

Detecting Misused Keys in Wireless Sensor Networks

Donggang Liu and Qi Dong
The University of Texas at Arlington
{dliu, qi.dong}@uta.edu

Abstract

Key management is the cornerstone for secure communication in sensor networks. Researchers have recently developed many techniques to setup pairwise keys between sensor nodes. However, these techniques allow an attacker to compromise a few sensor nodes and learn many pairwise keys used between non-compromised nodes. The attacker can then use these keys to impersonate non-compromised nodes and mislead the sensing application. To deal with this problem, this paper proposes to detect misused keys in sensor networks. The paper introduces a hidden layer of protection, which is designed for the security of pairwise keys rather than the messages in the network. It cannot be seen and will not be used by sensor nodes during normal communication. However, it can be checked by some special nodes to identify suspicious keys. With this idea, this paper develops a serial of techniques to detect misused keys. These techniques make it particularly difficult for an attacker to actively mislead the application using the compromised keys shared between non-compromised nodes. The paper also shows the effectiveness and efficiency of these techniques through analysis and experiments.

1 Introduction

Recent technological advances have made it possible to deploy wireless sensor networks consisting of a large number of low-cost, low-power, and multi-functional sensor nodes that communicate in short distances through wireless links [1]. Such sensor networks are ideal candidates for a wide range of applications in military and civilian operations such as health monitoring, data acquisition in hazardous environments, and target tracking. The desirable features of wireless sensor networks have attracted many researchers to develop protocols and algorithms to support these various applications.

Sensor networks may be deployed in hostile environments where enemies may be present. It is critical to ensure the integrity, availability, and at times confidentiality of the data collected by sensor nodes in hostile environments. However, providing security for sensor networks is particularly challenging due to the resource constraints on sensor

nodes [1] and the threat of node compromises [8].

Key management is the cornerstone of many security services such as authentication and encryption. Since the sensor nodes in the network usually communicate with other sensor nodes or base stations through their neighbors, a critical security service is a way to establish a pairwise key between two neighbors in the network. Research seeking low-cost pairwise key establishment techniques that can survive node compromises in sensor networks becomes quite active in the past three, four years, yielding several novel key pre-distribution schemes [2, 3, 5–7, 11, 12, 15].

To name a few, Eschenauer and Gligor proposed to distribute a random subset of keys from a key pool to every sensor node before deployment such that every pair of sensor nodes will have a certain probability of sharing at least one key after deployment [7]. Chan et al. extended this scheme by requiring two sensor nodes share at least q pre-loaded keys to establish a pairwise key [3]. Chan et al. also developed a random pairwise keys scheme [3]. This scheme pre-distributes random pairwise keys between a sensor node and a random subset of other sensor nodes, and has the property that the compromise of sensor nodes does not lead to the compromise of any key shared *directly* between two non-compromised sensor nodes. To further enhance the security of key pre-distribution, Liu and Ning and Du et al. independently developed two similar threshold-based schemes [6, 11]. Chan and Perrig also developed a protocol named PIKE for key establishment by using peer sensor nodes as trusted intermediaries [2].

Motivation: Despite the recent advances in pairwise key establishment, there are still many open problems. In particular, none of these existing schemes can guarantee source authentication between non-compromised nodes. An attacker can compromise a small number of sensor nodes and learn a quite large number of keys established *directly* or *indirectly* between non-compromised nodes. For example, in the basic probabilistic scheme [7], an attacker can learn a large number of keys in the key pool by compromising a small number of sensor nodes. As a result, it can quickly determine a large fraction of keys used between non-compromised sensor nodes. As another example, in the random pairwise keys scheme [3], a large fraction of pairwise keys are actually established indirectly

with the help of intermediate nodes. When an intermediate node is compromised, it is very likely that the keys established through this sensor node have been disclosed to the attacker. Hence, although this scheme provides perfect security guarantee for the keys established directly between sensor nodes, the compromise of sensor nodes will still reveal many keys established indirectly between non-compromised sensor nodes.

Generally, an attacker can always learn many pairwise keys shared between non-compromised sensor nodes unless a trust server generates and assigns a unique and random key for every two sensor nodes. For example, we can either use the trusted base station as a KDC to help the sensor nodes establish pairwise keys or pre-load each sensor node a unique key with every other sensor node. However, these ideas will introduce huge communication overhead or huge storage overhead and is not practical for resource-constrained sensor networks [7]. It is thus reasonable to assume that one of those recently developed key pre-distribution techniques is employed in the network. In this case, a few compromised sensor nodes will allow an attacker to know a quite large number of pairwise keys shared between non-compromised sensor nodes.

Once an attacker knows the pairwise key between two non-compromised nodes, it can forge and modify messages between these two nodes and actively mislead the sensing application. Clearly, the detection of these misused keys is of particular importance. This is a unique key management problem for sensor networks due to the resource constraints and the trade-offs we have to make between the storage, communication, computation and the security. However, this problem hasn't been studied by the researchers in sensor network security. The objective here is to detect misused pairwise keys in sensor networks.

