Available online at www.sciencedirect.com
ScienceDirect
Ad Hoc Networks 5 (2007) 24–34
Ad Hoc Networks
www.elsevier.com/locate/adhoc

An effective key management scheme for heterogeneous sensor networks

Xiaojiang Du [a,*], Yang Xiao [b], Mohsen Guizani [c], Hsiao-Hwa Chen [d]

[a] Department of Computer Science, North Dakota State University, Fargo, ND 58105, United States
[b] Department of Computer Science, The University of Alabama, Tuscaloosa, AL 35487, United States
[c] Department of Computer Science, Western Michigan University, United States
[d] Institute of Communications Engineering, National Sun Yat-Sen University, Taiwan

Available online 30 June 2006

Abstract

Security is critical for sensor networks used in military, homeland security and other hostile environments. Previous research on sensor network security mainly considers homogeneous sensor networks. Research has shown that homogeneous ad hoc networks have poor performance and scalability. Furthermore, many security schemes designed for homogeneous sensor networks suffer from high communication overhead, computation overhead, and/or high storage requirement. Recently deployed sensor network systems are increasingly following heterogeneous designs. Key management is an essential cryptographic primitive to provide other security operations. In this paper, we present an effective key management scheme that takes advantage of the powerful high-end sensors in heterogeneous sensor networks. The performance evaluation and security analysis show that the key management scheme provides better security with low complexity and significant reduction on storage requirement, compared with existing key management schemes.

Keywords: Heterogeneous sensor networks; Key management; Security

1. Introduction

Most existing research on sensor networks considers homogeneous sensor networks, i.e., all sensor nodes have the same capabilities in terms of communication, computation, memory storage, energy supply, reliability, etc. However, a homogeneous ad hoc network suffers from poor performance and scalability. Recent research has demonstrated its performance bottleneck both theoretically [1], and through simulation experiments and testbed measurement [2]. E.g., Gupta and Kumar [1] showed that the per node throughput in a homogeneous ad hoc network is $\Theta\left(\frac{1}{\sqrt{n \log n}}\right)$, where n is the number of nodes. Furthermore, recently deployed sensor network systems are increasingly following heterogeneous designs, incorporating a mixture of sensors with widely varying capabilities [3]. E.g., a

* Corresponding author.
E-mail addresses: Xiaojiang.Du@ndsu.edu (X. Du), yangxiao@ieee.org (Y. Xiao), mguizani@cs.wmich.edu (M. Guizani), hshwchen@ieee.org (H.-H. Chen).

1570-8705/$ - see front matter
doi:10.1016/j.adhoc.2006.05.012

sensor network in [3] includes small MICA2 sensors (manufactured by Crossbow Technology [4]) as well as more powerful robotic nodes. Several recent works have studied Heterogeneous Sensor Networks (HSN). In [5], Mhatre et al. studied the optimum node density and node energies to guarantee a lifetime in HSN. Duarte-Melo and Liu analyzed energy consumption and lifetime of HSN in [6]. Yarvis et al. [7] studied how to lay out heterogeneous sensors to increase network lifetime if sensor locations are controllable. However, we have not seen any security scheme specifically designed for HSN, which take advantage of more powerful nodes in HSN. Research [6–9] shows that HSN can significantly improve network performance. To achieve better performance and security, we adopt an HSN model that consists of a small number of powerful High-end sensors (H-sensors) (e.g., PDAs) and a large number of Low-end sensors (L-sensors), e.g., the MICA2-DOT [4].

Security is very important for sensor network applications in military, homeland security and other hostile environments. Several literatures have studied security issues in homogeneous sensor networks [10–14]. Key management is an essential cryptographic primitive upon which other security primitives are built. Most security requirements, such as privacy, authenticity and integrity, can be addressed by building upon a solid key management framework. Several key management schemes [10–13] have been proposed for homogeneous sensor networks. However, most existing key management schemes are designed for homogeneous sensor networks, and many of these schemes suffer from high communication or computation overhead, and/or high storage requirement. No key management scheme has been designed for HSN.

Probabilistic key pre-distribution is a promising scheme for key management in sensor networks. To ensure the scheme work well, the probability (referred to as key-sharing probability in the following) that each sensor has at least one shared key with a neighbor sensor should be high. For the key pre-distribution scheme in [10], each sensor randomly selects its key ring from a key pool with size P. When the key pool size is large, each sensor needs to pre-load a large number of keys to achieve a high key-sharing probability. For example, when P is 10,000, each sensor needs to pre-load more than 150 keys to achieve a key-sharing probability of 0.9 [10]. If the key length is 256 bits, then 150 keys require a storage space of 4800 bytes. Such a storage requirement is too large for many sensor nodes. For example, a smart dust sensor [15] has only 8 K bytes of program memory and 512 bytes of data memory.

In this paper, we present an effective and scalable key management scheme for HSN. The scheme utilizes strong capabilities of H-sensors in computation, communication, storage, energy supply and reliability. The rest of the paper is organized as following. In Section 2, we present the key management scheme for HSN. We present other security schemes for HSN in Section 3. Section 4 is the performance evaluation and Section 5 is the security analysis. Section 6 concludes the paper. Below are the notations used in the rest of the paper: (1) u, v, x, y and n are L-sensors; (2) H is an H-sensor; (3) $\{m\}_k$ denotes encrypting message m with key k.

2. The key management scheme for heterogeneous sensor networks

In this Section, we present a key management scheme specifically designed for Heterogeneous Sensor Networks. We consider an HSN consisting of two types of sensors: a small number of powerful H-sensors and a large number of L-sensors. First, we list the assumptions of HSN below.

