

國立交通大學

網路工程研究所

碩士論文

降低 node-compromised 攻擊法衝擊性

之方法設計



The Design of a scheme to mitigate
node-compromised attacks

研究生：李雅婷

指導教授：葉義雄 教授

中華民國九十六年六月

降低 node-compromised 攻擊法衝擊性之方法設計

研究生：李雅婷

指導教授：葉義雄 博士

國立交通大學網路工程學系碩士班

中文摘要：

無線感測器網路(Wireless Sensor Network)是近年來學術界越來越重視的技術；它可以應用在許多方面，例如：軍事方面、安全監控、醫療管理、…等等。無線感測器網路是由眾多的感測器所組成的，屬於低耗電與低數據量的短距離無線傳輸網路。因此無線感測器網路有一些先天特性與限制，例如：(1)感測器的記憶體、能量、傳輸距離、計算能力是有限的。(2)無線感測器網路是由大量感測器所組成，因此網路監控者考量成本後，通常不會監控所有的感測器。(3)感測器通常放置在易接觸的地方，因此是暴露在不安全的環境。(4)感測器主要是以低成本考量來設計的，所以不具有防竄改的硬體保護。…等等。由於以上的特性與限制，確保無線感測器網路安全是個重要的議題。然而，低成本的硬體元件限制了感測器的計算能力甚至能源，因此公共鑰匙基礎結構與許多已經成熟的安全機制是不適用於無線感測器網路。如何提供省能源的網路安全機制成為一個很大的挑戰。而無線感

測器網路環境最難解決的其中之一安全議題就是 node-compromised attack。為了解決此問題，本論文提出一個方法可以緩和此攻擊所造成的衝擊。

本論文提出一個利用「完美雜湊族」(Perfect Hash Families)的方式來實現 (k, n) 秘密分享機制，將私密的金鑰分散給在網路上的每個節點。當節點偵測到某事件發生，必須由 k 個節點使用到 Threshold MAC 機制來共同簽章該訊息，而負責傳遞的節點可用簡單的機制來驗證該訊息是否正確、是否要繼續傳遞；Base station 收到此訊息亦可驗證該訊息的正確性。因此可以緩和 Compromise node 攻擊。然而我們提出的方法，最複雜的運算是 one-way hash function，該運算速度快與不耗能源，因此該機制適合於無線感測器網路。

關鍵字：無線感測器網路、 (k, n) 秘密分享機制、完美雜湊族、node-compromised 攻擊、Threshold MAC。

The Design of a scheme to mitigate node-compromised attacks

Student : Ya-Ting Li

Advisor : Dr. Yi-Shiung Yeh

Institute of Network Engineering National Chiao Tung University

Abstract

In recent years, academics are paying increasing attention to the technology of Wireless Sensor Network. The technology can be used in many areas, such as military, security monitoring and control, and medical management. Wireless sensor network is composed of a large number of sensors, and it belongs to low-power, low-data and low-distance wireless transmission networks. Therefore, there are some characteristics and restrictions in wireless sensor networks. For example, (1) the memory, the power, the transmitting distance, the computing capability of sensors is limited. (2) Although the wireless sensor network consists of a large quantity of sensors, the network monitoring personnel usually do not monitor all sensors as they consider the cost. (3) Many sensor systems lay aside in places where are easy to contact, and they are therefore exposed in rather insecure environments. (4) For economic reasons, sensors are lack of tamper-resistant hardware. Due to the abovementioned characteristics and restrictions, guaranteeing the security of wireless sensor networks is an important subject. In addition, the low-cost sensor has a slow-speed processor and limited energy. Therefore, the public key infrastructure and many mature security mechanisms are not suitable for wireless sensor networks. How to provide the power-saving security mechanism for wireless sensor networks becomes a difficult challenge. The

node-compromised attack is the most difficult challenge related to security in the wireless sensor network. In order to solve this problem, this thesis proposes a mechanism to mitigate the impact of the node-compromise attacks.

Specifically, this thesis proposes a mechanism using the perfect hash families to implement the (k, n) threshold secret sharing. We distribute the private keys to every node in the network. When the sensors detect an event occurred, k sensors of them can sign the message using Threshold MAC mechanism. The forwarding nodes use simple method to verify whether this message correct, and to determine whether they should continue with the transmission process. When the base station receives this message, it can also confirm the validity of this message. In our mechanism, the most complicated operation is one-way hash function, which has a fast speed and does not consume much energy. Therefore, our mechanism can mitigate the node-compromised attack, and it is suitable for wireless sensor network.

Key words: Wireless Sensor Network, (k, n) threshold secret sharing, Perfect Hash Families, node-compromised attack, Threshold MAC.

致 謝

這篇論文可以順利完成，首先我必須誠摯的感謝我的指導教授，葉義雄教授，這兩年來老師教導我很多，讓我深入了解密碼與網路安全領域的奧妙，老師對學問的專攻是我學習的典範。此外，老師在待人處世方面的態度，更是令我受益良多。

我想要特別感謝銘智、定宇學長、靜文學姊，在論文的研究與寫作的過程中，不厭其煩的指出我的缺失與探討，在我迷網時為我開啟一盞明燈。接著要感謝已經畢業的鴻祥學長，給我許多寶貴的意見與支持。還有，要感謝在美國的怡君，特地幫我修改英文寫作，改正我寫作的錯誤。此外，還要感謝實驗室的各位學長、同學與學弟妹，韓禹、龍哥、欣諭、漢民、長榮、伯昕、文浩、佳君、子強、家明，這些日子的陪伴讓我研究所的生活變的多彩多姿。

最後，要感謝我摯愛的父母，由於你們的支持與鼓勵，才有今天的我，非常謝謝你們的栽培。

謹將我這篇論文獻給所有關心我與支持我的人，謝謝。

李雅婷

中華民國九十六年六月

中文摘要:	ii	
Abstract	iv	
致 謝	vi	
Figure List	viii	
Table List	ix	
Chapter 1	Introduction	1
1.1	Background	1
1.2	Motivation	4
1.3	Organization	6
Chapter 2	Related Work	7
Chapter 3	Primitives	10
3.1	Combinatorial Object	10
3.1.1	Perfect Hash Families	10
3.1.2	Construction methods of Perfect Hash Families	15
3.2	Threshold MACs	19
Chapter 4	System Architecture	22
4.1	Concept	22
4.2	Network Assumptions	22
4.2.1	Trust Requirements	22
4.2.2	Intrusion model	23
4.3	Details and Protocols	23
4.3.1	System initialization	24
4.3.2	Deployment Phase	29
4.3.3	Signature Phase	30
4.3.4	Verification Phase	32
Chapter 5	Evaluation and Analysis	35
5.1	Evaluation	35
5.1.1	Hardware Specifications	35
5.1.2	TinyOS	37
5.1.3	Performance Evaluation	38
5.2	Security Analysis	41
Chapter 6	Conclusion and Future Work	45
Chapter 7	Reference	47
Appendix A		49

Figure List

Figure 4.3.1	Deploying sensor nodes by blocks	29
Figure 4.3.2	An event occurs.....	30
Figure 4.3.3	Sending shares to AP.....	31
Figure 4.3.4	The forwarding phase	32
Figure 5.1.1	MicaZ mote	36
Figure 5.1.2	MIB510 Programming board.....	36
Figure 5.1.3	the relation with n , q^2 , and stored data in each node	38
Figure 5.2.1	$n_c = 0$, $w = 3$, the attacker generates a bogus report with 3 bad shares.	42
Figure 5.2.2	$n_c = 1$, $w = 3$, the attacker generates a bogus report with 2 bad shares.	43
Figure 5.2.3	$n_c = 2$, $w = 3$, the attacker generates a bogus report with 1 bad share.....	43



Table List

Table 1.1.1	A model of Wireless Sensor Network	2
Table 3.1.1	PHF(4; 9, 3, 3) [9]	11
Table 3.1.2	Verification that F is a PHF(4; 9, 3, 3) [9]	12
Table 3.1.3	Construction Methods	15
Table 4.3.1	PHF (4; 9, 3, 3)	26
Table 4.3.2	Block i corresponds to PHF (4; 9, 3, 3)	27
Table 5.1.1	The relation among w, q, and the packet overhead	40
Table 5.1.2	Energy numbers for SHA-1. The numbers were averaged over inputs ranging from 64 to 1024 bytes	41





Chapter 1 Introduction

1.1 Background

With the maturity of internet technology for data communication and the development of wireless transmission techniques, users can now operate their wireless devices, such as PDAs and laptops, to connect to the internet via wireless transmission protocol. Wireless network provides users with great flexibilities and convenience. For these reasons, applications for wireless networks are becoming popular and common. While wireless networking has become more available in recent years, there are, however, still many problems associated with wireless technology, and they should be discussed deeply by scholars and development teams.

Common techniques for wireless access control include IEEE 802.11 for wireless local area networks (WLANs), IEEE 802.16e (mobile WiMAX) for Wireless Metropolitan Area Networks (WMANs) that provides long-range links and supports the speed up to tens of Mbps, and IEEE 802.15 for wireless personal area networks (WPANs) that provides short-distance and low-power links. The following section focuses on the wireless sensor network in wireless personal area networks (WPANs).

Due to the improvement of miniature manufacturing, communication technology and battery technology, small detecting devices (e.g. sensor) have the ability to sense, communicate and process data information.

Sensors not only can monitor environmental situations, such as temperature, sound, light, movement, or seismic detections cooperatively, but also can process the collected data. Furthermore, after processing the collected data, it can send these data to an aggregation point or a base station using wireless transmission. These sensors can make up

a kind of network—namely, the Wireless Sensor Network (WSN). There are many areas of application for wireless sensor networks including, for example, home automation and military.

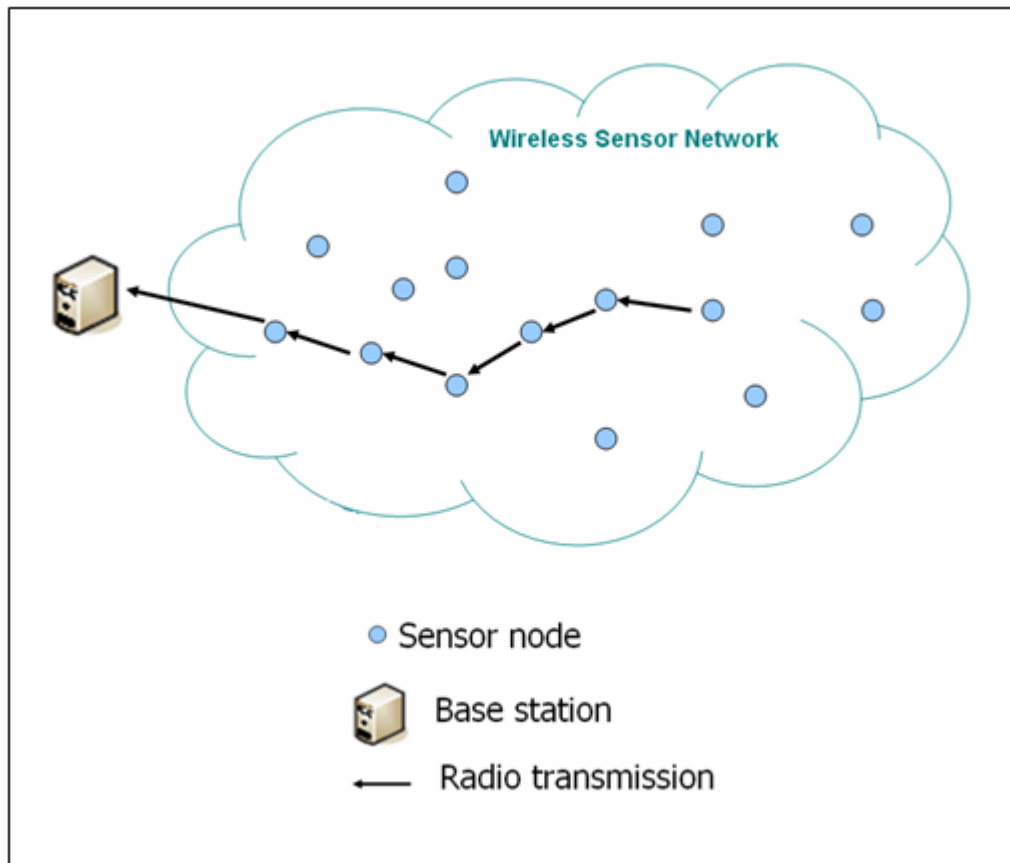
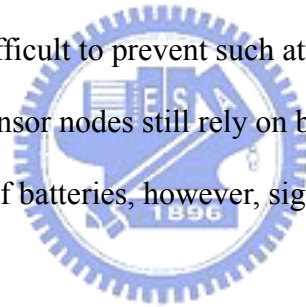


Table 1.1.1 A model of Wireless Sensor Network

In the following section, we will list characteristics and restrictions in wireless sensor network.

