_f) - z\sqrt{r_f^2 - z^2/4}.$$

Thus, the total number of nodes in the overlapped region is $\rho A_{overlap}(z)$. The probability distribution function of the distance between two nodes within network transmission range r_t is given $F(z) = (\text{ distance } < z) = z^2/r_t^2$. The probability density function is thus $f(z) = F'(z) = 2z/r_t^2$. The probability that i and j are either directly connected or i and j are not connected directly but there exists at least one common neighbor connecting them is:

$$p'' = \int_0^{r_t} (1-p)(1-(p')^{\rho A(z)})\frac{2z}{r_t^2}dz + p$$

where p' is the probability that the common neighbors of i and j is not securely connected to both of them.

$$p' = 1 - \frac{\binom{P}{k} - 2\binom{P-k}{k} + \binom{P-2k}{k}}{\binom{P}{k}}.$$

Table 1 shows the necessary b in EG according to the various r_t and k.

5.3 Overhead Comparison

The metric used to evaluate the performance of our protocol are the computational and communication costs of key establishment. Since the key set-up phase is in common, we compare the cost for path key identification only.

Computational cost

The main computational cost is the cost of encrypting and decrypting the newly generated key in both schemes. A single encryption in the sender and decryption in the receiver is used whenever the unicast message is transmitted. In FTTL, if the number of hops in the path between i and j is h, the total number of encryptions and decryptions are both h. In CSN, the number of hops the unicast message traverses is always 2, so a total of 2 encryptions and 2 decryptions are used.

Communication cost

We tested the bandwidth required for the path key set-up phase. For comparison purposes, we performed a simulation either using only FTTL or only CSN. In FTTL, the communication overhead consists of the broadcast packet for route discovery and unicast of the newly generated secret key. The cost of broadcast packets is large, especially when TTL=t is large. The bandwidth for broadcasting its own id and neighbor's id which are not connected in a network of size n is

$$(i+(n_t-d\times\frac{n_t}{n_f})\times i)\times(1+d+d\times(d-1)+\cdots+d\times(d-1)^{t-2})\times n$$

where i is the size of id. The bandwidth all the nodes unicasting its own id, destination's id and key in network size n is

$$(i + i + K_s) \times \overline{t'} \times (n_t - d \times \frac{n_t}{n_f}) \times n/2$$

where K_s is the key size, $\overline{t}$ is the average path length, where $t' \leq t$.
In CSN, there is no communication cost for route discovery. But a unicast message is used to transmit the newly generated secret key. Since the number of hops it traverses is always 2, the total bandwidth for unicasting its own id, destination's id and key in network size n is

$$(i + i + K_s) \times 2 \times (n_t - d \times \frac{n_t}{n_f}) \times n/2$$

Figure 9 shows the simulation results where the above computation is used. The basic scheme is used with a fixed network transmission range $r_t = 15$, $P = 100,000$ and $n = 10,000$. For a different key ring size k, r_f is determined by the two ways introduced in the previous section.

As can be observed in the Figure 9, the unicast cost of FTTL is bigger than the unicast cost of CSN. Moreover, while there is no broadcasting cost in CSN, the broadcasting cost in FTTL is not negligible. While CSN has much less communication cost, it has the following problem. First, if we use CSN, some fraction of neighboring nodes are not connected. For example, with the key ring size 140 and the 8 targeted neighbors, each node has about 9.12% neighbors that are not connected. (Of course, this was the theoretical bound we computed.) Second, CSN has an additional computation cost. As explained in the protocol, each node compares the key rings in the every broadcast message it receives with its own and the targeted neighbor's (in the small disk). Thus, if the key ring size is large or it receives a lot of broadcast messages, the computation complexity can be significant. In other words, CSN trades off communication with computation.