1. Due to cost constraints, L-sensors are not equipped with tamper-resistant hardware. Assume that if an adversary compromises an L-sensor, she can extract all key material, data, and code stored on that node.
2. H-sensors are equipped with tamper-resistant hardware. It is reasonable to assume that powerful H-sensors are equipped with this technology.
3. Each L-sensor (and H-sensor) is static and aware of its own location. Sensor nodes can use location services such as [16] to estimate their locations, and no GPS receiver is required at each node.
4. Base stations are trusted.

Clusters are formed in an HSN. Clustering-base schemes are promising techniques for sensor networks because of their good scalability and support for data aggregation. For an HSN, it is natural to let powerful H-sensors serve as cluster heads and form clusters around them. In Section 2.1, we discuss the cluster formation in HSN. In Section 2.2, we present the HSN key management scheme.

2.1. The cluster formation

In this subsection, we briefly describe the cluster formation scheme in HSN. Details of the clustering scheme can be found in [9]. Both L-sensors and H-sensors are distributed in the network. For simplicity, we assume that both L-sensors and H-sensors are uniformly and randomly distributed in the network. Note that our key management schemes also work for other sensor distributions. During sensor network initialization, each H-sensor broadcasts a Hello message to nearby L-sensors using the maximum power and with a random delay. The random delay is to avoid the collision of Hello messages from two neighbor H-sensors. A Hello message includes the ID and location of the H-sensor. Given the large transmission range of H-sensors and a sufficient number of H-sensors distributed in the network, most L-sensors can receive Hello messages from one or more H-sensors. Then each L-sensor chooses the H-sensor whose Hello message has the best signal strength as the cluster head. Each L-sensor also records other H-sensors from which it receives the Hello messages, and these H-sensors will serve as backup cluster heads in case the cluster head fails. A HSN is divided into multiple clusters, where each H-sensor serves as the cluster head. If the network is a two-dimension plane, each L-sensor will select the closest H-sensor as the cluster head (except when there is an obstacle in between), and this leads to the formation of a Voronoi diagram where the cluster heads are the nuclei of the Voronoi cells. An example of the cluster formation is shown in Fig. 1, where the small squares are L-sensors, large rectangular nodes are H-sensors, and the large square at the top-right corner is the base station (BS).

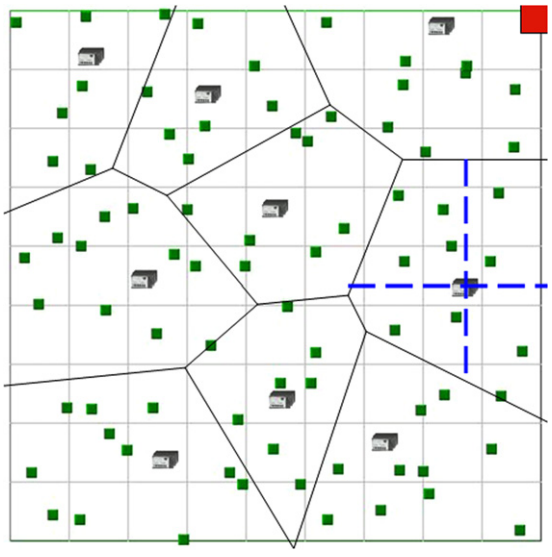


Fig. 1. The Cluster structure in HSN.

2.2. The asymmetric pre-distribution key management scheme

In this subsection, we present an effective key management scheme for HSN. The main idea is to pre-load only a small number of keys in each L-sensor while pre-load a relatively large number of keys in each H-sensor, since an H-sensor has much larger storage space than an L-sensor. Furthermore, H-sensors have tamper-resistant hardware to protect a large number of keys. Since the number of pre-distributed keys in an H-sensor is quite different from that in an L-sensor, we refer to this scheme as asymmetric pre-distribution (AP) key management scheme. The AP scheme includes three phases: key pre-distribution phase, shared-key discovery phase, and H-sensor based pairwise key setup phase. We discuss the three phases below.

Key pre-distribution phase includes several steps. First a large pool of P keys and the corresponding key IDs are generated. Then each L-sensor is pre-loaded with l keys, randomly selected from the key pool without replacement. The l keys form a key ring in each L-sensor. Each H-sensor is pre-loaded with M ($M \gg l$) keys, also randomly selected from the key pool without replacement. In addition, each H-sensor is pre-loaded with a special key K_H. K_H is also known to the BS, but not to any L-sensor.

Shared-key discovery phase can be done in either a centralized way or a distributed way. The shared-key discovery phase begins after cluster formation. In the distributed way, each L-sensor communicates with its neighbors and find out the shared keys (if any). The simplest way for any two L-sensors to discover if they share a key is that each node broadcasts, in clear text, the list of key IDs on its key ring. In the centralized way, each L-sensor (say u) sends to its cluster head (say H) a clear (un-encrypted) *Key-list* message, which includes the L-sensor ID – u, key IDs in u, and u's location. Then H discovers the shared-keys between each pair of neighboring L-sensors. H can determine if two L-sensor u and v are (one-hop) neighbors based on their locations: if the distance between u and v is less than the transmission range of an L-sensor, H assumes that u and

v are neighbors. Of course sometimes this is not true, e.g., there is an obstacle between two nodes. However, this will not affect the security of our key management scheme.

After discovering shared-keys between each pair of neighboring L-sensors, H disseminates the shared-key information to L-sensors using *Shared-key* messages. A *Shared-key* message includes a list of triple {shared-key ID s, u,v}, which means that L-sensor$u$ and v share key s. If u and v have more than one shared key, only one will be included in the *Shared-key* message to reduce overhead. If the number of L-sensors in the cluster is not very large, one *Shared-key* message can include the triples for all pairs of neighbors. Aggregating all the triples in one packet can reduce both the packet header overhead and delay caused by multiple transmissions. Otherwise, the H-sensor could send a short *Shared-key* message with one triple to each pair (e.g., multicasting to only u and v). Another way to distribute the shared-key information is to divide the cluster into several sections. For example, the center-right cluster in Fig. 1 is divided into four sections (by the dashed lines). Then the H-sensor can send to each section one Shared-key message, which includes the triples for all L-sensors in the corresponding section.

