Digital Ecosystems and Business Intelligence Institute
Curtin Business School

Key Management for Wireless Sensor Network Security

Biming Tian

This thesis is presented for the Degree of
Doctor of Philosophy
of
Curtin University

August 2011

Declaration

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgment has been made.

This work has not been submitted for any other degree or diploma in any other university.

Signature: [illegible]

Date: 8/11/2011

Abstract

Wireless Sensor Networks (WSNs) have attracted great attention not only in industry but also in academia due to their enormous application potential and unique security challenges. A typical sensor network can be seen as a combination of a number of low-cost sensor nodes which have very limited computation and communication capability, memory space, and energy supply. The nodes are self-organized into a network to sense or monitor surrounding information in an unattended environment, while the self-organization property makes the networks vulnerable to various attacks.

Many cryptographic mechanisms that solve network security problems rely directly on secure and efficient key management making key management a fundamental research topic in the field of WSNs security. Although key management for WSNs has been studied over the last years, the majority of the literature has focused on some assumed vulnerabilities along with corresponding countermeasures. Specific application, which is an important factor in determining the feasibility of the scheme, has been overlooked to a large extent in the existing literature.

This thesis is an effort to develop a key management framework and specific schemes for WSNs by which different types of keys can be established and also can be distributed in a self-healing manner; explicit/implicit authentication can be integrated according to the security requirements of expected applications. The proposed solutions would provide reliable and robust security infrastructure for facilitating secure communications in WSNs.

There are five main parts in the thesis. In Part I, we begin with an introduction to the research background, problems definition and overview of existing solutions. From Part II to Part IV, we propose specific solutions, including purely Symmetric Key Cryptography based solutions, purely Public Key Cryptography based solutions, and a hybrid solution. While there is always a trade-off between security and performance, analysis and experimental results prove that each proposed solution can achieve the expected security aims with acceptable overheads for some specific applications. Finally, we recapitulate the main contribution of our work and identify future research directions in Part V.

To my dear husband Ming (Alex) Lu and son Allen Ziye Lu. . . . . .

Acknowledgements

I would especially like to express my gratitude and heartfelt thanks to my supervisor, Dr Song Han, for his excellent supervision and support in this research. He is a motivator, a challenger, and above all a helpful friend. His constructive comments were invaluable for the completion of this thesis. His encouragement has been an inspiration to me.

I would like to express my sincere thanks to my co-supervisor Professor Tharam Dillon. His research expertise, his research community network within Australia and overseas, and his advice in setting up my research foundation have been a great help. I would also like to thank my associate supervisor Dr Alex Talevski who provided me an opportunity for Oil & Gas industrial project. I would also like to express deep gratitude to Professor Jiankun Hu at the University of New South Wales at the Australian Defence Force Academy. I am grateful to him for his publication support and suggestions for my career development.

I would like to thank Professor Elizabeth Chang and DEBII's funding to support my publication and conference trips. I would also like to thank Dr Omar Hussain who is the chairperson of my thesis committee for his time and support. My thanks also extend to staffs and students at DEBII for making it such a friendly work environment. It has been an enjoyable interacting with these wonderful and talented people. Their advice and friendship have helped me to enjoy and learn a great deal from my PhD experience.

I would like to express my heartiest thanks to my families. Although they are so far from me, they are always close to my heart. I gratefully acknowledge my siblings, and especially my father Jisheng Tian and my mother Suhua Jin for all they have given me. They have stood by me at all times, providing constant support and love.

Finally, I would like to dedicate this dissertation to my husband Ming (Alex) Lu and my son Allen Ziye Lu. It was a little tough but amazing experience to complete PhD thesis during pregnancy. I am lucky to have them by my side and enrich my life with love and happiness. Without my husband's consistent love and support and motivation from my baby, I would not have been able to overcome the difficulties and challenges of this undertaking. I also thank my father-in-law Xudong Lu and mother-in-law Jijun Chen for their understanding and encouragement.

List of Publications from the Thesis

- **Refereed Journal Articles**

  1. Biming Tian, Song Han, Liu Liu, Saghar Khadem, and Sazia Parvin, *Towards Enhanced Key Management in Multi-Phase ZigBee Network Architecture*, Elsevier, Fast Track: TrustCom2010: Computer Communications; Accepted, in press.
  2. Biming Tian, Song Han, Sazia Parvin, Jiankun Hu, and Sajal Das, *Self-healing Key Distribution Schemes for Wireless Networks: A Survey*, Special Focus on Multimedia Security and Privacy (MSP): The Computer Journal, Vol.54, Issue 4, pp.549-569, April 2011.
  3. Miao Xie, Song Han, Biming Tian, and Sazia Parvin, *Anomaly Detection in Wireless Sensor Networks: A Survey*, Elsevier, Journal of Network and Computer Applications, Vol.34, Issue 4, pp.1302-1325, July 2011.
  4. Biming Tian, Song Han, Jiankun Hu, and Tharam S. Dillon, *A Mutual-Healing Key Distribution Scheme in Wireless Sensor Networks*, Elsevier, Journal of Network and Computer Applications, Vol.34, Issue 1, pp.80-88, January 2011.
  5. Song Han, Biming Tian, Mingxing He, and Elizabeth Chang, *Efficient Threshold Self-healing Key Distribution Scheme with Sponsorization for Infrastructureless Wireless Networks*, IEEE Transactions on Wireless Communications, Vol.8, No.4, pp.1876-1887, April 2009.
  6. Song Han, Tharam S. Dillon, Elizabeth Chang, and Biming Tian, *Secure Web Service Using Two-way Authentication and Three-party Key Establishment for Resource Delivery*, Elsevier, Journal of Systems Architecture, Vol.55, Issue 4, pp.233-242, April 2009.

7. Biming Tian, Song Han, Miao Xie, and Sajal Das, *Fully Anti-collusive Self-healing Key Distribution Scheme Using Hash Chain and Vector Space Secret Sharing*, IEEE Transactions on Parallel and Distributed Systems; (Submitted for Acceptance).

- **Refereed Conference Articles**

1. Biming Tian, Song Han, Sazia Parvin, and Miao Xie, *Generalized Hash-Binary-Tree Based Self-healing Key Distribution with Implicit Authentication*, the 7th International Wireless Communication and Mobile Computing Conference (IWCMC 2011), Istanbul, Turkey, July 4-8, 2011.
2. Sazia Parvin, Song Han, Farookh K. Hussain, and Biming Tian, *A Combinational Approach for Trust Establishment in Cognitive Radio Networks*, the 5th International Conference on Complex, Intelligent, and Software Intensive Systems(CISIS 2011), Seoul, Korea, June 30th-July 2nd, 2011.
3. Biming Tian, Song Han, Sazia Parvin, Tharam S. Dillon, *A Key Management Protocol for Multiphase Hierarchical Wireless Sensor Networks*, the 6th IEEE/IFIP International Symposium on Trusted Computing and Communications (TrustCom 2010), Hongkong, December 11-13, 2010.
4. Sazia Parvin, Song Han, Biming Tian, Farookh K. Hussain, *Trust Based Authentication for Secure Communication in Cognitive Radio Networks*, the 6th IEEE/IFIP International Symposium on Trusted Computing and Communications (TrustCom 2010), Hongkong, December 11-13, 2010.
5. Sazia Parvin, Song Han, Biming Tian, Miao Xie, *Authenticated Spectrum Sharing for Secondary Users in Cognitive Radio Networks*, the 2nd IEEE International Conference on Network Infrastructure and Digital Content (IC-NIDC 2010), Beijing, China, September 24-26, 2010.
6. Song Han, Biming Tian, Yanchun Zhang, Jiankun Hu, *An Efficient Self-Healing Key Distribution Scheme with Constant-*

Size Personal Keys for Wireless Sensor Networks, IEEE International Conference on Communications (ICC 2010), Cape Town, South Africa, May 23-27, 2010.

7. Biming Tian, Song Han, and Tharam S. Dillon, *A Key Management Scheme for Heterogeneous Sensor Networks Using Keyed-Hash Chain*, the 5th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN 2009), Wuyi Mountain, China, December 14-16, 2009.
8. Biming Tian, Elizabeth Chang, Farookh K. Hussain, Tharam S. Dillon, and Song Han, *An Authenticated Self-Healing Key Distribution Scheme Based on Bilinear Pairings*, the 6th Annual IEEE Consumer Communications & Networking Conference (CCNC 2009), Las Vegas, Nevada, pp.1-5, January 10-13, 2009.

Contents

List of Figures

List of Tables

List of Acronyms

WSN: Wireless Sensor Network

HWSN: Hierarchical Wireless Sensor Network

DWSN: Distributed Wireless Sensor Network

ADC: analog-to-digital converter

I/O: Input/Output

RAM: Random-access memory, which is a form of computer data storage

PKC: Public Key Cryptography

MAC: Message Authentication Code

ECC: Elliptic Curve Cryptography

TESLA: Timed Efficient Stream Loss-Tolerant Authentication

AID: Anomaly-based Intrusion Detection

MID: Misuse Intrusion Detection

IDS: Intrusion Detection System

LEAP: Localized Encryption and Authentication Protocol

EBS: Exclusion basis system

MSKP: Multiple-space Key Predistribution

BIBD: Balanced Incomplete Block Design

LOCK: Localized Combinatorial Keying

LKH: Logical Key Hierarchy

OFT: One-way Function Tree

PKI: Public Key Infrastructure

CA: Certificate Authority

KGC: Key Generation Center

KDC: Key Distribution Center

CRL: Certificate Revocation List

KEK: Key Encryption Key

TEK: Traffic Encryption Key

BS: Base Station

GM: Group Manager

SPINS: Security Protocols for Sensor Networks

SNEP: Secure Network Encryption Protocol

CBC-MAC: Cipher Block Chaining Message Authentication Code

CRC: Cyclic Redundancy Check

SEF: Statistical Enroute Filtering

RFID: Radio-frequency Identification

SDR: Subset Difference Rekeying

DDHC: Dual Directional Hash Chains

PRNG: Pseudo-random Number Generator

CSPRNG: Cryptographically Secure Pseudo-random Number Generator

DLP: Discrete Logarithm Problem

LR-WPAN: Low-rate Wireless Personal Area Networks

BDH Assumption: Bilinear Diffie-Hellman Assumption

HBT: Hash Binary Tree

SME: Security-Memory-Energy

RGM: Random Generation Material

TCL key: Trust Center Link key

LBKs: Location-based Keys

MIC: Message Integrity Code

NIST: National Institute of Standards and Technology

Part I

The Research Background, Literature Review, Problems Definition and Overview of Existing Solutions

1

Introduction

1.1 Introduction

Recent advances in tiny microprocessors, low-power circuit designs, and radio technologies have made a new technological vision referred to as "Wireless Sensor Networks (WSNs)" possible. WSNs generally consist one or several sinks (base stations) and perhaps tens of thousands of sensor nodes. The sensor nodes are equipped with low-cost sensing devices, a mini processor, and a battery-powered module. Although the price and size of sensors vary according to applications, the price of a general sensor is less than one US$ and the size would be a few cubic millimeters. When large quantities of sensors are scattered in a physical space, they are self-organized into a new form of network with sinks as interfaces to remote end users.

The typical tasks for WSNs could be sensing or monitoring physical phenomena, such as temperature, light, pressure, radiation, etc. from the surrounding environments. Sensor nodes report the crude data or aggregated data to the sink. The sink makes decisions according to the aggregated data and in turn can inject commands into the network to assign tasks to sensors. However, except for some common threats, WSNs are additionally vulnerable to other security breaches because they are usually deployed in unattended environments and use unreliable radio communication. Due to various attacks, end users may receive incorrect sensing data and correspondingly make wrong decisions, which may be dangerous in scenarios such as battlefield surveillance and environment monitoring. Therefore, proper security mechanisms have to be used in order to keep networks secure.

In Section 1.2, we introduce the background to this research, including the classification, application, constraints, vulnerabilities, security requirements of WSNs, and current hot research topics in WSN security. In Section 1.3, we identify our motivation for carrying out this research. In Section 1.4, we clearly define the research scope and highlight the areas that will be addressed in this thesis. In Section 1.5, we list the general objectives of the thesis. In Section 1.6, we summarize the significance of this research. In Section 1.7, we outline the main structure of thesis. Finally, we conclude this chapter in Section 1.8.

1.2 Background of the Research

1.2.1 Classification of WSNs

WSNs can be classified by their architectures into two categories: hierarchical networks and distributed networks [8]. A hierarchical wireless sensor network (HWSN) which is shown in Figure 1.1 typically consists of a large number of low-cost sensor nodes (L-nodes), a few powerful nodes (we alternatively call them H-nodes or cluster heads in this thesis), and a base station. In contrast, a distributed wireless sensor network (DWSN) which is shown in Figure 1.1 contains only L-nodes. In an HWSN, only the base station and H-nodes manage the network. The base station is usually assumed to be trusted. H-nodes are responsible for cluster management. The effect of nodes addition and revocation operations can be localized into one or several clusters. In addition, because only H-nodes bear the responsibility of aggregating and transmitting data to the base station, the L-nodes can be equipped with simple hardware. In a DWSN, since each node has to be capable of aggregating and forwarding data, it is necessary for each node to have complex hardware. In addition, DWSNs are not practical in many applications where strict ranks exist, such as in the military. In contrast, HWSNs take advantage of node diversity and have better scalability. Therefore, HWSNs are more applicable in some practical scenarios.

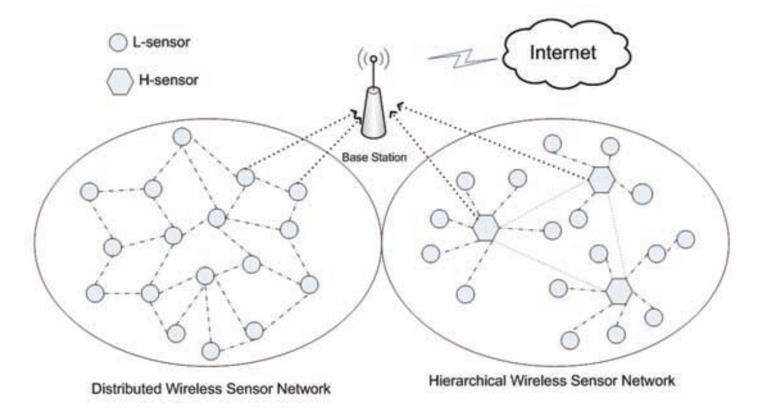

L-sensor
H-sensor
Internet
Base Station
Distributed Wireless Sensor Network
Hierarchical Wireless Sensor Network

Figure 1.1: The distributed and hierarchical wireless sensor network topologies

1.2.2 Application of WSNs

WSNs are event-driven networks widely used for military and civilian operations. One of the key advantages of WSNs is that they are potentially low-cost solutions to a variety of real-world challenges. Sensors can be deployed to continuously report environmental data for a long period of time. This is a very important improvement with respect to previous operating conditions where human operators had to move to the fields and take manual measurements periodically, resulting in less data, higher errors, higher costs and non-negligible interference with life conditions of the observed species. WSNs can reduce or eliminate the need for human involvement in information gathering in certain applications, including agriculture, health, environment, military, etc. [9].

In agricultural application, sensors are scatted over a large tract of farmland to test any changes in water or chemicals. If the fertility points are below the requirement, the farmer can obtain the information from sensors. Correspondingly, the soil would be fertilized appropriately. Sensors can also be used to monitor the nutrition and health indexes of cattle on farms. Wireless sensor networks can accurately report animal species and collect data concerning their habits, population, or position to farmers.

The development of cross-disciplinary networking within various medical fields makes in-home pervasive networks possible. For example, one of the projects undertaken by the Laboratory for Assisted Cognition Environments at University of

Washington is concerned with the continuous medical monitoring of degenerative diseases like Alzheimer, Parkinson, or similar cognitive disorders [10]. The medical networks may assist residents by continuously monitoring physiological parameters such as heartbeat or blood pressure of patients, and reporting to the hospital when any parameters are altered or an emergency occurs [11].

There are numerous examples of environment monitor applications of WSNs. Sensors can be installed on bridges or buildings to collect data about earthquake vibration patterns. They are used to detect marine ground floor erosion because wired facilities cannot research the deeply embedded points. Pollution detection systems can also benefit from WSNs. Sensors are deployed to monitor the current levels of polluting substances in a town or a river and identify the source of anomalous situations, if any. Similar detection systems can be employed to monitor rain and water levels and forecast flooding [12].

The military can also take advantage of sensor network technology. They can deploy such networks behind enemy lines and observe movements/presence of troops and/or collect geographical information on the deployment area [13]. In addition, sensor networks can be used to detect any signs of life in rescue activities.

1.2.3 Constraints of WSNs

Sensor nodes are low-cost and have very limited resources. These nodes are usually scattered randomly in a designated field and self-organized into a network after the deployment. Sensor nodes are usually densely deployed in unattended and even harsh environments and the scale of WSNs varies from hundreds to thousands of sensor nodes. The topology of WSNs may frequently change due to the mobility of nodes in some applications. Wireless channels are open and unreliable and may suffer from many kinds of attacks. The transmission of data packets may be delayed or they may not reach their destination at all. Indeed, security challenges in WSNs stem from these constraints. Here we examine a number of constraints which make the design of security mechanisms for wireless sensor networks more complicated and challenging. In order to facilitate understanding, we categorize the constraints into device constraints, communication constraints, and deployment constraints.

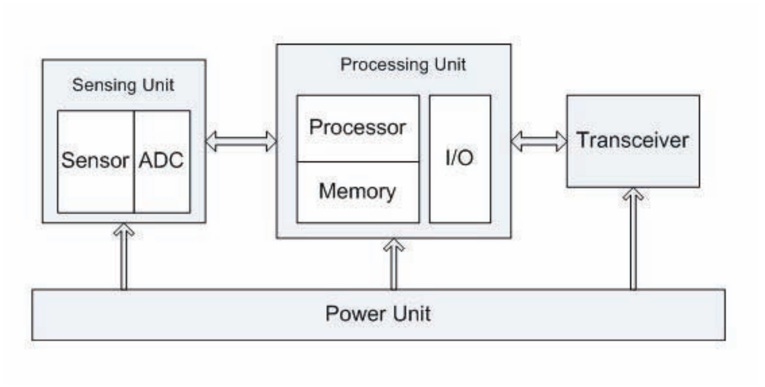


Figure 1.2: The components of a typical sensor node

Device Constraints

A typical sensor node/device comprises four basic components (as shown in Figure 1.2): a sensing unit, a processing unit, a transceiver unit, and a power unit [14].

- *Sensing unit*. Sensing unit is usually composed of two subunits: sensor and an analog-to-digital converter (ADC) [14]. The sensor produces analog signals of the observed physical phenomenon and the ADC converts the measurements into digital signals which are further processed by the processing unit.
- *Processing unit*. As a part of its processing unit, it has processor, memory, and I/O components. The main function of this unit is to analyze and process sensor data.
- *Transceiver unit*. A transceiver unit transmits and receives data. It connects a sensor node to a network.
- *Power unit*. Power unit of sensor node is typically a battery or may be supported by power scavenging units such as solar cells [14].

Current sensor nodes are tiny devices with very limited resources and unrechargeable battery. Therefore, the complicated and resource-consuming security algorithms used in other networks cannot be used to protect WSNs. The security solutions designed for sensor networks should take into consideration of the following device constraints.

- *Limited memory*. MICA2 is a widely used sensor node with 128KB program memory and 4KB RAM. Such limited memory requires short software code and communication packets. The total code space of TinyOS, which is a specialized operating system for programming with a small embedded device, is approximately 4KB [15].
- *Limited energy*. MICA2 is powered by only two AA batteries. Energy usage is another major constraint to consider when designing security mechanisms for WSNs. Since sensor nodes are usually powered by batteries so that they are physically small and economical, the unattended deployment environment makes replacing the batteries impracticable. Thus, it is very important to keep energy consumption at a reasonable level as increased expenditure of energy may decrease overall network performance.
- *Limited processor*. MICA2 has a 8-bit ATmega128L CPU. Except for the constraints of the processor itself, the data processing capability is constrained by limited energy.