Contributions: In this paper, we present a serial of techniques to detect the misused keys in the network. To the best of our knowledge, this is the first paper in this research direction. The main idea is to add a *hidden* layer of protection for the security of the keys rather than the messages in the network. It cannot be seen and will not be used by any sensor node during the normal communication. However, it can be checked by some special nodes (e.g., dedicated sensor nodes or base station) to identify misused keys.

The proposed methods have a number of advantages. First, even if an attacker has learned the keys shared between non-compromised sensor nodes, it is still difficult for him to impersonate non-compromised nodes using these keys and actively mislead the sensing application without being detected. Second, the proposed methods are independent from the detection of compromised nodes. Therefore, they can be used as stand-alone techniques to evaluate whether a given network is under attack. Third, the results generated by our detection approaches can actually help detect compromised nodes. For example, many existing pairwise key establishment techniques such as the grid-

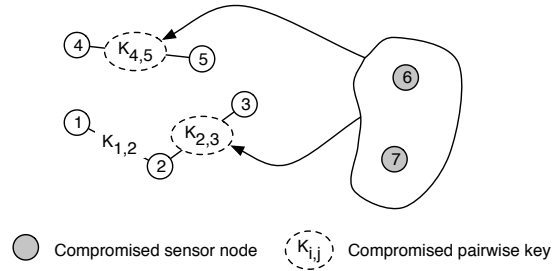


Figure 1. Example of compromised keys

based scheme [11] and PIKE [2] are deterministic. In these schemes, we can often figure out which sensor node is involved in the establishment of a given pairwise key. Thus, if a given pairwise key is detected to be misused, we can immediately narrow down the set of suspicious nodes.

Organization: This paper is organized as follows. The next section gives the system model and the design goals. Section 3 presents the technical detail as well as the analysis for the proposed approaches. Section 4 discusses the implementation issues. Section 5 reviews related work on sensor network security. Section 6 concludes this paper and discusses some future research directions.

2 System Models and Design Goals

In this paper, we consider wireless sensor networks consisting of a large number of resource-constrained sensor nodes and a powerful and resourceful base station. The sensor nodes are randomly scattered to sense and report the conditions and events in the field. The base station is used to collect or process the sensed results or act as a gateway to the traditional networks.

An attacker can launch a wide range of attacks against the network. For example, he can simply perform a denial-of-service (DoS) attack to jam the wireless channel and disable the network operation. Such DoS attack is simple but common to the protocols in every sensor network. There are no effective ways to stop the attacker from mounting such attack. However, in this paper, we are more interested in the attacks whose goal is to actively mislead the sensing application by inserting forged or modified messages into the network since generating a wrong result or making a wrong decision often causes more damage to the application.

In this paper, we assume that an attacker can eavesdrop, modify, forge, replay or block any network traffic. We assume that it is computationally infeasible to break the cryptographic primitives such as encryption and decryption. We assume the attacker can compromise a few sensor nodes and learn the keys between non-compromised sensor nodes. For example, in Figure 1, the compromise of nodes 6 and 7 will allow the attacker to learn the two keys $K_{2,3}$ and $K_{4,5}$. By checking the messages in the network, the attacker can easily determine the pairs of non-compromised nodes that share these two keys. Based on our previous discussion, we

know that this strategy is true for all the existing pairwise key establishment schemes.

Design Goals: As we mentioned, by compromising a few sensor nodes, an attacker can easily obtain many keys established between non-compromised nodes. For all existing pairwise key establishment protocols, there is no further protection for these keys. An attacker can use these compromised keys to impersonate non-compromised nodes and mislead the sensing application. Our overall goal is to provide additional protection for non-compromised nodes even if their shared keys have been disclosed. Specifically, we want to make sure that *the attacker cannot misuse the keys shared between non-compromised nodes.*

3 The Detection of Misused Keys

This section provides the technical detail on how to detect misused keys in the network. We will start with a simple and naive approach, where the base station makes the detection decisions in a centralized manner. We will then develop a serial of techniques to deal with the limitations of this naive approach and finally generate an efficient distributed detection approach.

The proposed detection schemes are independent from the pairwise key establishment protocol. They can be used for improving the resilience of any pairwise key establishment protocol. Hence, we skip the detail of pairwise key establishment and assume that the pairwise key between any two communicating sensor nodes has already been established. We assume that every message between two sensor nodes are protected by their shared pairwise key. For the sake of presentation, let $K_{u,v}$ be the key established between two nodes u and v .

3.1 Scheme I: The Naive Approach

In the naive approach, every sensor node adds a secret value, called the *committing value*, to every outgoing message in addition to the protection achieved by the pairwise key. This committing value will not be used by sensor nodes during the normal communication in sensor networks. However, it can be verified by the base station to detect if the key is misused. Simply speaking, the pairwise keys in the network are used for providing secure communication between sensor nodes, while the committing values are used for providing protection for the usage of keys.