2.2.1. H-sensor based pairwise key setup phase

Some L-sensors may not share any pre-loaded key with neighbors. For each pair of L-sensors (denoted as x and y) that do not share any key, H first obtains a shared-key between H and x and a shared-key between H and y, and then H generates a pairwise key for each pair (e.g., x and y) and sends the key to them securely. First H checks if it has a pre-loaded key shared with the L-sensor (e.g., x). H is pre-loaded with a large number of keys so there is a high probability that H can find at least one shared key with x. If H does not share any key with x, the following scheme is used to set up pairwise key for x and y.

Definition 1. An L-sensor that shares at least one pre-loaded key with its cluster head is referred to as a 1st-*degree neighbor* of the cluster head.

In case that an L-sensor (say x) does not share any pre-loaded key with the cluster head H, H will check if any 1st-*degree neighbor* shares a key with x. Since every L-sensor sends the *Key-list* message to H, H knows the pre-loaded keys in each L-sensor in its cluster. If there is one (or more) 1st-*degree neighbor* (say z) that shares a key (say K_x) with x, then H can ask z to send the key K_x to H, encrypted by the shared key between z and H (say K_z), i.e., $z \rightarrow \mathrm{H}:\{K_x\}K_z$. Then H has a shared-key (K_x) with x.

Definition 2. An L-sensor that shares a key with its cluster head by the help of a *jth-degree neighbor* ($j = 1$ as above) is referred to as a $(j+1)$*th-degree neighbor* of the cluster head, where $j = 1, 2, 3, \ldots$

If none of the 1st-*degree* neighbors have a shared key with x, H will try the 2nd-*degree neighbors*, the 3rd-*degree neighbors*, up to dth-*degree neighbors*, where d is a system parameter. If none of the 1st $\sim$ dth degree neighbors have a shared key with x, H sends to the BS a *Request* message, which includes one key ID of node x; then the BS sends H the corresponding key, encrypted by K_H. To reduce the communication overhead, H can collect the IDs of the keys that need to be obtained from the BS, and sends only one *Request* message to the BS. After obtaining the keys from the BS, H can generate a pairwise key $K_{x,y}$ for each pair of L-sensors x and y, and unicasts the key $K_{x,y}$ to node x and y, encrypted by the shared key between H and x, and H and y, respectively, i.e., $\mathrm{H} \rightarrow x:\{K_{x,y}\}K_x$ and $\mathrm{H} \rightarrow y:\{K_{x,y}\}K_y$. Then x and y have a pairwise shared key $K_{x,y}$, and they can start secure communications.

3. Other security issues in HSN

In this Section, we discuss other security issues in HSN, including setting up broadcast keys, key revocation, and setting up keys for newly deployed sensor nodes.

3.1. Setting up broadcast keys

A broadcast key is used by an L-sensor to securely broadcast messages to its neighbors. After discovering shared-keys or setting up pairwise keys with each neighbor, it is easy to establish a broadcast key. An L-sensor u generates a broadcast key $K_{u\mathrm{B}}$, and unicasts to each neighbor the key, encrypted with the corresponding shared-key. For example, u sends the following packet to a neighbor v:$\{K_{u\mathrm{B}}\}K_{u,v}$, where $K_{u,v}$ is the shared-key between u and v.

3.2. Key revocation

When an L-sensor is compromised by an adversary, all the keys in that L-sensor need to be

revoked. Assume that the compromised node is detected by a scheme and is reported to the cluster head H. H disseminates a *Revocation* message containing a list of IDs of the keys to be revoked. A cluster head H shares keys with many L-sensors, including $1 \sim d$th neighbors. H computes one Message Authentication Code (MAC) using each of the shared keys. The *Revocation* message is formed as: (list of key IDs)$\|(u + MAC(K_{H,u},\text{list}))\|(v + MAC(K_{H,v},\text{list}))\|\cdots\|(y + MAC(K_{H,y},\text{list}))$, where u, v, and y are L-sensor IDs; $K_{H,u}$, $K_{H,v}$, and $K_{H,y}$ are the keys shared between H and u, v and y respectively; $\|$ is the operation of concatenation; and list is the "list of key IDs". When an L-sensor (which is included in the *Revocation* message, e.g., u) receives the *Revocation* message, it can check the integrity of the message by verifying the corresponding MAC.

3.3. Establishing keys for newly deployed sensors

After initial deployment, some areas of a sensor network may not be covered by any sensor node due to the randomness of sensor locations, e.g., when sensors are distributed from an airplane. Furthermore, low-end sensors are unreliable devices with limited power supply, and they may fail or run out of power over time. These factors can cause severe coverage and connectivity problems in sensor networks, and significantly degrade network performance and shorten network lifetime. To solve these problems, one solution is to deploy new sensor nodes in the field after some operation time. However, the additional deployment of sensors poses challenge on security schemes. Specifically, the newly deployed sensors need to establish security keys with existing sensors. This is a non-trivial problem because of the following reasons: 1. An existing sensor could have been compromised; 2 A newly deployed sensor could be an adversary node.