To solve the problems of FTTL and CSN, we have designed a protocol, the details of which (including the simulation and theoretical analysis) will be provided in the extended version of this paper. First, key set-up phase is the same as the original protocol. Second, after the key set-up phase, each node i broadcasts its own id and the id of unconnected neighbors (in the small disk). Its secure neighbor node j checks if one of the unconnected neighbors of i is its neighbor or not. If so, it replies to i to connect the unconnected node through itself. After the first broadcast message of each node, some of the neighboring nodes will be connected. Node i, on the other hand, checks its neighbors' unconnected list to see if it can find others to help. Note that this may not require any additional broadcast message after the first broadcast of unconnected node list set. As time goes by, more neighbor nodes are connected.

6. CONCLUDING REMARKS

This paper analyzes relation between connectivity, memory size, and security of key pre-distribution schemes for the sensor networks. Using the giant component analysis, we have shown that we can trade-off connectivity, memory size, and security in the various density of sensor networks. We can reduce the key ring size, improve the security or the scalability of the previous schemes. Our simulation results show that the actual performance of the path key identification can be significantly improved by using cascade-on counting method. Lastly, we introduce ways to narrow the gap between the "optimal" network degree given by the networking community and the "required" network degree given by the security community. We believe our techniques can improve the practicality of the existing key pre-distribution schemes for sensor networks.

7. REFERENCES

[1] μamps: μ-adaptive multi-domain power aware sensors. `http://www-mtl.mit.edu/research/icsystems/uamps`.

[2] Wireless integrated network sensors, ucla. `http://www.janet.ucla.edu/WINS`.

[3] N. Alon and J. Spencer. The probabilistic method. In *Wiley-Interscience*, 2000.

[4] R. Blom. An optimal class of symmetric key generation systems. In *EUROCRYPT 84*, 1985.

[5] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung. Perfectly-secure key distribution for dynamic conferences. In *Crypto*, 1992.

[6] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Research in Security and Privacy*, 2003.

[7] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. In *Conference of the IEEE Communications Society (Infocom)*, 2004.

[8] W. Du, J. Deng, Y. S. Han, and P. K. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *ACM Conference on Computer and Communications Security (CCS)*, 2003.

[9] P. Erdös and A. Rényi. On the evolution of random graph. *Institute of Mathematics Hungarian Academy of Sciences*, 1959.

[10] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *ACM Conference on Computer and communication Security (CCS)*, 2004.

[11] T. Hou and V. Li. Transmission range control in multihop packet radio networks. In *IEEE Transaction on Communications*, 1986.

[12] S. Janson, T. Luczak, and A. Rucinski. *Random Graphs*. Wiley, 2000.

[13] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Next century challenges: Mobile networking for smart dust. In *Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 1999.

[14] L. Kleinrock and J. Silvester. Optimum transmission radii for packet radio networks or why six is a magic number. In *IEEE National Telecommunication Conference*, 1978.

[15] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *ACM Conference on Computer and Communications Security (CCS)*, 2003.

[16] A. M. R. D. Pietro and L. V. Mancini. Efficient and resilient key discovery based on pseudo-random key pre-deployment. In *4th IEEE Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks*, 2004.

[17] J. Spencer. The strange logic of random graphs. In *Algorithms and Combinatorics*, 2000.

[18] H. Takagi and L. Kleinrock. Optimal transmission ranges for randomly distributed packet radio terminals. In *IEEE Transaction on Communications*, 1984.

[19] D. D. Wentzloff, B. H. Calhoun, R. Min, A. Wang, and N. Ickes. Design considerations for next generation wireless power-aware microsensor nodes. In *International Conference on VLSI Design*, 2004.

[20] F. Xue and P. R. Kumar. The number of neighbors needed for connectivity of wireless networks. In *Wireless Networks*, 2004.

[21] S. Zhu, S. Setia, and S. Jajodia. Leap: Efficient security mechanisms for large-scale distributed sensor networks. In *10th ACM Conference on Computer and Communication Security (CCS)*, 2003.