Communication Constraints

Sensor nodes in WSNs communicate over wireless channels which can be accessed by anyone within the cover range. This openness feature of wireless communication makes security an important issue. In addition, wireless channels are unreliable and heavily affected by environmental conditions. This property calls for fault-tolerant countermeasures.

- *Unreliable channels and limited bandwidth*. Wireless communication channels are inherently unreliable. They are susceptible to various abnormalities, such as channel error and congestion, which can cause the packets to be tampered with or lost. More serious situations occur if the channels are intentionally interfered with adversaries' radio signals. The limited bandwidth makes the transmission of large blocks of data over a wireless channel impossible.
- *Collision*. WSNs are susceptible to packet collision in the wireless channel as they utilize a dense arrangement of nodes. Collisions happen when two or more sensor nodes within communication range of each other transmit packets at the same time. The wireless protocols have to handle traffic collision as

the retransmission of packets is impracticable due to the energy limitation of sensor nodes [14].

- *Latency*. Latency arises from the nature of WSN communication, such as multi-hop routing, network congestion, and node processing capability. This latency makes it difficult to achieve synchronization among sensor nodes and further impacts on WSN security in terms of event reporting and cryptographic key distribution and updating.

Deployment Constraints

One of the main benefits of WSNs is their ability to collect information from public, even potentially hostile, environments without supervision. Just as a coin has two sides, the unattended deployment environments render WSNs vulnerable to various types of attacks and make some physical protection measures, such as infrastructure support and tamper-proof components, infeasible.

- *Unattended environments*. WSNs are usually deployed in unattended areas open to physical attacks or harsh environments. An attacker may compromise one or a number of sensor nodes without being noticed. In addition, extreme natural phenomena-such as storms and flash flooding may also impede their functioning.

- *No fixed infrastructure*. After deployment, sensor nodes are self-organized into a network without support of a fixed infrastructure. Therefore, it is difficult to implement continuous surveillance after network deployment. If the WSN is not properly designed, the self-organized network may be inefficient, even fragile.

- *Remote management*. Although sensor nodes can be deployed in inaccessible environments and managed remotely. There are disadvantages. On the one hand, it is difficult to remotely detect attacks against WSNs , replace the batteries, and redeploy cryptographic keys; on the other hand, insecure WSNs might be used by attackers to gain access to the remote control center.

- *Application-specific property*. WSNs are application-specific networks. No single security mechanism is ideal for all the scenarios where WSNs are used.

It is impossible to design a “one-size-fits-all” solution for every different type of application.

1.2.4 Vulnerabilities of WSNs

In general, WSNs are susceptible to various security attacks due to the aforementioned constraints. The attacks can be categorized according to different criteria, such as the origin of attacks, or the techniques used in attacks. The attacks can be divided into passive attacks and active attacks according to the operation mode, or external attacks and internal attacks according to their origin [16].

Passive and active attacks

Passive attacks include eavesdropping on, or monitoring of transmissions. The openness of the wireless communication medium makes passive attacks easier than those in traditional wired environments. In a passive attack, the attacker merely attempts to learn or utilize information that is being transmitted without being detected. The attacker analyzes the large volume of collected information in order to extract any secret information. Such secret information might be used later to launch an active attack. In a passive attack, the attacker may knows the communications and follow the protocols like normal nodes. It is very difficult to detect passive attacks because the attacker does not leave much evidence.

However, in active attacks, the attacker actively launches various attacks typically through system faults or security holes. Some of the well-known active attacks include packet modification, injection, or replaying. The impact of active is more severe than passive attacks. However, additional anomalies can show evidence of malicious attacks because the attacker is actively involved in network communications [16]. Most of these attacks result in a Denial of Service, which is a degradation of or a complete halt to network service.

External and internal attacks

Generally, all the nodes in the network can be seen as honest and cooperative entities, whereas unauthorized nodes have no right to access the network. External attacks are carried out by an external adversary from outside the scope of the network. The impact of external attack is limited if external adversaries just eavesdrop on the public traffic. Such external attack can be easily prevented with effective authentication techniques. However, great harm may be caused if the external adversary gains access to the network as an authorized user by means of physical attack or capturing a few legitimate nodes, replicating them and then deploying them throughout the network. These attacks are of highly possible because WSNs are often left unattended after deployment.

Once an attacker obtains authorization from compromised nodes which are actually part of the network, it becomes an internal attacker. In this case, the attacker can easily launch a large class of attacks because it knows some private information and is seen as a valid member [16]. The replicas of the compromised nodes can be severely destructive to the functioning of a network [17]. For example, an replica can overhear the secret passing traffic, or inject bogus data into the network, defame other nodes and even revoke legitimate nodes [18]. Unfortunately, conventional cryptographic techniques, such as encryption and authentication, are not impregnable to internal attacks, because the internal attackers hold the legitimate cryptographic keys.

Message-based, node-based, and network-based attacks

Li *et al.* in [19] classified the security requirements into three security levels: message-based level, node-based level, and network-based level. Here, we classify the attack goals according to whether they are message-based attacks, node-based attacks, or network-based attacks. Message-based attacks attempt to break data confidentiality, integrity and freshness. Typical message-based attacks includes eavesdropping attack, replay attack, and traffic analysis attack. Node-based attacks target on the valid nodes to obtain secret information stored by the nodes and make further attacks using the obtained information. Node compromise, node replication, resource exhaustion, and node misbehavior are the four most common node-based attacks. Network-based attacks try to reduce network connectivity or availability. Network-based attacks can be launched locally or globally. The most common network-based attacks include routing attack and time synchronization attack. These three types of attacks are not isolated from each other. Some message-based and node-based attacks may result in Denial of Service (DoS), which degrades a network's performance.

1. Message-based attacks

    - *Eavesdropping attack.* Eavesdropping is the most common attack on privacy. By overhearing the communication, the attacker could easily obtain critical data such as sensing data and routing information. When the traffic conveys the control information about the sensor network configuration, which contains potentially more detailed information than is accessible through the location server, the eavesdropping can effectively breach privacy protection [20].
    - *Replay attack.* Replay attack is when an adversary replays the old messages at a later time. Replay attack disrupts data freshness which is very important, especially when shared keys need to be changed over time. A commonly used defence is to add a nonce or time-related counter to every message and reject messages with old nonce or counter values.
    - *Traffic analysis attack.* Traffic analysis typically combines with monitoring and eavesdropping [20]. Traffic analysis attacks provide a way to identify the network topology and special roles of some sensor nodes. Once the structure of the targeted WSN is known, the adversary can intentionally attack some critical nodes.

2. Node-based attacks

    - *Node compromise attack.* Node compromise is one of the most fruitful attacks on a WSN. Because sensor nodes are usually deployed in an open area without monitoring and protection of infrastructure and low-cost sensor nodes are not installed with tamper-proof component, an adversary can extract all secrets from a captured sensor node. The extracted secrets make the adversary more capable of launching other severe attacks [16].
    - *Node replication attack.* In the node replication attack [18], an attacker intentionally adds replicas of a compromised node to an existing sensor network. These replicas can produce inconsistent or severe disrupted

ongoing network communication: packets can be corrupted or even misrouted. A cryptographic key related defence was proposed in [21] whereby each node uses a location-based key to detect node replication attack.

- *Resource exhaustion attack*. In this attack, an adversary intends to deplete resources of the victim nodes. The targeted resources include battery power, bandwidth, and computational power. The attacks could be in the form of unnecessary requests for routes, very frequent generation of beacon packets, or forwarding of stale packets to other nodes. As a result of the attack, the base station denies access to sensory readings [22].
- *Misbehavior*. Nodes in a network misbehave when they become malicious or selfish. Malicious nodes may do harm to a network. Srinivasan *et al.* [23] classify malicious behavior into two types of misbehavior: forwarding and routing. Some common forwarding misbehavior includes packet dropping, modification, fabrication, timing attacks, and silent route changes. A malicious node with routing misbehavior attacks during the route discovery. Black hole, grey hole, and worm hole are the common routing misbehavior attacks. Selfish behavior may result in unintentionally causing damage to other nodes. The aim of a selfish behavior node is not to launch an attack, but it may have other aims such as wanting to obtain an unfair advantage over other nodes. Selfish behavior is classified as either self exclusion or non-forwarding [23]. In self-exclusion misbehavior, a selfish node does not participate in the network. This helps the selfish node conserve its power. A non-forwarding node behaves selfishly by simply not forwarding the packets it receives in order to save its resources.

3. Network-based attacks

   - *Routing attack*. Many sensor network routing protocols are so simple that they are vulnerable to attacks. Karlof and Wang in [24] documented a list of attacks which may be applied to compromise routing protocols that have been proposed in the literature for WSNs. Most attacks against sensor network routing protocols fall into one of the categories listed in Table 1.1.

Table 1.1: Routing attacks in wireless sensor networks

Attack type	Attack method
Spoofed, altered, or replayed routing information	Create routing loop, attract or repel network traffic, extend or shorten source routes,generate false error messages, partition the network, increase end-to-end latency etc.
Selective forwarding	A malicious node refuses to forward certain messages or behaves like a black hole and refuses to forward every packet it receives.
Sinkhole attacks	Make a compromised node look especially attractive to surrounding node with respect to the routing algorithm. It makes selective forwarding trivial.
Sybil attacks	A single node presents multiple identities to other nodes in the network to defeat the redundancy mechanisms.
Wormhole attacks	An adversary tunnels messages received in one part of network over a low latency link and replays them in a different part. It can be used in combination with selective forwarding or eavesdropping.
HELLO flood attacks	An adversary sends or replays a routing protocol's HELLO packets with more transmission power and convinces nodes in the network that the adversary is its neighbor.
Acknowledgement spoofing	An adversary can spoof link layer acknowledgement for "overhead" address to neighboring nodes. The goal is convincing the sender that a weak link is strong or a disabled node is alive.

- *Time synchronization attack.* The main goal of a time synchronization attack is to convince the victim node that its clock is at a different time than it actually is. In such an attack, the adversary repeatedly forges messages to one or both end nodes. These messages carry sequence numbers or control flags that cause the end nodes to request retransmission of missed frames. If the adversary can maintain proper timing, it can prevent the end points from exchanging any useful information, causing them to waste energy in an endless synchronization-recovery protocol [25].

1.2.5 Security Requirements of WSNs

Many survey papers [16, 20] examined a great number of security requirements of WSNs. We have shown in Figure 1.2 that security requirements can be classified into three security levels: message-based level, node-based level, and network-based level. Here we address the security requirements according to these three levels.

1. *Message-based level.* WSNs are more vulnerable to various attacks than traditional networks as sensor nodes are resource-constrained and deployed in unattended environments. Correspondingly, WSNs have the following fundamental security requirements which also appear in conventional networks, namely:

    - *Confidentiality*, which ensures that the data, either unicast, multicast, or broadcast messages, should be understood by the expected recipients alone [26].
    - *Integrity*, which ensures that the transferred message is not modified by intermediate nodes [26].
    - *Authentication*, which ensures that the entities with whom one communicates are the expected ones and the received data is the original one sent by the counterparts. That is, the authentication includes authentication of both the data source and data itself [26].
    - *Refreshness*, which ensures that content-correlative information reaches its destination within the specified time [26].

2. *Node-based level.* The security services at this level deal with node addition, revocation, as well as node compromise or failure. Security requirements at this level includes:

    - *Forward secrecy*: A sensor node should not be able to break the confidentiality of any future messages after it leaves the network [26].
    - *Backward secrecy*: A sensor node should not be able to break the confidentiality of any previously transmitted messages before it joins the network [26].
    - *Fault-tolerance*: The security mechanism should continue to provide security services in the presence of faults such as nodes compromise or failure [26].

- *Efficiency*: The storage, computation, and communication overheads introduced by security mechanisms should comply with the resource constraint of sensor nodes.

3. *Network-based level.* The security requirements at this level elaborate on the network-related issues. Securing the whole network is completely different from securing a single sensor node. All the network parameters such as routing, nodes' energy consumption, communication range, network density should be discussed correlatively [19]. The network-related security requirements should include the following functions:

    - *Survivability*: The security mechanism should provide a minimum level of service in the presence of power exhaustion or attacks. In fact, survivability means tolerating faults from a network-wide perspective.
    - *Scalability*: The security mechanism should be able to scale when the network needs extension after deployment.
    - *Self-healing*: The authorized sensors can obtain the services on their own, without requesting additional transmission from the higher level, in case packets loss occurs during transmission.

1.2.6 Research Interest in WSN Security

In the previous subsections, we mentioned that WSNs are vulnerable to numerous security threats due to their deployment environments. And due to resource constraints of sensor nodes, traditional security mechanisms with large overheads of computation and communication are infeasible in WSNs. Moreover the use of WSNs must ensure that the network is protected from unauthorized access and adversarial attacks. Security in WSNs is, therefore, a particularly challenging task and attracts great interest from world-wide researchers . In this subsection, we review the current popular research interest in WSN security. It is important to note that we list the current hot research topics in the field of WSN security and highlight the importance of key management in the whole security framework, rather than giving a comprehensive coverage of existing security techniques being researched.

Key management

Key management is one of the important security aspects of WSNs as it is crucial for providing data authentication, confidentiality and integrity and almost all WSN security mechanisms rely on solid encryption. Even though key management has been intensively studied in broadcast communication and is not a unique issue to wireless sensor networks, traditional key management techniques cannot be used for WSNs directly or even with minor revision due to the constraints of sensor nodes and application environments.

Symmetric Key Cryptography based techniques are attractive for WSN application because they are energy-efficient. Symmetric keys are predistributed to sensors before network deployment. After deployment, sensors perform operations of neighbors discovery and shared key establishment to establish secure communications between them. However, due to the limited memory of sensor nodes, these Symmetric Key Cryptography based techniques are not able to achieve both perfect connectivity and perfect resilience for large-scale WSNs. Instead, the use of Public Key Cryptography (PKC) would eliminate the above problem. Due to their asymmetric property, sensors do not need to carry the predistributed keys. Any two sensors can establish a secure communication channel between themselves. Because of the key independence of each other's public key, the capture of some sensors will not affect the security of others. However, it is universally acknowledged that the essential cryptographic primitives for WSN are Symmetric Key Cryptography (e.g. RC5, RC6, AES), Message Authentication Code (MAC), and hash function (e.g. MD5, SHA-1). Public Key Cryptography has long been considered infeasible in WSNs due to resource constraints of sensor nodes. There is almost no quantitative analysis that supports this widely accepted conclusion [19]. To the best of our knowledge, the first challenge on common perception is the literature [27] which proposed a hybrid authentication key establishment scheme based on Elliptic Curve Cryptography (ECC).

Authentication

Authentication guarantees that the entities with whom one communicates are the expected ones and the received data is the original one sent by the counterparts.

Generally speaking, an asymmetric key mechanism is required to authenticate messages. However, due to the resource constraints at sensor nodes, solutions based on Public Key Cryptography (e.g. RSA) have intolerable storage and computation overheads on WSNs. Current research on authentication in WSNs focuses on broadcast authentication.

Local broadcast is an essential service in HWSNs because the networks are decoupled into small clusters and each cluster has a high degree of autonomy. In an HWSN, the base station or cluster heads broadcast commands and data within the cluster. The authenticity of such commands and data is critical for the normal operation of WSNs [28]. In a hostile environment, if sensor nodes are convinced by the forged or modified commands, they may operate in an inverse way, and cannot fulfill the intended target of the network. However, providing broadcast authentication in hierarchical WSNs is by no means a trivial task. On the one hand, PKC-based digital signatures consume too much energy to be practical in WSNs. On the other hand, secret key cryptography based mechanisms cannot be directly applied to broadcast authentication, since otherwise a compromised receiver can easily forge any messages from the sender [28]. Perrig *et al.* [2] proposed a broadcast authentication mechanism named μTESLA. Many techniques are used to extend the capabilities of μTESLA in [28, 29, 30]. The scheme in [29] talked about how to tailor μTESLA to local broadcast authentication. The scheme in [28] overcomes the length limit of the hash chain. The scheme in [30] extends μTESLA to support a multicast scenario.

Secure routing

Routing protocols, to some extent, have received maximum attention from the researchers both in wired networks and wireless networks. Therefore, most current research primarily focuses on providing the most energy efficient routing scheme. In WSNs, the in-network processing characteristic is one of the challenges of designing a routing protocol. Intermediate nodes have access to the data. Once one of these intermediate nodes is compromised, it can eavesdrop and even modify the data, thus threatening the entire network. Another challenge is that it is very easy for a single node to disrupt the entire routing protocol by simply disrupting the route discovery process [20]. So, the routing protocols in WSNs should provide not only reliable delivery, but also security services.

The routing security in WSNs summarizes attacks against the current proposed routing protocols and discusses countermeasures and design considerations for secure routing protocols. The attacks can be classified into two categories: (1) Those that try to manipulate user data directly or (2) Those that try to affect the underlying routing topology. Both kinds of attacks can consume valuable resources to cause a DoS attack. The author claimed that it is unlikely to find effective countermeasures against those attacks after the design of a protocol has been completed. So, it is crucial to consider security issues at the beginning of routing protocol design [24].

Intrusion detection

Traditional intrusion detection methods fall into two main categories: Anomaly-based Intrusion Detection (AID) and Misuse Intrusion Detection (MID). The AID technique assumes that intruders will demonstrate unusual system behavior compared to the legitimate nodes and any unusual network behavior is an indication of an attack. With that in mind, a profile of the system in normal use is developed and used to evaluate the system when intruders emerge. The advantage of the AID system is that it is able to detect previously unknown attacks. However, the AID system has two obvious disadvantages. Firstly, it is susceptible to false positives since it is difficult to define normal system behavior. Secondly, the AID system has high computational cost when comparing the current system activity to the base profile. Such high computation cost can severely impact upon the longevity of the network.

The MID technique maintains a database of intrusion signatures. Using these signatures, the system can easily detect intrusions on the network. This approach is less prone to false positives but is unable to detect unknown attacks. The advantage of this technique is that it requires less computation in order to identify intruders as the comparison of network events with the available signatures is relatively low cost [31].

Even though effective intrusion detection systems are essential for WSN security, a perfect solution has not yet been devised.

Secure data aggregation

As WSNs continue to grow in size, so does the amount of data that the WSNs are capable of sensing. This data is collected by individual sensor nodes that have limited storage and sensing capabilities. In order to obtain meaningful information from this data, the raw stream of data must be securely processed first. This is typically done using a series of aggregators which are responsible for collecting the raw data from a subset of nodes and processing/aggregating the raw data from the nodes into more usable data. However, these aggregators are a single point of failure. In the event that an aggregation node is compromised, then all of the data delivered from the WSN to the control server may be forged. The end user may make an incorrect decision based on the forged data. Therefore, secure data aggregation techniques should be developed for WSNs.

Wagner analyzes the resilience of existing aggregation techniques in [32], and argues that current aggregation schemes were designed without consideration of security and that there are easy attacks against them. To date, a great number of secure data aggregation protocols have been proposed for WSNs. Wang *et al.* in [26] presented a taxonomy of secure data aggregation protocols, plaintext-based protocols and ciphertext-based data aggregation protocols. A conflict in data confidentiality and data aggregation exists in almost all of literature. Confidentiality requires the data to be transmitted after encryption, whereas data aggregation is usually done after decryption. Frequent encryption and decryption operations involve unacceptable computation overhead. An alternative method is to aggregate concealed data.

Secure localization

The location information of nodes in WSNs plays a important role in understanding the application environment. There are three visible advantages of knowing the location of sensor nodes [33]. First, location information is needed to identify the location of an event of interest, such as the location of an intruder. Second, location awareness facilitates numerous application services. Third, location information can assist in various system functionalities, such as geographical routing. Due to these advantages and specific characteristics of WSNs, it is natural that secure localization in WSNs has become a major focus of research in recent years.

Time synchronization

Time synchronization is an important component in all distributed systems, and WSNs are no exception. With the proliferation of WSNs, time synchronization in WSNs has attracted increasing attention in the last few years. In order to conserve power, an individual sensor's radio may be turned off for periods of time. Furthermore, sensors may wish to compute the end-to-end delay of a packet as it travels between two pairwise sensors. A more collaborative sensor network may require group synchronization for tracking applications, etc.