- Since wireless sensor network is expected to consist of hundreds or even thousands of sensor nodes, network management is undoubtedly difficult. Also, it is unrealistic and uneconomical to deploy these sensors one by one and monitor them all.

- Unlike the traditional network, many sensor systems are deployed in unattended or insecure environments such as military and forest. A sensor node is typically equipped with a radio transceiver and communicates with one another via wireless short-distance technique. For these reasons, wireless sensor networks encounter new security problems. Application service designers should, therefore, focus on issues pertinent to security in wireless sensor network.
- For economic reasons, sensor nodes are designed for low-cost purposes. Thus, they have the following characteristics: (1) resource constrained, such like low-power, low-transmission speed, small-memory, narrow-bandwidth and small-microcontroller and (2) lack of tamper-resistant hardware. The second factor may cause sensor nodes to suffer node-compromised attacks, and the first limitation makes it difficult to prevent such attacks.
- Currently, wireless sensor nodes still rely on batteries as their source of power. The limited lifetime of batteries, however, significantly impedes the usefulness of such devices.



Wireless sensor networks are vulnerable to many kinds of attacks. In addition to the traditional wireless security threats such as secret information leakage, modified data, replay attack, and denial of service, the WSN can easily face physical attacks. Specifically, the adversary may gain full control over a sensor node through direct physical access (node capture attack) to threaten the network.

The following section discusses attacks in detail. Attacks associated with wireless sensor network can be divided into outside attacks and inside attacks.

In the former case, attackers have no cryptographic keying materials to participate in network as legitimate nodes. They might just passively eavesdrop on radio transmissions

or actively inject bogus data to consume network resources. Using cryptography such as authentication and key management can secure the communication among sensors and discard packets from unauthenticated nodes.

Inside attacks, on the other hand, refer to the adversary having full control over the sensor nodes, including their cryptographic keys. There are many kinds of inside attacks—for example, wormhole attacks, Sybil attacks, identity replication attacks, and injecting bogus data into network. The cryptography cannot prevent inside or node-capture attacks by itself because legitimate nodes are unable to identify malicious nodes that carry correct cryptographic keying material. In fact, up to now, coping with compromised nodes remains to be one of the most difficult challenges on wireless sensor network security.

1.2 Motivation



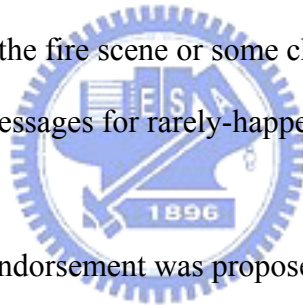
The most difficult challenge faced in developing a security mechanism for wireless sensor network is due to the fact that the characteristics of sensor render traditional security mechanisms insufficient and impractical. When we design a security mechanism for wireless sensor networks, we must, therefore, consider these two problems: sensors' life is limited, and the computing ability of sensors is deficient.

We take the Crossbow's Micaz as an example. The Micaz has the 4MHz Atmega128L microprocessor, 128Kb of program flash memory, and 512Kb of measurement (serial) flash, and uses with an AA battery. Under this environment, it is difficult to use the asymmetric cryptography. The asymmetric (also called public/private key) cryptography requires a large amount of energy to do the computation. Thus, the public-key based schemes are not suitable for the resource-limited wireless sensor network. Many security mechanisms used in conventional networks are not optimal for WSNs. For example,

neither SSL nor IPSec is suitable for wireless sensor network. Therefore, research into how to design security mechanisms specifically for the WSNs is needed.

This thesis mainly focuses on the node-compromised attack, which is the most difficult challenge to security in wireless sensor network.

Z Yanchao, L Wei, L Wenjing, F Yuguang in [1] and K. Bıçakcı, C. Gamage, B. Crispo, A. Tanenbaum in [7] proposed the schemes to mitigate node-compromised attack respectively. The method suggested in [7] is effective to minimize the influence of the node-capture attack, and the operation is simple. However, the sensors in this scheme can only be used once. For this reason, the applicability of this method is limited and impractical. It only suits applications where sensors lose their functionality after the first sensing. For example, when sensors can be used only one-time because of external conditions (e.g., fire sensors in the fire scene or some chemical detectors) or when the sensor network carries alarm messages for rarely-happened events (e.g., nuclear attacks), this method may be used.



The concept of threshold-endorsement was proposed in [1] to mitigate node-compromised attacks. Their mechanisms are based on the pairing technique on Elliptic Curve Cryptography to design the signature and verification mechanisms. Although this method achieves high security, the Elliptic Curve Cryptography is exceeding complex, too slow, and power-consuming. It is, therefore, doubtful whether this scheme can be used in current WSNs.

In this thesis, we propose a compromise-tolerant security mechanism. Specifically, we use the perfect hash family (PHF) [9] approach to perform the key distribution in our scheme. Furthermore, in order to implement the threshold-endorsement, we use the threshold MAC. Finally, an important feature of our proposed mechanism is that even the most complicated operation requires only one-way hash function. Since the computation cost of the one-way hash function is relatively minimal, it is suitable for current wireless

sensor networks.

1.3 Organization

This thesis is organized as follows. In the next chapter, we will discuss two papers which proposed to mitigate node-compromised attacks [1] [7], compare them, and point out their deficiency and restriction. Then chapter 3 will introduce the primitives. It includes backgrounds on the PHF and the threshold MAC. In chapter 4, we will focus on the specific system architecture and describe the detailed protocol of our scheme. We will then present the analysis of our scheme in Chapter 5. Finally, chapter 6 will provide conclusions and suggestions for some future research directions based on this thesis.



Chapter 2 Related Work

There has been much research focused on securing wireless sensor network. One of the most difficult issues related to security in the wireless sensor network environment is how to minimize the impact of node-compromised problems.

In [7], K. Bıçakcı, C. Gamage, B. Crispo, and A. Tanenbaum proposed one-time sensors mechanism to mitigate node-capture attacks. Below is the description of their mechanism.

First, the base station preloads every sensor node with a unique ID value and a single cryptographic token. All sensor nodes are also preloaded with a sufficient amount of verification data to enable them to check the validity of tokens received. In every node there is also a memory space reserved to store the revocation list, which is initially empty.

Then the operation is performed as follows :

1. Based on its local routing information, when an one-time sensor senses the target event, such as a fire, it sends an alarm message to one node or multiple nodes.

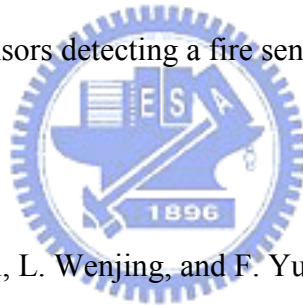
Through this routing path, the alarm message can be sent to the base station. The alarm message is basically consists of the ID and cryptographic token of the sensing node.

2. When a node receives an alarm message, it would first check if it has already received a valid alarm message from the same node by comparing the ID value with the entries in its revocation list. If yes, it had received the other alarm message from the node which generated the alarm message. If not, it then ensures that the cryptographic token it received is indeed valid. Only if the cryptographic token is verified correctly, then the following two actions are taken. First, the alarm message is forwarded to the node(s) on the way to the base station. Second, the ID of the sender is added to the

revocation list for future reference.

3. The second step repeats itself with other nodes until the alarm message is received by the base station. The base station verifies that the alarm message is valid and has not been received before. Based on the threshold value and the number of messages it received before, the base station either decides to notify an alarm or waits for additional alarm messages.

From the above description, we know that when a sensor node detects an event and sends the alarm message out, the forwarding nodes can deliver this alarm message at most once. It means that each sensor can be used only one time. Therefore, the applicability of this method has great restriction. It only suits the applications where the sensors can be used only one-time, such as sensors detecting a fire sense, nuclear attacks, and chemical outbreak.



In [1], Z. Yanchao, L. Wei, L. Wenjing, and F. Yuguang developed a location-based threshold-endorsement scheme to thwart the bogus data that attackers use the captured node to inject. This mechanism is based on the Tate Pairing technique on Elliptic Curve Cryptography to provide authentication, key establishment, endorsement and verification. It can prevent malicious nodes from joining the wireless sensor network and diminish the node-compromised problem efficiently. Since sensors need to compute the Elliptic Curve Cryptography operations when they sign or verify messages, it is comparatively too power-consuming.

Z. Yanchao, L. Wei, L. Wenjing, and F. Yuguang in [1] thought that the Tate Pairing technique can be workable in wireless sensor networks because K. Bıçakcı, C. Gamage, B. Crispo, and A. Tanenbaum in [8] computed the Tate Pairing with the similar parameters as theirs. Also, these researchers in [8] quoted that the execution cost of the Tate pairing

operation was only 62.04ms and 25.5mJ.

However, G. Bertoni, L. Chen, P. Fragneto, K. Harrison, and G. Pelosi in [8] implemented the Tate Pairing operation in 32-bit ST22 smartcard microprocessor at 33MHz. The sensors presently do not have the chip of the Tate pairing operation. If this scheme implements on current WSNs, it is unclear whether the low-microprocessor sensor nodes can sign messages together quickly, verify them fast, and send them to the base station immediately or not. If the base station cannot be notified immediately without any delay, the monitor system is, in fact, useless. Additionally, sensor nodes rely on batteries as their source of power. The power-consuming Elliptic Curve operation is easy to make sensors have no power, which, in turn, would lead sensor nodes become ineffective. For these reasons, we doubt if this scheme in [1] is suitable for use in the current WSN.

Some researchers in [2]-[6] discussed the feasibility of using the public-key cryptography architecture, such as RSA or Elliptic Curve Cryptography, in WSNs. The investigators in [5] thought that if the public-key cryptography architecture should be feasible in WSNs, then the sensor nodes can embed the chip with the operation of public-key cryptography. However, sensor nodes embed the chip of tamper resistant hardware better than the chip of these public-key operations. When attackers invade the sensor nodes with tamper resistant hardware, these sensor nodes can prevent their data from being obtained. Thus, we do not need to consider node-compromised attacks, and simple secure mechanisms can guarantee the security in networks.

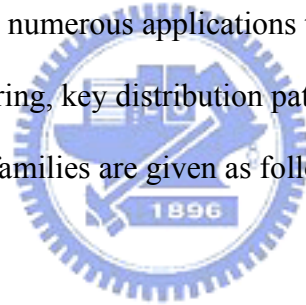
Chapter 3 Primitives

3.1 Combinatorial Object

In this section, we will describe the combinatorial objects, affine plane and perfect hash families which are used in the thesis.

3.1.1 Perfect Hash Families

Perfect hash families are basic combinatorial structures. They have applications to operating system, language translation system, file managers, and compiler constructions. More recently, they have found numerous applications to cryptography, such as broadcast encryption schemes, secret sharing, key distribution patterns, and cover-free families, etc. The definition of perfect hash families are given as follows [9]:



Definition 3.1.1 [9]

An (n, m, w) - perfect hash family is a finite set of hash function F such that

$$h: A \rightarrow B$$

For each $h \in F$, where $|A| = n$ and $|B| = m$ (where $n, m > 0$), with the property that for any $X \subseteq A$ such that $|X| = w$, there exists at least one $h \in F$ such that $h|_X$ is injective.

□

We use the notation $\text{PHF}(N; n, m, w)$ to denote an (n, m, w) - perfect hash family with $|F| = N$. We can think of a $\text{PHF}(N; n, m, w)$ as an $N \times n$ array of m symbols, where each row of the array corresponds to one of the functions in the family. This array has the

property that, for any subset of w columns, there exists at least one row such that the entries in the w given columns of that row are distinct.