Many existing sensor network key management protocols [10–13] did not consider this issue. In the following, we present an efficient scheme to verify newly deployed sensors and establish pairwise keys between new L-sensors and existing L-sensors. Recall that each deployed H-sensor has a special key K_H. A newly deployed L-sensor (say n) is preloaded with l keys plus a special key $K_{L,n}$, where $K_{L,n}$ is generated by an one-way hash function F applied to the XOR of K_H and n, i.e., $K_{L,n} = F(K_H \oplus n)$. Note that node ID n can be padded or truncated to have the same length as K_H. After an L-sensor n is deployed in the field, n sends to the cluster head H a *Join* message: $n\|\text{location}\|MAC(K_{L,n},n\|\text{location})$. When H receives the *Join* message, it can generate $K_{L,n}$ by using n and K_H, and then H can verify if this *Join* message is from a legitimate new sensor. If yes, H determines the neighbors of n and generates one shared-key for n and each of its neighbors. Then H unicasts the shared-keys to $n$and its neighbors, and the transmissions are protected by the corresponding key.

4. Performance evaluation

In this Section, we evaluate the performance of the AP key management scheme.

4.1. Key pool size

For the AP scheme, the key pool size P is a critical parameter. A large P provides better security for the sensor network. Suppose that the number of preloaded keys is fixed. The larger the P value, the smaller the impact on other sensors' communications is when a fixed number of L-sensors are compromised. On the other hand, as P increases, the probability that two sensors share one common key decreases. In [11], Chan et al. propose the q-composite keys scheme that requires that two sensors share at least q keys in order to have a secure connection. The q-composite keys scheme provides better security for sensor networks. The two key management schemes in this paper can be easily extended to require at least q shared keys. In this subsection, we discuss the general case where q shared keys are required to set up secure connections.

We want to find the largest key pool size such that the probability of an L-sensor and an H-sensor sharing at least q keys is no less than a threshold p. Let $p(j)$ be the probability that an L-sensor and an H-sensor have exactly j keys in common. Recall that an L-sensor and an H-sensor are pre-loaded with l and M keys, respectively. An L-sensor has $\binom{P}{l}$ different ways of picking l keys from a key pool with the size P, and an H-sensor has $\binom{P}{M}$ different ways of picking M keys from the key pool. Thus, the total number of ways for an L-sensor and an H-sensor to pick l and M keys, respectively, is $\binom{P}{l}\binom{P}{M}$. Suppose that the two nodes have j keys in common. There are $\binom{P}{j}$ ways to pick j common keys. After the j common keys are picked, there remain $M + l - 2j$ distinct keys in the two key rings which are to be

picked from the remaining pool of $P-j$ keys. The number of ways to do so is $\binom{P-j}{M+l-2j}$. The $M+l-2j$ distinct keys must then be partitioned between the L-sensor and the H-sensor. The number of such partitions is $\binom{M+l-2j}{l-j}$. Hence the total number of ways to choose two key rings with j keys in common is the product of the three terms, i.e., $\binom{P}{j}\binom{P-j}{M+l-2j}\binom{M+l-2j}{l-j}$.

Thus, we have:

$$p(j) = \binom{P}{j}\binom{P-j}{M+l-2j}\binom{M+l-2j}{l-j} \Big/ \left[\binom{P}{l}\binom{P}{M}\right]. \quad (1)$$

Let p_c be the probability that an L-sensor and an H-sensor share sufficient keys to form a secure connection. If q shared-keys are required, we have: $p_c = 1-(p(0)+p(1)+\cdots+p(q-1))$. For given key ring size l and M, key overlap q, and minimum connection probability p, the largest key pool size P can be computed such that $p_c \geqslant p$.

4.2. Significant storage saving

The AP scheme can significantly reduce sensor storage requirement for key pre-distribution, compared to the key pre-distribution schemes [15,16] for homogeneous sensor networks. The probability (p_{s1}) of sharing at least one key between an H-sensor and an L-sensor in the AP scheme is

$$p_{s1} = 1-\Pr[\text{do not share any key}] = 1-\frac{\binom{P}{M}\binom{P-M}{l}}{\binom{P}{l}\binom{P}{M}} = 1-\frac{(P-M)!(P-l)!}{P!(P-M-l)!}. \quad (2)$$

For comparison, we also list the probability of sharing at least one key between two homogeneous sensors. This probability is given in [15], and is listed below:

$$p_{s2} = 1-\frac{\binom{P}{m}\binom{P-m}{m}}{\binom{P}{m}\binom{P}{m}} = 1-\frac{((P-m)!)^2}{P!(P-2m)!}, \quad (3)$$

where m is the size of the key ring in each sensor.

For different values of P, M, l and m, we plot p_{s1} – the probability of sharing at least one key, under the AP scheme and p_{s2} – the key pre-distribution scheme proposed by Eschenauer and Gligor [15], which is referred to as E–G scheme in the following. In Fig. 2, the key pool size ranges from 1,000 to 10,000, with an incremental of 1000. The solid lines with circles are the probabilities of the AP scheme, and the dotted lines with crosses are the probabilities of the E–G scheme. There are four pairs of curves in Fig. 2. From bottom to top, the corresponding parameters $[M,l,m]$ for the four pairs of probability curves are [125, 5, 25]; [250, 10, 50]; [375, 15, 75]; and [500, 20, 100], respectively. We observe that the probability of sharing key increases when the number of pre-loaded keys increases. For the same parameters $[M,l,m]$, the probability of sharing key decreases as the key pool size becomes large. Furthermore, we notice that the two probability curves of each pair are very close to each other. The relationship between these parameters $[M,l,m]$ is $M\times l = m^2$. I.e., if an H-sensor and an L-sensor is pre-loaded with M and l keys, respectively, and $M\times l = m^2$, then p_{s1} is roughly the same as p_{s2} with m keys pre-loaded. Since an H-sensor has much larger memory space and can be pre-loaded with a relatively number of keys, i.e., M can be large, each L-sensor only needs to be pre-loaded a small number of keys, while achieving similar key sharing probability as that in homogeneous sensor networks.