To detect misused pairwise keys, we have the base station check the communication in the network and see if the messages transmitted in the network include incorrect committing values. A message including an incorrect committing value clearly indicates a misused key. Thus, every sensor node samples the messages received from other sensor nodes and generates a *report* to the base station for every sampled message. When the base station receives such report, it can then check if the committing value matches the sampled message. If not, the pairwise key is used improperly. In this way, we can easily detect those misused keys if

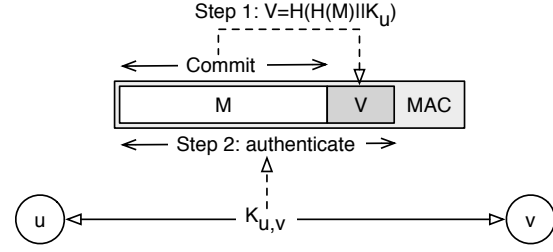


Figure 2. Example packet format in Scheme I

the attacker forges or modifies the messages between non-compromised sensor nodes to actively mislead the sensing application. The detail of this approach is given below.

- **Committing Messages:** Before deployment, each sensor node u is pre-loaded with a unique random key K_u . This key is only known by the base station and node u . In the naive approach, a sensor node simply adds a committing value to every outgoing message *other than the report message*. Let M be the message needs to be transmitted by node u . The committing value V is computed by $V = H(H(M)||K_u)$, where H is a one-way hash function and “||” denotes the concatenation operation. The final message $\{M, V\}$ will be protected by a proper pairwise key and transmitted over the wireless channel. For convenience, we call this message $\{M, V\}$ the *committed message*. Figure 2 shows an example of the packet format after applying our detection approach. For simplicity, we only provide integrity for the messages in this paper. However, confidentiality can be easily achieved in our method whenever it is required by the sensing application.
- **Sampling and Detection:** When a sensor node v receives a committed message $\{M, V\}$ from another sensor node u , it will first authenticate this message using the pairwise key shared with u . If the authentication fails, this message will be dropped; otherwise, it will construct a report message $M' = \{u, H(M), V\}$ at a probability of p . This report message M' will be protected by the key K_v and delivered to the base station for the detection of misused keys.

When the base station receives a report $\{u, H(M), V\}$ from node v , it checks if V equals $H(H(M)||K_u)$. If not, the key shared between u and v is not trustworthy. The base station will then notify v that its shared key with u is not secure anymore. The notification message may also include a new pairwise key for the secure communication between u and v .

In Scheme I, we can clearly see that an attacker cannot generate correct committing values for those forged or modified messages between two non-compromised sensor nodes u and v even if it has compromised their shared key $K_{u,v}$. The reason is that the attacker has to know K_u in order to generate a correct committing value for any message from

u to v . While this key is only known by the base station and the non-compromised sensor node u . Hence, as long as the report message for one forged or modified message between two non-compromised sensor nodes is successfully delivered to the base station, we can detect that the pairwise key has been misused.

A critical parameter in the proposed approach is the probability p of sending a report to the base station. Assume the attacker has forged or modified x messages between two non-compromised nodes, the probability of this misused key being detected can be estimated by $1 - (1 - p)^x$ when every report message can successfully reach the base station. Clearly, the more often the attacker misuses a pairwise key, the higher the probability of being detected. For example, if $p = 0.05$, when the attacker forges 10 messages, the misused key will be detected at a probability of 0.4; when the attacker forges 100 messages, the misused key will be detected at a probability of 0.994.

Since every message between two sensor nodes will lead to a report to the base station at a probability of p , the percentage of additional messages introduced by our detection approach can be roughly estimated by $p \times d_{avr}$, where d_{avr} is the average hop distance to the base station. For example, if $p = 0.05$, when the average distance to the base station is 10 hops, the number of messages transmitted in the network will be increased by 50 percent. Therefore, we usually prefer a small p for a reasonable communication overhead.

From the above discussion, we can clearly see the limitations of the naive approach. For security purpose, we prefer a large p so that we can detect misused keys with a high probability even if the attacker only forges or modifies a few messages between non-compromised nodes. However, a large p will lead to substantial communication overhead. In the next subsection, we will present an improved method which guarantees the detection of misused keys as long as one report can reach the base station.

3.2 Scheme II: Cumulative Commitment

In the previous approach, a committing value can only be used for the verification of one message. In the improved approach, we propose to *cumulatively commit* the communication between two nodes u and v so that a committing value can be used to verify all the previously transmitted messages. More specifically, a committing value is always generated based on the hash image of all the previous messages to the destination node. In this way, as long as one report can reach the base station, we can tell whether the pairwise key has been misused so far.

For simplicity, we assume that the communication between two *neighbor* sensor nodes is always reliable. This can be easily achieved by using standard fault tolerance techniques such as re-transmission. The details will be skipped in this paper for simplicity. We also assume that a sequence number is included in every message to deal with the replay attacks. This sequence number is increased by

one for every outgoing message. The main improvement we made in Scheme II is the way we generate the committing value. The sampling and detection part will be similar to the naive approach.