Let N be the minimum number of functions such that a $\text{PHF}(F; A, B, w)$ would exist. That is, $N = \min \{|F|\}$ is the optimal solution [9].

Below is a simple example of a perfect hash family – $\text{PHF}(4; 9, 3, 3)$.

Example 3.1.1.

We have a $\text{PHF}(4; 9, 3, 3)$. Consider the matrix:

$$M = \begin{bmatrix} 1 & 1 & 1 & 2 & 2 & 2 & 3 & 3 & 3 \\ 1 & 2 & 3 & 1 & 2 & 3 & 1 & 2 & 3 \\ 1 & 2 & 3 & 3 & 1 & 2 & 2 & 3 & 1 \\ 1 & 2 & 3 & 2 & 3 & 1 & 3 & 1 & 2 \end{bmatrix}$$

Let $A = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ and $B = \{1, 2, 3\}$. Hence $|A| = 9, |B| = 3$.

Define $f_i(x) =$ the value of entry (i, x) of M

Let F be a set of hash functions $f_i(x), i = 1, 2, 3, 4$, as shown in Table 3.1.1.

X	1	2	3	4	5	6	7	8	9
$f_1(x)$	1	1	1	2	2	2	3	3	3
$f_2(x)$	1	2	3	1	2	3	1	2	3
$f_3(x)$	1	2	3	3	1	2	2	3	1
$f_4(x)$	1	2	3	2	3	1	3	1	2

Table 3.1.1 PHF(4; 9, 3, 3) [9]

From these four functions, we can see that for any subset of $X \subseteq A$ with $|X| = 3$, we have at least one function of F that separates X . The verification of that F is a $\text{PHF}(4; 9, 3, 3)$, which is demonstrated in Table 3.1.2.

X	<i>i</i>	X	<i>i</i>	X	<i>i</i>	X	<i>i</i>	X	<i>i</i>	X	<i>i</i>
123	2,3,4	124	3	125	4	126	2	127	4	128	3
129	2	134	4	135	2	136	3	137	3	138	2
139	4	145	4	146	3	147	1,3,4	148	1	149	1
156	2	157	1	158	1	159	1,2,4	167	1	168	1,2,3
169	1	175	3	179	4	189	2	234	2	235	3
236	4	237	2	238	4	239	3	245	3	246	4
247	1	248	1	249	1,2,3	256	1	257	1	258	1,3,4
259	1	267	1,2,4	268	1	269	1	278	4	279	2
289	3	345	2	346	4	347	1	348	1,2,4	349	1
356	3	357	1,2,3	358	2	359	1	367	1	368	1
369	1,3,4	378	2	379	3	389	4	456	2,3,4	457	3
458	4	459	2	467	4	468	2	469	3	478	4
479	3	489	2	567	2	568	3	569	4	578	3
579	2	589	4	678	2	679	4	689	3	789	2,3,4

Table 3.1.2 Verification that F is a PHF(4; 9, 3, 3) [9]

Note: X is a 3-subset of A and i represents a function $f_i \in F$. For convenience, we write a subset X in the form of 123, instead of $\{1, 2, 3\}$.

Two propositions of perfect hash families are discussed in [9] and [10]. Proposition 3.1.3 is the partition according to perfect hash families. Proposition 3.1.4 describes a relationship between a perfect hash family and an array.

Definition 3.1.2 w -partition of A

$w \mid |A|$, Π is a partition of A . $P_i \in \Pi$, $\{P_1, P_2, \dots, P_{\frac{|A|}{w}}\}$, $|P_i| = w$.

The order of each subset is w . □

Note: A set $X \subseteq A$ is separated by a partition π of A if the elements of X are in distinct part of π .

Proposition 3.1.3 [9][10]

Suppose that Π is a set of partitions of a set A with $|\Pi| = N$. For all sets $X \subseteq A$ with $|X| = w$, X is separated by at least one $\pi \in \Pi$. Then it exists a $\text{PHF}(N; n, m, w)$.
Conversely, a $\text{PHF}(N; n, m, w)$ gives rise to such w -partition set Π of A . \square

The proof for Proposition 3.1.3 is included in Appendix A. The following is an example of Proposition 3.1.3.

Example 3.1.2.

Let $A = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Applying the process described in Proposition 3.1.3 to the $\text{PHF}(4; 9, 3, 3)$ constructed in Example 3.1.2. We can then get the following results:

$$\begin{aligned} \pi_1 &= \{\{1, 2, 3\}, \{4, 5, 6\}, \{7, 8, 9\}\}, & \pi_2 &= \{\{1, 4, 7\}, \{2, 5, 8\}, \{3, 6, 9\}\} \\ \pi_3 &= \{\{1, 5, 9\}, \{2, 6, 7\}, \{3, 4, 8\}\}, & \pi_4 &= \{\{1, 6, 8\}, \{2, 4, 9\}, \{3, 5, 7\}\} \end{aligned}$$

Thus, $\Pi = \{\pi_1, \pi_2, \pi_3, \pi_4\}$ is the most desired set of partitions of A .

Conversely, we can find a function family $F = \{f_i : 1 \leq i \leq 4\}$, such that $f_i(x)$ is regarded as π_i and for each $x \in A$, labeling the part for each partition π_i according to the given order. \square

Proposition 3.1.4 [9][10]

Suppose that there exists a $\text{PHF}(N; n, m, w)$. Then there exists an array M , where its size is $N \times n$ and which has entries in a set B of size m , such that for any subset X of columns of M with $|X| = w$, there is at least one row of M that separates the subset X of columns of M .

Conversely, such an array gives rise to a $\text{PHF}(N; n, m, w)$. \square

The proof for Proposition 3.1.4 is also included in Appendix A. A simple example of Proposition 3.1.4 is given below.

Example 3.1.3.

Refer to Table 3.1.1 PHF (4; 9, 3, 3) provided in Example 3.1.1.

We randomly take a subset $X = \{1, 2, 3\}$, and there are $f_2(x)$, $f_3(x)$, $f_4(x)$ rows to separate the subset X . Then, we take another subset $X = \{1, 4, 8\}$, and there is $f_1(x)$ row to separate the subset X .

Next section provides a summary of construction methods of perfect hash families proposed in other studies.



3.1.2 Construction methods of Perfect Hash

Families

There are many kinds of method in constructing perfect hash families from combinational structure and algebraic structure. Table 3.1.3 lists the approaches included in both combinational and algebra structures.

Combinatorial Structures	Algebra Structures
Design Theory	Special Global Function Field
Error-Correcting Codes	Algebraic Curves
Recursive Constructions	
Orthogonal arrays	

Table 3.1.3 Construction Methods

In the thesis, we use the method to construct perfect hash families based on the design theory method. The PHF is constructed by the affine plane and resolved BIBD (balanced incomplete block design). First, we will provide some definitions of an affine plane and a resolvable BIBD.

An affine plane is a PBD (P, B) . Therefore, we will state PBD (P, B) , the affine plane and the corresponding properties in order.

Definition 3.1.3 [11]

A **pairwise balanced design** (PBD) is an ordered pair (S, B) , where S is a finite set of symbols, and B is a collection of subsets of S called blocks, such that each pair of distinct elements of S occurs together in exactly one block of B .

Following we will call the blocks of a PBD as lines. If several points belong to the

same line, we will say that they are collinear; if two lines fail to intersect we will state that they are parallel.

Definition 3.1.4 [11]

An affine plane is a PBD (P, B) which satisfies the following properties :

- (1) Given two different points p_1 and p_2 , there is exactly one line of B containing p_1 and p_2 points.
- (2) P contains at least one subset of 4 points, and no 3 of which are collinear.
- (3) Given a line l and a point p not on l , there is exactly one line of B containing p which is parallel to l .

Example 3.1.4.

An affine plane.

$$P = \{1, 2, 3, 4\}$$

$$B = \{ \{1, 2\} \quad \{1, 3\} \quad \{1, 4\} \\ \{3, 4\} \quad \{2, 4\} \quad \{2, 3\} \}$$



□

In an affine plane (P, B), the number of points in each block is called the order of the affine plane.

Definition 3.1.5

A (v, b, r, k, λ) balanced incomplete block design (BIBD) is a pair (X, A) , where X is a non-empty set of points; A is a collection of k -element subsets (blocks) of set X . Let v, k, λ be positive integers such that $v \geq k \geq 2$. Following properties are satisfied:

- 1. $|X| = v$,
- 2. Every point occurs in r blocks, and
- 3. Every pair of points occurs in exactly λ blocks.

□

For simplicity, we will write blocks in the form of abc , rather than $\{a, b, c\}$, in the following examples.

Example 3.1.5.

A $(10, 15, 6, 4, 2)$ – BIBD.

$X = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, and

$A = \{0123, 0145, 0246, 0378, 0579, 0689, 1278, 1369, 1479, 1568, 2359, 2489, 2567, 3458, 3467\}$. □

Theorem 3.1.6

A (v, b, r, k, λ) – BIBD follows from elementary counting that $vr = bk$ and $\lambda(v - 1) = r(k - 1)$. □



The proof of Theorem 3.1.6 is included in Appendix A.

A parallel class in (X, A) is a set of blocks that forms a partition of the point set X . A BIBD is resolvable if A can be partitioned into r parallel classes, and each of which consists of v/k disjoint blocks. Obviously, a BIBD can have a parallel class only if $v \equiv 0 \pmod k$.

Example 3.1.6. A resolvable $(6, 15, 5, 2, 1)$ – BIBD.

Let $X = \{0, 1, 2, 3, 4, 5\}$, and $r = 5$. Hence there are 5 parallel classes, and each consists of 3 blocks.

So parallel classes = $\{01, 25, 34\}$,

{02, 13, 45},

{03, 24, 15},

{04, 35, 12},

{05, 14, 23}

□

It is well-known that an affine plane of order q is an $(q^2, q(q+1), q+1, q, 1) - \text{BIBD}$. It is also a resolvable BIBD. Thus, the following theorem can be derived: For any prime power q , there exists an affine plane of order q . That is, there exists a $(q^2, q(q+1), q+1, q, 1) - \text{BIBD}$.

Theorem 3.1.7

If there exists a resolvable $(v, b, r, k, \lambda) - \text{BIBD}$ with $r > \lambda \binom{w}{2}$, then there exists a PHF($r; v, v/k, w$).



□

Based on this theory and the above description, we then can derive the following corollary.

Corollary 3.1.8 [11]

Let w be an integer such that $w \geq 2$. Suppose q is a prime power and $q+1 > \binom{w}{2}$. Then there exists a PHF $(q+1; q^2, q, w)$.

□

Therefore, we can use an affine plane to construct a PHF.

In this thesis, we construct the perfect hash families according to Corollary 3.1.8. The detail of our construction is described in the next chapter. By observations, we find that

formats of the BIBD and the PHF, which are constructed from an affine plane, are determined by only one parameter – prime power q . Thus, we give a special name for these kinds of BIBDs and PHFs – namely, $(q, 1)$ – BIBD and (q, w) – PHF.

3.2 Threshold MACs

In this section, we introduce threshold MACs that combines a secure MAC and a combinational object, called a *cover-free-family* (CFF). This mechanism is proposed by K. M. Martin, J. Pieprzyk, R. S. Naini, H. Wang, and P. R. Wild [13].

Definition 3.2.1 [14]

A set system (X, \mathcal{B}) with $X = \{x_1, \dots, x_n\}$ and $\mathcal{B} = \{B_i \subseteq X \mid i = 1, \dots, n\}$ is called an (n, v, t) -cover free-family (or (n, v, t) -CFF for short) if for any subset $\Delta \subseteq \{1, \dots, n\}$ with $|\Delta| = t$ and any $i \in \Delta$,

$$\left| B_i \setminus \bigcup_{\substack{j \in \Delta \\ j \neq i}} B_j \right| \geq 1.$$

The elements of X are called *points* and elements of \mathcal{B} are called *blocks*. In other words, in a (n, v, t) -CFF (X, \mathcal{B}) the union of any $t-1$ blocks in \mathcal{B} cannot cover any other remaining one. Cover-free families were introduced by P. Erdős, P. Frankl, and Z. Füredi [14].