The authors in [34] presented an overview of the time synchronization problem in WSNs, defining the requirements, and various issues for designing synchronization algorithms for wireless sensor networks. The authors argue that time synchronization should be multi-modal, tiered, and tunable, so that it can satisfy the diverse needs of various sensor network applications.

Trust management

Trust management can solve some problems in WSNs that traditional cryptographic security mechanisms cannot deal with. For example, Trust mechanisms are effective in judging the quality and reliability of sensor nodes and wireless links, data aggregation reliability and correctness of aggregator nodes. However, it is not easy to build a good trust model within a sensor network given the resource constraints [20]. Many existing security mechanisms assume that a trust relationship between nodes exists in advance. This assumption ignores the independence of sensor nodes.

Trust establishment techniques in wireless networks even in ad hoc networks are not fresh. However these techniques cannot be applied directly to WSNs due to the capacity of sensor nodes. The specific techniques for trust management in sensor networks were proposed only recently. Ganeriwal *et al.* proposed a reputation-based framework for a high integrity sensor network in [35]. A beta reputation system is employed in this framework for reputation representation, updates, and integration. The trust model in [36] presents a method of location-centric isolation of nodes exhibiting misbehavior and trust-based routing in WSNs. The trust value is a function of the capacity of the cryptographic suite being used. If the trust value falls below a specific trust threshold, then the location of the node is considered insecure and the node is avoided when forwarding packets.

Trust management usually involves high computation overhead, so that building an efficient scheme for resource-constrained sensor network is a very challenging task.

1.3 Motivation of the Research

It is not easy to implement security defences in WSNs. One of the major obstacles in deploying security on WSNs is that the current WSNs have limited computation and communication capabilities and it is impossible to manually replace the battery due to the unattended nature and hazardous sensing of environments. The constraints make the provision of adequate security countermeasures even more difficult.

Key management is the provision made in a cryptography system design that is related to key generation, key distribution, key updating, and key revocation [37]. Key management is the essential building block for most cryptographic solutions. The proper management of cryptographic keys determines the effective use of cryptography for security. Key management is built on several cryptosystems and cryptographic primitives, such as PKC and MAC, and provides support to other security mechanisms, such as authentication and secure routing. The position of key management in a security architecture is displayed in Figure 1.3, part of which has been shown in [19].

However, the unique characteristics render the key management schemes of wired and general wireless networks ineffective for WSNs. With that in mind, some researchers have begun to safeguard WSNs with lightweight key management mechanisms. Compared with general key management schemes, lightweight key management schemes provide security support with reduced overhead and thus are more suitable for WSNs. Although considerable developments have been made in general key management, the research on lightweight key management is still in its infancy. Most of the available literature follow a routine whereby they firstly discover the potential attacks and then counteract the security problems with corresponding countermeasures. None of these solutions take into consideration the specificities of WSNs in practical application terms.

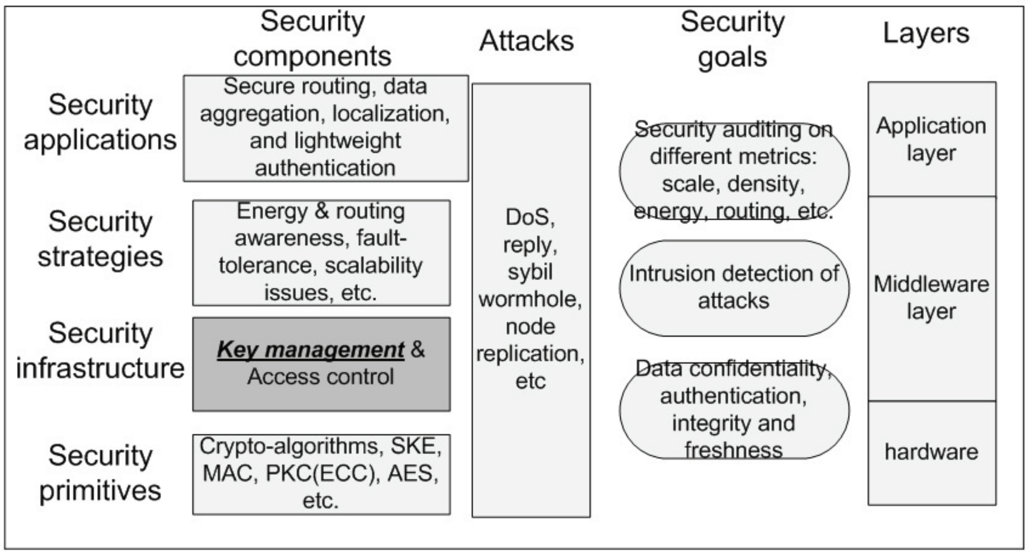

Security components
Attacks
Security goals
Layers
Security applications
Secure routing, data aggregation, localization, and lightweight authentication
Security strategies
Energy & routing awareness, fault-tolerance, scalability issues, etc.
Security infrastructure
Key management & Access control
Security primitives
Crypto-algorithms, SKE, MAC, PKC(ECC), AES, etc.
DoS, reply, sybil wormhole, node replication, etc
Security auditing on different metrics: scale, density, energy, routing, etc.
Intrusion detection of attacks
Data confidentiality, authentication, integrity and freshness
Application layer
Middleware layer
hardware

Figure 1.3: The position of key management in a security architecture

WSNs are application-specific networks. Except for some common features, a sensor network for a specific application has some unique features and correspondingly has some unique security requirements. The solution proposed for one particular application is unlikely to be readily applicable for another environment. Suppose a sensor network is deployed in the military surveillance environment and another in an agricultural base; the requirements of security should be different based on the resource that nodes possess and the risks they face. It is impossible to reach a one-fits-all key management solution. A key management mechanism would lose its practical value in applications when it is separated from the considerations of network-related issues [19]. Our designs are driven by specific applications and their security requirements; this principle makes the designed scheme more practical. In this work, we integrate prior research results and investigate the open problems from the following subcategories: establishment of different types of keys, implicit or lightweight authentication mechanism, self-healing key distribution mechanism, and PKC-based or hybrid key management for WSNs.

1.3.1 Support for Different Types of Communication

Most of the key management schemes for group communication in WSNs address only session key establishment and renewal. Almost all the key predistribution schemes consider only the establishment of pairwise keys between sensor nodes. There are still several unsolved problems in the key predistribution field. For example, path keys do not exclusively belong to two end nodes. All the intermediate nodes on the path know the pairwise keys forwarded by them; the compromise of a limited number of nodes may lead to the exposure of the whole key pool. Most importantly, there is no way to establish different types of keys for different kinds of communication requirements. A single key cannot meet different communication requirements of WSNs, especially of HWSNs.

1.3.2 Embedding Lightweight/Implicit Authentication to Key Management

Existing authentication schemes have at least one or several of the following shortcomings: high computation or communication overhead, no resilience to nodes compromised, delayed authentication, loose or even strict time synchronization, and absence of scalability. Lightweight authentication and PKC-based authentication are ideal alternative options for WSNs, while the unique characteristics of WSNs should be given full consideration. Current literature mainly talks about authentication of communication content which happens after key establishment. In fact, authentication is also vital during the process of key establishment.

1.3.3 Development of Secure and Efficient Self-healing Mechanisms

Due to the nature of WSNs, they are inherently susceptible to packets loss and nodes failure. Much current research on group management has been dedicated to minimizing the storage, computational and communication overhead to meet scalability. However, little attention has been paid to the robustness of these protocols. Most of the current key management protocols are not designed with the capability of tolerating failures. However, such failures may block the normal run of key management protocols which are the building blocks of the whole security architecture. Self-healing key distribution is an ideal mechanism for dealing with packets loss in session key distribution. However some problems, such as robustness inconsistence, still remain unsolved. The existing countermeasures against node failure either make stringent assumption, or are too costly to be feasible. The node failure tolerance property of pairwise key establishment protocols has not been addressed to date. How to maintain a trade-off between fault-tolerance and redundancy is also an open problem.

1.3.4 Development of PKC-based/Hybrid Key Management Mechanisms

Although much literature has demonstrated the feasibility of ECC-based public key generation from the hardware or software point of view, rarely do current works proposed complete key management infrastructure using PKC. Traditional key management schemes for wired network cannot be directly transplanted to HWSNs given the unique attributes of the latter. Public Key Cryptography has incomparable advantages over Symmetric Key Cryptography in key management and authentication. Nevertheless, great effort has to be applied to make PKC-based key management a reality in HWSNs.

1.4 Scope of the Research

In the previous sections, we highlighted the advantages of HWSNs over DWSNs. Most importantly, the standards which imply the current trends of WSNs, IEEE 802.15.4b [38] and the ZigBee "enhanced" standard [13], support cluster-based operations. The two standards give group support a high priority. In addition, the new target broadcast feature that can reach a specific subset of nodes is important to key management mechanism selection because it adds a new functional requirement. Therefore, those purely random or pairwise keying scheme like [3] and [39] would not be commercially viable. With that in mind, this thesis focuses on proposing key management schemes for HWSNs. The proposed keying schemes offer broadcast or multicast abilities and thus are much more compatible with industry trends.

It should be noted that the methodology proposed in this thesis relates purely to the domain of key management. It encompasses cryptographic algorithms and protocols which focus on issues involving the management of cryptographic keys: their generation, distribution, transmission and destruction. Related topic, including algorithm selection and appropriate key size, cryptographic policy, and cryptographic module selection, are also included in this thesis. This thesis does not address the implementation details for key generation. We assume that the key materials are generated utilizing existing approaches presented in the literature.

Key management is not a separate component of security architecture for WSNs. Key management provides a security infrastructure for other security services. At the same time, it relies on other security services. Key management and other security services together make up the security architecture of wireless sensor networks (shown in Figure 1.3). A comprehensive consideration is compulsory when designing a key management scheme for a sensor network. However, in our solutions, we do not assume the existence of too many other security components, such as Intrusion Diction Systems (IDS) and secure routing.

1.5 Objectives of the Thesis

This thesis focuses on key management from four perspectives, including: the establishment of different types of keys, integrating lightweight authentication, involving self-healing property, and exploring the feasibility of PKC-based or hybrid key management protocols. The aim of this thesis is to design, specify, and analyze key management frameworks in HWSNs.

The main research objectives of this thesis are as follows:

1. We present the security requirements and challenges of WSNs. We highlight the properties that directly affect key management protocols. The information includes the densities and the size of network, the available hardware and software resources, and some special knowledge that can be used in a particular real-time scenario, such as location information. We also illustrate the performance indicators that can be used to assess specific key management schemes. These performance parameters allow key management schemes for the same architecture and parameter settings to be directly compared. These parameters include the desired key connectivity, scalability, resilience, and

computation/communication overheads. Based on the acquired information and with a clear understanding of the challenges involved in implementing security mechanisms, one can define a threat model and then clarify the security requirements. The security model is defined according to security requirements and will work as a performance metrics to evaluate the proposed key management schemes. The development of security metrics, measurements, and evaluation of approaches are of great importance in order to establish a scientific methodology for the field of entire wireless network security research.

2. We propose a methodology whereby different types of keys, including network key, cluster key, pairwise keys, can be established to support different communication modes. We also fix the security problems in the famous Localized Encryption and Authentication Protocol (LEAP) [29] which supports the establishment of four types of keys. Our mechanism disperses the damage resulting from the disclosure of the initial key.

3. We propose a methodology which combines implicit authentication with the proposed schemes. Such authentication methodology produces lightweight authentication without needing time synchronization.

4. We propose a methodology whereby nodes can establish keys over an unreliable channel. By subtly introducing random numbers, the proposed scheme can resist collusion attack while existing schemes can only maintain forward and backward secrecy. We also define a security model and explore the technical details of mutual-healing key distribution.

5. We propose a methodology which takes full advantage of the heterogeneity of two-tiered nodes in HWSNs. It uses PKC-based cryptosystem (ECC) in cluster head level communication. Our scheme is very flexible and can be used in a group communication dominated environment and peer communication dominated environment by selecting the different keying algorithm at a lower level.

1.6 Significance of the Research

1.6.1 Social and Economic Significance

The significance of the proposed research, from a social and economic point of view, includes:

1. Saving resources. Current key management schemes consume valuable resources like memory, network bandwidth, and power. The proposed approaches are efficient and can reduce the consumption of resources and thus can be used in large-scale HWSNs.
2. Low-cost network maintenance. The introduced fault-tolerant property of key management schemes makes the establishment of keys over an unreliable network possible. Hence, it can address the network maintenance problem without increasing hardware costs.
3. Provision of lightweight key management. The applications of HWSNs calls for corresponding security mechanisms. The lack of matching lightweight security mechanism hinders the wide applications of HWSNs. However, the challenges of HWSNs render most of security mechanism infeasible. The key management schemes in this research pave the way of access control and establishment of secure communication channels in HWSNs.

1.6.2 Scientific Significance

The significance of the proposed research, from a scientific point of view, includes:

1. proposing robust key management schemes for HWSNs. Each of them has some advantages over existing key management schemes of the same kind. More specifically, this thesis proposes a robust key management scheme which fixes security problems of LEAP and can be seamlessly used to enhance key management services of ZigBee networks; this thesis proposes a hash chain based key predistribution scheme which reduces storage overhead while increases key connectivity, and is resilient to node capture attack; this thesis proposes a unified approach for collusion attack against hash chain based self-healing key distribution schemes and develops a fully anti-collusive self-healing key distribution scheme.

2. proposing a formal definition of mutual-healing key distribution and developing a practical mutual-healing key distribution for WSNs. It is the first time to develop mutual-healing key distribution and explore its technical details.
3. integrating implicit authentication mechanism and PKC-based authentication mechanism with self-healing key distribution schemes to filter the false broadcasts out. It is the first time to realize authenticated self-healing key distribution.
4. proposing a practical hybrid key management solution for HWSNs, whereby the entities can take full advantage of their heterogeneity and merits of Symmetric Key Cryptography and Public Key Cryptography.

1.7 The Structure of the Thesis

As mentioned earlier, in this thesis we propose to develop a methodology for key management whereby keys can be distributed and updated with lightweight overhead. In the case of there being packets loss during transmission, the key management mechanism can proceed in a fault-tolerant manner. In order to achieve the above objectives, this thesis is organized as follows:

Part I: Research background, problem definition, and major contributions

- In Chapter 1 we introduce the background on sensor network security. We outline the motivation, the significance, the scope, and main objectives of this research. Finally, we present the structure of the thesis.
- In Chapter 2, we present an in-depth survey of the state-of-the-art techniques for key management in WSNs with a focus on five aspects. We review, compare, and contrast the proposed techniques in terms of each aspect and summarize the weaknesses and existing problems in each aspect. The literature review in this chapter provides the foundation for the problem definition in the following chapter.
- In Chapter 3, we formally present the problem definition. We then break down the problem definition into different research issues. We also define the terms and terminologies that will be used while solving each issue. Finally in this chapter, we overview the solutions.

Part II: Symmetric Key Cryptography based solutions

- In Chapter 4, we analyze the security problem of LEAP [29]. The security is threatened by attacks launched after the initialization phase. We then propose a key management scheme which disperses the damage resulting from the disclosure of the initial key.
- In Chapter 5, we propose a methodology whereby nodes in a hierarchical network can establish five types of keys. The methodology adopts a keyed hash chain based approach. A new cluster mechanism is used to improve the probability of key sharing between sensors and their cluster heads. Unlike existing schemes where a node capture attack might lead to the disclosure of several key chains, our method can avoid this drawback without storing network-wide generating keys in low-cost sensors.
- In Chapter 6, we propose a unified approach for collusion attack against hash chain based self-healing key distributions schemes. Then we propose a fully anti-collusive mechanism to defend against collusion attack in hash chain based self-healing key distribution schemes.
- In Chapter 7, we propose an implicitly authenticated self-healing key distribution scheme. The scheme is based on a hash binary tree. Implication authentication is subtly introduced to detect tamper attack against broadcast messages during transmission. The scheme achieves the same security level of existing schemes with reduced storage and computation overhead.

Part III: PKC-based solutions

In Chapter 8, we propose a methodology for mutual-healing key distribution which enables a node in a WSN to recover its new session key although its last broadcast message was lost. A formal definition for mutual-healing key distribution is proposed. The methodology is based on bilinear pairings and is collusion-free for any coalition of non-authorized nodes. We also propose an authenticated self-healing key distribution scheme.

Part IV: A hybrid solution

In Chapter 9, we propose a methodology which takes full advantage of the heterogeneity of nodes in hierarchical networks. It uses Elliptic Curve Cryptosystem in cluster head level communication and an improved one-way function tree based key management in general nodes lever. The proposed scheme has good flexibility and can be used in a group communication dominated environment and peer communication dominated environment by selecting the different keying algorithm at a lower level.

Part V: Conclusion and future work

In Chapter 10, we conclude the thesis by summarizing the work developed in this thesis and then identify the potential for further work.

1.8 Conclusion

The proliferation of WSNs has driven the research into sensor network security, with key management being the cornerstone of the whole research. In this chapter, we provided a comprehensive overview of WSN security, including, the application, classification, constraints, vulnerabilities, security requirements, and current hot research topics in the field of sensor network security. The overview helps in understanding of the characteristics of WSNs and the importance of key management research. Then we introduced the motivation that drives us to carry out this research, followed by a description of the significance of our work. The scope and objectives of this study were pointed out and discussed and the structure of the thesis was presented.

2

Literature Review

2.1 Introduction

We highlighted that security is a crucial issue in wireless sensor networks due to inherent network characteristics in Chapter 1. In this chapter, we examine and present an overview of existing literature on key management techniques. Substantial progress has been made on providing a sound, practical basis for a number of problems that are associated with key management for sensor network security. A number of key management techniques have been documented in the literature. In the following sections, we will discuss the work that has been previously undertaken to solve some of the issues related to key management for WSNs. The research areas investigated by this dissertation can be grouped into four categories:

1. Key management scheme for WSNs
    - Symmetric Key Cryptography based key management
    - PKC-based key management
    - Hybrid key management
2. Establishment of different types of keys
3. Authentication mechanism in key management
4. Self-healing key distribution mechanism

Our research focuses on key management in hierarchical wireless sensor networks. Different types of keys have to be established in order to meet different types of communication. Therefore, the problem of establishment of different types of keys is especially discussed in this research. We do not consider separate authentication and self-healing mechanisms. Our research lays emphasis on integration of these two properties with key management. Both Symmetric Key Cryptography and Public Key Cryptography are used to realize these two mechanisms.

2.2 Key Management Schemes for WSNs

The majority of the literature is based on the generic characteristics of WSNs. The researchers were devoted to discovering numerous vulnerabilities and addressing the corresponding countermeasures. The countermeasures mainly rely on Symmetric Key Cryptography, Public Key Cryptography, or a hybrid of these. This motivates us to review the state-of-the-art key management techniques and categorize the schemes according to: Symmetric Key Cryptography based schemes, PKC-based schemes, and hybrid schemes. Figure 2.1 presents a collection of representative key management solutions proposed in the literature for wireless sensor networks.