An example is given below to show the storage saving achieved by the AP key management scheme. Suppose there are 1000 L-sensors and 10 H-sensors

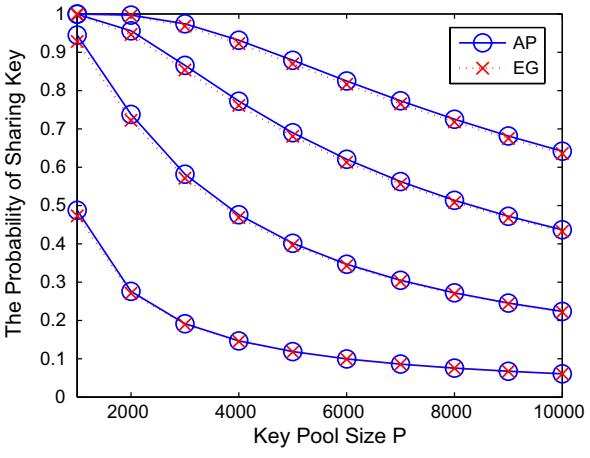


Fig. 2. The key sharing probability for different P.

in an HSN. In the HSN, each L-sensor is pre-loaded with $l = 20$ keys, and each H-sensor is pre-loaded with $M = 500$ keys. For comparison, consider a homogeneous sensor network with 1000 L-sensors, and each L-sensor is pre-loaded with $m = 100$ keys to have similar key sharing probability. In the HSN, the total memory used to store the pre-loaded keys is $20 \times 1000 + 500 \times 10 = 25{,}000$ (in the unit of the key length). In the homogeneous sensor network, the total memory used to store the pre-loaded keys is $100 \times 1000 = 100{,}000$, which is much larger than (four times of) that in the HSN. As we can see from the above example, our HSN key management scheme significantly reduces the total storage requirement in sensor nodes, which achieving a similar key sharing probability.

In Fig. 3, we also plot the key sharing probability for different numbers of pre-loaded keys. The solid lines with circles are the probabilities of the AP scheme, and the dotted lines with crosses are the probabilities of the E–G scheme. The x-axis is the number of pre-loaded keys – m in the E–G scheme, where m varies from 25 to 200, with an incremental of 25. For HSN, the number of pre-loaded keys in an L-sensor and an H-sensor are l and M, respectively, and they are set in the following way: $l = m/5$ and $M = 5m$ such that $M \times l = m^2$. As we can see from Fig. 3, the probability of sharing key increases as more keys are pre-loaded in a sensor node. For the four pairs of curves, from top to bottom, the corresponding key pool size P is 1000, 2000, 5000, and 10,000, respectively. Fig. 3 shows that the larger the key pool size, the smaller the probability of sharing key for given number of pre-loaded keys is.

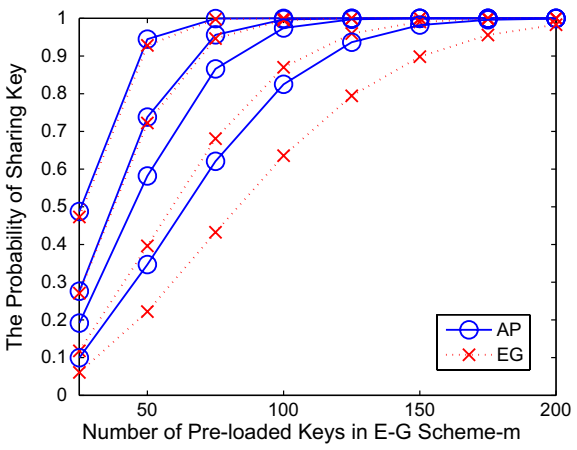


Fig. 3. The key sharing probability for different m.

4.3. The probability of being a Kth-degree neighbor

In each cluster, there are several L-sensors and one H-sensor. As discussed in Section 2, the L-sensors in a cluster are classified into 1st-degree neighbors, 2nd-degree neighbors, …, etc., depending on how they share keys with the H-sensor. In the following, we derive the probability that an L-sensor being a 1st-degree neighbor, a 2nd-degree neighbor, and so on. Recall that an L-sensor u and an H-sensor H are pre-loaded with l and M keys, respectively, from a key pool of size P. In order to be a 1st-degree neighbor, an L-sensor (say u) needs to share at least one pre-loaded key with the H-sensor. The probability that u is a 1st-degree neighbor of H is given in (2), i.e.,

$$p_1 \equiv 1 - \frac{(P-M)!(P-l)!}{(P-M-l)!P!}. \tag{4}$$

To be a 2nd-degree neighbor, an L-sensor v should share at least one key with one of the 1st-degree neighbors and v is not a 1st-degree neighbor. Let n_1 denote the number of 1st-degree neighbors, and let N denote the number of L-sensors in a cluster. The average number of 1st-degree neighbors in the cluster is $p_1 \times N$. Thus, we have an estimation of n_1 as below:

$$n_1 = \lfloor p_1 \times N \rfloor. \tag{5}$$

Let us first derive the probability that v shares at least one key with one of the 1st-degree neighbors.

$$\begin{aligned} p_{12} &\equiv \Pr(v \text{ shares at least 1 key with a 1st-degree neighbors}) \\ &= 1 - \Pr(v \text{ does not share any key with 1st-degree neighbors}) \\ &= 1 - \left(\binom{P}{l} \binom{P-l}{l} \Big/ \binom{P}{l}^2 \right)^{n_1} \\ &= 1 - \left(\binom{P}{l} \binom{P-l}{l} \Big/ \binom{P}{l}^2 \right)^{\lfloor p_1 \times N \rfloor}. \end{aligned} \tag{6}$$