Threshold CFF MAC. [14]

Suppose (X, \mathcal{B}) is an (n, v, t) -CFF and $F : \{0,1\}^k \times \{0,1\}^L \rightarrow \{0,1\}^l$ is a secure MAC, we construct a (t, n) threshold MAC $M \left[\begin{smallmatrix} n \\ t \end{smallmatrix} \right] = (KGEN, MAC, VF)$ as follows:

1. **KGEN** : The receiver randomly chooses v keys in $\{0,1\}^k$, $X = \{k_1, \dots, k_v\}$, and securely sends a subset $B_i \subseteq X$ of keys to sender P_i for $1 \leq i \leq n$, such that (X, B) is a (n, v, t) -CFE, where $B = \{B_1, \dots, B_n\}$.
2. **MAC** : Suppose t senders $A = \{P_{i_1}, \dots, P_{i_t}\}$ want to authenticate message m . The senders in A first compute the set of indices for their keys, that is, they compute $I = \{j \mid k_j \in B_{i_1} \cup \dots \cup B_{i_t}\}$. Then the senders in A jointly compute $\sigma = F_I^{||l}(m) = \bigoplus_{j \in I} F_{k_j}(m)$ and send (m, σ, I) to the receiver.
3. **VF**: Upon receiving a message (m, σ, I) , the receiver recomputes $F_I^{||l}(m)$, using the keys $\{k_j \mid j \in I\}$, to verify the authenticity of that message.

Definition 3.2.2 [14]

Let X_1, \dots, X_l be l disjoint subsets of a set X such that $X = \bigcup_{j=1}^l X_j$. Let

$B = \{B_i, 1 \leq i \leq n\}$ be a family of subsets of X . We call $(X_1, X_2, \dots, X_l; B)$ an (n, t) generalized cumulative array (*GCA*) if the following conditions are satisfied :

1. For any t blocks B_{i_1}, \dots, B_{i_t} in B , there exists an j such that $X_j \subseteq \bigcup_{s=1}^t B_{i_s}$.
2. For any $t-1$ blocks $B_{i_1}, \dots, B_{i_{t-1}}$, and for any j , $1 \leq j \leq l$, $X_j \not\subseteq \bigcup_{s=1}^{t-1} B_{i_s}$.

If $|X_1| = \dots = |X_l| = \alpha$ for some integer α , we say $(X_1, X_2, \dots, X_l; B)$ is an (n, α, l, t) -*GCA*

It is easy to see that a *GCA* is a *CFE*. Now we slightly modify the previous threshold *CFE MAC* scheme as follows, if the underlying *CFE* is a *GCA*.

Threshold GCA MAC. [14]

Let $(X_1, X_2, \dots, X_l; B)$ is an (n, α, l, t) -*GCA* and $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^l$ be a MAC. We construct a threshold MAC, called threshold *GCA MAC*, as follows.

1. **KG**EN : The receiver randomly chooses a set of αl keys from $\{0,1\}^k$,

$X = \{k_1, \dots, k_{\alpha l}\}$, and partitions X into l disjoint subsets X_1, \dots, X_l with $|X_i| = \alpha$ for all i . The receiver then securely gives to sender P_i a subset of keys $B_i \subseteq X$ in such a way that $(X_1, X_2, \dots, X_l; B)$ is an (n, α, l, t) -GCA, where $B = \{B_i, 1 \leq i \leq n\}$

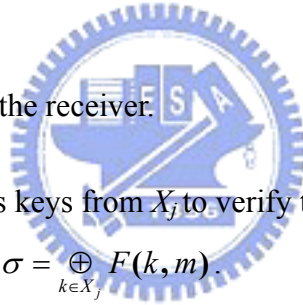
2. **MAC**: Suppose a t -subset of P , $A = \{P_{i_1}, \dots, P_{i_t}\}$, wants to authenticate a message m . For each index j , $1 \leq j \leq l$, they determine the set I_j of indices of their keys

X_i and put J equal to the smallest index j such that $\{k_i \mid i \in I_j\} = X_j$. Note that since $(X_1, X_2, \dots, X_l; B)$ is a GCA, such J exists. They then compute

$$\sigma = \bigoplus_{k \in X_j} F(k, m),$$

and send (m, σ, J) to the receiver.

3. **VF** : The receiver uses keys from X_j to verify the authenticity of (m, σ) by checking the equality $\sigma = \bigoplus_{k \in X_j} F(k, m)$.



Chapter 4 System Architecture

4.1 Concept

In this chapter, we will introduce a (w, q^2) threshold signature scheme with shared verification. This scheme can mitigate the node- compromised attacks, and it also adapts to the wireless sensor network.

We utilize perfect hash families' properties to design the key distribution protocol in this scheme. Furthermore, we use threshold MAC mechanism to implement the threshold signature and the verification. Below we will introduce our network system and the operational procedure for our proposed mechanism in details.

4.2 Network Assumptions

4.2.1 Trust Requirements

Since a base station serves as a gateway between a sensor network and the outside world, if a base station is compromised, then it may cause the entire network to crash. For this reason, we operate under the assumption that base stations, indeed, are credible entities.

All sensor nodes may be deployed to unattended or insecure environments, including aggregation points. Thus the attackers may try to dispose malicious aggregation points, try to turn compromised nodes into aggregation points, or directly attempt to capture aggregation points. Aggregation points are, therefore, regarded as incredible entities.

4.2.2 Intrusion model

In this section, we briefly discuss the type of intrusion models that our scheme can resist. There are many kinds of attacks in wireless sensor networks. Some people consider attacks as inside attacks and outside attacks. In outside attacks, the attack nodes do not have authorized information to participate in the sensor network as legitimate nodes. They might just passively eavesdrop on radio transmissions or actively inject bogus data to consume network resources.

Different from outside attacks, inside attacks refer to the adversary having full control over the sensor nodes, including their cryptographic keys. With node compromised, an adversary can perform an inside attack. In contrast to disabled node, compromised nodes activity seeks to disrupt the network. A compromised node may a subverted sensor node or a more powerful device, like laptop, with more computational power, memory, and powerful radio. It may be running some malicious code and seek to steal secrets from the sensor network or inject a lot of bogus reports to the sensor network. Then, our scheme aims to tolerate the node-compromised attack, which is fatal attack in wireless sensor networks.

Finally, we assume that the intrusion attacker has more power to tamper, eavesdrop, or even drop any information he obtains.

4.3 Details and Protocols

This section is divided based on the following four stages. In 4.3.1 initialization phase, the first stage, we define the network environments, by, for example, defining some variables. Then, the base station divides key shares among sensor nodes by utilizing the

PHF characteristics. Finally, these key shares and hash values are preloaded to sensor nodes prior to deploying these nodes. In 4.3.2 deployment phase, the second stage, we describe the method of deploying these sensor nodes.

In 4.3.3 signature phase, the third stage, sensor nodes perform in a cooperative monitoring environment and report the sensed events to the base station. A base station is a data collection center within the entire sensor network and reports the data to an end user. It also has a sufficient amount of powerful processing capabilities and resources.

We will depict how these detecting nodes use the threshold MAC concept to endorse messages when an event has happened. In 4.3.4 verification phase, the fourth stage, all sensor nodes can act as forwarding nodes. We will describe the forwarding nodes and illustrate how the base station verifies signature messages.

4.3.1 System initialization



In our scheme, we assume that the network system consists of many blocks. Therefore, we divide sensor nodes into many blocks and assign each block an index number, starting at one and going up from there. Each sensor node is marked with two index numbers. One represents the block that the sensor node belongs to; the other is the serial number of that sensor node. For example, the network system is comprised of four blocks, and each block has four sensor nodes. We assign each of the four blocks with index B_1 , B_2 , B_3 , and B_4 respectively. Furthermore, we mark the index of sensor nodes in the B_1 block as $n_{1,1}$, $n_{1,2}$, $n_{1,3}$, $n_{1,4}$, those in the B_2 block as $n_{2,1}$, $n_{2,2}$, $n_{2,3}$, $n_{2,4}$, and so on and so forth.

From section 3.1.2, we know that for any prime power q , there exists an affine plane of order q . In addition, an affine plane of order q can construct a $(q^2, q(q+1), q+1, q, 1)$ — BIBD. Let w be an integer such that $w \geq 2$. Suppose q is a prime power and

$q + 1 > \binom{w}{2}$. Then, based on Corollary 3.1.8, there exists a PHF $(q+1; q^2, q, w)$. When we want to initialize the system, we therefore first consider the prime power q , which is related to the block size. Then, another important parameter is the security parameter w , which means how many sensor nodes in a block will sign the message when an event happens. After both variables are determined, we get a suitable prime power q such that $q + 1 > \binom{w}{2}$. Then, we use this prime power q to create a PHF $(q+1; q^2, q, w)$.

Below, we focus on the design of a block. In our scheme, every parameter in the PHF $(q+1; q^2, q, w)$ is explained as follows : q^2 means the number of sensor nodes in each block. $(q+1)$ indicates that the PHF has $(q+1)$ key sets, and there are q key shares per set. Besides, it also represents the number of key shares that each sensor node should hold. These key shares belong to different key sets separately. Finally, w implies the number of sensor nodes needed to endorse the message when an event happens. If less than w sensor nodes sign it, the message would be an invalid one.

Since each block has different key shares, in order to clearly identify them, we will mark all key shares. Each key share has three index numbers; the first index indicates that the key share belongs to which block, the second index represents the key share belongs to which key set, and the third index means that the key share belongs to which key share.

Next, our scheme uses one-way hash functions. The hash function takes key shares as input and produces a fixed-length hash value as output. Each hash value also has three index numbers, which is the same as the key share. For instance, we use a one-way hash function to calculate the hash value $h_{i,j,k}$ from the key share $K_{i,j,k}$. Finally, we use the PHF to distribute key shares and hash values to sensor nodes in each block.

Take the PHF $(4; 9, 3, 3)$ in Table 4.3.1 that we mentioned above as an example. From this PHF, each block has nine sensor nodes. Then, we randomly generate three key shares for each key set and there are four key sets for each block. We, thus, create a total of 12

(equal 3×4) key shares for each block. Each node would have four key shares from different key sets. According to the PHF, we distribute key shares to the nine sensor nodes.

Table 4.3.1 is a simple example.

x	1	2	3	4	5	6	7	8	9
$f_1(x)$	1	1	1	2	2	2	3	3	3
$f_2(x)$	1	2	3	1	2	3	1	2	3
$f_3(x)$	1	2	3	3	1	2	2	3	1
$f_4(x)$	1	2	3	2	3	1	3	1	2


Table 4.3.1 PHF (4; 9, 3, 3)

In the B_1 block, we assume the four key sets are $S_{1,1}$, $S_{1,2}$, $S_{1,3}$, and $S_{1,4}$. Furthermore, we mark the key shares in $S_{1,1}$ as $K_{1,1,1}$, $K_{1,1,2}$, $K_{1,1,3}$, those in $S_{1,2}$ as $K_{1,2,1}$, $K_{1,2,2}$, $K_{1,2,3}$, those in $S_{1,3}$ as $K_{1,3,1}$, $K_{1,3,2}$, $K_{1,3,3}$, and finally those in $S_{1,4}$ as $K_{1,4,1}$, $K_{1,4,2}$, and $K_{1,4,3}$. In the B_2 block, we assume another four key sets as $S_{2,1}$, $S_{2,2}$, $S_{2,3}$, and $S_{2,4}$. Similarly, we also mark the key shares in $S_{2,1}$ as $K_{2,1,1}$, $K_{2,1,2}$, $K_{2,1,3}$, those in $S_{2,2}$ as $K_{2,2,1}$, $K_{2,2,2}$, $K_{2,2,3}$, those in $S_{2,3}$ as $K_{2,3,1}$, $K_{2,3,2}$, $K_{2,3,3}$, and those in $S_{2,4}$ as $K_{2,4,1}$, $K_{2,4,2}$, and $K_{2,4,3}$. In other blocks, we use the same method to label the key sets and key shares.