- *Committing Messages:* Consider two communicating sensor nodes u and v . Both of them maintain a variable $c_{u,v}$ to track the history of the communication from u to v . (Another variable $c_{v,u}$ will be needed for the communication from v to u .) Initially, we have $c_{u,v} = 0$. For every message M from u to v , node u simply updates $c_{u,v} = H(c_{u,v}||M)$ and computes the committing value V as $V = H(c_{u,v}||K_u)$. The final message from u to v is $\{M, V\}$, which will be protected by the key $K_{u,v}$ and delivered to node v .
- *Sampling and Detection:* This step is similar to the naive approach in the previous subsection. The only difference is: when a sensor node v receives a committed message $\{M, V\}$ from another sensor node u , it updates $c_{u,v}$, which will be the same as the value $c_{u,v}$ at node u under the reliable communication, and then sends the report message $\{u, c_{u,v}, V\}$ to the base station at a probability of p .

Scheme II significantly improves the performance of the naive approach. As we mentioned, every report message can be used to tell whether the pairwise key has been misused in any previous communication from the source node to the destination node due to the cumulative commitment. In contrast, the report message in the naive approach can only provide evidences for a single communication. It is very likely that the base station will miss those forged or modified messages even if every report message can be successfully delivered.

Assume every report from a regular sensor node can reach the base station. Consider two non-compromised nodes u and v . If an attacker has compromised their shared key $K_{u,v}$ and forged a message from u to v , the probability that it will be detected during the transmission of the next x messages from u to v can be estimated by $1 - (1 - p)^{x+1}$. Different from the naive approach, these x messages do not have to be forged or modified messages due to the cumulative commitment. Hence, even if the attacker only forged or modified a single message, the misused key will be detected as long as one report prepared by the destination node reaches the base station.

Certainly, an attacker may choose to forge a few messages and then block the communication from the source node to the destination node. It is very likely that the destination node has not sent any report to the base station yet. In this case, the misuse of this key will not be detected. A simple way to fix this is to have every sensor node buffer the last message received from any neighbor node and always generate a report to the base station if it has been a long time since the receipt of the last message. With this fix, the detection of misused keys between non-compromised sensor

nodes is almost guaranteed as long as the destination nodes do not run out of battery.

However, the cumulative commitment approach also has limitations that may affect its application in real-world scenarios. First, it still introduces considerable communication overhead to the base station due to the centralized detection. Second, although it guarantees to detect a misused key when at least one report for the usage of such key can reach the base station, the delay introduced in the detection may still cause problems, especially when we need a small p to reduce the communication to the base station. In the next subsection, we will present a scheme to address these two issues by making the detection part distributed.

3.3 Scheme III: Distributed Detection

A simple solution for distributed detection is to duplicate the base station’s keying materials at a number of tamper-resistant nodes and deploy these tamper-resistant nodes in the target field to monitor the network. These tamper-resistant nodes can thus perform the detection of misused keys in the same way as the base station. As long as a sensor node can find a tamper-resistant node in its local area, it can use this node for the detection. However, tamper-resistant packaging is still expensive and can not provide a high level of security. In this paper, we do not assume the tamper-resistant nodes. We are more interested in the algorithmic solutions.

The main idea in this subsection is to make the detection part of the previous scheme *distributed*. The communication cost to the base station only happens when a misused key is detected. This will greatly reduce the communication overhead and the detection delay. Indeed, we can have a large p to reduce the detection delay without increasing the communication overhead to the base station. Specifically, in the proposed approach, we use a number of dedicated sensor nodes, called the *detecting sensor nodes*, to sample the communication in the network. These detecting sensor nodes have the keying materials that allow them to check the committing values included in the messages and detect the misused keys. Once a key is detected to be compromised, the detecting node will notify the base station to revoke this key.

We assume that the attacker may compromise a few detecting sensor nodes. Once a detecting node is compromised, the attacker can discover all its keying materials. Thus, the keying materials in the detecting sensor nodes are different from that in the base station; but they should still allow the detecting sensor nodes to check the committing values. We assume that the base station is well protected and will not be compromised. The detailed protocol is described below.

- *Initialization:* Let m be the total number of detecting sensor nodes and n be the total number of regular sensor nodes in the network. Every regular sensor node u is pre-loaded with a unique and random key K_u .

This key is only known by the base station and node u . Every detecting sensor node i stores a hash value $H(K_u||i)$ for every regular sensor node u . Therefore, every detecting sensor node will be pre-loaded with n hash images, one for each regular sensor node. Figure 3(a) shows an example of pre-loaded keys for the regular and detecting sensor nodes when $n = 4$ and $m = 2$.

Clearly, the above key assignment will introduce substantial storage overhead for the detecting sensor nodes in case of a large sensor network. However, we believe that this is often feasible since the main function of the detecting sensor nodes is to monitor the usage of keys in the network. They can thus use all their memory, including the flash memory, to store these keys. For example, the TelosB motes have 1024Kbytes flash memory [4], which allows a detecting sensor node to store keys for a network of 128,000 sensor nodes if every key is 8-byte long.

- *Committing Messages:* After deployment, every sensor node u discovers the set $\mathcal{M}(u)$ of detecting sensor nodes in its neighborhood. The IDs in this list are ranked in an ascend order. For any two communicating neighbor sensor nodes u and v , they both need to get the other node’s list of neighbor detecting sensor nodes through the secure channel established with the key $K_{u,v}$. Hence, as long as the pairwise key $K_{u,v}$ is still secure, u will know the correct version of $\mathcal{M}(v)$, and v will also know the correct version of $\mathcal{M}(u)$.