2.2.1 Symmetric Key Cryptography Based Key Management Schemes

Existing literature [40, 41] thoroughly examine the state-of-the-art techniques for key management for WSNs. A key predistribution mechanism provides a nice tradeoff between storage overhead and processing power, and thus is considered as the most suitable mechanism for WSNs. Solutions to key predistribution can be classified into the following five types according to the mathematical construction on which they are based:

1. Random graph theory based key predistribution schemes

2. Polynomial-based key predistribution schemes

3. Matrix-based key predistribution schemes

4. Combinatorial design based key predistribution schemes

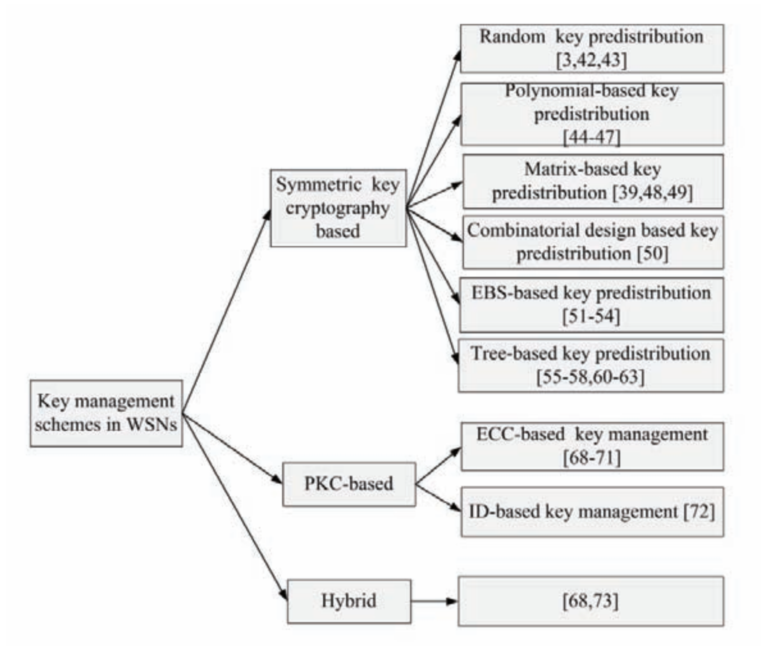

Key management schemes in WSNs
Symmetric key cryptography based
PKC-based
Hybrid
Random key predistribution [3,42,43]
Polynomial-based key predistribution [44-47]
Matrix-based key predistribution [39,48,49]
Combinatorial design based key predistribution [50]
EBS-based key predistribution [51-54]
Tree-based key predistribution [55-58,60-63]
ECC-based key management [68-71]
ID-based key management [72]
[68,73]

Figure 2.1: A collection of representative key management solutions for WSNs

5. Exclusion Basis System (EBS) based key predistribution schemes.

In this subsection, we introduce the representative schemes in each category. In addition, we survey tree-based key management schemes as tree models are widely used in key management.

Random key predistribution schemes

Eschenauer and Gligor proposed the original random key predistribution scheme for WSNs that relies on probabilistic key sharing among nodes of a random graph in [3]. The scheme includes three phases: key predistribution, shared key discovery, and path key establishment. In the key predistribution phase, a large key pool of P keys and their identities are generated. Each sensor randomly draws k ($k \ll P$) keys from the key pool without replacement. These k keys and their identities form the key-chain for a sensor node. In the shared key discovery phase, two neighbor nodes exchange and compare the list of identities of keys in their key chains. If two sensors have some keys in common, they can set up a secure link directly. Otherwise, the path key establishment procedure needs to be triggered to set up a link between two neighbors. If a node is compromised, the base station broadcasts a message to all other nodes to revoke the compromised node's key chain. Re-keying follows the same procedure as revocation. The probability that any two nodes share at least one common key is $p = 1 - ((P-k)!)^2/((P-2k)!P!)$. Eschenauer *et al.* showed that if p reaches a critical value, the connectivity of the entire network can be obtained with a high probability. The advantage of the random key predistribution scheme is that there is no computational overhead to generate pairwise keys between sensor nodes. The main shortcoming of the scheme in [3] is that a large number of keys could be disclosed by compromising a few selected nodes. For one compromised node, the attacker has a probability of k/P to successfully attack a link. Because $k \ll P$, a compromised node does not affect the security of the scheme too much.

The basic scheme [3] that two nodes share a unique key for establishing a secure communication link was further improved by Chan *et al.* in [42] from two different aspects. Two variations are proposed: q ($q \geq 2$)-composite scheme and the multi-path key reinforcement scheme. In the q-composite scheme, two neighboring nodes can build a secure link only if they have at least q common keys. They may form a new communication link using the hash of all the q keys, i.e., $K = hash(k_1 \parallel k_2 \parallel \ldots \parallel k_q)$.

The multi-path key reinforcement scheme is used for rekeying. Suppose nodes A and B share a key k. Since k may be residing in the key ring memory of some other nodes in the network, the security of k is jeopardized if node A or node B is captured. Thus, it is necessary to update the shared key k between nodes A and B instead of using a key in the key pool. In the multi-path key reinforcement scheme, if node A updates the shared key with node B, all possible disjointed paths to node B must be used. Assume that there are h such disjointed paths from node A to node B. Then node A generates h random values $(g_1, g_2, \ldots, g_h)$ with the same size of the shared key and sends one down each available disjointed path to node B. When node B has received all h random values, it computes the new key $k' = k \oplus g_1 \oplus g_2 \oplus \ldots \oplus g_h$ where k is the original key. Node A does the same computation. The new key k' is used to secure communication link. An adversary in this case has to eavesdrop on all the disjoint paths between node A and node B if he wants to reconstruct the new shared key k'.

The q-composite scheme makes it more difficult to compromise a node. It is secure under small scale attacks while being vulnerable under large scale attacks. The scheme does not satisfy scalability requirements. Also, it is hard to choose an optimal value for q. The multi-path key reinforcement scheme increases the resilience of the network against node capture at the cost of more CPU use and power consumption. It can be used for applications where security is more of a concern than is bandwidth or power drain.

Du *et al.* combined a random key predistribution scheme with node deployment knowledge to improve resilience [43]. In this scheme, deployment knowledge is modeled using non-uniform probability density functions. Nodes are grouped based on their deployment knowledge and each group has a small key pool. As nodes from the same group are more likely to be neighbors, their keys can choose from the corresponding key pool. As the size of key pool becomes smaller, both local connectivity and global connectivity increase. As each node has fewer keys, resistance to node capture attack and network overhead for pairwise key establishment are improved.

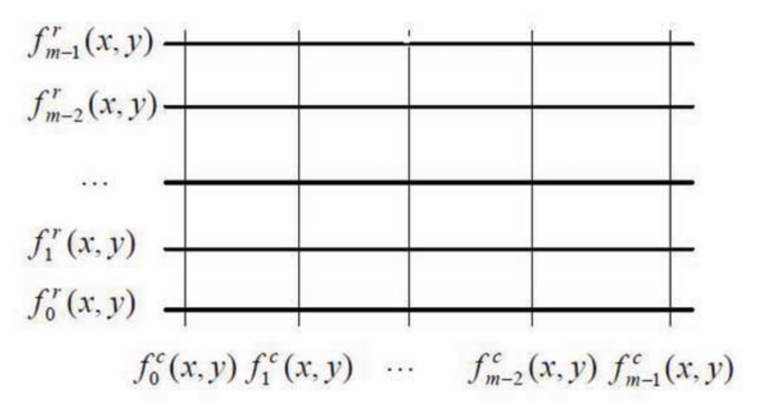
Polynomial-based key predistribution schemes

Blundo *et al.* proposed a polynomial-based key predistribution scheme in [44]. In this polynomial-based scheme, the setup server randomly generates a bivariate t-degree polynomial $f(x, y) = \sum_{i,j=0}^{t} a_{i,j}x^i y^j$ over a finite field F_q, where q is a prime number that is large enough to accommodate a cryptographic key, such that the equation has the property $f(x, y) = f(y, x)$. The setup server then computes a polynomial share of the equation for each node in the network; e.g., a node with ID i receives $f(i, y)$ and a node with ID j receives $f(j, y)$. When node i and node j want to establish a key, node i can compute the same key $f(i, j)$ by evaluating $f(i, y)$ at point j, and node j can compute the same key $f(j, i) = f(i, j)$ by evaluating $f(j, y)$ at point i. Each node in this approach has to store a t-degree polynomial $f(i, x)$, which occupies $(t+1)logq$ storage space. This scheme is secure and reveals nothing for less than t compromised nodes. t value depends upon the memory available of sensors. There is no communication overhead during the pairwise key establishment process. The computation overhead is incurred from the calculation of the polynomial at the ID of the other sensor node.

Liu *et al.* extended the scheme in [44] to a polynomial pool based key predistribution scheme in [45]. Instead using a single t-degree polynomial in [44], the setup server generates a pool of polynomials over F_p. Each polynomial has an ID. The setup server picks up a subset of these polynomial shares and places it in each node. In the key discovery stage, each node finds a node with which it shares the same bivariate polynomial and calculates the shared key as that in [44]. After the key discovery stage, if two nodes do not find a common polynomial, similarly they can establish a path key. The polynomial pool-based key predistribution scheme allows for network growth after deployment. The collision resistance is constrained by t. More than t compromised polynomials would lead to whole network compromise. The experiment shows that the scheme has a better resilience property than [3] and q-composite scheme in [42] if some nodes are compromised. However, when the number of compromised nodes is higher than a threshold (for example 60% nodes are compromised), the number of compromised paths in [45] is higher than that of [3] and q-composite scheme in [42].

Liu *et al.* [45] further developed a random subset assignment scheme which is an extension of the polynomial pool based scheme. In this scheme, each node is assigned with a small set of random polynomials rather than keys as that in [3]. The following stages are similar to the polynomial pool-based scheme. The probability of two sensors sharing the same bivariate polynomial is the same as the probability of the two sharing a common key in [3]. While in this scheme, the established pairwise keys generated by any two nodes are unique. If no more than t shares of the same polynomial have been disclosed, the pairwise keys computed from the polynomial are secure.

Liu *et al.* then proposed a grid-based key predistribution scheme [45]. Suppose a sensor network has at most N nodes, the server constructs a $m \times m$ grid with a set of $2m$ polynomials $\{f_i^c(x,y), f_i^r(x,y)\}_{i=0,\dots,m-1}$, where $m = \lceil\sqrt{N}\rceil$. Each column i in the grid is associated with a polynomial $f_i^c(x,y)$, and each row i is associated with a polynomial $f_i^r(x,y)$. The server assigns each sensor in the network to a unique intersection in this grid. For the sensor at the coordinate (i,j), the server assigns polynomial shares of $f_i^c(x,y)$ and $f_j^r(x,y)$ to the sensor, as shown in the Figure 2.2. After deployment, the nodes in the same row or column can establish pairwise keys directly. Therefore, each node can establish a pairwise key



$f^r_{m-1}(x,y)$
$f^r_{m-2}(x,y)$
$\cdots$
$f^r_1(x,y)$
$f^r_0(x,y)$
$f^c_0(x,y)$ $f^c_1(x,y)$ $\cdots$ $f^c_{m-2}(x,y)$ $f^c_{m-1}(x,y)$

Figure 2.2: Grid-based key predistribution scheme

with $2(m-1)$ other nodes directly. According to the path discovery method, any two nodes in different rows and columns can establish a pairwise key with the help of intermediate nodes. The grid-based scheme offers all the attractive properties of the polynomial pool based scheme and guarantees that the nodes can establish a pairwise key with each other and they can judge which polynomial should be used for key establishment. This scheme has the same resilience as does the basic scheme [3] and the q-composite scheme [42] with less communication and computation overhead; this scheme offers the same degree of security as the random pairwise scheme when the same number of sensors and storage overhead are considered. The grid-based approach offers a greater probability of key establishment than any other scheme does when there are no compromised nodes, as well as a greater probability of key establishment when some nodes are compromised.

The grid-based scheme was further extended to a hypercube-based scheme in [46]. Suppose a network has N nodes and key predistribution is performed in a n-dimensional hypercube. Each node is assigned with an exclusive coordinate in the hypercube. The setup server computes $m^{n-1}(m = \lceil\sqrt{N}\rceil)$ bivariate polynomials for each dimension and assigns each node a set of polynomials according to its coordinate. i.e. a node at coordinate $(j_1, \ldots, j_n)$ is assigned with polynomials $\{f^1_{j_1,\ldots,j_n}(x,y), \ldots, f^n_{j_1,\ldots,j_n}(x,y)\}$. If nodes i and j share a common polynomial, they can make a direct connection and compute a pairwise key to communicate. If the two nodes do not share a common polynomial, they have to use the path discovery method to compute an indirect key [46]. The storage cost at most is

$n(t+1)logq + ntl(l = \lceil log_2 m \rceil)$ bits including n polynomials and t number of compromised node IDs. The computation cost is $2(L+1)$ polynomial evaluations where L equals the key path [46]. There is no communication overhead if the two nodes have a common polynomial.

Similarly to the scheme in [43], many key predistribution schemes can be combined with deployment knowledge. Liu *et al.* proposed two location-aware key predistribution schemes in [47]. They firstly integrated the expected location with the random pairwise keys scheme [42]. Then they integrated the expected location information with the random subset assignment scheme [45]. By exploiting the location information, these two schemes have a higher probability of establishing pairwise keys between neighbor nodes with smaller storage overhead than that of the original schemes. They also provide better resistance against node compromise attacks and greatly improve the performance of key predistribution.

Matrix-based key predistribution schemes

In the original matrix based key predistribution scheme [48], the base station first constructs a $(\lambda+1) \times N$ public matrix G over a finite field $GF(q)$, where N is the size of the network. An example of matrix G is as follows:

$$\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ s & s^2 & s^3 & \dots & s^N \\ s^2 & (s^2)^2 & (s^3)^2 & \dots & (s^N)^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s^\lambda & (s^2)^\lambda & (s^3)^\lambda & \dots & (s^N)^\lambda \end{pmatrix}.$$

The base station also creates a random symmetric matrix $D_{(\lambda+1)\times(\lambda+1)}$ over $GF(q)$ and calculates $A_{N\times(\lambda+1)} = (DG)^T$. Both $D_{(\lambda+1)\times\lambda+1)}$ and $A_{N\times(\lambda+1)}$ should be kept secret. A matrix $K_{N\times N} = (DG)^T G$ stores all pairwise keys of a group of N nodes, where each element k_{ij} is the key of node i for securing the link with node j. We can verify that $K_{N\times N}$ is a symmetric matrix as follows:

$$K_{N\times N} = (DG)^T G = G^T D^T G = G^T DG = K^T_{N\times N}.$$

If each node i stores the ith row of secret matrix and the ith column of public matrix G, each pair of nodes i and j can individually compute a pairwise key $k_{ij} =$

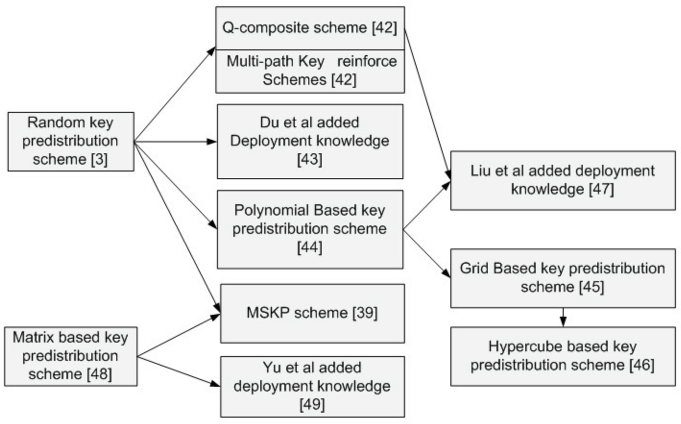

Q-composite scheme [42]
Multi-path Key reinforce Schemes [42]
Random key predistribution scheme [3]
Du et al added Deployment knowledge [43]
Polynomial Based key predistribution scheme [44]
Liu et al added deployment knowledge [47]
Grid Based key predistribution scheme [45]
MSKP scheme [39]
Matrix based key predistribution scheme [48]
Yu et al added deployment knowledge [49]
Hypercube based key predistribution scheme [46]

Figure 2.3: The relationship of several representative key predistribution schemes

k_{ji} by exchanging only their columns in plain text because the key is a product of their own row and the column of another's. This scheme guarantees a complete key graph, that is, any pair of nodes in the network is able to establish a pairwise key. This scheme has λ-security. In other words, if no more than λ nodes are compromised, the entire secret matrix is perfectly secure.

Du *et al.* combined Blom's scheme [48] with the random key distribution scheme [3] to a multiple-space key predistribution (MSKP)scheme [39]. A key pool with ω symmetric matrices $D_1, \ldots, D_\omega$ of size $(\lambda + 1) \times (\lambda + 1)$ is generated by the base station. Each node is assigned with G and τ distinct matrices D. After deployment, if any pair of nodes discovers that they have a common space, then they follow Blom's scheme to establish a pairwise key. The MSKP scheme is scalable and flexible. It also achieves better resilience against node capture while it reduces full connectivity to a connected graph. However, in this scheme, each node has to store a lot of key materials in order to ensure a connected network. This produces a heavy storage overhead to memory-constrained nodes. The combination of Blom's scheme with deployment knowledge was proposed in [49].

We summarize the relationship of the reviewed key predistribution schemes in Figure 2.3.

Combinatorial design based key predistribution schemes

Camtepe applied combinatorial design theory to deterministic key distribution for WSNs in [50]. Assume that a network has N sensor nodes. A finite projective plane with order q (q is a prime and satisfies $q^2 + q + 1 \geq N$) is a symmetric Balanced Incomplete Block Design (BIBD). It supports a network with q^2+q+1 nodes. $b \geq N$ blocks needs to be constructed by using a set S with $\mid S \mid= v = b = q^2 + q + 1$ distinct random key and each block can be looked as a key chain with $k = q + 1$ keys. The symmetric design guarantees that any pair of key chains has at least $\lambda = 1$ key in common. That is, the probability of key connectivity of any pair of sensors is 1. However, the prime q cannot be used for any network size. For example, when $N > q^2 + q + 1$, a smaller q' must be chosen for the network. Otherwise, too big a q causes the size of key chains to dramatically increase. Once the size of the key chain is larger than the memory of the sensors, the scheme is infeasible. Generalized Quadrangles (GQ) support different network size. For example, $GQ(q, q)$, $GQ(q, q^2)$ and $GQ(q^2, q^3)$ support networks with $O(q^3)$, $O(q^5)$ and $O(q^8)$ respectively. However, the prime q is not easy to find. Camtepe combined symmetric BIBD and GQ to design a hybrid key predistribution scheme. The hybrid construction combines a deterministic core design with probabilistic extensions to achieve key distribution for any network size. They generate $q^2+q+1 < N$ key chains with BIBD or GQ. Then they select uniformly at random remaining $N-(q^2+q+1)$ key chains among ($k = q + 1$)-subsets of the Complementary Design of BIBD or GQ. These $N-(q^2+q+1)$ key chains and q^2+q+1 key chains generated in the first step together construct N key chains. The hybrid key predistribution scheme improves the scalability and resilience of networks, while it does not guarantee that the key share probability is 1. The analysis indicates that such a deterministic approach has advantages over a randomized one [3] in two aspects. Firstly, it increases the probability that two nodes share a key, and secondly, it decreases the average key-path length while providing scalability with hybrid approaches.

EBS-based key predistribution schemes

Eltoweissy *et al.* in [51] proposed a low energy hierarchical key management for WSNs based on an Exclusion Basis System, which is an abstraction of the problem of constructing a logical structure for key management. The definition of an EBS is as follows:

Definition 2.1. [52] Let n, k and m be positive integers, such that $1 < k, m < n$. An Exclusion Basis System of dimension (n, k, m), denoted by $EBS(n, k, m)$, is a collection Γ of subsets of $[1, n] = \{1, 2, \ldots, n\}$ such that for every integer $t \in [1, n]$ the following two properties hold:

1. t is in at most k subsets of Γ, and
2. there are exactly m subsets, say $A_1, A_2, \ldots, A_m$, in Γ such that $\cup_{i=1}^{m} A_i$ is $[1, n] - \{t\}$. (That is, each element t is excluded by a union of exactly m subsets in Γ.)

In the $EBS(n, k, m)$ key distribution system, n is the size of the network (number of members), k is the number of keys stored by each group member, and m is the number of rekey messages. A canonical enumeration of all possible ways to form subsets of k objects from a set of $k+m$ objects is employed in the construction of $EBS(n, k, m)$ for feasible values of n, k, and m. In the selected enumeration method, each element of the sequence is a bit string of length $k + m$ where a 1 in the ith position of a string means that object i is included in that subset, for any i $(1 \leq i \leq k + m)$.

For any k and m, let $Canonical(k, m)$ be the canonical enumeration of all $C(k + m, k)$ ways to form a subset of k elements from a set of $k + m$ objects. The sequence of bit strings in $Canonical(k, m)$ can form a canonical matrix A whose $C(k + m, k)$ columns are the successive bit strings of $k + m$ length, each with k bits 1. Table 2.1 displays a canonical matrix A for $EBS(10, 3, 2)$. There are $C(5, 3)$ ways to form a subset of 3 keys from 5 keys.