Thus, the probability of v being a 2nd-degree neighbor is:

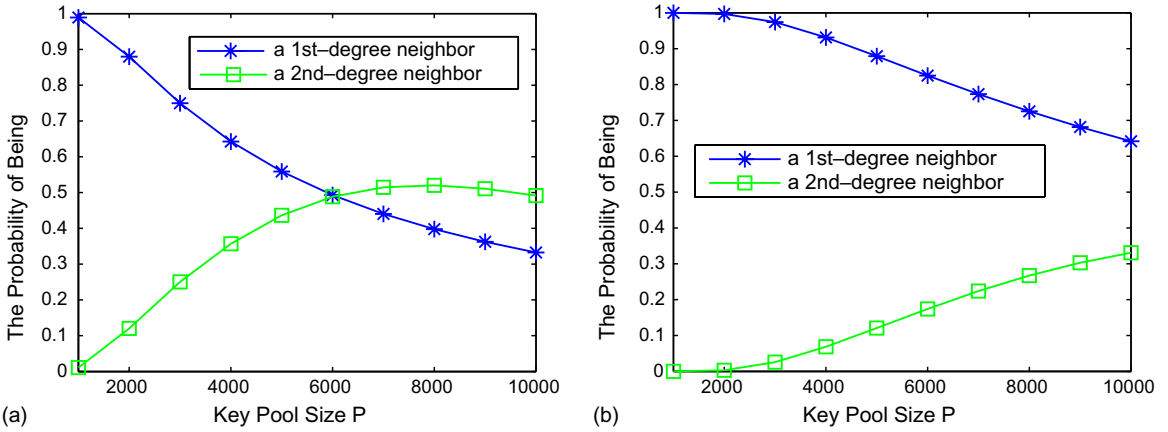


Fig. 4. (a) The probability of being a 1st/2nd-degree neighbor. (b) $M = 500$.

$$p_2 = \Pr((v \text{ is not a 1st-degree neighbor}) \cap (v \text{ shares at least 1 key with a 1st-degree neighbor})) = (1 - p_1) \times p_{12} = \frac{(P-M)!(P-l)!}{(P-M-l)!P!} \times \left(1 - \left(\frac{\binom{P}{l}\binom{P-l}{l}}{\binom{P}{l}^2}\right)^{\lfloor p_1 \times N \rfloor}\right). \quad (7)$$

Similarly, we can calculate the probability of an L-sensor being a 3rd-degree neighbor, 4th-degree neighbor, etc. We plot the probability of an L-sensor being a 1st-degree and 2nd-degree neighbor, i.e., p_1 and p_2 in Fig. 4, respectively, for different value of P, M, l and N. In Fig. 4, the key pool size P varies from 1000 to 10,000, with an incremental of 1000. The parameters of Fig. 4(a) are set as $M = 200$, $l = 20$ and $N = 100$, and the parameters of Fig. 4(b) are set as $M = 500$, $l = 20$ and $N = 100$. The solid line with circles represents the probabilities of an L-sensor being a 1st-degree neighbor, and the dotted line with crosses represents the probabilities of an L-sensor being a 2nd-degree neighbor. As we can see from Fig. 4, when the number of pre-loaded keys in an H-sensor – M is larger, the probability of an L-sensor being a 1st-degree is very large. For example, when $M = 500$, the probability is 0.88 and 0.64 for a key pool size of 5000 and 10,000, respectively. This means that with a high probability an L-sensor directly shares a key with its cluster head, and this significantly reduces the communication overheads of finding shared key between L-sensors and H-sensors. I.e., for most L-sensors, an H-sensor does not need to communicate with other L-sensors or the BS to obtain a shared key.

5. Security analysis

In this Section, we analyze the resilience of the AP key management scheme against node compromise attack. We want to find out the effect of c L-sensors being compromised on the rest of the network. For example, for any two L-sensors u and v which are not compromised, what is the probability that the adversary can decrypt the communications between u and v when c L-sensors are compromised? Note that keys in H-sensors are protected by tamper-resistant hardware, so they are safe. Recall that each L-sensor is pre-loaded with l keys. The probability of a given key K belonging to an L-sensor's key ring is l/P, where P is the key pool size. Therefore, the probability of K NOT in an L-sensor's key ring is $\left(1 - \frac{l}{P}\right)$. Furthermore, the probability of K NOT in any of the key rings of the c L-sensors is $\left(1 - \frac{l}{P}\right)^c$. Thus, the probability of a given key K in any of the key rings of the c L-sensors is:

$$1 - \left(1 - \frac{l}{P}\right)^c \quad (8)$$

Thus, Eq. (8) provides the probability of a given key K known to the adversary when c L-sensors are compromised. As we can see, the smaller l is, the smaller the probability of compromising another link is (i.e., better resilience against node compromise attack) for given key pool size P and the number of compromised L-sensors-c. This is another important

reason that we want to pre-load only a small number of keys in each L-sensor. Note that after key setup stage, two neighboring L-sensors u and v can use the established secure links to agree on a new random key which is only known to them and use the new key for later communications. Since the new key does not depend on the initial key pool P, the compromises of other L-sensors will not affect the security of the new key. Of course, if an adversary can record all the communications, she can still find out the new key if she knows the previous shared key between u and v from the compromised L-sensors.