Next, we describe how to distribute these key shares to sensor nodes. In B_1 block, supposing the distribution of the key shares in $S_{1,1}$ corresponds to $f_1(x)$, and those in $S_{1,2}$, $S_{1,3}$ and $S_{1,4}$ are based on $f_2(x)$, $f_3(x)$ and $f_4(x)$ respectively. Also, in B_2 block, the key shares in $S_{2,1}$, $S_{2,2}$, $S_{2,3}$ and $S_{2,4}$ correspond to $f_1(x)$, $f_2(x)$, $f_3(x)$ and $f_4(x)$ and are orderly to be distributed. In other blocks, we use the same way to perform the distribution of the key shares. Then, the value of X represents the number of sensor nodes. In other words, it means that $n_{i,1}$ implies 'x = 1' and $n_{i,2}$ implies 'x = 2' in the B_i block ($i = 1, 2, \dots$). Table

4.3.1 shows that for the polynomial function $f_l(x)$ (corresponds to key set $S_{i,l}$), $X = 1$ (corresponds to $n_{i,1}$ sensor node), $X = 2$ (corresponds to $n_{i,2}$ sensor node), and $X = 3$ (corresponds to $n_{i,3}$ sensor node) map to the same number — that is, 1 (corresponds to $K_{i,1,1}$ key share). Consequently, these three sensor nodes have the same key share $K_{i,1,1}$ from $S_{i,1}$. For the same reason, sensor nodes $n_{i,4}$, $n_{i,5}$ and $n_{i,6}$ map to the same number of the key set $S_{i,1}$, which is 2, so they get the same key share $K_{i,1,2}$ from $S_{i,1}$. Sensor nodes $n_{i,7}$, $n_{i,8}$ and $n_{i,9}$ have the same key share $K_{i,1,3}$ from $S_{i,1}$, because these sensor nodes have the same number of $S_{i,1}$, which is 3.

Table 4.3.2 describes the relationship between each block and the PHF. We can clearly understand the distribution of key shares. In B_i block, node $n_{i,1}$ has four key shares, $K_{i,1,1}$, $K_{i,2,1}$, $K_{i,3,1}$, and $K_{i,4,1}$, and node $n_{i,6}$ has four key shares, $K_{i,1,2}$, $K_{i,2,3}$, $K_{i,3,2}$, and $K_{i,4,1}$.



	$n_{i,1}$	$n_{i,2}$	$n_{i,3}$	$n_{i,4}$	$n_{i,5}$	$n_{i,6}$	$n_{i,7}$	$n_{i,8}$	$n_{i,9}$
$S_{i,1}$	$K_{i,1,1}$	$K_{i,1,1}$	$K_{i,1,1}$	$K_{i,1,2}$	$K_{i,1,2}$	$K_{i,1,2}$	$K_{i,1,3}$	$K_{i,1,3}$	$K_{i,1,3}$
$S_{i,2}$	$K_{i,2,1}$	$K_{i,2,2}$	$K_{i,2,3}$	$K_{i,2,1}$	$K_{i,2,2}$	$K_{i,2,3}$	$K_{i,2,1}$	$K_{i,2,2}$	$K_{i,2,3}$
$S_{i,3}$	$K_{i,3,1}$	$K_{i,3,2}$	$K_{i,3,3}$	$K_{i,3,3}$	$K_{i,3,1}$	$K_{i,3,2}$	$K_{i,3,2}$	$K_{i,3,3}$	$K_{i,3,1}$
$S_{i,4}$	$K_{i,4,1}$	$K_{i,4,2}$	$K_{i,4,3}$	$K_{i,4,2}$	$K_{i,4,3}$	$K_{i,4,1}$	$K_{i,4,3}$	$K_{i,4,1}$	$K_{i,4,2}$

Table 4.3.2 Block i corresponds to PHF (4; 9, 3, 3)

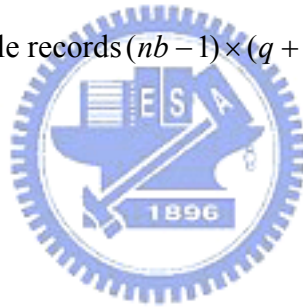
Next, we focus on the distribution of hash values. An one-way hash function takes an input $K_{i,j,k}$ and returns a fixed-size string, which is called the hash value $h_{i,j,k}$ (that is, $h_{i,j,k} = H(K_{i,j,k})$). The hash values that correspond to the key shares in certain block will be stored in other blocks. With respect to the distribution of these hash values, it is related with the PHF; The hash values generated for the key shares of sensor node $n_{i,p}$ in B_i block will be distributed to sensor node $n_{j,p}$ of B_j block ($i \neq j$). Following up with the aforementioned example, we assume that there are three blocks of B_1, B_2, B_3 in the network. In B_1 block,

node $n_{1,1}$ has key shares of $K_{1,1,1}$, $K_{1,2,1}$, $K_{1,3,1}$, and $K_{1,4,1}$. Then, the hash values of $h_{1,1,1}$, $h_{1,2,1}$, $h_{1,3,1}$, and $h_{1,4,1}$ are held by node $n_{2,1}$ of B_2 block and node $n_{3,1}$ of B_3 block. In B_2 block, node $n_{2,1}$ has key shares of $K_{2,1,1}$, $K_{2,2,1}$, $K_{2,3,1}$, and $K_{2,4,1}$. The hash values of $h_{2,1,1}$, $h_{2,2,1}$, $h_{2,3,1}$, and $h_{2,4,1}$ are distributed to node $n_{1,1}$ of B_1 block and node $n_{3,1}$ of B_3 block. In B_3 block, node $n_{3,1}$ has key shares of $K_{3,1,1}$, $K_{3,2,1}$, $K_{3,3,1}$, and $K_{3,4,1}$. The hash values of $h_{3,1,1}$, $h_{3,2,1}$, $h_{3,3,1}$, and $h_{3,4,1}$ are distributed to node $n_{1,1}$ of B_1 block and node $n_{2,1}$ of B_2 block.

Other hash values are distributed by the same method.

Finally, each node contains the following information (Assuming the network has nb blocks):

1. Block number, node ID.
2. The key table is stored in $(q+1)$ key shares.
3. The hash value table records $(nb - 1) \times (q + 1)$ hash values.



4.3.2 Deployment Phase

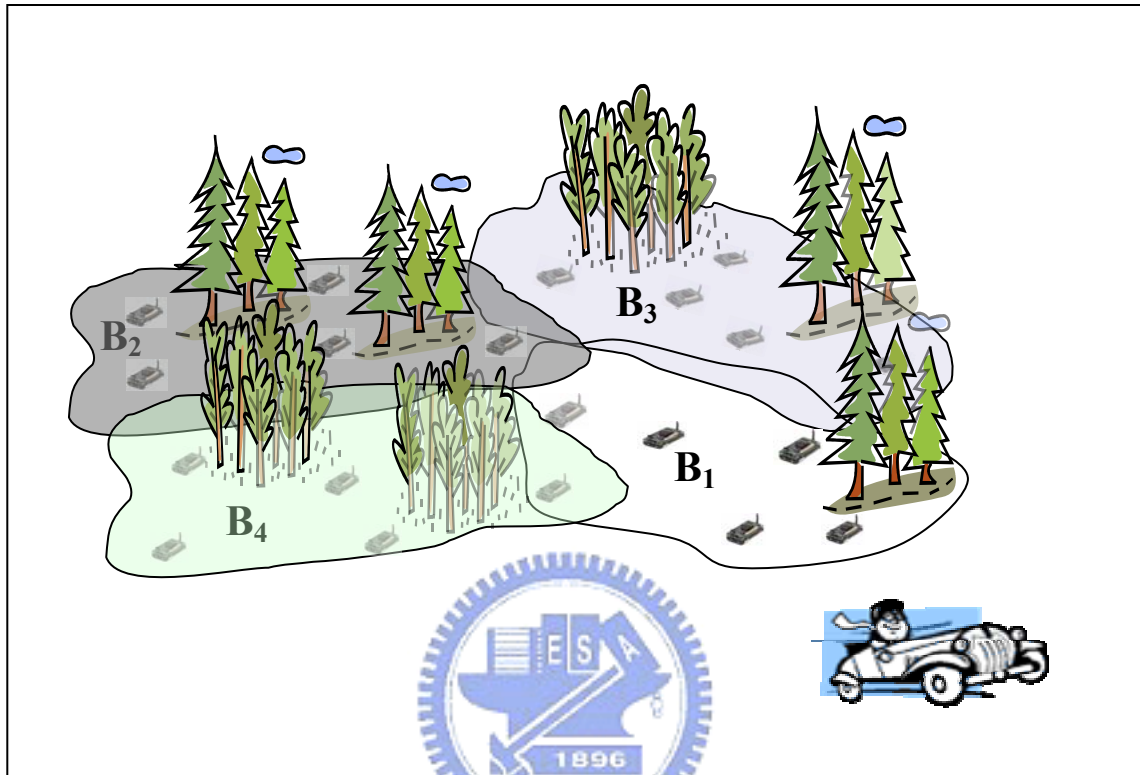


Figure 4.3.1 Deploying sensor nodes by blocks

Since wireless sensor network is expected to consist of hundreds, or even thousands of sensor nodes, it is unrealistic and uneconomical to deploy these sensors one by one.

For this reason, we came up with a workable alternative method by dividing sensor nodes into many blocks, and then deploy these blocks one by one. Therefore, each block monitors one field. Figure 4.3.3 shows that some sensor blocks are deployed at a woodland location. The sensing range of blocks could overlap.

4.3.3 Signature Phase

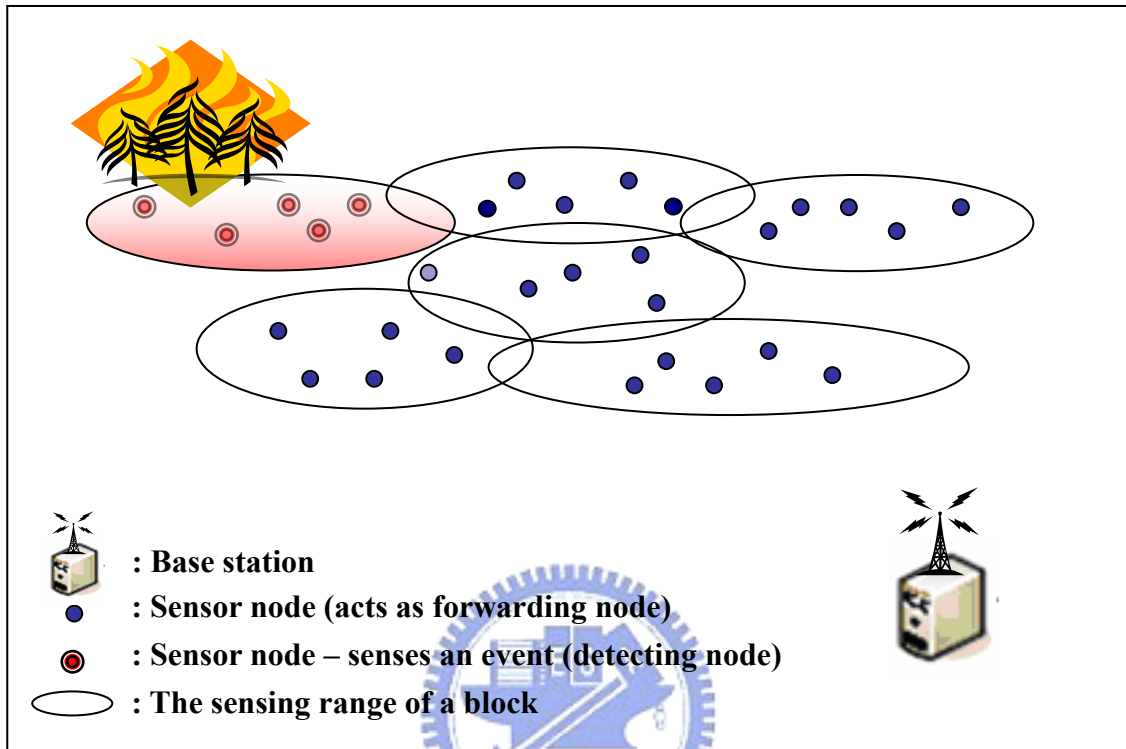


Figure 4.3.2 An event occurs

When an event occurs, some sensor nodes in some blocks may detect the event. We call these sensor nodes that can sense an event happened as detecting nodes. If the event occurs close to the block boundary, these detecting nodes may be in different adjacent blocks. Then, only the nodes in the same block could sign this event by themselves. Therefore, there might have many different blocks to sign the same event. Figure 4.3.4 illustrates a fire event occurred and one block has sensed it. In this block (shown as the red block), the detecting nodes can come to a consensus on a message, called m which contains application-dependent information such as the type, occurrence time and the location of the event.