Similar to the previous scheme, both u and v maintain a variable $c_{u,v}$ to track the history of the communication from u to v . (Another variable $c_{v,u}$ will be needed for the communication from v to u .) Initially, we have $c_{u,v} = 0$. For every new message M from u to v , we update $c_{u,v} = H(c_{u,v}||M)$. Let S be the size of $\mathcal{M}(v)$ and s be the sequence number included in M . Node u will use the detecting node i at the position $1 + (s \bmod S)$ in $\mathcal{M}(v)$ for detection. The purpose of choosing the detecting sensor node i in this way is to make sure that an attacker cannot always pick a compromised detecting node for its messages.

Node u then computes two committing values: $V_1 = H(c_{u,v}||H(K_u||i))$ and $V_2 = H(c_{u,v}||K_u)$. V_1 will be used by the detecting node i to check the committing value, while V_2 will be used only when the detecting node i notices the misuse of the key and needs to notify the base station to revoke the key. The final message from u to v is thus $\{M, V_1, V_2\}$, which will be further protected by the key $K_{u,v}$ and delivered to node v .

- *Sampling and Detection:* When v receives a committed message $\{M, V_1, V_2\}$ from u , it first gets the ID i of the detecting node at position $1 + (s \bmod S)$ in $\mathcal{M}(v)$ once this message is authenticated, where s is

the sequence number included in M . Node v then updates the variable $c_{u,v} = H(c_{u,v}||M)$ and send a report to the detecting node i at a probability of p . This report includes two protected pieces of information, $\{u, c_{u,v}, V_1\}$ and $\{u, v, c_{u,v}, V_2\}$. The first one is for the detecting node i , while the second one is for the base station. The second piece of information will be first protected by K_v , and then the whole report will be protected by $H(K_v||i)$. Figure 3(b) shows an example of the final scheme, where MAC1 is a message authentication code generated using the key $K_{u,v}$, MAC2 is generated using $H(K_v||i)$, and MAC3 is generated using K_v .

When the detecting node i receives the report from node v , it checks if V_1 equals $H(c_{u,v}||H(K_u||i))$ after authenticating the report. If not, the key $K_{u,v}$ is misused. In this case, if the misuse of $K_{u,v}$ hasn't been reported to the base station before, the detecting node i will send the second piece of information $\{u, v, c_{u,v}, V_2\}$ to the base station to revoke the key $K_{u,v}$; otherwise, it will ignore the report. When the base station receives this message, it will further check if V_2 equals $H(c_{u,v}||K_u)$ after the message is authenticated using the key K_v . If not, it will notify v that its shared key with u is not secure anymore. The notification message may include a new pairwise key for the secure communication between u and v .

Security Analysis: In the security analysis, we will focus on our final scheme, the distributed detection approach. Clearly, if a sensor node cannot find any detecting node in its local area, it is not possible to perform the detection of misused keys. As a result, we want to make sure that a regular sensor node can find at least one detecting node in its local area with a high probability. A critical parameter for this requirement is the number m of detecting sensor nodes in the network. For simplicity, we assume that the sensor nodes are evenly deployed in the field. Let b be the average number of neighbor nodes for every regular sensor node before the deployment of detecting sensor nodes. After the detecting sensor nodes are evenly deployed in the network, the probability that a regular sensor node cannot find any detecting node in its neighborhood can be estimated by $(1 - \frac{b}{n})^m$. The probability of finding at least one detecting sensor node in any given regular sensor node's neighborhood can be estimated by

$$p_{cover}(m, n) = 1 - (1 - \frac{b}{n})^m \approx 1 - (\frac{1}{e})^{\frac{bm}{n}}.$$

Clearly, for densely deployed sensor networks, we can usually deploy a small fraction of additional sensor nodes for detection so that most regular sensor nodes are covered by at least one detecting node. For example, when $b = 50$, we only need to deploy 10% additional sensor nodes ($m = \frac{n}{10}$) to make sure that a regular sensor node can find a detecting sensor node with a probability of 0.99.

Additionally, deploying a reasonable fraction of additional sensor nodes are usually acceptable for security sensitive operations. We thus assume that the number m of detecting sensor nodes is large enough so that every sensor node has at least one neighbor detecting sensor node in its neighborhood.

Consider two particular non-compromised sensor nodes u and v . There are two cases in terms of the pairwise key $K_{u,v}$. It is either compromised or not compromised. When the key $K_{u,v}$ is still secure, we can clearly see that neither of u and v is compromised. In addition, a non-compromised detecting sensor node will never report a false alarm against this key as shown in the following theorem.

Theorem 1 *A non-compromised detecting sensor node will never report a false alarm against any non-compromised pairwise key.*

Proof: Assume the detecting node i receives an authenticated report from v . Let $\{u, c_{u,v}, V_1\}$ be the first piece of information in this report. Since the key between u and v is still secure, and the communication between them is always reliable, we know that both of u and v will have the same copy of $c_{u,v}$. It is therefore guaranteed that V_1 equals $H(c_{u,v}||H(K_u||i))$. This means that a non-compromised detecting sensor node will never report false alarms against non-compromised pairwise keys. \square

The above theorem indicates that a non-compromised detecting sensor node will never introduce communication overhead to the base station unless it detects a misused key. However, when a detecting node is compromised, it can certainly report a false alarm. *Fortunately, this compromised detecting sensor node is still unable to affect the communication between non-compromised nodes u and v .* This can be explained in the following theorem.