If at least q keys are required to setup a secure link between two L-sensors, the probability of compromising a new link, given c L-sensors are compromised, is even smaller. If the key is the hash of j shared keys between two L-sensors, the probability of the link being compromised is $\left(1-\left(1-\frac{l}{P}\right)^c\right)^j$. The probability of setting up a secure link in an HSN is $p = p(q) + p(q+1) + \cdots + p(l)$, where $p(j)$ is given in (1). Thus, the probability that a secure link between two L-sensors is compromised when c L-sensors are captured is:

$$C(l) = \sum_{j=q}^{l} \left(1 - \left(1 - \frac{l}{P}\right)^c\right)^j \frac{p(j)}{p}, \tag{9}$$

In [11], Chan et al. give the probability that two sensors have exactly j keys in common: $p'(j) = \binom{P}{j}\binom{P-j}{2(m-j)}\binom{2(m-j)}{m-j}\Big/\binom{P}{m}^2$, where m is the number of keys pre-loaded in each sensor. Chan et al. provide the probability of compromising a secure link in a homogeneous sensor network as:

$$C(m) = \sum_{j=1}^{m} \left(1 - \left(1 - \frac{m}{P}\right)^c\right)^j p'(j) \Bigg/ \sum_{j=1}^{m} p'(j), \tag{10}$$

where m is the number of pre-loaded keys in each sensor. The function $C(l)$ in (9) is an increasing function of l. Since the number of pre-loaded keys in an L-sensor – l is much smaller than m, the probability in (9) is much smaller than that in (10). This means that the HSN key management scheme is more resilient to node compromise attack than key management schemes for homogeneous sensor networks, such as those in [10] and [11].

In Fig. 5, we plot the probability that an adversary can decrypt the communications between two sensors u and v when c L-sensors (other than u and v) are compromised (referred to as compromising probability). In Fig. 5, the number of keys required to setup a secure link-$q = 1$, the key pool size P is 10,000, and the number of compromised sensors – c varies from

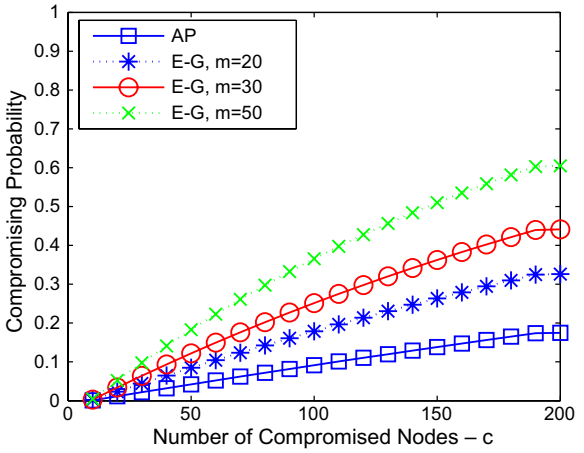


Fig. 5. The compromising probability when $q = 1$.

1 to 200, with an incremental of 10. For the E–G scheme, we calculate the probabilities for three different values of m: 20, 30, and 50. For the AP key management scheme, the parameters are set as: $M = 100$, $l = 10$. The value of M and l makes the AP scheme has similar key sharing probability with the E–G scheme for $m = 30$, since $M \times l \approx m^2$. Fig. 5 shows that the compromising probability of the AP scheme is smaller than all the three E–G schemes, for any given number of compromised sensors. I.e., the AP key management scheme is more resilient to node compromise attack than E–G scheme. We can see from Fig. 5, for the E–G scheme, the more pre-loaded keys in a sensor, the larger the compromising probability is, i.e., the larger the impact on other links for given number of compromised sensors.

In Fig. 6, we plot the compromising probability when $q = 3$, and all other parameters are the same

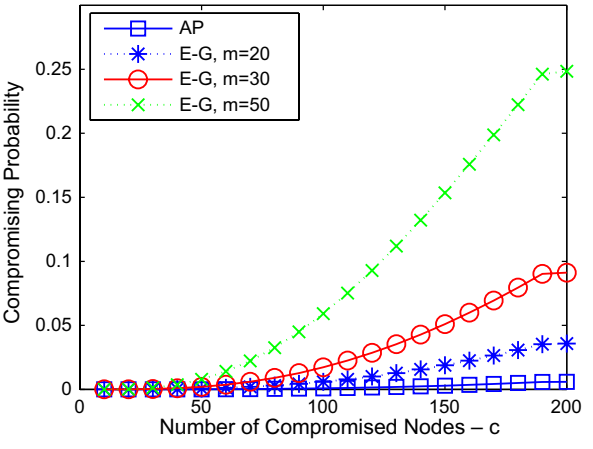


Fig. 6. The compromising probability when $q = 3$.

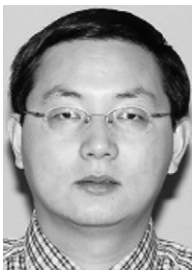
as those in Fig. 5. Note that for illustration purpose, the maximum vertical coordinate in Fig. 6 is 0.3 (it is 1.0 in Fig. 5). Comparing Figs. 5 and 6, we can see that the compromising probability when $q = 3$ is much smaller than the corresponding one when $q = 1$. For example, in the E–G scheme with $m = 50$, the compromising probability is only 0.25 for $q = 3$ and 200 sensors are compromised, while it is more than 0.61 when $q = 1$. This result shows that q-composite keys scheme can significantly improve the resilience against node compromise attack, as shown in [11].

6. Conclusions

In this paper, we present an effective key management schemes – the asymmetric pre-distribution (AP) scheme for heterogeneous sensor networks. The powerful H-sensors are utilized to provide simple, efficient and effective key set up schemes for L-sensors. Although tamper-resistant hardware is too expensive for L-sensors, it is reasonable to assume that powerful H-sensors are equipped with this technology. The performance and security analysis shows that the AP key management scheme can significantly reduce the sensor storage requirement while achieving better security (e.g., better resilience against node compromise attack) than existing sensor network key management schemes.