Now, we discuss the detecting nodes in B_i block. These nodes are required to select an

AP (aggregation point) among themselves. We already know that each node has $(q+1)$ key shares. Except for the AP, other detecting nodes will generate $(q+1)$ shares, which are $(j, k) \parallel h(m, h(m, K_{b,j,k})) \parallel h(m, K_{b,j,k})$, that $K_{b,j,k} \in$ the detecting node. Then these detecting nodes send these shares to AP. Figure 4.3.5 depicts this process.

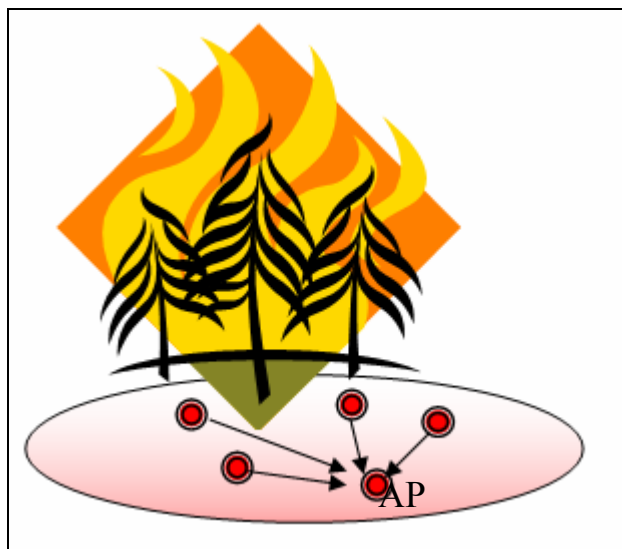


Figure 4.3.3 Sending shares to AP

In our scheme, based on the PHF characteristic of Proposition 3.1.4 and Corollary 3.18, there are $(q+1)$ key sets and each key set has q key shares in a block. Additionally, according to the PHF, we distribute keys to each node. For any subset X of nodes with $|X| = w$, there exists at least one key set that the nodes of subset X have no identical key shares. Therefore, the AP receives shares from more than w detecting nodes, and it can pick w shares among them. These shares are generated by different key shares from the same key set, and it means that $(q+1)$ shares are $\langle (j, k) \parallel h(m, h(m, K_{b,j,k})) \parallel h(m, K_{b,j,k}) \rangle$, and that b is the number of blocks, j is the number of key sets, and

$K_{b,j,k} \in \{K_{b,j,k1}, K_{b,j,k2}, \dots, K_{b,j,kw}\} \subseteq S_{b,j}, \forall m \neq n, km \neq kn, 1 \leq km, kn \leq q$. Then, we get

$h(m, K_{b,j,k})$ parts of the share and calculate $h(m, K_{b,j,k1}) \oplus \dots \oplus h(m, K_{b,j,kw})$, which called

Threshold MAC. Finally, we generate the threshold-signature report Λ , that is

$$\langle b \parallel j \parallel (k1, k2, \dots, kw) \parallel m \parallel h(m, h(K_{b,j,k1})) \parallel \dots \parallel h(m, h(K_{b,j,kw})) \parallel \text{Threshold MAC} \rangle.$$

$b \parallel j \parallel (k1, k2, \dots, kw)$ of Λ is index parameters whose purpose is used while forwarding

nodes verify the Λ . Forwarding nodes check $h(m, h(K_{b,j,k1})) \parallel \dots \parallel h(m, h(K_{b,j,kw}))$ to

determine whether the report Λ is indeed correct. Finally, the base station verifies the Λ

by Threshold MAC.

4.3.4 Verification Phase

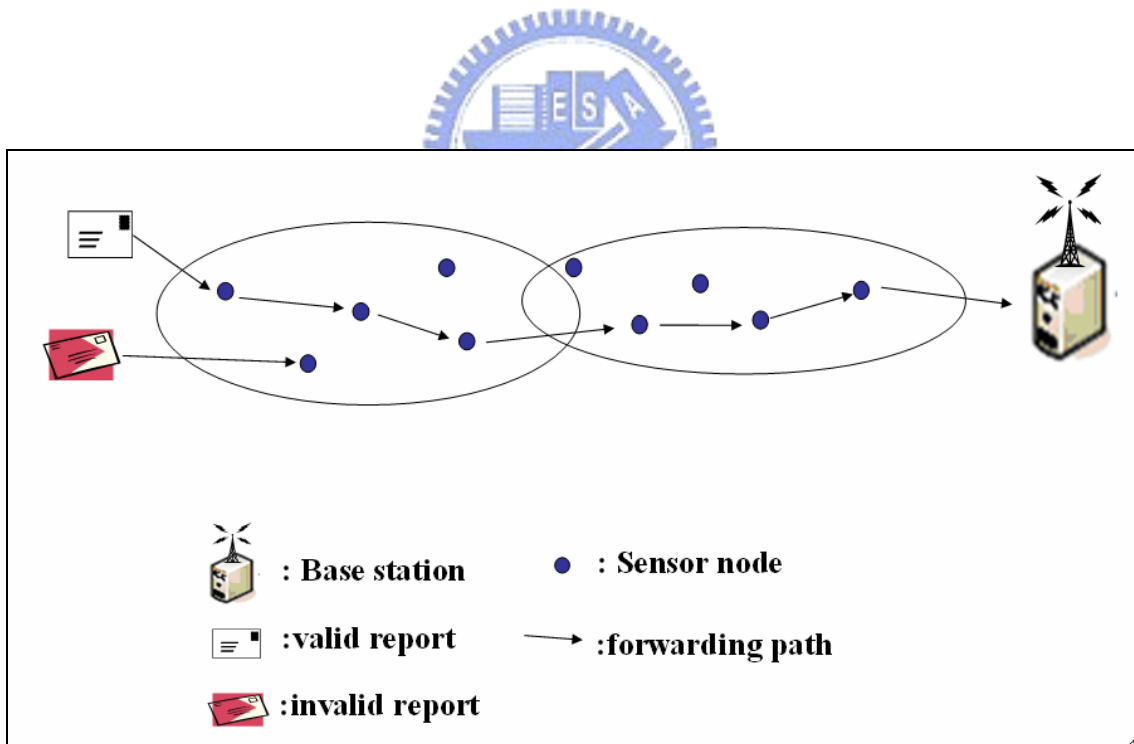


Figure 4.3.4 The forwarding phase

In our scheme, all sensor nodes can function as forwarding nodes. In Figure 4.3.4, blue nodes could act the role of forwarding nodes. The AP sends a report Λ to the base

station along a multi-hop path. The content of such a report is

$$\langle b \parallel j \parallel (k_1, k_2, \dots, k_w) \parallel m \parallel h(m, h(K_{b,j,k_1})) \parallel \dots \parallel h(m, h(K_{b,j,k_w})) \parallel \text{Threshold MAC} \rangle.$$

The verifications of a forwarding node and the base station are different. Now, we describe the verification of a forwarding node. The operation is performed as follows:

1. Upon receipt of a report Λ to be forwarded, an intermediate node, say A, fetches $b \parallel j \parallel (k_1, k_2, \dots, k_w)$ from the report Λ . Node A will then check whether it has a hash share $h_{b,j,z}$ ($z = k_1, k_2, \dots, k_w$) among these hash shares which it stores.
2. If yes, node A will compute $h(m, h_{b,j,z})$ and compare the value of $h(m, h_{b,j,z})$ and $h(m, h(K_{b,j,z}))$.
 - If $h(m, h_{b,j,z}) = h(m, h(K_{b,j,z}))$ – node A would consider report Λ as correct and then would forward it to the next hop.
 - Otherwise – node A would conclude that report Λ is a fabricated one and then would simply disregard it.
3. If no, node A does not have the hash share to verify report Λ . It only forwards it to the next hop.
4. Repeat the first step to the third step with other nodes until report Λ is received by the base station.

Since a base station is a data collection center with sufficiently powerful processing capabilities and resources, we assume that the base station stores all key shares that are eventually distributed to sensor nodes. When the base station receives report Λ , it verifies whether the report is valid or not. Then, the base station does the following operations:

1. It fetches $b \parallel j \parallel (k_1, k_2, \dots, k_w)$ from the report Λ , gets $K_{b,j,k_1}, K_{b,j,k_2}, \dots, K_{b,j,k_w}$, and calculates $X = h(m, K_{b,j,k_1}) \oplus h(m, K_{b,j,k_2}) \oplus \dots \oplus h(m, K_{b,j,k_w})$.

2. Compare the value of X and Threshold MAC from the report Λ
 - If equal – report Λ is valid.
 - Otherwise – report Λ is a fabricated one, and it then is thrown away.

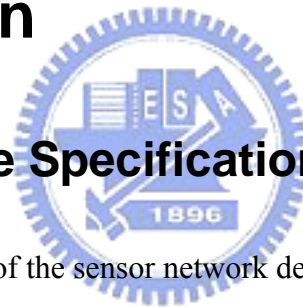


Chapter 5 Evaluation and Analysis

In the current chapter, we discuss the evaluation and analysis of our scheme. We first introduce the environment of our implementation. Second, in our scheme, the bogus report may be verified correct and be forwarded by several forwarding nodes. The bogus report is not always instant filtered. So we discuss the probability of filtering one bogus report.

5.1 Evaluation

5.1.1 Hardware Specifications



At present, manufactures of the sensor network devices include Crossbow Motes, Berkeley Piconodes, Sensoria WINS, MIT uAMPs, Smart Mesh Dust Mote, Intel iMote, Intel Xscale Nodes, and others.

We use Crossbow's MIB510 Programming board and MicaZ motes which include sensor boards and programming boards. The characteristics of MicaZ motes are as follows : [15]

- Wireless platform for low-power sensor networks
- 2.4 GHz, IEEE 802.15.4 compliant
- Offers a 250 kbps high data rate and utilizes a direct sequence spread spectrum radio that is resistant to RF interference
- Wireless communications with every node as router capability
- An 8-bit Atmel ATmega processor, 128KB instruction memory (FLASH) and

4KB RAM. The CPU is clocked at 7.37MHz.

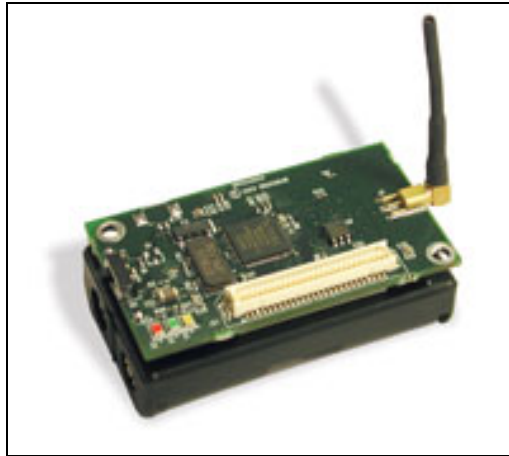


Figure 5.1.1 MicaZ mote

MIB510 Programming board specifications are as follows :

- It allows for the aggregation of sensor network data on a PC as well as other standard computer platforms. It also provides a serial programming interface for all MicaZ hardware platforms.
- It can act as a base station for wireless sensor networks via standard MicaZ processor radio board



Figure 5.1.2 MIB510 Programming board

5.1.2 TinyOS

TinyOS[19] is an open-source operating system designed for wireless embedded sensor networks. It is designed by the component-based architectures that are able to incorporate rapid innovation and operate with very limited resources. TinyOS's component library includes network protocols, distributed services, sensor drivers, and data acquisition tools – all of which can be used as-is or be further refined for a custom application.

TinyOS uses the NesC language, an extension of C, with similar syntax, that attempts to embody the structuring concepts and execution model [16]. As an embedded operating system, the TinyOS is event-driven concurrency model at interrupts and tasks



5.1.3 Performance Evaluation

In this section, we evaluate the performance of our scheme.