Theorem 2 *A non-compromised pairwise key will never be revoked.*

Proof: According to the protocol description, we know that only the base station can revoke the pairwise keys in the network. Let $R = \{u, v, c_{u,v}, V_2\}$ denote the report from a compromised detecting sensor node to the base station. Since R is protected by K_v , which is only known by the base station and node v , the attacker cannot forge or modify any report. As a result, when the report is authenticated, it is always guaranteed that V_2 equals $H(c_{u,v}||K_u)$ as long as the key $K_{u,v}$ is still secure. This means that a non-compromised pairwise key will never be revoked. \square

When the pairwise key $K_{u,v}$ is compromised, we will study the *detection rate*, which is the probability of a misused key being detected. Let f_c be the fraction of those detecting sensor nodes that have been compromised. We assume that a report message can always reach the intended detecting node. We note that the keys stored on different detecting nodes are different from each other due to the one-way hash function H . We can clearly see that no matter how many detecting nodes are compromised, the secrets on those non-compromised detecting nodes are still safe.

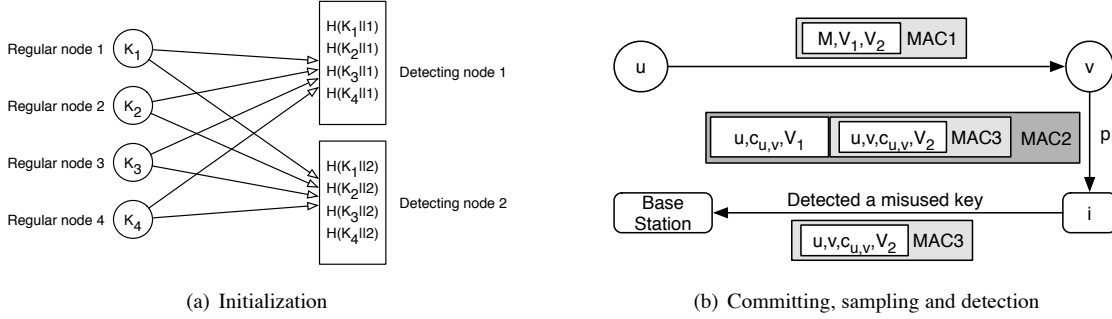


Figure 3. Illustration of the steps in Scheme III

Theorem 3 Let u and v be two non-compromised sensor nodes and x be the number of authenticated messages v received since the first misuse of the key $K_{u,v}$. The probability of this misuse being detected is at least $(1 - (1 - (1 - f_c) \times p)^{x+1}) \times p_{cover}(m \times (1 - f_c), n)$.

Proof: Clearly, when all the detecting nodes v can reach are compromised, we are unable to protect the usage of the keys for the communication to this sensor node. Based on our previous analysis, the probability that v finds at least one benign detecting node is $p_{cover}(m \times (1 - f_c), n)$. In addition, we note that v will generate a report to a detecting node at a probability of p for every authenticated message from u . When this detecting node is compromised, the misuse of $K_{u,v}$ will not be detected; otherwise, this misuse of $K_{u,v}$ will be detected at a probability of p . Since every detecting node in $\mathcal{M}(v)$ will be used by u at the same probability, the probability of detecting the misuse of $K_{u,v}$ after transmitting a single message can be estimated by $(1 - f_c) \times p$. Thus, after the transmission of x messages, the probability of detecting the misuse of this pairwise key is at least $1 - (1 - (1 - f_c) \times p)^{x+1}$. Hence, the overall probability of detecting a misused key is at least $(1 - (1 - (1 - f_c) \times p)^{x+1}) \times p_{cover}(m \times (1 - f_c), n)$. \square

Figure 4 shows the probability of detecting a misused key under different situations. From Figure 4(a), we can clearly see that the detection rate increases quickly with x , the number of messages transmitted from the source node to the destination node. It means that any misused pairwise key will be detected with a very high probability as long as the source node is still communicating with the destination node. We also note that increasing the probability p of sending a report will improve the probability of detecting a misused key given only a small number of messages. In other words, increasing p has a positive impact on reducing the detection delay. Since a large p implies more communication overhead, we thus have to make a trade-off between the detection delay and the communication overhead.

Figure 4(b) shows that when there are more compromised detecting sensor nodes, we will have less chance of detecting misused keys. In particular, when all the detecting nodes are compromised, we are not able to detect any misused key. Nevertheless, we can clearly see that the per-

formance of our detection approach degrades slowly with the number of compromised detecting sensor nodes.

We would like to mention that the proposed scheme can only detect the misuse of the keys shared between non-compromised sensor nodes. A compromised sensor node can certainly misuse its own keys to mislead the sensing application without being detected. Nevertheless, our detection method prevents the attacker from impersonating those non-compromised sensor nodes or non-existing sensor nodes. As a result, the attacker can only cause damage to the network using the compromised sensor nodes. For example, a compromised sensor node can only communicate with a non-compromised sensor node as “itself”. This greatly mitigates the impact of node compromises on the security of network operations.