References

[1] P. Gupta, P.R. Kumar, The capacity of wireless networks, IEEE Transactions on Information Theory T-46 (2) (2000) 388–404.
[2] K. Xu, X. Hong, M. Gerla, An ad hoc network with mobile backbones, in: Proceedings of IEEE ICC 2002, New York, NY, April 2002.
[3] L. Girod et al., A system for simulation, emulation, and deployment of heterogeneous sensor networks, in: Proceedings of ACM SenSys, 2004.
[4] Crossbow Technology Inc., www.xbow.com.
[5] V. Mhatre, C.P. Rosenberg, D. Kofman, et al., A minimum cost heterogeneous sensor network with a lifetime constraint, IEEE Transactions on Mobile Computing 4 (1) (2005) 4–15.
[6] E. Duarte-Melo, M. Liu, Analysis of energy consumption and lifetime of heterogeneous wireless sensor networks, in: Proceedings of IEEE Globecom, November 2002, Taipei, Taiwan.
[7] M. Yarvis, N. Kushalnagar, H. Singh, et al., Exploiting heterogeneity in sensor networks, in: Proceedings of the IEEE INFOCOM 2005, Miami, FL, March 2005.
[8] X. Du, Y. Xiao, Energy efficient chessboard clustering and routing in heterogeneous sensor network, International Journal of Wireless and Mobile Computing (IJWMC), in press.
[9] X. Du, F. Lin, Maintaining differentiated coverage in heterogeneous sensor networks, EURASIP Journal on Wireless Communications and Networking (4) (2005) 565–572.
[10] L. Eschenauerm, V.D. Gligor, A key management scheme for distributed sensor networks, in: Proceedings of the Ninth ACM CCS, November 2002, pp. 41–47.
[11] H. Chan, A. Perrig, D. Song, Random key predistribution schemes for sensor networks, in: Proceedings of the 2003 IEEE Symposium on Security and Privacy, May 11–14, pp. 197–213.
[12] D. Liu, P. Ning, Establishing pairwise keys in distributed sensor networks, in: Proceedings of the 10th ACM CCS, Washington, DC, October, 2003.
[13] S. Zhu, S. Setia, S. Jajodia, LEAP: Efficient security mechanisms for large-scale distributed sensor networks, in: Proceedings of the 10th ACM CCS, Washington, DC, October, 2003.
[14] W. Du, R. Wang, and P. Ning, An efficient scheme for authenticating public keys in sensor networks, in: Proceedings of the sixth ACM MobiHoc, May, 2005, UIUC.
[15] J.M. Kahn, R.H. Katz, K.S.J. Pister, Mobile Networking for Smart Dust, in: Proceedings of ACM/IEEE MobiCom 99, Seattle, WA, August, 1999, pp. 271–278.
[16] A. Savvides, C. Han, M. Strivastava, Dynamic fine-grained localization in ad hoc networks of sensors, in: Proceedings of ACM/IEEE MobiCom'01, 2001, pp. 166–179.



Xiaojiang (James) Du is an assistant professor in Department of Computer Science, North Dakota State University. He received his B.E. degree from Tsinghua University, Beijing, China in 1996, and his M.S. and Ph.D. degrees from University of Maryland, College Park in 2002 and 2003, respectively, all in Electrical Engineering. His research interests are wireless sensor networks, mobile ad hoc networks, wireless networks, computer networks, network security and network management. He is an associated editor of Wiley Journal of Wireless Communication and Mobile Computing. He is the program chair of Computer and Network Security Symposium of IEEE International Wireless Communication and Mobile Computing Conference (IWCMC) 2006.

Yang Xiao is an assistant professor in Department of Computer Science at The University of Alabama. He was a voting member of IEEE 802.11 Working Group from 2001 to 2004. He is an IEEE Senior Member. He currently serves as Editor-in-Chief for International Journal of Security and Networks (IJSN) and for International Journal of Sensor Networks (IJSNet). He serves as an associate editor or on editorial boards for the following refereed journals: International Journal of Communication Systems (Wiley), Wireless Communications and Mobile Computing (WCMC), EURASIP Journal on Wireless Commu-

nications and Networking (WCN), and the International Journal of Wireless and Mobile Computing (IJWMC). His research areas are wireless networks, mobile computing, and network security. He has published more than 140 papers in major journals and refereed conference proceedings related to these research areas.

Hsiao-Hwa Chen is currently a full Professor at National Sun Yat-Sen University, Taiwan. He has authored or co-authored over 160 technical papers in major international journals and conferences, and five books and three book chapters in the areas of communications. He served as symposium co-chair of major international conferences, including IEEE VTC, ICC, Globecom, WCNC, etc. He served or is serving as an Editorial Board member or/and Guest Editor of IEEE Communications Magazine, IEEE JSAC, IEEE Wireless Communication Magazine, IEEE Networks Magazine, IEEE Trans. on Wireless Communications, IEEE Vehicular Technology Magazine, Wireless Communications and Mobile Computing (WCMC) Journal and International Journal of Communication Systems, etc. He is a Guest Professor of Zhejiang University, Shanghai Jiao Tung University, China.

Mohsen Guizani is currently a full professor and chair of the Computer Science Department at Western Michigan University. His research interests include computer networks, design and analysis of computer systems, wireless communications and computing, and optical networking. He currently serves on the editorial boards of many national and international journals, such as *IEEE Transactions on Vehicular Technology*, *IEEE Communications Magazine*, *Journal of Parallel and Distributed Systems and Networks*, and *International Journal of Computer Research*. He has served as a gu*est editor for IEEE Communications Magazine, IEEE JSAC, Journal of Communications and Networks*, and several other publications. He is the founder and Editor-In-Chief of Wiley *Wireless Communications and Mobile Computing Journal*.