(1) Analysis of the size of the stored data :

Our scheme requires that each node needs to store the materials which are $(q+1)$ key shares and $(q+1) \times (\text{number of blocks} - 1)$ hash values. We assume the size of key share is 64 bytes. We assume that one-way hash function h implemented using SHA-1[17] with a 20-byte output. So the size of hash value is 20 bytes; n is the number of sensor nodes of the network, q^2 is the number of sensors in each block, so the number of blocks is $\frac{n}{q^2}$. S means the size of the materials that each sensor stores.

$$S = \left(\frac{n}{q^2} - 1\right) \times (q + 1) \times 20 + (q + 1) \times 64 \text{ bytes}$$

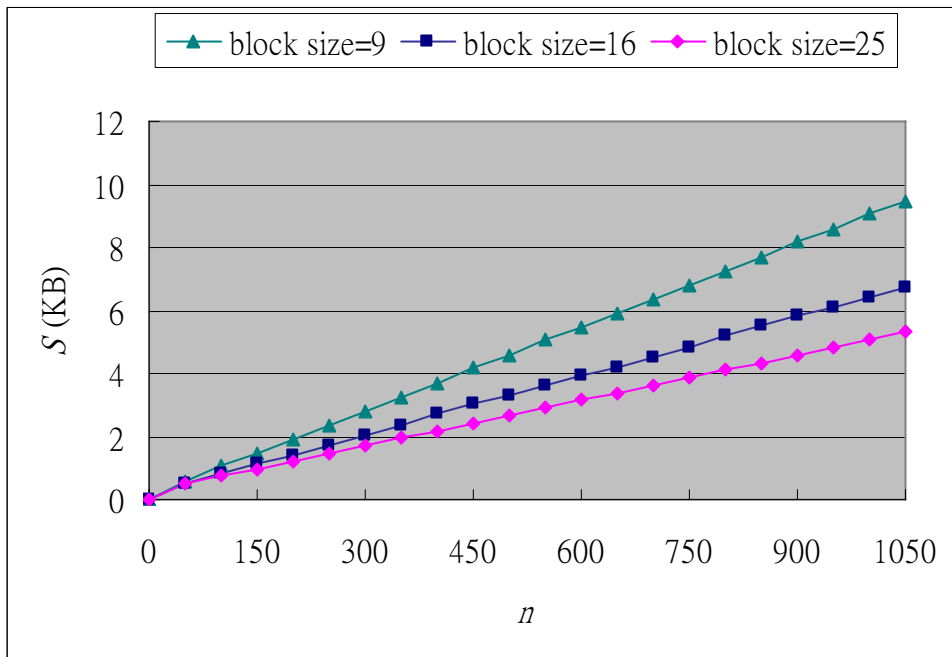


Figure 5.1.3 the relation with n , q^2 , and stored data in each node

Figure 5.1.3 shows that as the number of sensors in the network increases, each sensor node also needs to store more materials. If the number of sensor nodes in the network is less than 1000, the size of materials that each sensor node needs to store is less than 10KB when the number of sensors in a block is 9, 16 or 25. The MicaZ mote has 128KB instruction memory, and therefore the size of materials each sensor node stores is accepted.

(2) Overhead Analysis :

Our scheme requires that the format of each report Λ is

$$\langle b \parallel j \parallel (k_1, k_2, \dots, k_w) \parallel m \parallel h(m, h(K_{b,j,k_1})) \parallel \dots \parallel h(m, h(K_{b,j,k_w})) \parallel \text{Threshold MAC} \rangle$$

$\langle b \parallel j \parallel (k_1, k_2, \dots, k_w) \rangle$ represents key share indices and the overhead is

$(w + 2)$ bytes. We assume that one-way hash function h implemented using SHA-1[17] with a 20-byte output.

$\langle h(m, h(K_{b,j,k_1})) \parallel \dots \parallel h(m, h(K_{b,j,k_w})) \parallel \text{Threshold MAC} \rangle$ is used for verifying

report Λ , and the size is $20 \times (w + 1)$ bytes. Therefore, the overhead of total packet introduced by our scheme is $20 \times (w + 1) + (w + 2)$ bytes, and it depends

on w . The w value is based on $q + 1 > \binom{w}{2}$, and the q^2 value corresponds to

the number of sensors in a block. Therefore, Table 5.1.1 shows the relation among w , q , and the packet overhead.

The packet overhead is 84 bytes in [1]. In our scheme, w represents the number of sensor nodes needed to sign a report. We could know that as the number of sensor nodes sign a report increases, the degree of environmental security also increases. Then, Table 5.1.1 shows that as w gets larger, the size of packet overhead also gets larger. Therefore, it is a tradeoff between the degree

of environment's security and the packet overhead.

q	3	5	7	9	11	13	16
Block size (q^2)	9	25	49	81	121	169	256
w	3	3	4	4	5	5	6
Overhead of packet (Bytes)	85	85	106	106	127	127	148

Table 5.1.1 The relation among w , q , and the packet overhead

(3) Energy Analysis :

When an event occurs, each detecting node sends $(q+1)$ shares to AP. The AP needs to receive a lot of shares and handle them to generate threshold-endorsement reports. Therefore, AP's energy consumption is high. In wireless sensor networks, it is possible that every node in the network functions to act as an aggregation point. For this reason, if sensor nodes can act as the role of the AP by turns, the energy cost can be dispersed on sensor nodes.

Regardless of threshold-signature or verification, sensor nodes need to calculate one-way hash function. In [18], A.S. Wander, N. Gura, H. Eberle, V. Gupta, and S.C. Shantz analyzed the energy cost with SHA-1 for hashing, and the result is in Table 5.1.2. Because they used Mica2dot mote to implement the operation and the microprocessor was the same on Mica2dot mote and MicaZ mote, we could know that the energy cost of SHA-1 operation was similar to $5.9 \mu\text{J}/\text{byte}$. There are some special elliptic curves based on the Tate Pairing concept. The energy of one elliptic curve operation is 200 times more than the energy of one SHA-1 operation.

Therefore, our scheme is a low-power mechanism and it suits the wireless

sensor networks.

Algorithm	Energy
SHA-1	5.9 μ J /byte

Table 5.1.2 Energy numbers for SHA-1. The numbers were averaged over inputs ranging from 64 to 1024 bytes.

5.2 Security Analysis

We first calculate the probability of filtering the fabricated data reports.

The $\langle h(m, h(K_{b,j,k1})) \parallel \dots \parallel h(m, h(K_{b,j,kw})) \rangle$ part of report Λ is used to verify report Λ . If the forwarding node does not have $h_{b,j,z} (z = k1, k2, \dots, kw)$, it cannot check whether the report is correct or incorrect. If the forwarding node has $h_{b,j,z} (z = k1, k2, \dots, kw)$, it just can check $\langle h(m, h(K_{b,j,z})) \rangle$ to judge whether report Λ valid or not. If the attacker compromises n_c sensor nodes ($n_c < w$) in one block, he can generate a bogus report Λ' with n_c correct shares, $(w - n_c)$ wrong shares, and false Threshold MAC. Assume that the $n_c = |\text{KC}|$ shares are is generated by $\text{KC} = \{K_{b,j,m1}, K_{b,j,m2}, \dots, K_{b,j,mc}\}$. If the detecting node has $h_{b,j,k} (k \in \{m1, m2, \dots, mc\})$, it will consider report Λ' is correct and forward it. Therefore, we focus on this situation and further discuss it.

We assume that n is the number of sensor nodes in the network, q^2 is the number of sensor nodes in a block; w is the number of nodes endorsing the report together, and n_c is the number of compromised nodes signing the report. We can know that $\frac{n}{q^2}$ means the number of blocks in the network.

Each hash value has $(\frac{n}{q^2} - 1) \times q$ sensor nodes to have in the network, so there are $(\frac{n}{q^2} - 1) \times q$ nodes to check one share. Let $p(t)$ be the probability when a bogus report is not filtered through t hops exactly.

$$p(t) = \prod_{i=0}^{t-1} \frac{(n - q^2) - n_c \times [(\frac{n}{q^2} - 1) \times q] - i}{n - q^2 - i} \quad , n_c = 0, 1, 2, \dots, w - 1$$

Thus, the probability of a bogus report filtered and dropped is $P_f = 1 - p(t)$.

Now, we want to examine the probability of a bogus report filtered when n is about 640 and $q=3, 4, 5$. The following figures (Figure 5.2.1, Figure 5.2.2, and Figure 5.2.3) show the results of the above analyses.

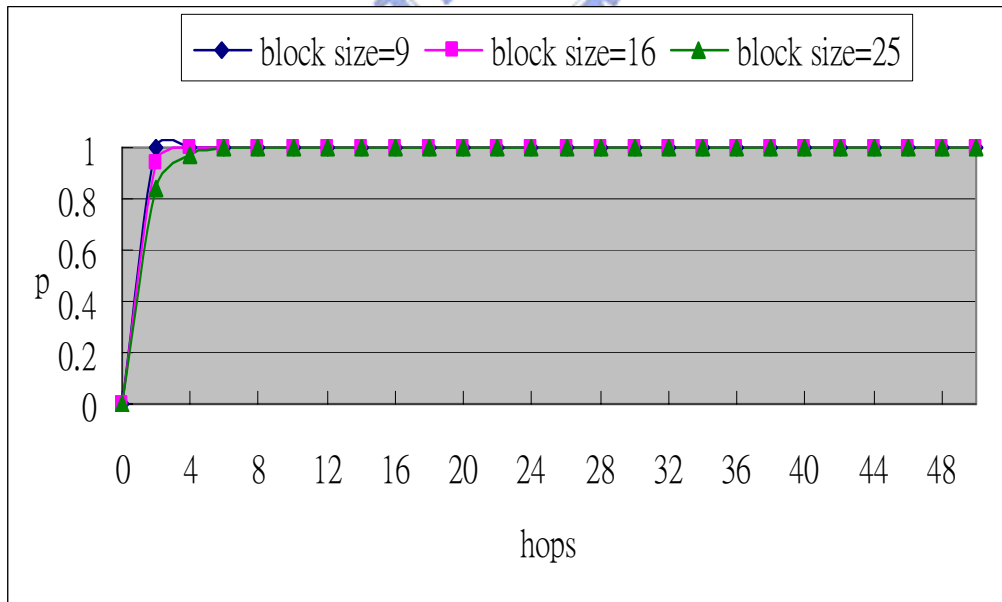


Figure 5.2.1 $n_c = 0$, $w = 3$, the attacker generates a bogus report with 3 bad shares.

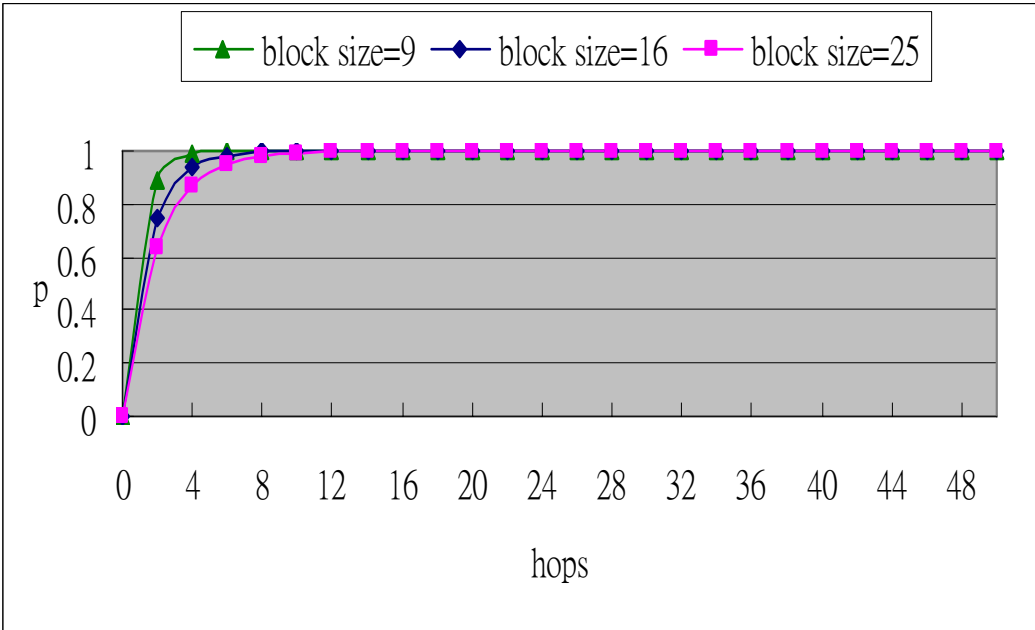


Figure 5.2.2 $n_c = 1, w = 3$, the attacker generates a bogus report with 2 bad shares.