Overheads: For every regular sensor node, the additional storage overhead introduced by our detection approach is a shared key with the base station and the space for the lists of nearby detecting sensor nodes and the variables to track the communication history with its neighbor nodes. Note that in most applications, a sensor node only needs to communicate with a few other nodes and only needs to save a few list of neighbor detecting nodes. Thus, the storage overhead at sensor nodes will not be a big problem in our approach. For every detecting node, it needs to store n keys for a network of n sensor nodes. As discussed before, this is often feasible for a reasonable size sensor network since all the sensor memory can be used to store these keys.

As for the computation overhead, our detection approach only involves a few efficient symmetric key operations for the regular sensor nodes. For the detecting sensor nodes, they need to access the flash memory for the keys needed to verify the messages. Certainly, accessing flash memory will be much slower than accessing RAM, and will likely cost more energy. However, because of the limited number of neighbors for every sensor node, a detecting sensor node only needs to retrieve a few keys from its flash memory during the entire lifetime. A detecting sensor node also needs to check every report message. However, this only involve a few number of efficient symmetric key operations.

As for the communication overhead, our detection approach needs two additional fields for the committing val-

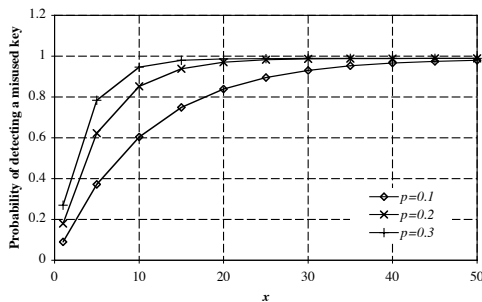
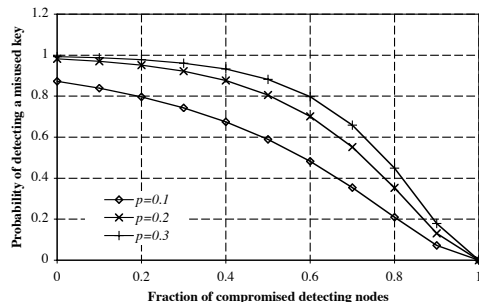
(a) $f_c = 0.1$ (b) $x = 20$

Figure 4. Probability of detecting a misused key. Assume average number of neighbors $b = 50$ and the total number of detecting nodes $m = \frac{n}{10}$.

ues in every outgoing message. A typical size of a committing value is 8-byte long. Hence, this is usually not a problem for the current generation of sensor networks. In addition to this, every sensor node will generate a report message at a probability of p for every message other than the report from its neighbor nodes. Thus, the total number of messages transmitted in the network will be approximately increased by a fraction of p . However, these report messages are only transmitted between neighbor nodes and will unlikely generate big impact on the performance of the sensing application. Note that a detecting sensor node also needs to notify the base station for revoking a misused key. However, this only happens when a misused key is detected. We therefore believe that our detection approach is practical for the current generation of sensor networks in terms of the storage, communication and computation overheads.

Advantages: The main advantage of our approach is the ability of determining whether a particular pairwise key has been misused before. This is particularly meaningful for us in many cases. In the following, we will show the benefits of our detection approach in improving the resilience of a pairwise key establishment technique. We will consider the basic probabilistic key pre-distribution scheme [7], the random subset assignment scheme [11], the matrix-based scheme [6], the random pairwise keys scheme [3], the grid-based scheme [11] and PIKE [2].

For the basic probabilistic key pre-distribution scheme [7], the random subset assignment [11] and the matrix-based scheme [6], a small number of compromised sensor nodes reveal a large fraction of keys in the network. As a result, the attacker will learn a large number of pairwise keys shared between non-compromised sensor nodes. Additionally, the attacker can easily create new sensor nodes using the secrets learned from compromised sensor nodes, leading to a Sybil attack [13]. These newly created sensor nodes can establish keys with legitimate sensor nodes and generate negative impact on the network at a large scale. By applying our detection method, both problems can be mit-

igated significantly. The misused keys between those non-compromised or newly created sensor nodes can be detected with a high probability.

For the random pairwise keys scheme [3], the grid-based scheme [11] and PIKE [2], an attacker learns nothing about or only a very small fraction of the keys shared directly between non-compromised sensor nodes. However, a large fraction of keys will be established indirectly with the help of one or a few intermediate sensor nodes. In this case, a few compromised nodes will leak many indirect keys. By applying our detection method, we can achieve additional security property. In particular, those indirect pairwise keys between non-compromised sensor nodes can be monitored for security purpose. Once they are misused by the attacker to mislead the sensing application, we are able to detect the misuse at a very high probability as shown in Theorem 3.

4 Implementation Issues

The security and overheads of our proposed detection approach has been thoroughly discussed in Section 3. The result shows that our scheme can *effectively* detect misused keys with a high probability. In this section, we will study the implementation issues. The purpose is to show that our detection approach can be efficiently implemented on resource-constrained sensor nodes.