Table 2.1: The canonical matrix for $EBS(10, 3, 2)$.

	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9
K_1	1	1	1	1	1	1	0	0	0	0
K_2	1	1	1	0	0	0	1	1	1	0
K_3	1	0	0	1	1	0	1	1	0	1
K_4	0	1	0	1	0	1	1	0	1	1
K_4	0	0	1	0	1	1	0	1	1	1

The canonical matrix A serves rekeying. Suppose M_4 in $EBS(10, 3, 2)$ in Table 2.1 is compromised. Since M_4 possesses keys K_1, K_3, and K_5 and session key S, these keys need updating. At this point, only $K_2 \cup K_4$ is unknown by M_4. Therefore, $K_2 \cup K_4$ can be used for rekeying. The following messages will be generated for rekeying:

$$Message \quad 1 : E(K_2(S', E(K_1(K_1')), E(K_3(K_3')), E(K_5(K_5'))))$$

$$Message \quad 2 : E(K_4(S', E(K_1(K_1')), E(K_3(K_3')), E(K_5(K_5'))))$$

where $E()$ is an encryption function and S' is the new session key. In this way, the compromised keys are renewed and the compromised node M_4 is evicted.

The EBS-based key management scheme uses symmetric encryption and rekeying to support current, forward and backward secrecy. An important contribution is that it yields optimal results with respect to the parameters n, k, and m. The solution allows for the trade-offs between the number of keys per group member, k, and the number of rekeying messages. It also separates administrative keys used for key management from communication keys used for secure data transmission resulting in efficient key management and multi-granularity secure communications. In a large-scale group, the cost for key establishment and key renewing is usually relevant to the group size and subsequently becomes a performance bottleneck in achieving scalability. Another drawback of this scheme is that it does not resist collusion attack.

A new approach called SHELL [53] has been developed to resist collusion attack in the EBS-based key management system. SHELL is a hierarchical scheme and can be used for clustered sensor networks. SHELL uses post-deployment location information to assign keys and an EBS framework to perform rekeying in each cluster. The keys are stored by gateways and are assigned to each node by the corresponding gateway. Due to the location-based key distribution mechanism, the number of keys revealed by compromised nodes can be minimized.

Eltoweissy *et al.* presented a hierarchical key management scheme for WSNs called 'localized combinatorial keying' (LOCK) [54]. As shown in Figure 2.4, the

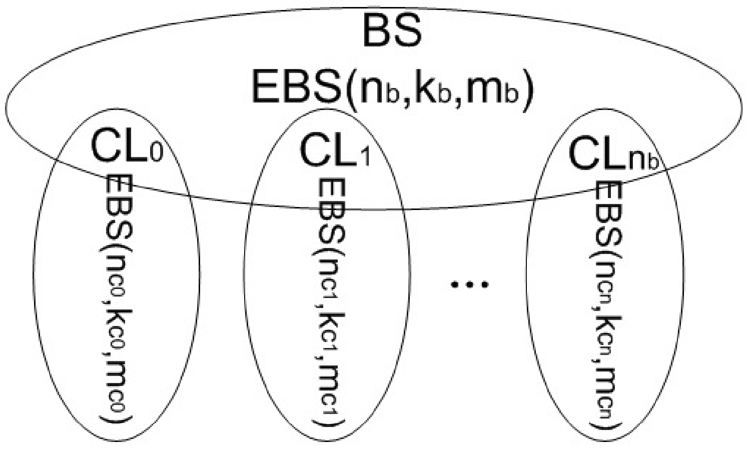


Figure 2.4: Physical network architecture of LOCK

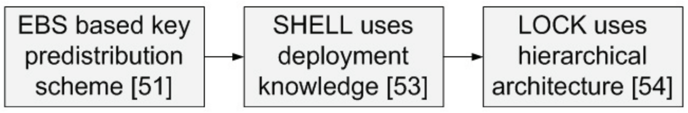

EBS based key predistribution scheme [51]
SHELL uses deployment knowledge [53]
LOCK uses hierarchical architecture [54]

Figure 2.5: The relationship of several EBS-based key predistribution schemes

physical network model of LOCK consists of three tiers, including a base station, cluster heads, and normal sensor nodes. LOCK employs two-layer EBS to perform localized rekeying to minimize communication overhead. However, LOCK uses neither location information to generate keys nor location-based key assignment as in SHELL. LOCK uses key polynomials to improve network resilience to collusion so that a compromised node does not affect the operations of other nodes in other clusters.

The relationship of EBS-based key predistribution schemes [51, 53, 54] is shown in Figure 2.5

Tree-based key management schemes

Two tree models are commonly used in key management. One is the Logical Key Hierarchy (LKH) model [55] which was proposed in 1999 and the other is the One-

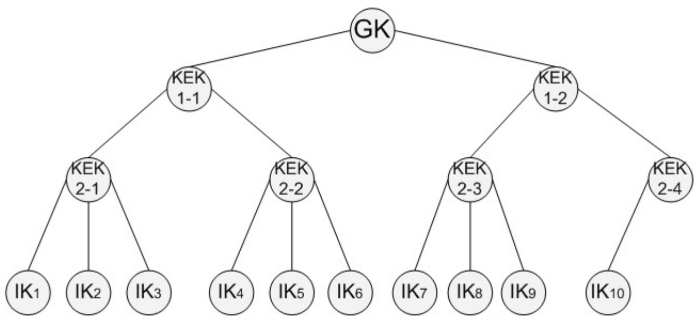


Figure 2.6: An example of the LKH model with 10 nodes

way Function Tree (OFT) model [56] which was proposed in 2003.

LKH model based key management schemes

The LKH model was introduced to efficiently update the group key when the membership changes for group communication. In the LKH model, a key tree is maintained at the Key Distribution Center (KDC) which is used for group key distribution and updating. The group key which is the root of the key tree is used to encrypt all messages broadcast in the group. Key Encryption Keys (KEKs) which are intermediate nodes of the key tree are used to update the root GK and other KEKs. Each leaf of the key tree is an Individual Key (IK) which associates with a group member. Each group member knows all the keys from its leaf node up to the root. As a result, each member in a group possesses three kinds of keys: its own IK, KEKs (on the path to the root), and a root GK. The set of these keys is referred to as the key path. Figure 2.6 is an example of LKH model consisted of 10 nodes. Without loss of generality, we take node 1 as an example. Node 1 stores key set $\{GK, KEK_{1-1}, KEK_{2-1}, IK_1\}$. In the LKH model, the number of keys required to update is $O(logN)$ (N is the number of members) which is the same as the number of keys on the key path.

The LKH model has been extended to a directed diffusion secure multicast scheme for WSNs (LKHW) in [57]. The directed diffusion is a data-centric, energy-efficient data dissemination technique for WSNs [58]. In directed diffusion, a query would be transformed into an interest and then diffused throughout the network nodes (called source nodes). The source node will activate its sensors to collect information with propagated interest. The dissemination technique also sets up certain gradients designed to draw data matching the interest. The collected data is then sent back to the sink along the reverse path of the interest propagation. The LKHW scheme take advantages of both LKH model and the directed diffusion technique. The LKHW protocol is described as follows:

1. The base station, which works as a key distribution center, sends out “interest about interest to join”;

2. The interested nodes reply with “interest to join”;

3. The base station supplies key set and then secures interest and data encrypted with the group key.

LKHW allows nodes to leave and join. When a node joins the network, a key set is generated for the new node based on the keys within the existing key hierarchy. When a node leaves the network, the key hierarchy is used to effectively re-establish keys for the nodes below the node that has left the group. In a multi-hop WSN where each node routes messages of other nodes, rekeying messages generated from the logical key tree should be forwarded to many irrelevant nodes before reaching their destinations. In other words, logical key tree-based schemes can incur heavy communication overhead in multi-hop WSN environments since the key tree structure does not reflect the underlying network topology [59]. LKHW is vulnerable to single point of failure attack as all keys logically root at the base station.

Lazos and Poovendran improved LKHW [57] in [60] and [61], respectively. In [60], a routing-aware tree is constructed in which the leaf nodes are assigned keys based on all relay nodes above them. As the scheme takes advantage of routing information to construct the key hierarchy, it is more energy-efficient than routing schemes that arbitrarily arrange nodes into a routing tree. In [61], the nodes are grouped into different clusters according to geographical location. The nodes within the same cluster are able to reach each other with one-hop communication. Using the cluster information, a key hierarchy is constructed in a manner similar to that proposed in [60].

OFT model-based key management schemes

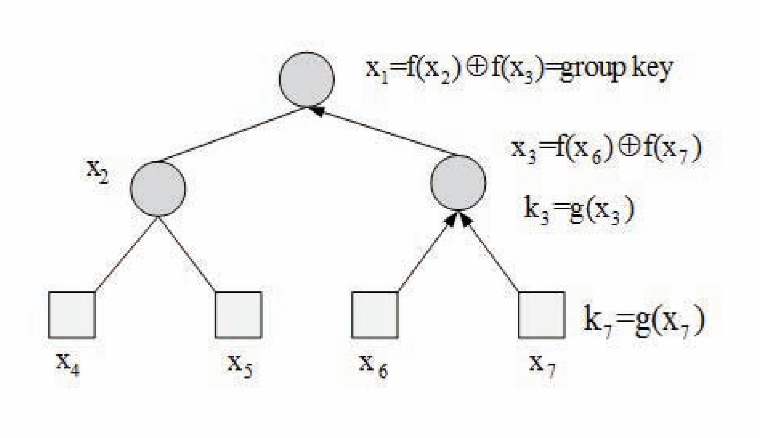


Figure 2.7: An OFT key tree and the functional relationships among its node secrets and node keys

In the OFT model, a group manager maintains a balanced binary tree for a group. The node secret of the root node is the group key. A one-way function key tree is computed in a bottom-up manner using the pseudo-random function f and a bitwise exclusive-or operation denoted by $\oplus$. Each internal node v in the key tree has a left child L and right child R. Each internal node v in the key tree is associated with a node secret x_v and a node key k_v. The node secret associated with v is computed from its children's secrets. Specifically, as shown in Figure 2.7 , the node secret for an internal node v is computed by $x_v = f(x_L) \oplus f(x_R)$. For each node v, the node key k_v is computed by $k_v = g(x_v)$ where g is also a pseudo-random function.

In comparison with the top-down LKH model [55], the bottom-up OFT model approximately halves the broadcast overhead in rekeying after membership change. It is the first time to achieve an approximate halving in broadcast length. It is claimed that the OFT algorithm provides complete forward and backward security: newly admitted group members cannot read previous messages, and evicted members cannot read future messages, even with collusion by arbitrarily many evicted members [56]. However, collusion attacks against original OFT key establishment algorithm were proposed in [62, 63] shortly after its publication.

2.2.2 PKC-based Key Management Schemes

In Chapter 1, we have mentioned that the first challenge to a common perception that PKC is infeasible in WSNs was made in [27] which proposed a hybrid authentication key establishment scheme based on ECC. In fact, sensors are also assumed unable to perform PKC operations in [27]. The computation intensive elliptic curve scalar multiplication of a random point at senor nodes is removed. Alternatively, the scheme shifts high-cost operations to a new entity called a 'security manager'. In order to prevent the node impersonation attack, the scheme authenticates the two identities based on Elliptic Curve implicit certificates, solves the key distribution and storage problems, which are typical bottlenecks in pure symmetric key based protocols. Lately, the first known implementation of ECC [64] over F_{2^p} for sensor networks based on MICA2 show that ECC-based public key scheme is practical for popular sensors.

TinyPK [65] is a mechanism for authentication and key exchange between sensor nodes and a third-party. It is based on the RSA cryptosystem. The protocols were implemented on MICA2 motes with a TinyOS platform. TinyPK does not support dynamic networks where multiple session keys are inevitable due to membership change.

The authors of [66] claimed that PKC is feasible in WSNs provided with the right selection of algorithms and associated parameters. It was envisioned that NtruEncrypt and ECC are the most promising candidates for low-power WSNs implementation. In order to validate their claim, they made an in-depth comparison of three different PKC implementations (Rabin's scheme, NtruEncrypt and Elliptic Curve) particularly target on WSNs. The authors of [67] investigated several additive homomorphic public key encryption schemes as well as their applicability to WSNs. The authors recommended that selecting the most suitable PKC should comply with the topologies and the scenarios of WSNs. They concluded that public key cryptographic schemes are required for specific WSN settings where symmetric encryption schemes prove to be insufficient due to their sensitivity to node compromise.

ECC-based key management schemes

It is important to note that all of the schemes [27, 64, 65, 66, 67] of PKC implementation are key generation, authentication, or encryption/decryption algorithms but none is a complete PKC-based key management framework. As far as we know, the first trial of PKC key management for WSNs is the work in [68]. This paper presents three key management schemes for HWSNs: one is based purely on Symmetric Key Cryptography (SACK), another is based purely on PKC (SACK-P), and the other is the hybrid of both the symmetric and public approaches (SACK-H). These schemes are based on a typical hierarchical network model which compose of a base station (BS), a number of cluster heads, and numerous nodes. The network is arranged in a cluster-based topology. Each entity has a unique Identity (ID). All sensor nodes are loosely time synchronized with the BS. Key management starts after all the nodes have joined the network and before other communication has started.

SACK supports the establishment of three types of keys which helps to minimize the impact of any key's compromise to only a certain number of nodes. The key K_{NB} is an unique pairwise key shared by each node and the BS. A routing key (K_{CL}) is used by a cluster head to communicate with the BS. Cluster Key K_{C_i} is used by sensor nodes of the i-th cluster to communicate with their cluster head and other nodes within the cluster. Before network deployment, each sensor node is preloaded with one unique key (K_{NB}) and one master key K_M. The BS stores (SN_{ID_j}, K_{NB}) pair for each node and uses it to authenticate and establish a pairwise symmetric key for each sensor node when the node joining in the network. The BS also stores routing keys (K_{CL}) and cluster keys (K_{C_i}) in a database. After the formation of clusters, the BS sends a key generation seed S_{CL} to the cluster heads. Each cluster head then computes K_{CL} using S_{CL} and K_M. Once K_{CL} is generated, every cluster head generates a seed S_{C_i} different from S_{CL} and broadcasts it to its respective nodes. Each node of cluster i then computes K_{C_i} by using S_{C_i} on K_M.

SACK-P is based on Public Key Infrastructure (PKI). Before deployment, a public key pair $(K_{pub}[CN_j], K_{pri}[CN_j])$ is generated for each node using ECC. The BS maintains $[SN_{ID_j}, K_{pub}[CN_j]]$ pair for each node and each node is preloaded with its respective private key $K_{pri}[CN_j]$. During the phase of network setup, each sensor node sends a JOIN request message encrypted with $K_{pri}[CN_j]$ and its ID to the nearest cluster head. The cluster head forwards the message to the BS. The BS verifies the node authenticity by decrypting the message with the public key registered against the node ID stored in its database. If the node is authenticated as a legitimate node, an OK message is sent to the cluster head; otherwise a REVOKE message is directed to the cluster head. If the cluster head receives an OK message, it stores the node's public key $K_{pub}[CN_j]$ for future references; otherwise the message is discarded. After completion of this process, the BS broadcasts its public key to all the cluster heads. On receiving the BS's public key, each cluster head broadcasts its public key to its subordinate sensor nodes.

To the best of our knowledge, the first completed PKC-based key management scheme for HWSNs was proposed in [69]. The scheme is a routing-driven scheme, which establishes a shared key only for neighbor nodes that communicate with each other. Suppose a network consists of M H-nodes and N L-nodes. On the condition that each node is aware of the tree structure of the network and location, each H-node has to store $(N + 3)$ keys and each L-node has to store 2 keys. Otherwise, much more storage and communication overhead and some security problems may arise from an incorrect assumption. The scheme also requires that all nodes must be deployed by strictly following the known tree structure; otherwise, some L-nodes cannot find the keys matched with their cluster head. The scheme proposed in [70] is more secure than the scheme in [69]. Each L-node is preloaded only with its secret point and its random number. The secret point is used to dynamically establish pairwise keys with other communicating nodes. The random number is periodically renewed by the BS and forwarded to L-nodes through the corresponding cluster head. The random number is used for authentication between an L-node and its cluster head. Most of the security services of the scheme are based on the security of random numbers. Therefore, in each session, the renewed random number for a specific L-node is encrypted with the shared key between the L-node and the cluster head. If an L-node is not directly connected to the cluster head, the cluster head performs multiplex encryption with the key shared with L-nodes in the path from the cluster head to destination L-node. Obviously, the renewal of random numbers incurs large communication overhead and computation overhead at each cluster head.

Another pairing-based key management scheme for wireless sensor networks was proposed in [71]. The scheme not only exploits the means of pairwise key establishment, but also the method of cluster key and network key establishment. The scheme is built on a static network model. It is assumed that each node knows its location. It is claimed that the scheme is robust against several typical attacks. However, the robustness relies on a strong assumption that each cluster head is equipped with a intrusion detection system to detect misbehavior. It is well known that any intrusion detection system is too complex to be installed in a cluster head.

ID-based key management schemes

The basic idea of an ID-based encryption system is that a user's public key can be derived from its identity information. E.g. a user can use email as ID to calculate a public key, rather than extracting it from the certificate issued by a Certificate Authority (CA). ID-based encryption can eliminate the communication overhead for transmission of public key certificates.

Zhang proposed an ID-based key predistribution scheme for wireless sensor networks in [72]. The scheme comprises two phases: system initialization and pairwise key establishment. The procedure is as follows:

- System Initialization. A trusted Key Generation Center (KGC) constructs two groups G_1, G_2 and an admissible bilinear map $e : G_1 \times G_1 \rightarrow G_2$ where G_1 is a cyclic additive group with order q and G_2 is a cyclic multiplicative group with order q. KGC also chooses two cryptographic hash functions $h : \{0, 1\}^* \rightarrow Z_q^*$, $h_1 : G_2 \rightarrow Z_q^*$.

  KGC computes $g = e(P, P)$ and $P_{pub} = kP$ where P is a random generator of G_1 and $k \in Z_q^*$ is a random integer which works as the network master secret.

  KGC assigns a unique $ID_u \in \{0, 1\}^l$ to each node u. Then KGC calculates $S_u = (h(ID_u) + k)^{-1}P$ and $P_u = h(ID_u)P + P_{pub}$. Each node u is preloaded with its ID_u, its key pair (P_U, S_u) and the public system parameters $(p, q, G_1, G_2, h, h_1, P, P_{pub}, g)$.

- pairwise key establishment. Each node u can establish a pairwise key with its neighbor node v within three steps.

1. $u : r_u \in_R Z_q^*, r_u P_v = r_u(h(ID_v)P + P_{pub}), h_1(g^{r_u})S_u;$

   $u \rightarrow v : \{r_u P_v, h_1(g^{r_u})S_u\};$

2. $v : P_u = h(ID_u)P + P_{pub}, e(r_u P_v, S_v) = g^{r_u}; e(h_1(g^{r_u})S_u, P_u) = g^{h_1(e(r_u P_v, S_v))};$

   If the equality holds, $r_v \in_R Z_q^*; K_{v,u} = e(r_u P_v, S_v)^{r_v} = g^{r_u r_v};$

   $u \rightarrow v : \{r_v P_u, h_1(g^{r_v})S_v\};$

3. $u : e(h_1(g^{r_v})S_v, P_v) = g^{h_1(e(r_v P_u, S_u))};$

   If the equation holds, $K_{u,v} = e(r_v P_u, S_u)^{r_u} = g^{r_v r_u}.$

In this scheme, the established pairwise key can be known only by the two end nodes. The storage overhead is acceptable in this scheme. In this scheme, each node has to store its ID, its key pair, system parameters, and the established pairwise keys. In each pairwise key establishment, each node has to perform two elliptic curve discrete logarithm operations, two bilinear maps operations, and three discrete logarithm operations based on a finite field. If the network is large, the cost for computation may exhaust sensor nodes.