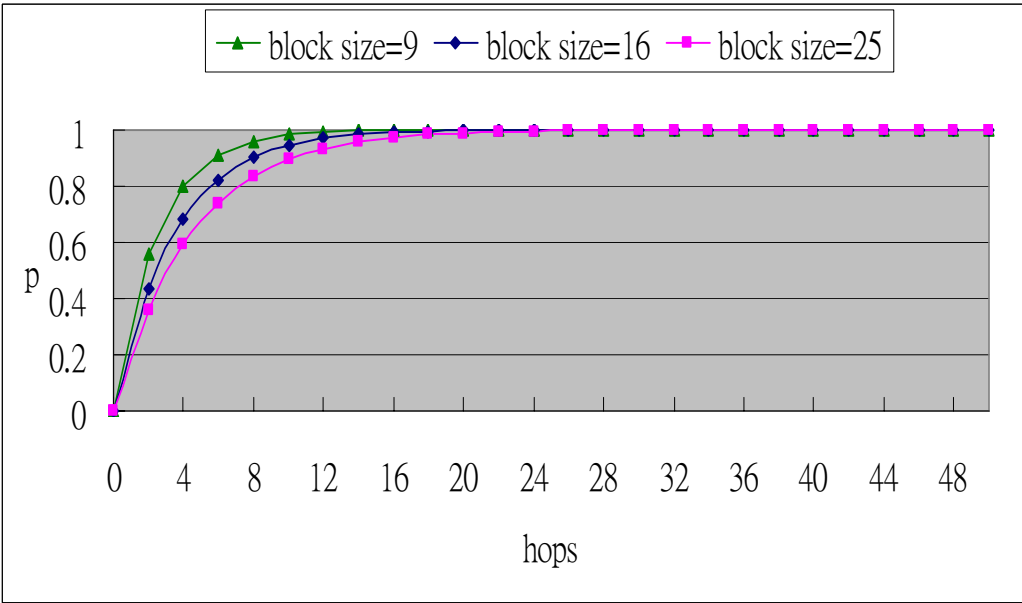


Figure 5.2.3 $n_c = 2, w = 3$, the attacker generates a bogus report with 1 bad share.

In wireless sensor networks, we can not keep sensor nodes away from attacks. If attackers compromise some nodes, they can generate bogus reports and inject them into the networks. Therefore, it is crucial to filter these bogus reports at an earlier phase. In our scheme, the forwarding nodes check $\langle h(m, h(K_{b,j,k1})) \parallel \dots \parallel h(m, h(K_{b,j,kw})) \rangle$ to judge the report correct or not. If the forwarding node does not have the $h_{b,j,z}$ ($z = k1, k2, \dots, kw$), it can not verify the report. Besides, if the attacker compromises the sensor node that has $K_{b,j,m}$ and he generates the bogus data Λ' , the forwarding nodes which has $h_{b,j,m}$ will assume the bogus data Λ' is correct and they will forward the report Λ' . Therefore, the bogus reports might not be instant filtered in our scheme.

In Figure 5.2.1, the attacker does not compromise any sensor nodes and he injects the bogus reports into the network. The bogus reports are detected through approximately two forwarding nodes, when the number of sensor nodes in each block is 9, 16, or 25. In Figure 5.2.2, the attacker compromises one sensor node and he injects the bogus reports into the network. The bogus reports are filtered through about eight forwarding nodes, when the number of sensor nodes in each block is 9, 16, or 25. In Figure 5.2.3, the attacker compromises two sensor nodes and he injects the bogus reports into the network. The bogus reports are filtered through about fourteen forwarding nodes, when the number of sensor nodes in each block is 9, 16, or 25. Consequently, as the attacker compromises more sensor nodes and injects bogus reports, these bogus reports are filtered through more forwarding nodes. The forwarding nodes just execute one hash operation at most, so the energy cost that the forwarding nodes required is relatively low.

Also, if there are more than w sensor nodes which are compromised in the same block, the adversaries can successfully generate bogus reports; the forwarding nodes or the base station can believe them. Hence, our scheme has w -degree toleration.

Chapter 6 Conclusion and Future

Work

Nowadays, since the applications of wireless sensor network have become increasingly popular and there are many hidden potential danger attacks in wireless sensor networks, there is a high demand in developing a secure wireless sensor network.

In order to reduce the impact of the node-compromised attacks, we proposed a key distribution approach using the perfect hash family (PHF) and a signature/verification mechanism using Threshold MAC.

Listed below is a summary of our scheme's advantages.

1. In our scheme, we utilize simple mechanisms to mitigate node-compromised attacks.
2. The computing power of the verification and the signature is low.
3. The computing speed of the verification and the signature is fast.
4. Our scheme is easy to implement and suitable for wireless sensor networks.
5. Our scheme is w -degree tolerance of the node-capture attacks.
6. Despite the limitations of wireless sensor networks, our proposed mechanism offers a potential solution to secure the wireless sensor networks.

The following descriptions are our scheme's disadvantages.

1. We can not dynamically inject sensor nodes into networks, and this means that the network is a static one.
2. The forwarding nodes might not instant filter the bogus reports, and it means that the attacker can inject the bogus reports into the network in order to consume the

forwarding nodes' power.

Future research is needed to reduce the probability of the nodes being compromised.

1. We would investigate the update-key aspect. If the key shares and hash value are updated before the intruder compromises more than w sensor nodes, our scheme could have more abilities to resist these kinds of adversaries.
2. We would build security tunnel that can guarantee data to be transmitted securely, and it also can guarantee that data not be eavesdropped.
3. We would like to strengthen our scheme's algorithm.

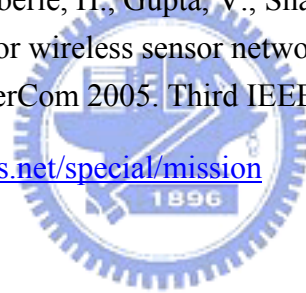


Chapter 7 Reference

- [1] Z. Yanchao, L. Wei, L. Wenjing , F. Yuguang, “Location-based compromise-tolerant security mechanisms for wireless sensor networks,” IEEE Journal on Selected Areas in Communications, vol.24,No.2,247-260,Feb. 2006
- [2] D. J. Malan, M. Welsh, and M. D. Smith, “A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography,” in Proc. IEEE SECON, Santa Clara, CA, Oct. 2004, pp. 71–80.
- [3] N. Gura, A. Patel, A. Wander, H. Eberle, and S. C. Shantz, “Comparing elliptic curve cryptography and RSA on 8-bit CPUS,” in Proc. CHES, Boston, MA, Aug. 2004, pp. 119–132.
- [4] R. Watro, D. Kong, S. fen Cuti, C. Gardiner, C. Lynn, and P. Kruus, “Tinypk: Securing sensor networks with public key technology,” in Proc. ACM SASN, Washington, DC, Oct. 2004, pp. 59–64.
- [5] G. Gaubatz, J. Kaps, and B. Sunar, “Public keys cryptography in sensor networks—revisited,” in Proc. ESAS, Heidelberg, Germany, Aug. 2004, pp. 2–18.
- [6] G. Bertoni, L. Breveglieri, M. Venturi, “ECC Hardware Coprocessors for 8-bit Systems and Power Consumption Considerations” in Information Technology: New Generations, 2006. ITNG 2006. Third International Conference on 10-12 April 2006 Page(s):573 – 574
- [7] K. Bıçakcı, C. Gamage, B. Crispo, and A. Tanenbaum, “A. One-Time Sensors: A Novel Concept to Mitigate Node-Capture Attacks.” in ESAS 2005: 2nd European Workshop on Security and Privacy in Ad hoc and Sensor Networks. It will be published as postproceedings in LNCS, Springer.
- [8] G. Bertoni, L. Chen, P. Fragneto, K. Harrison, and G. Pelosi1, “Computing Tate pairing on smartcards,” *White Paper STMicroelectronics*, 2005. [Online]. Available: http://www.st.com/stonline/products/families/smartcard/ast_ibe.htm.
- [9] K. Kyung-Mi, “Perfect Hash Families: Constructions and Applications,” a thesis of University of Waterloo, 2003.
- [10] 2. S. R. Blackburn, Combinatorics and threshold cryptography. In F. C. Holroyd, K. A. S. Quinn, C. Rowley and B. S. Web (eds), Combinatorial Designs and their

Applications, Chapman and Hall/CRC Research Notes in Mathematics, Vol. 403, CRC Press, London (1999) pp. 49-70.

- [11] C. C. Lindner, C. A. Rodger, "Design Theory," CRC Press, 1997.
- [12] A. Shamir, "How to Share a Secret," Communications of ACM, 1979.
- [13] K. M. Martin, J. Pieprzyk, R. S. Naini, H. Wang, and P. R. Wild, "Threshold MACs", ICISC 2002: 237-252
- [14] P. Erdős, P. Frankl, and Z. Füredi, Families of finite sets in which no set is covered by the union of r others, Israel Journal of Mathematics, 51(1985), 79-89.
- [15] Crossbow, Inc., <http://www.xbow.com/>
- [16] NesC, <http://nescc.sourceforge.net/papers/nesc-ref.pdf>
- [17] Digital Hash Standard, Federal information processing standards publication 180-1, Apr. 1995.
- [18] Wander, A.S., Gura, N., Eberle, H., Gupta, V., Shantz, S.C., "Energy analysis of public-key cryptography for wireless sensor networks", Pervasive Computing and Communications, 2005. PerCom 2005. Third IEEE International Conference on.
- [19] TinyOS, <http://www.tinyos.net/special/mission>



Appendix A

1. The proof of Proposition 3.1.3 [9]

Proof :

Let $\Pi = \{\pi_1, \pi_2, \dots, \pi_n\}$ be a family of w -partition of a set \mathbf{A} . We can construct a collection F of functions by labelling the parts of each partition π_i with distinct elements of \mathbf{B} , and then defining f_i to map each $x \in \mathbf{A}$ to the label of the part of π_i containing x . Then resulting set of functions, say $F = \{f_1, f_2, \dots, f_n\}$, is an (n, m, w) -perfect hash family.

Conversely, suppose that $F = \{f_1, f_2, \dots, f_n\}$ is a PHF($N; n, m, w$). We can construct a set of partitions of \mathbf{A} , say $\Pi = \{\pi_1, \pi_2, \dots, \pi_n\}$, by setting π_i to f_i for all $i = 1, 2, \dots, n$. And then for any π_i , $x, y \in \mathbf{A}$ in the same part of π_i whenever $f_i(x) = f_i(y)$. Hence Π is the desired set of partitions of \mathbf{A} .

2. The proof of Proposition 3.1.4 [9]

Proof :

For given an (n, m, w) -perfect hash family $\mathbf{F} = \{f_1, f_2, \dots, f_n\}$, we can produce an array M of size $N \times n$ with entries in \mathbf{B} as follows : Index the columns of M by the elements $x \in \mathbf{A}$, and index the rows of M by the set $\{1, 2, \dots, n\}$, i.e., each row of the array corresponds to one of the functions in the family \mathbf{F} . Setting the value of the entry (i, x) in M to be $f_i(x)$, the resulting array satisfies the desired conditions .

In the reverse direction, suppose that M is an array of size $N \times n$, having entries in \mathbf{B} . For $i = 1, 2, \dots, n$ and $x \in \mathbf{A}$, we define $f_i(x)$ to be the value of the entry (i, x) of M . Hence $f_i(x) = f_i(y)$ for $f_i \in F$ whenever the (i, x) th and (i, y) th entries of M are equal. Then we have a desired set $F = \{f_i : 1 \leq i \leq n\}$, which is a PHF($N; n, m, w$).