We have implemented the distributed detection approach using TinyOS [9]. Since the base station is much more resourceful than sensor nodes, adding an additional piece of revocation mechanism is usually not a problem. Hence, we focus on the implementation for the detecting sensor nodes and for the regular sensor nodes. We use RC5 [14] to implement the security primitives such as hash and MAC operations. All the keys and hash values are 8-byte long.

For a detecting sensor node i , it needs to listen to the channel for the report message from regular sensor nodes. The detecting node may need to access the flash memory to get the key to check the committing value. If a misused key is detected, it will notify the base station to revoke the key.

For a regular sensor node, our detection method is implemented in the communication module. Specifically, whenever a new message other than the report needs to be delivered to other sensor nodes, we add two committing values to the message. Whenever a new message is received from other sensor nodes, we will generate and send a report message to a detecting sensor node at a certain probability after authenticating the message.

From the above discussion, we can see that the proposed scheme can be easily implemented on resource-constrained sensor nodes. The following table summarizes some important characteristics for our implementation on TelosB [4] motes. The RAM size for the regular nodes also include the space for 10 neighbor regular nodes, and each of them has 5 neighbor detecting nodes.

Detecting node	Code size with OS	ROM: 13086 bytes RAM: 1205 bytes
	# of hash operations	Detect: 2
		Report: 1
Regular node	Additional code size	ROM: 1668 bytes RAM: 494 bytes
		# of hash operations

5 Related Work

The closest related work to the techniques studied in this paper is pairwise key establishment. Many techniques have been proposed along this research direction, including the basic probabilistic scheme [7], the q -composite scheme [3], the random pairwise keys scheme [3], the two threshold-based schemes [6, 11] and PIKE [2]. Additionally, the prior deployment and post deployment knowledge were also used to improve the performance of key pre-distribution in various situations [5, 10, 12]. Our techniques in this paper address an important issue that hasn't been studied yet. The proposed techniques can improve the resilient of sensor network by detecting misused keys.

6 Conclusion and Future Work

This paper identifies an important security problem for key management in sensor networks, the detection of misused keys. In this paper, we present a serial of solutions to detect the misuse of pairwise keys in the network. The analysis indicates that our proposed approaches are efficient and effective for the current generation of sensor networks.

There are still a number of open issues. First, the proposed schemes cannot detect the misused key when one of the sensor nodes related to this key is compromised. It is interesting to see what else we can do to deal with this situation. In addition, we are also interested in the detection of compromised nodes that behave maliciously.

Acknowledgment The authors would like to thank the anonymous reviewers for their valuable comments.

References

- [1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *Computer Networks*, 38(4):393–422, 2002.
- [2] H. Chan and A. Perrig. PIKE: Peer intermediaries for key establishment in sensor networks. In *Proceedings of IEEE Infocom*, Mar. 2005.
- [3] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Research in Security and Privacy*, pages 197–213, 2003.
- [4] Crossbow Technology Inc. Wireless sensor networks. http://www.xbow.com/Products/Wireless_Sensor_Networks.htm. Accessed in February 2006.
- [5] W. Du, J. Deng, Y. S. Han, S. Chen, and P. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. In *Proceedings of IEEE INFOCOM'04*, March 2004.
- [6] W. Du, J. Deng, Y. S. Han, and P. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *Proceedings of 10th ACM Conference on Computer and Communications Security (CCS'03)*, pages 42–51, October 2003.
- [7] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 41–47, November 2002.
- [8] C. Hartung, J. Balasalle, and R. Han. Node compromise in sensor networks: The need for secure systems. Technical Report CU-CS-990-05, U. Colorado at Boulder, Jan. 2005.
- [9] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. S. J. Pister. System architecture directions for networked sensors. In *Architectural Support for Programming Languages and Operating Systems*, pages 93–104, 2000.
- [10] D. Huang, M. Mehta, D. Medhi, and L. Harn. Location-aware key management scheme for wireless sensor networks. In *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks (SASN '04)*, pages 29–42, October 2004.
- [11] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *Proceedings of 10th ACM Conference on Computer and Communications Security (CCS'03)*, pages 52–61, October 2003.
- [12] D. Liu and P. Ning. Location-based pairwise key establishments for static sensor networks. In *2003 ACM Workshop on Security in Ad Hoc and Sensor Networks (SASN '03)*, pages 72–82, October 2003.
- [13] J. Newsome, R. Shi, D. Song, and A. Perrig. The sybil attack in sensor networks: Analysis and defenses. In *Proceedings of IEEE International Conference on Information Processing in Sensor Networks (IPSN 2004)*, Apr 2004.
- [14] R. Rivest. The RC5 encryption algorithm. In *Proceedings of the 1st International Workshop on Fast Software Encryption*, volume 809, pages 86–96, 1994.
- [15] P. Traynor, R. Kumar, H. B. Saad, G. Cao, and T. L. Porta. Liger: Implementing efficient hybrid security mechanisms for heterogeneous sensor networks. In *ACM/USENIX Fourth International Conference on Mobile Systems Applications and Services (MobiSys)*, 2006.