2.2.3 Hybrid Key Management Schemes

SACK-H system in [68] takes advantage of the difference in the computational capabilities of different nodes. The scheme shifts the burden of more complex, computationally expensive algorithms on the BS and cluster heads which have greater resources. It uses the public cryptography in inter-cluster level communication, and symmetric cryptography in intra-cluster communication. Before deployment, each node is installed with two types of cryptosystems: one is PKC-based cryptosystem like ECC and the other is Symmetric Key Cryptography based cryptosystem like RC5 or AES. The same as for SACK, each node is given a unique key K_{NB} and preloaded with a master key K_M. Instead of K_{CL}, each cluster head has separate publick/private key pair which can be denoted as $K_{pri}[CL_i]$ and $K_{pub}[CL_i]$. The key generation process is initiated by the BS. It calculates its public/private key pair and broadcasts its public key to cluster heads. Subsequently, each cluster head generates its public/private key pair and sends its public key along with its ID to the BS. The BS saves this public key and also broadcasts it along with the cluster head's ID so that all the cluster heads receive the public keys of all the other cluster heads.

After the completion of the key generation at cluster head level, each cluster head generates a seed and broadcasts it to all of its subordinate sensor nodes. Each sensor node uses this seed to generate the cluster-wide common symmetric key.

The performance analysis is undertaken from the perspective of scalability, key connectivity, revocation, resilience, and mobility. Network scalability is determined by security overhead in terms of data packet length. The experiment shows that SACK is best at scalability, followed by SACK-H, and SACK-P is worst. In fact, the operations of scalability in these three schemes are trivial. Node addition involves interaction between cluster heads and the BS. If new nodes join the network frequently, the authentication operations will collapse the BS. Key connectivity refers to the number of keys required to be stored on each node for a specified level of required connectivity. Both SACK and SACK-H guarantee 100% network connectivity. As far as SACK-P is concerned, it also provides complete network connectivity with the help of intermediate nodes. Secure node revocation is realized by TASER, a node revocation mechanism which was addressed in detail in [68]. For SACK-P, although the compromised nodes do not reveal any important keying information, removal of other sensor nodes' public key from compromised node will prevent the compromised node from generating any kind of attack on sensor nodes whose public key it possesses. The BS takes the role of trusted CA and maintains the Certificate Revocation List (CRL). SACK-P employs a hierarchical mode of communication, which makes the certificates verification proceed in a hierarchical manner. That is, for the communication between cluster heads, CRL is performed at the BS to check whether the intended cluster head has been revoked. In a similar way, for communications between sensor nodes, the validity of the node's certificate is checked by each cluster head. The resilience capability indicates the network's resistance against node capture. The higher degree of resilience, the more secure of the network. For SACK, one node compromise reveals its cluster or routing key and master key K_M which, in turn, makes the whole network vulnerable. For SACK-H, one node compromise reveals the common symmetric key for cluster-wide communication. Any node compromise in SACK-P does not reveal any keying information except its own private key and a few public keys. Therefore, SACK-P has the best resilience capability while SACK-H has intermediate resilience capability, and SACK has the worst resilience capability.

A more detailed hybrid key management scheme was proposed in [73]. In the upper tier, the pairing-based Public Key Cryptography is used to distribute pairwise keys. In the lower tier, a unbalanced Blom key predistribution scheme [48] is used to establish pairwise keys. The unbalanced key predistribution scheme provides perfect key connectivity. However, it inherits the weakness of the Blom key predistribution scheme. That is, the resilience is constrained by a threshold λ. More than λ compromised nodes within a cluster can work out all the pairwise keys for the specific cluster. The larger the size of λ, the larger is the broadcast message and the complexity of computation involved. In addition, this scheme considers only how to establish pairwise keys.

2.2.4 Security Issues in Existing Key Management for WSNs

Even though a lot of work has been done in this field, there are some drawbacks to overcome. The main disadvantage of random key predistribution schemes is that a large number of keys could be disclosed by compromising a few nodes. The polynomial-based key predistribution scheme has better attack resilience capability than do random key predistribution schemes when a small number of nodes are compromised. However, when the number of compromised nodes is higher than a certain threshold, the number of compromised paths increases dramatically. The scalability of EBS-based schemes is constrained by the cost of key establishment and key renewing which increase with group size. The combinatorial design key predistribution schemes do not guarantee that the key share probability is 1. LKH-based schemes incur too much communication overhead in multi-hop WSNs and OFT-based schemes are vulnerable to collusion attack.

In contrast to Symmetric Key Cryptography, Public Key Cryptography has advantages in addressing issues like key management, authentication, non-repudiation and digital signatures. The public key cryptosystem is being deployed in WSNs as is evidenced by the recent interest in [27, 64, 66, 67] from different aspects. The first trial of the PKC-based key management scheme was SACK-P [68]. However, it is a direct transformation of the basic SACK scheme which is based on Symmetric Key Cryptography and correspondingly cannot be seen as an original PKC-based key management scheme. Even though several PKC-based key management schemes have been proposed, the algorithms rely too much on some strict assumptions or incur heavy overhead. PKC-based key management in WSNs is still in the infant stage and lightweight PKC-based for WSNs is an open problem.

The hybrid scheme is a better approach for WSNs because it exploits the heterogeneous property of networks. It shifts the cryptographic burden to the base station or the cluster heads where the resources are less constrained. The hybrid scheme has the advantages of both the asymmetric and symmetric key cryptosystems. Therefore, it is suitable for the large HWSNs.

2.3 Establishment of Different Types of Keys

Most of existing key management schemes for group communication in WSNs address only session key establishment and renewal. Key predistribution is considered as the most suitable key assignment mechanism for WSNs. However, key predistribution schemes are concerned only with the establishment pairwise keys which can only be used for peer-to-peer communication between neighboring nodes.

Generally speaking, a HWSN includes three types of entities: the base station, cluster heads, and normal sensor nodes. These entities comprise a hierarchical network. In such a network, various types of communication may occur. The base station broadcasts control commands throughout the whole network. Each cluster head multicasts messages within the cluster. A node communicates with its neighboring nodes by unicasting. In one word, a single key cannot meet different communication requirements in WSNs, especially in hierarchical WSNs. Different types of keys should be established in WSNs to secure unicast, multicast, and broadcast communications.

Most of existing symmetric schemes for WSNs mainly consider a distributed network topology. They introduce different mechanisms for pairwise key establishment. Some centralized schemes [52, 55, 56] only discuss how to distribute and update globally shared group key. Research shows that such a network topology is poor at performance and scalability. HWSNs have better scalability while they involve various communication patterns, including unicast of single nodes, multicast within a group and broadcast in the whole network. Therefore, different keys must be established to encrypt different types of packets.

2.3.1 LEAP

Zhu *et al.* [29] devised a scheme called LEAP for hierarchical WSNs. LEAP supports the establishment of individual keys, pairwise keys, cluster keys, and a global key. Different keys are used to handle the different types of packets.

- *Individual key* [41]: This is a unique key that is shared between the base station and each sensor node [29]. The key is preloaded into each node's memory before being deployed. The individual key is calculated as $K_u^m = f_{K^m}(u)$ where f is a pseudo-random function and K^m is the master key known only to the base station. Each sensor node uses the individual key to calculate MAC on the messages to the base stations. In the same way, a base station can use the shared individual key to encrypt messages to each node. The base station does not store all the individual keys. The base station can generate the individual key whenever it attempts to communicate with a node.

- *Pairwise shared key* [41]: Each node shares a pairwise key with each of its immediate neighbors. The pairwise key is used to secure communications that requires privacy or source authentication. Similar to the scheme in [3], there are four stages of pairwise key establishment: key predistribution, neighbor discovery, pairwise key establishment, and key erasure. During the initial stage of key predistribution, node U is loaded with a key K_i by the controller and drives the master key K_u using it. For neighbor discovery, node U first initializes a timer to activate at time t_{min}, then it broadcasts a HELLO message containing its ID. The neighboring node V responds to node U with an acknowledgement (ACK) containing its ID if it receives node U's HELLO message. The ACK of V is authenticated using its master key K_v which is derived from K_i. Node U verifies the authentication of V by generating the master key K_v with K_i. The neighbor discovery stage can be denoted as:

$$U \to * : U;$$

$$V \to U : V, MAC(K_v, U \mid V).$$

In the stage of pairwise key establishment, node U computes the pairwise key K_{uv} shared with node V, as $K_{uv} = f_{K_v}(u)$. Node V can also compute K_{uv} in the same way. K_{uv} serves as their pairwise key. In the final stage, when its timer expires after t_{min}, node U erases K_i and all the masters keys of its neighbors, which is computed in the neighbor discovery stage. Even though an adversary captures a node, the communications between it and another node cannot be decrypted without the key K_i.

- *Cluster key* [41]: This is a key shared between a node and its neighboring nodes. Cluster key establishment follows the pairwise key establishment phase. Suppose a node U wants to establish a cluster key with all its immediate neighbors $V_1, V_2, \ldots, V_m$. Node U first generates a random key K_u^c, then encrypts this key with the pairwise key shared with its neighbors, and then transmits the encrypted key to each neighbor $V_i (1 \leq i \leq m)$.

  $$U \rightarrow V_i : (K_u^c)_{K_{u,v_i}}.$$

  Each node V_i decrypts the key K_u^c with the key K_{u,v_i} and stores it in a table, and then sends its own cluster key to node U in the same way. When node U is revoked, every neighbor node generates a new cluster key and transmits it to all the other neighbors in the same way.

- *Global key* [41]: This key is shared between the base station and all the sensor nodes in the network. It is mainly used by the base station to distribute confidential messages to the whole network. A simple method of bootstrapping a group key is to preload each node with the global key before the deployment. An important issue that arises immediately is the need to securely update the global key once the membership changes or a compromised node is detected. Such key renewal involves much communication overhead. However, Zhu *et al.* [29] proposed an efficient scheme based on cluster keys for which the transmission cost will be only one key. In WSNs, all messages sent by the base station must be authenticated; otherwise, an adversary may impersonate them. μTESLA, based on a one-way key chain and delayed the disclosure of keys, is an efficient method to broadcast messages into a WSN. To bootstrap μTESLA, each node should be preloaded with the commitment of the key chain. If K_g is the new group key and U is the node to be revoked, the base station broadcasts the following message M:

$$Controller \rightarrow * : u \parallel f_{K'_g}(0) \parallel MAC(k_i^T; u \mid f_{K'_g}(0)),$$

where $f_{K'_g}(0)$ is the authentication key which enables a node to verify the authenticity of the global key $f_{K'_g}$ that it will receive later. The server then distributes the MAC key K_T^i after one μTESLA interval. After a node V receives the message M, it verifies the authenticity of the message using μTESLA. If node V is a neighbor of U, V will remove its pairwise key shared with U and update its cluster key. Each node encrypts the renewed global key with its cluster key and transmits it to neighbors. This algorithm continues recursively until all the nodes have received the renewed key.

2.3.2 The Time-based Key Management Scheme

In order to minimize the portion of compromised network when the initial key IK is disclosed, Jang *et al.* [1] split the lifetime of a sensor network into P time slots and each time slot is assigned with an initial key. As depicted in Figure 2.8, T_j and N_j represent a time slot and a group of node deployed during that time slot T_j, respectively. If a node is deployed at time slot T_j, the sensor node is preloaded with the initial key IK_j and m master keys of randomly chosen time slots. Then the newly deployed node can establish pairwise keys with nodes which are deployed at the same or different time slots. Three situations exist for the establishment of pairwise keys.

1. All nodes in the same group N_j ($1 \leq j \leq P$) are able to establish pairwise keys with each other using the initial key IK_j during the time slot T_j.

2. Then, they are able to establish pairwise keys with other nodes which are deployed at different time slots, but have the master key derived from the current initial key. Suppose u is a node deployed at time slot T_j and v is a node deployed before T_j. If the node v has the master key K_{vj} which is derived from the initial key IK_j for the time slot T_j, the node v can compute

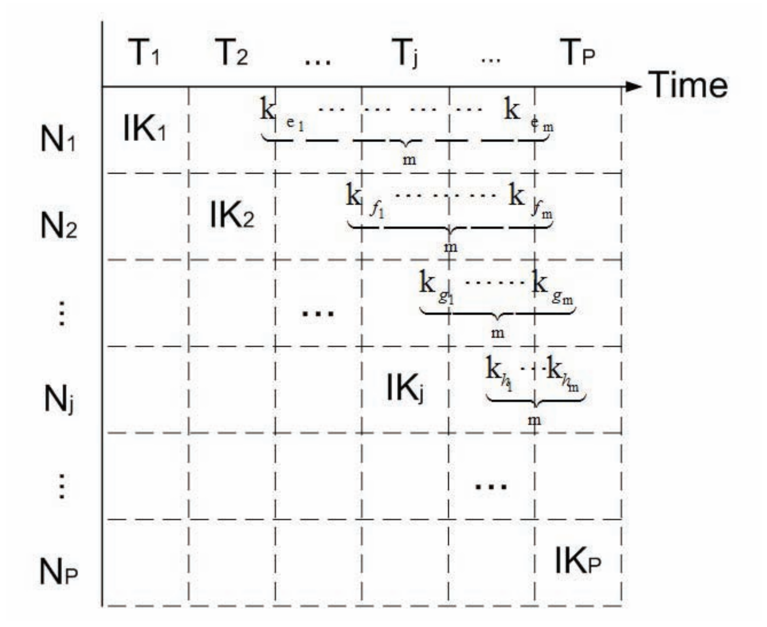


Figure 2.8: Key materials preloaded to nodes at different time slots in the scheme [1]

a pairwise key $K_{uv} = f_{K_{vj}}(u)$. The node u is also able to generate a master key of the node v, $K_{vj} = f_{IK_j}(v)$.

3. Finally, a pair of sensor nodes that do not share any keying material but are within each other's communication range, can establish pairwise keys via proxy nodes.

2.3.3 Security Issues

LEAP efficiently establishes multi-level keys. LEAP is energy-efficient since it supports an in-network processing technique which greatly reduces network communication. LEAP can minimizing the effects of selective forwarding attack by restricting this problem to a local area. LEAP can also prevent a HELLO attack since the nodes accept packets only from authenticated neighbors. However, LEAP suffers from sinkhole attack. In a sinkhole attack, a compromised node attracts packets by advertising information like high battery power, etc., then later drops all the packets [41]. Like many other key management protocols, LEAP assumes that sensor nodes are secure during the initialization phase and can be compromised after the phase. However, such an assumption could be incorrect. The security of

LEAP depends mainly upon the initial key which is erased from sensor nodes after the initialization phase. However, the same initial key IK should be used again for node addition after that phase, while the new node can be captured before removing the initial key. Therefore, the initial key IK should never be used for node addition in LEAP after the initial time T_{min}. Different initial keys are used for different time slots in the time-based key management scheme [1]. The threat caused by the disclosure of the initial key is eliminated. However, the key connectivity is constrained by the number of preloaded master keys m and the order of the current time slot. If m is far less than the lifetime P of the network, the key connectivity $\frac{m}{P-j}$ is far less than 1 at the time slots j ($j \leq P/2$). On the contrary, if m is close to P, higher key connectivity can be achieved with a heavy burden on storage.

We consider the security problem of the established pairwise key between two nodes. The pairwise key does not exclusively belong to the two end nodes and threat against confidentiality and authentication may arise from it. As shown in Figure 2 in [1], Nodes of group N_1, N_2, and N_6 are preloaded with master key K_{u7}, the pairwise keys between any two groups of them are known by the other group. In addition, m master keys of randomly-chosen time slots are preloaded to the nodes when they are deployed to the network without taking the lifetime of nodes into consideration. Suppose a node which can survive at most G_w time slots is deployed at the j-th time slot with m master keys of randomly-chosen time slots. Those master keys of the time slots from the $(j + G_w)$-th to the P-th would never be used. They waste the scant memory of sensor nodes.

2.4 Authentication Mechanisms in Key Management

Authentication guarantees that the entities with which one communicates are the expected ones and the received data is the original one sent by the counterparts. Generally speaking, an asymmetric key mechanism is required to authenticate messages. However, due to the resource constraints at sensor nodes, solutions based on RSA-based authentication have unacceptable storage and computation overhead in WSNs. Current research on authentication in WSNs focuses on unicast authentication, broadcast authentication, and filtering false data with Symmetric Key Cryptography. Only in very recent years, has research been carried out on ECC-based authentication and lightweight authentication.

2.4.1 Unicast Authentication

Unicast authentication ascertains the origin and integrity of the transferred messages when two nodes establish communication. If the two nodes share a secret symmetric key, a MAC can be generated with the shared key to ensure secure unicast authentication. Perrig et al. proposed a suite of security protocols called Security Protocols for WSNs (SPINS) optimized for WSNs [2]. SPINS includes two protocols: secure network encryption protocol (SNEP) and μTESLA. SNEP provides unicast authentication, confidentiality, and replay protection through authentication with MAC and encryption. μTESLA offers a solution for broadcast authentication.

With the SNEP protocol, two communicating nodes maintain a shared counter to prevent replay attacks. The counter is initialized when two nodes begin communicating and increases it after each communication block. The counter aids both the encryption process and the calculation of the MAC. SNEP generates the MAC by running the plaintext message and the counter through the RC5 block cipher in CBC-MAC mode. Since the MAC provides message integrity, the CRC field is dropped. Thus, total packet length increases only by 6 bytes. SNEP provides origin authentication and message integrity by appending an 8-byte MAC to the ciphertext. It is unnecessary to send the counter along with each message, thereby minimizing communication overhead. For those cases where strong freshness of message is required, a nonce is employed.

SNEP has a number of unique advantages. It packages multi-functions into one cryptographic protocol. SNEP achieves semantic security which prevents adversaries from reading the plaintext even from multiple ciphertexts of the same message. The use of the counter in encryption means that repeatedly encrypting a message M would produce different ciphertexts. In addition, use of the counter in calculation of the MAC provides weak message freshness and protection against replay attack. However, Karlof *et al.* in [74] state that SNEP was neither fully specified nor fully implemented, motivating the creation of TinySec.

The TinySec was designed as a replacement for the incomplete SNEP. TinySec aims to provide similar security: origin integrity, message integrity, replay protection, and message confidentiality. In SNEP, each node has to maintain a counter for each node communicating with it. Since such a mechanism consumes a great deal of memory,, TinySec removes the application of counters. TinySec secures wireless sensor networks in a way that facilitates integration into sensor network applications. It uses CBS mode with ciphertext stealing for encryption and CBC-MAC mode for authentication. TinySec provides two packet formats: TinySec-Auth for origin and message authentication only, and TinySec-AE for authentication and encryption. Since TinySec drops the TinyOS Group ID field, TinySec-Auth produces only 1-byte of additional communication overhead.

TinySec makes use of MAC to provide authentication and message integrity. The security of CBC-MAC is directly related to the length of the MAC. The secure length of MACs usually ranges from 8-bytes to 16-bytes, while TinySec specifies a 4-byte MAC for the trade-off between security and cost. Karlof *et al.* argue that a 4-byte MAC meets the adequate level of security for WSNs [74]. This implies that 2^{31} packets must be generated in order to forge just one malicious packet. As a compensating control, it is required that nodes signal the base station when the rate of MAC failures exceeds a predefined threshold.

The TinySec MAC evolved from CBC-MAC block cipher mode. This construction fails to secure variably sized messages. Three alternatives for generating MACs of variably sized messages were provided later. TinySec is implemented with Skipjack rather than RC5 because the latter requires additional RAM to store a pre-computed key schedule. The implementation shows that the operational costs of TinySec are relatively low.

2.4.2 Broadcast Authentication

Broadcast authentication guarantees that multiple recipients of a message can verify its origin and integrity. Perrig *et al.* [2] proposed a broadcast authentication mechanism named μTESLA. Many techniques are used to extend the capabilities of μTESLA in [28, 29, 30]. The scheme in [29] tailors μTESLA to local broadcast authentication. The scheme in [28] overcomes the length limit of the hash chain. The scheme in [30] extends μTESLA to support multicast scenarios.

μTESLA divides the time period for broadcasting into multiple time intervals and assigns different authentication keys to different time intervals. μTESLA employs a key chain of authentication keys. Each key in the key chain is the image of the next key under a pseudo random function. μTESLA achieved broadcast authentication through delayed disclosure of authentication keys in the key chain. The sender selects a random value K_n as the last key in the key chain and repeatedly performs the pseudo random function F to compute all the other keys: $K_i = F(K_{i+1})$ $(0 \leq i \leq n-1)$, where K_i is assigned to the i-th time interval. With the pseudo random function F, given K_i in the key chain, anybody can compute all the previous keys K_i $(1 \leq i \leq j)$, but nobody can compute any of the later keys K_i $(j+1 \leq i \leq n)$. Thus, with the knowledge of the initial key K_0, which is called the 'commitment of the key chain', a receiver can authenticate any key in the key chain by merely performing pseudo random function operations. When a broadcast message is available in i-th time interval, the sender generates MAC for this message with the key derived from K_i and then broadcast this message along with its MAC and discloses the key K_{i-d} assigned to the time interval I_{i-d}, where d is the disclosure lag of the authentication keys. The sender prefers a long delay in order to make sure that all or most of the receiver s can receive its broadcast messages. But, for the receivers, a long delay could result in a high storage overhead to buffer the messages. As far as authentication is concerned, μTESLA is efficient because only a one-way random function and Symmetric Key Cryptography based cryptographic operations are needed to authenticate a broadcast message. However, the base station has to unicast the parameters to the sensor nodes individually in the initialization phase. Such a method for bootstrapping new receivers in μTESLA does not scale to large WSNs. Therefore, the major barrier to using μTESLA is the mismatch between the unicast-based distribution of key chain commitments and the authentication of broadcast messages.

μTESLA is not suitable for local broadcast because μTESLA does not provide authentication immediately. In μTESLA, nodes need to keep the packets in their buffers until the authenticating key arrives. Local broadcast usually requires immediate authentication. Pairwise keys cannot be used for local broadcast authentication because, if a node has n neighbors, the approach requires the sender node to calculate n MACs for each message. Local broadcast needs a method where a node can broadcast a message to all its neighbors using a single MAC and cluster keys, with a problem as follows. If an adversary can compromise a node, the cluster key from that node is available and can be used to attack the network by impersonating that node or a neighboring node. If nodes X, U, and V are three vertices of a triangle, X is compromised, and U wants to send messages to X and V, X can use node $U's$ cluster key to impersonate it and send false messages to V [41].

Zhu *et al.* [29] designed a one-way key chain based authentication scheme which is based on μTESLA for defeating this attack. In the proposed authentication scheme, each node generates a one-way key chain and sends the commitment of it to their neighbors. If a node wants to send a message to its neighbors, it attaches the next authorization key from its key chain to the message. The receiving node can verify the validation of the key based on the commitment it has already received. The one-way key chain based authentication is designed based on two observations [41]: a node only needs to authenticate to its neighbors and that a node V will receive a packet before a neighboring X receives it and re-sends it to V. This observation is true because of the triangular inequality among the distances of nodes involved. An adversary may still try to attack the nodes by shielding node V while U is transmitting a message, and then later send a modified packet to V with the same authorization key; but this attack can be prevented by combining the authorization keys with the cluster keys. When this is done, the adversary does not have the cluster key and so cannot impersonate node U. However, this scheme does not provide a solution for attacks from inside where the adversary knows U's cluster key.

The original idea of [28] is to predetermine and broadcast the initial parameters required by μTESLA instead of unicast-based message transmission. The authors presented a multi-level key chain scheme to efficiently distribute the key chain commitments for μTESLA. Several techniques are also proposed to improve the survivability of the scheme and defeat some DoS attacks. By using pre-determination and broadcast, the final multi-level approach removes μTESLA's requirement of a unicast-based distribution of initial key chain commitments and satisfies several nice properties, including low overhead, tolerance of message loss, scabridity to large networks, and resistance to replay attacks as well as DoS attacks. Despite these advantages, several issues are not yet properly addressed. None of μTESLA and multi-level μTESLA are scalable in terms of the number of senders. Though multi-level μTESLA schemes are scalable in terms of receivers, they either use substantial bandwidth and storage at sensor nodes, or require significant resources at senders to deal with DoS attacks. Subsequently, Liu *et al.* presented efficient techniques in [30] to support a potentially large number of broadcast senders using μTESLA instances as building blocks. The scheme has three advantages over the multi-level μTESLA schemes. Firstly, the scheme allows broadcast authentication with a large number of senders. Secondly, the scheme is not subject to the DoS attacks against the distribution of μTESLA parameters. Thirdly, the proposed scheme revokes the broadcast authentication capabilities from compromised senders with two complementary approaches: a Merkle hash tree based scheme and a proactive distribution based scheme. The former approach removes the authentication delay as well as the vulnerability to DoS attacks during the distribution of μTESLA parameters, while the latter proactively controls the distribution of the broadcast authentication capability of each sender to allow the revocation of compromised senders.

2.4.3 Filtering False Data

WSNs are susceptible to false data injection attacks because they are often deployed in unattended environments. In false data injection attacks, an adversary may compromise several sensor nodes, and then use the compromised node and inject false data into the network with the goal of deceiving the base station or depleting the resource of the relaying nodes. Once a node is compromised , all the security parameters stored in that node become accessible to the attacker. The compromised node can successfully authenticate a bogus report to a neighbor, which has no way of differentiating such false reports from legitimates ones. This attack falls in the category of insider attacks. Standard authentication mechanisms cannot prevent such insider attack. A symmetric group key-based Statistical Enroute Filtering (SEF) of injected data scheme, which allows both the base station and en-route nodes to detect and drop false fata with certain probability, is proposed in [75]. SEF requires that each sensing report be validated by multiple keyed MACs, each generated by a node that detects the same event. As the report is forwarded, each node along the way verifies the correctness of the MACs probabilistically and drops those with invalid MACs at earliest points. The probability of detecting incorrect MACs increases with the number of hops the report travels. Depending on the path length, there is a non-zero probability that some reports with incorrect MACs may escape en-route filtering and be delivered to the sink. In any case, the sink will further verify the correctness of each MAC carried in each report and reject false ones. SEF exploits the network scale to determine the truthfulness of each report through collective decision-making by multiple forwarding nodes. The basic components of the scheme include a key assignment method designed for en-route detection of false reports in the presence of compromised nodes and a mechanism for collective data report generation, enroute report filtering, and sink verification. The analysis and simulation results show that SEF can effectively detect false reports even when the attacker has obtained the security keys from a number of compromised nodes, as long as those keys belong to a small number of the key pool partitions. With an overhead of 14 bytes per report, SEF is able to filter out $80\% \sim 90\%$ false data by a compromised node within 10 forwarding hops. SEF is the first step towards filtering false data. SEF achieves this goal by carefully limiting the amount of security information assigned to each individual node.

Three interleaved hop-by-hop authentication schemes was proposed in [76]. These schemes guarantee that the base station can detect injected false data immediately when no more than a certain threshold nodes are compromised. Moreover, these schemes enable attempts to filter out false data packets injected into the network by compromised nodes before they reach the base station, thus saving the energy for relaying them. These schemes based on strong security assumptions. In the security model, the base station has a mechanism to authenticate its broadcast message and every node can verify the broadcast message. Such a broadcast mechanism can be achieved by μTESLA [2]. The schemes also assume that every node shares a master security key with the base station and every node knows the authenticated set of its one-hop neighbors and has established a pairwise key with each of them. Such security assumptions can be achieved by the pairwise key establishment scheme in LEAP [29]. In a sense, these schemes can be seen as the complement of LEAP [29]. Though the local broadcast authentication mechanism is robust against outsider attacks, it is vulnerable to insider attacks in which an adversary only needs to compromise a single node to inject false data. In contrast, the interleaved hop-by-hop authentication scheme is robust against insider attacks involving a certain number of compromised nodes. Indeed, the local broadcast authentication mechanism can be considered as a special case of the interleaved hop-by-hop scheme where $t = 0$.

2.4.4 PKC-based Authentication

Even some public key cryptosystems are viable to WSNs. However, compared to the secret key cryptography and one-way hash function, PKC is still much more expensive. To maximize the life time of batteries, the use of PKC in WSNs must be limited and optimized. Authenticating the public key is essential for PKC, and it may happen many times. In general networks, public key authentication is achieved by certificates. This operation involves an expensive signature verification on a certificate. Therefore, it is necessary to substitute the PKC authentication with algorithms that consume much less energy. Du *et al.* [77] proposed to use one-way hash function to conduct public key authentication in wireless sensor networks. Seeing that sensors usually belong to one administrative entity before the deployment, they can exchange the one-way hash values of their public keys securely prior to the deployment. The Merkle tree technique is used such that each sensor only needs to save one hash value while being able to authenticate all the public keys. Since the communication is proportional to the height of the Merkle tree, they trim down the single Merkle tree to a number of shorter trees to reduce the communication overhead. The trimming scheme is based on sensor deployment knowledge and can be described as follows: if B is more likely to be A's neighbor, B should be in a shorter tree of A, so the communication overhead to authenticate B's public key is lower; if B is less likely to be A's neighbor, B can be put in a taller tree of A, because the authentication is less likely to occur. The performance evaluation shows that the scheme can save up to 86% of the energy for the public key authentication operation.

2.4.5 Lightweight Authentication

μTESLA [2] and its variants [28, 29, 30] achieved delayed authentication with the requirement of loose time synchronization among nodes as well as the sharing of initial secrets between authenticators and verifiers. However, in some scenarios, especially when the network size is large, it is hard to determine the bound of normal delay, and this can be exploited by the adversary to launch DoS attacks. SEF [75] and interleaved hop-by-hop authentication scheme [76] use hash functions to produce multiple MACs to authenticate messages. These two schemes are more efficient than the PKC-based approaches. However, because each secret key is shared by multiple nodes, these schemes become ineffective or even useless if a large number of nodes are compromised. In addition, these schemes [2, 28, 29, 30, 75, 76] cannot achieve non-repudiation, either.

Lightweight authentication is a new authentication mechanism designed for RFID [78]. Zhang *et al.* lent this idea to WSNs [72]. They provided a lightweight and compromise-resilient message authentication scheme which adopts a perturbed polynomial-based technique to simultaneously accomplish the goals of lightweight, resilience to a large number of node compromises, immediate authentication, scalability and non-repudiation. The scheme achieves a good level of security by local collaboration method. The perturbed polynomial-based message authentication provides higher adaptability over existing authentication techniques based on multiple MACs [75, 76]. The proposed method also keeps the advantage of immediate authentication held by those techniques. In addition, independent and random factors are employed to perturb polynomial shares that preloaded to individual nodes, which significantly increases the complexity for the intruder to break the secret polynomial, and therefore renders the proposed approach to be resilient to node compromises. The authors claimed that the scheme is lightweight. However, the higher level security is achieved at the cost of very high computation and communication overheads.

2.4.6 Security Issues of Authentication Mechanisms

The main problem of SNEP is that it was not fully specified and implemented. TinySec is not perfectly resilient to attacks. The security of TinySec relies on a single key, and therefore is unable to securely perform a rekey if necessary [64]. As far as broadcast authentication is concerned, all μTESLA-like schemes [2, 28, 29, 30] require loosely time synchronization and the nodes suffer from severe energy-depletion DoS attacks due to the nature of delayed message authentication. As far as filtering false data is concerned, collaborative filtering of false reports in [75] requires that nodes share a certain amount of security information. The more security information each forwarding node has, the more effective the en-route filtering can be, but also the more secret the attacker can obtain from a compromised node. However, the problem of how to evaluate the trade-off between these two conflicting goals is not explored. The three hop-by-hop authentication schemes in [76] can only be applied to data collection. They do not work for hop-by-hop data aggregation applications and the applications in which more complex data reports other than Boolean values are generated. Lightweight authentication and PKC-based authentication are ideal alternative options for WSNs while the unique characteristics and constraints of WSNs should be given full consideration. To date, no feasible lightweight authentication and PKC-based authentication schemes have been proposed for resource-constrained WSNs.

2.5 Self-healing Key Distribution Mechanisms

Faults in WSNs may stem from a variety of things, including hardware and software design, radio interference, de-synchronization, unreliable communication channel, battery exhaustion, or dislocation. It is a common assumption that both hardware and software are well designed and implemented in sensor networks. Current research considers node failure or compromise, and corrupted or undelivered packets. The failure nodes may reduce the availability of the network and even lead to network failure. Corrupted or undelivered packets compromise the effectiveness of the aggregated results. Therefore, fault-tolerance is an essential attribute that should be integrated into WSNs. To date, fault- tolerance techniques have been widely used in WSNs from hardware to software, from the physical layer to the application layer.

We have mentioned that sensor nodes can be compromised or disabled and messages may be delayed or lost. These faults considerably affect the performance of key management protocols. Most of existing solutions have been dedicated to minimizing the computational and the communication overhead of key management protocols to meet scalability. However, most of the current research does not take these faults into consideration and the majority of current key management protocols are not designed to tolerate failures. In particular, node failure during key establishment is not addressed. However, such failure may block the key establishment process resulting in significant communication overhead needed to re-synchronize the participants to the shared key. Fault-tolerant property of key management guarantees the establishment of a key between correct members even if service is halted due to high traffic or system failure. Here we consider a self-healing key distribution countermeasure to combat packet loss or corruption during key transmission.

Self-healing key distribution is a type of key distribution. Key distribution schemes should meet availability, integrity, confidentiality, authentication and non-reputation traditional security requirements. The same is true of self-healing key distribution schemes. In addition, according to the feature and application environment of self-healing key distribution, some particular evaluation measurements should be highlighted. The first one is security. The security here includes not only forward and backward secrecy, but also collusion resistance between the newly joined nodes and the revoked nodes. Considering the wireless application environment, storage and bandwidth are regarded as constrained resources moreover, computational power, storage, communication and computation overheads should be as low as possible. Furthermore, the group may become too large to be managed by a single party, thus raising the issue of scalability. A self-healing key distribution scheme used in a resource-limited environment seeks to minimize the workloads of both the group manager and end users in order to augment the scalability.

2.5.1 Classification of Self-healing Key Distribution Schemes

The success of a self-healing key distribution scheme is determined in part by its capability of securely and efficiently recovering the session key in low-cost wireless networks. Self-healing key distribution schemes can be broadly classified into unconditionally secure and computationally secure schemes. Unconditionally secure schemes are based on information theory, which is generally considered to have been founded in 1948 by Claude Shannon in his seminal work [79] and are defined by entropy function $H(\cdot)$. On the other hand, computationally secure schemes are based on one or several open hard problems. The first and several subsequent self-healing key distribution schemes are unconditionally secure. In recent years, more and more research works have dealt with computational secure schemes. Existing

Table 2.2: Classification of self-healing key distribution schemes

	Unconditionally secure	Computationally secure
Theory basis	Information theory	Open problems in cryptography
Polynomial secret sharing	[80, 81, 82, 83, 84] [87, 88, 89]	[6, 85, 86]
Vector space secret sharing	[90, 91]	[7, 92, 93]
Hash chain	None	[6, 7, 86, 92, 93, 94, 95]
SDR	None	[96, 97]
Bilinear pairing	None	[98]

self-healing key distribution schemes can be classified into more classes with regard to different cryptographic primitives on which they are based. They are polynomial secret sharing, vector space secret sharing, hash chain, Subset Difference Rekeying (SDR), and bilinear pairings based self-healing key distribution schemes.

Table 2.2 shows a classification based on the two criteria: unconditionally secure VS. computationally secure and different cryptographic primitives on which they are based.

2.5.2 The Model of Self-healing Key Distribution Schemes

The life span of a network is partitioned into time intervals called sessions. m denotes the number of sessions, which should be determined in advance in some schemes. Let $U = \{U_1, \ldots, U_n\}$ be the finite universe of users. Each user U_i has a unique identifier ID_i. A broadcast unreliable channel is available, and time is defined by a global clock. The group manager sets up and manages, by means of joining and revoking operations, a communication group which is a dynamic subset of U. All the operations take place within a fixed range (here we suppose the range is F_p, where p is a sufficiently large prime number). Let $G_j \subseteq U$ be the communication group established by the group manager in session $j(j = 1, \ldots, m)$. Each user $U_i \in G_j$ holds a personal key S_i, received from the group manager before or when joining G_j. The personal key is used to recover the session keys as long

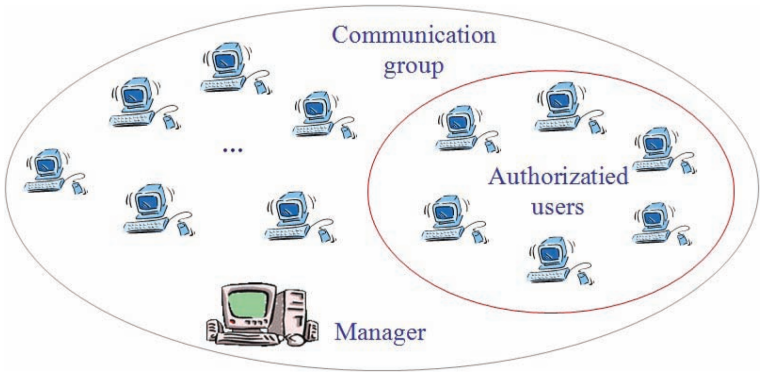


Figure 2.9: The general communication model of self-healing key distribution schemes

as U_i is not removed from the group by the group manager. Figure 2.9 depicts a general communication model.

Let $R_j \subseteq G_{j-1}$ denotes the set of revoked group users in session j and $J_j \subset U \setminus G_{j-1}$ denotes the set of users who join the group in session j with $R_j \cap J_j = \phi$. Hence, $G_j = (G_{j-1} \cup J_j) \setminus R_j$ for $j \geq 2$ and by definition $G_1 = U$. Moreover, for session $j \in \{1, \ldots, m\}$, the session key K_j is chosen independently and according to the uniform distribution on F_p. For any non-revoked user $U_i \in G_j$, the j-th session key K_j is determined by broadcast information B_j and personal key S_i. All the aforementioned notions are used throughout the thesis.

Self-healing key distribution schemes are defined under security models. The pioneering self-healing key distribution scheme [80] is unconditionally secure and defined by entropy function $H(\cdot)$ (See [99] for more details). The subsequent models are slightly modified versions of the basic one.

To clarify the performance of the schemes and facilitate the later analysis, we mention the following formal security model of the self-healing key distribution scheme with revocation capability. Here we consider the original communication model. Suppose there are n users and a group manager. The life-time of the network is divided into m sessions. t is the maximum number of users that can be revoked in the life-time of the network. We define the general unconditionally secure self-healing key distribution scheme from two aspects. *Definition 1* defines self-healing key distribution scheme with revocation capability. *Definition 2* defines the properties of security including forward secrecy, backward secrecy and the capability of collusion resistance.

Definition 1. Let U be the universe of users of a network, m be the maximum number of sessions, and t be the maximum number of users that can be revoked by the group manager.

1. The scheme is a session key distribution scheme if the following conditions are true:

   (a) For any user $U_i \in G_j$, the key K_j is determined by B_j and S_i. Formally, it holds that:

   $$H(\mathbf{K}_j|\mathbf{B}_j, \mathbf{S}_i) = 0. \tag{2.1}$$

   (b) What users learn from the broadcast B_j and their own personal key cannot be determined from the broadcasts or personal keys alone. That is:

   $$H(\mathbf{K}_1, \ldots, \mathbf{K}_m|\mathbf{B}_1, \ldots \mathbf{B}_m) = H(\mathbf{K}_1, \ldots, \mathbf{K}_m|\mathbf{S}_{G_1 \cup \ldots \cup G_m}) = 0. \tag{2.2}$$

2. The scheme has t-revocation capability if, for each session j, let $R = R_j \cup \ldots \cup R_1$, such that $|R| \leq t$, the group manager can generate a broadcast B_j such that the collusion of all revoked users in R cannot recover K_j. Formally, it holds that:

   $$H(\mathbf{K}_j|\mathbf{B}_1, \ldots \mathbf{B}_j, \mathbf{S}_R) = H(\mathbf{K}_j) \tag{2.3}$$

   where $\mathbf{S}_R$ denotes the set of personal keys of all users in R.

3. The scheme is self-healing if the following property is satisfied:

   Every $U_i \in G_r$, who has not been revoked after session r and before session s, from broadcast messages B_r and B_s, where $1 \leq r < s \leq m$, can recover all keys K_l, for $l = r, \ldots, s$. Formally, it holds that:

   $$H(\mathbf{K}_r, \ldots, \mathbf{K}_s|\mathbf{B}_r, \mathbf{B}_s, \mathbf{S}_i) = 0. \tag{2.4}$$

Definition 2. Let $U = \{U_1, \ldots, U_n\}$ and $j \in \{1, \ldots, m\}$. The scheme guarantees both forward and backward secrecy if the following properties are satisfied:

1. For any set $R \subseteq U$, and all the users in R are revoked before session j, it is computationally infeasible for the $U_i \in R$ together to get any information about K_j, even with the knowledge of group keys $K_1, \ldots, K_{j-1}$ before session j. Formally, it holds that:

$$H(\mathbf{K}_j|\mathbf{B}_1, \ldots, \mathbf{B}_m, \{S_i\}_{U_i \in R}, K_1, \ldots, K_{j-1}) = H(K_j) \tag{2.5}$$

2. For any set $J \subseteq U$, and all $U_l \in U$ join after session j, it is computationally infeasible for the users in J together to get any information about K_j, even with the knowledge of group keys $K_{j+1}, \ldots, K_m$ after session j. Formally, it holds that:

$$H(\mathbf{K}_j|\mathbf{B}_1, \ldots, \mathbf{B}_m, \{S_i\}_{U_i \in R}, K_{j+1}, \ldots, K_m) = H(K_j) \tag{2.6}$$

3. Let $C \subseteq R_r \cup \ldots \cup R_1$ be a coalition of users removed before session r and let $D \subseteq J_s \cup \ldots \cup J_m$ be a coalition of users who join the group from session s. Let $|C \cup D| \leq t$. Then, such a coalition does not get any information about keys K_j, for any $r \leq j < s$. Formally, it holds that:

$$H(\mathbf{K}_r, \ldots, \mathbf{K}_{s-1}|\mathbf{B}_1, \ldots, \mathbf{B}_m, \mathbf{S}_C, \mathbf{S}_D) = H(\mathbf{K}_r, \ldots, \mathbf{K}_{s-1}) \tag{2.7}$$

In *Definition 1*, Condition (1) states that every user U_i, from the broadcast B_j and its own personal key S_i, recovers the current session key K_j; however, personal keys and broadcasts alone do not give any information about any session key. Condition (2) means the group manager is able to revoke at most t users from the group. Conditions (3) characterizes the self-healing property: any two broadcasts are enough to recover all lost session keys for the "sandwich" sessions. In *Definition 2*, Condition (1) and Condition (2) describes the forward secrecy and backward secrecy respectively. Condition (3) defines the feature of resisting collusion. It states that a coalition of or less t revoked users and the newly joined users cannot obtain

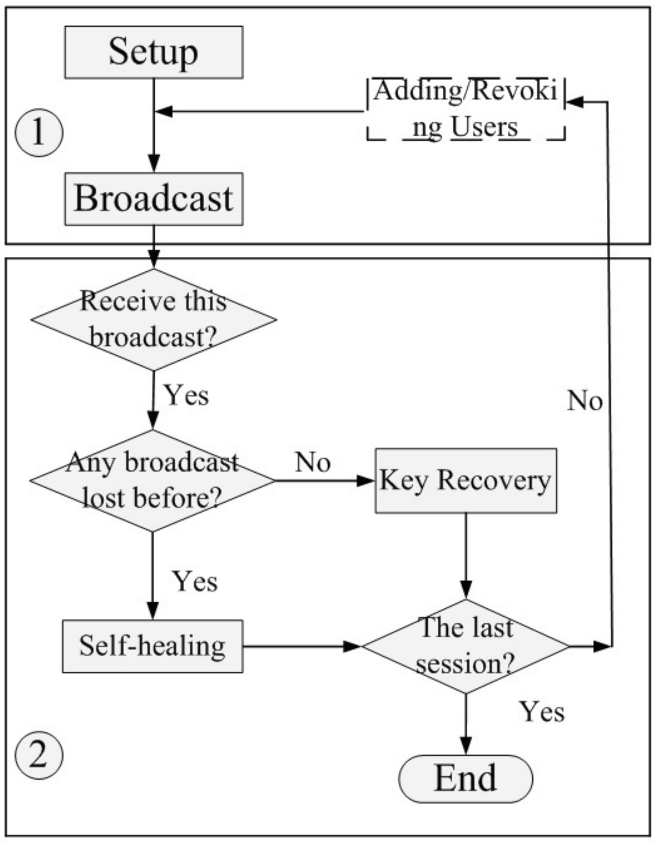


Figure 2.10: The basic procedures in a general self-healing key distribution

any information about the current session key.

2.5.3 The Procedures of Self-healing Key Distribution Schemes

Self-healing key distribution entails five basic procedures, namely *Setup*, *Broadcast*, *Key Recovery*, *Adding/Revoking Users*, and *Self-healing*. The basic procedures in a general self-healing key distribution are described in Figure 2.10. The entities in a self-healing key distribution scheme include a group manager and a great number of end users. The group manager is responsible for the procedures of *Setup*, *Broadcast* and *Adding/Revoking Users*. The end users perform *Key Recovery* and *Self-healing*. In Figure 2.10, The large pane 1 includes the operations performed by the group manager, while the large pane 2 includes all the operations performed by the end users. The small panes are operations which must be executed in each round, while the small dashed frames represent the operations which cannot be executed in some rounds of the scheme.

1. *Setup.* The group manager chooses and releases system parameters. It also generates and privately distributes personal keys to current group users. It is supposed that the personal keys are distributed in an off-line manner or there is a secure channel between the group manager and each user in this procedure. At last, the group manager chooses m session keys $K_1, \cdots, K_m$ from a fixed key space (for example F_p). These session keys are independent to each other and according to uniform distribution.

2. *Broadcast.* Suppose $|G_j|$ denotes the number of users in session j. For each session $1 \leq j \leq m$, according to the session group G_j, the group manager computes and broadcasts message B_j so that only authorized users can recover the session key from the broadcast message and their personal keys. At the same time, the construction of the broadcast message should meet the properties of forward and backward secrecy even the property of collusion resistance.

3. *Key Recovery.* When an authorized user $U_i \in G_j$ receives the broadcast message B_j, it can recover the session key for the j-th session using its personal key and the received broadcast message.

4. *Adding/Revoking Users.* When the group manager wants to add a new user from session j, it will generate and privately distribute a personal key to the new user so that the new user can recover session keys in subsequent sessions. Similarly, if the group manager wants to revoke a user from session j, it will add or delete some information related to the revoked user in the procedure of *Broadcast* so that the revoked user cannot recover the session keys in subsequent sessions. Both forward and backward secrecy, and the property of collusion resistance, should be guaranteed in this procedure.

5. *Self-healing.* If a user misses some broadcast messages before session j, when it receives the broadcast message in session j, it can recover all the lost session keys using its personal key and the broadcast B_j. The only requirement that must be satisfied, in order for the user to recover the lost keys,

is the membership in the group both before and after the sessions in which the broadcast messages containing the keys are sent.

Among these procedures, *Setup* occurs only during the period of initialization. *Broadcast* and *Key Recovery* are performed in each session. While *Adding/Revoking Users* does not necessarily occur in each session. *Self-healing* is triggered by the loss of broadcast messages. It only happens in the sessions after the broadcast messages have been lost.

2.5.4 Typical Self-healing Key Distribution Schemes

In this section, we present an overview of existing works in the area of self-healing key distribution according to their classification. Here we omit the proof of security of each scheme because the security of each scheme was proved in the correspondingly proposed paper. Nevertheless, we thoroughly analyze and compare the performance of schemes in the same classification.

Polynomial secret sharing based self-healing key distribution schemes

The idea of self-healing key distribution was introduced by Staddon *et al.* in the pioneering work [80]. Formal definitions, lower bounds on the resources as well as some constructions of self-healing key distribution scheme were proposed. The group manager, at the beginning of each session, in order to provide a key to each user of the group, sends packets over an unreliable broadcast channel. Combined with the pre-distributed secret key, every authorized user can recover the session key from the packets. In this way, the group manager can launch multiple sessions during a certain time interval, by adding/removing users to/from the initial group. The self-healing feature in the key distribution scheme enables a group manager to distribute session keys to a dynamic group over an unreliable network. If at the beginning of a certain session some broadcasted packets get lost, then users are still capable of recovering the session key for that session simply by using the packets they have received at the beginning of a previous session and the packets they will receive at the beginning of a subsequent one, without requesting additional transmission from the group manager. The only requirement that a user must satisfy in order to recover the lost keys through self-healing is its membership in the group both before and after the sessions in which the broadcast messages containing the keys are sent. This approach reduces network traffic and the work load on the group manager as well as the risk of user exposure through traffic analysis. This has made self-healing key distribution a hot research topic.

Liu *et al.* generalized the definitions in [80] and presented several constructions in [81]. The scheme introduced a novel key distribution scheme firstly by using both a revocation polynomial and a mask polynomial. Then they added the self-healing feature to the basic key distribution scheme, thereby greatly reducing communication and storage overhead. Furthermore, they developed two techniques that allow a trade-off between the broadcast message size and the recoverability of lost session keys. The first technique deals with possibly frequent but short-term communication failures and the second technique targets situations where there are relatively long-term but infrequent communication failures. The two methods further reduced the size of broadcast messages in these two conditions respectively. An unreasonable requirement of [81] is that the set of revoked group users must change monotonically. That is, $R_{j_1} \subseteq R_{j_2}$ for sessions j_1 and j_2 $(1 \leq j_1 < j_2 \leq m)$. Otherwise, a group user who is revoked in session j_1, rejoins the group in a later session j_2, can recover the key for session j_1. Therefore, the scheme is prone to a rejoin-collusion attack and does not allow temporary revocation.

More *et al.* in [89] addressed the three problems in [80] using a sliding window mechanism. The three problems are inconsistent robustness, high overhead and expensive maintenance cost. The scheme achieved good performance. Above all, the use of a sliding window mechanism makes error recovery consistently robust. In addition, the group manager is entitled to spread the cost of personal key distribution over multiple sessions, rather than having to distribute new personal keys to all users at the same time. Furthermore, by reusing the masking polynomial, personal key storage and broadcast size are dramatically reduced.

Blundo *et al.* in [83] proposed a new mechanism for implementing self-healing key distribution. Moreover, they described a secure and efficient construction which has optimal memory storage and communication complexity. Shortly afterwards, in [87] they presented an attack on the first construction in [80]. Then, they proposed a new self-healing key distribution scheme, which requires low storage and communication overhead. Finally, they slightly modified the security model, in order to extend the self-healing key distribution model, and proposed a scheme which enables a user to recover from a single broadcast message all those keys associated with sessions in which it is a user of the communication group. They pointed out two problems of the long-lived *Construction 5* in [80]. In [80], the values that are used for computing the instances of the personal key are sent by means of a single broadcast message at the beginning of each new set of m sessions. Since the network is not reliable, if some user does not receive such a message, it will retreat from the corresponding m sessions. This problem is solved by sending with each broadcast the values that are used for evolving the personal key in each session. For the solution of the first problem, the involved cost of modular exponentiation operations is too high to be affordable for low-cost wireless networks. This solution turned out to be infeasible. The second problem lies in the join operation in the presence of new users. It cannot be solved by slightly modifying *Construction 5* given in [80] and *Scheme 4* given in [87] in order to enable a secure join. It is still an open problem.

Hong *et al.* proposed a new self-healing key distribution scheme in [82]. It is one of the efficient unconditionally secure self-healing key distribution schemes. As the original self-healing key distribution scheme, it includes five procedures:

- *Setup.* The group manager randomly chooses m t-degree polynomials $s_1(x)$, $\ldots, s_m(x) \in F_p[x]$ and m session keys $K_1, \ldots, K_m \in F_p$. Both polynomials and session keys are independent on each other and according to uniform distribution. Then, the group manager privately sends personal key $S_i = \{s_1(i), \ldots, s_m(i)\}$ to U_i ($i = 1, \ldots, n$).

- *Broadcast.* Let $W_j = \{r_1^j, \ldots, r_{w_j}^j\}$ be the set of revoked user IDs for sessions in and before j such that $|W_j| = w_j \leq t$, and let $r_j(x) = (x - r_1^j), \ldots, (x - r_{w_j}^j)$. The broadcast message for the j-th ($j = 1, \ldots, m$) session is in this form:

$$B_j = \{P_1(x), \ldots, P_j(x)\} \cup \{W_i\}_{i=1,\ldots,j},$$

where $P_j(x) = r_j(x)K_j + s_j(x)$.

- *Key Recovery.* When a user $U_i \in G_j$ receives the j-th broadcast message, it evaluates the polynomial $r_j(x)$ at point i, and computes the current session key by computing

$$K_j = \frac{P_j(i) - s_j(i)}{r_j(i)}.$$

- *Adding/Revoking Users.* When the group manager wants to add a new user starting from the $r-$th session, it gives an unused unique identity $i' \in F_p$, computes personal keys corresponding to the current and future sessions $\{s_k(i')\}$ ($k = r, \ldots, m$) and privately sends the keys to this new user. When the group manager wants to revoke a user $U_{i'}$ starting from session s, it adds the identity $i' \in F_p$ to W_j for $j = s, \ldots, m$.

- *Self-healing.* For any U_i that is a user in session r and s ($1 \leq r < s \leq m$), it can recover $\{P_r(i), \ldots, P_s(i)\}$. Using the method described in Key Recovery, U_i can subsequently recover the whole sequence of session keys $K_r, \ldots, K_s$.

Compared with previous schemes, this scheme has good combined performance. It is optimal in terms of user memory storage and more efficient in terms of communication complexity.

The schemes in [80, 81, 82, 83, 84, 87, 88, 89] focus on unconditionally secure self-healing key distribution. The definitions and constructions were stated in terms of the entropy function. Blundo *et al.* in [100] analyzed current definitions of self-healing key distribution. They showed that no protocol can achieve the security requirements stated in [80, 81] and identified where the proposed schemes fail. They also showed that a previously derived lower bound on the size of the broadcast messages that the group manager has to send in order to establish session keys, proved in [80] and also used in [81], does not hold. After analysis, they proposed a new definition of self-healing key distribution and showed that it could be achieved by concrete schemes. Some lower bounds on the resources, such as user memory storage and communication complexity, required for implementing such schemes, were given finally and showed that these lower bounds were tight through simple constructions.

Dutta claimed that they realized unconditionally secure self-healing key distribution schemes in [101, 102]. The storage overhead in these two papers is $(t + 1)logp$, where t is the maximum number of compromised users and p is the key space. However, this claim cannot hold. Indeed, in [80, 87, 90], it has been proved that a lower bound on the size of the storage overhead is equal to $(m - j + 1)logp$, where m is the life-time of the network, j is order of the current session, and $logp$ is the size of the distributed key. According to the above-mentioned bound, one cannot design an unconditional secure self-healing scheme for m sessions by giving to each user a $(t + 1)logp$-size secret key which is updated by a broadcast message. The schemes in [80, 87, 90] are referenced by both [101, 102], but it seems that the authors missed this lower bound.

Zou and Dai in [88] proposed a new self-healing scheme based on a novel concept of access polynomial. It overcomes some shortcomings of existing schemes yet still possesses all their advantages. We should point out that the communication overhead in [88] comes from broadcast B_j which is comprised of $2(m+1)$ polynomials. As far as the polynomial $P_j(x)$ is concerned, $P_j(x) = A_j(x) \cdot S_j(x) + H(x)$. Both $S_j(x)$ and $H(x)$ are t degree polynomials, and the degree of $A_j(x)$ amounts to $(|G_j| + 1)$ where $|G_j|$ is the number of users in current communication group. Generally speaking, $|G_j|$ is larger than t. Therefore, the claim that communication overhead is $O(mt)$ is incorrect.

Tian *et al.* introduced and improved an secret sharing scheme in [85]. Then, they applied the secret sharing scheme to the design of a self-healing key scheme. The new scheme possesses several desirable properties. Firstly, the scheme reduces storage overhead of personal key to a constant. Secondly, the scheme cancels the requirement of secure channel in the *Setup* phase. In addition, the long-lived scheme is much more efficient than those in [80, 87]. However, efficiency is improved at the cost of relaxing the security slightly. The scheme is a computationally secure scheme. By introducing a one-way key chain, Dutta *et al.* proposed two constructions of scalable self-healing key distribution with t revocation capability in [86]. The schemes greatly reduce both communication and computation overheads without increasing storage overhead. At the same time, forward and backward secrecy are achieved. However, there is a fatal defect in their constructions. The collusion between the newly joined users and the revoked users can recover all the session keys to which they were not authorized.

In terms of storage overhead, the schemes require that each user stores a personal key for each session. It comes from the procedure of *Setup* and after receiving the session key distribution broadcast. Communication overhead comes from the procedure of *Broadcast*. Generally speaking, the broadcast message for the j-th session consists of identifiers of a set of revoked users. Since the user identities can be selected from a small finite field, one can ignore the communication overhead incurred by the identifies of all the revoked users set [82]. Computation overhead is introduced by the procedures of *Key Recovery* and *Self-healing*. It is supposed that the group manager has more computation resources while end users have limited computational ability. Therefore, we consider the computation cost only at user end. For the different cryptographic primitives on which the schemes are based, the computation overhead varies.

To clarify the performance of the proposed schemes, we present a thorough comparison of them. Table 2.3 summarizes the comparison in terms of storage, communication, computation overheads and several security parameters. We use C to denote Construction and S to denote Scheme, for example, $C3$ in [80] denote the *Construction 3* in [80]. We suppose the life-span of schemes is m sessions. j represents the order of a session, t represents the maximum number of users that can be revoked and $logp$ represents the length of private and session keys. The storage overhead is measured by the length of the personal key. The communication overhead comes from broadcast messages. For a user U_i at the j-th session, the computation overhead is incurred by recovering all previous session keys up to the j-session (worst case) by self-healing mechanism [86]. The key recovery operations include computing one or more points on polynomials (such as operations in [81, 82]) or recovering a t degree polynomial by using Lagrange formulation (such as operations in [83, 85, 86, 87]) or both (such as operations in [80]). Computing a point on a t-degree polynomial requires at most t multiplication operations because division can be regarded as multiplication when we calculate computation overhead. Compared with the overhead produced by multiplication, the overhead produced by addition and hash operations can be ignored. Therefore, the computation cost for each user is $2t + 1$ of *Construction 1* in [86], whereas the computation complexity of the scheme of *Construction 1* in [82] is $j(2t + 1)$. Recovery of a t degree poly-

Table 2.3: Performance comparison of the different polynomial secret sharing-based self-healing key distribution schemes

Method	Storage overhead	Communication overhead	Computation overhead	Forward and backward secrecy	Collusion resistance	Secure channel in initialization
C3 in [80]	$(m-j+1)^2 logp$	$(mt^2+2mt$ $+m+t)logp$	$2mt^2+3mt-t$	Both, t-wise	At most t	Yes
S3 in [81]	$2(m-j+1)logp$	$[(m-j+1)t+$ $(m+1)]logp$	$mt+t+2tj+j$	Both, t-wise	At most t	Yes
S2 in [87]	$(m-j+1)logp$	$(2tj+j)logp$	$2j(t^2+t)$	Both, t-wise	At most t	Yes
C1 in [82]	$(m-j+1)logp$	$(tj+j-t-1)logp$	$j(2t+1)$	Both, t-wise	At most t	Yes
[83]	$(m-j+1)logp$	$tj(t+1)logp$	$j(2t^2+3t)$	Both, t-wise	At most t	Yes
[85]	$logp$	$j(t+1)logp$	$j(logp+2t^2+2t)$	Both, t-wise	At most t	No
C1 in [86]	$(m-j+1)logp$	$(t+1)logp$	$2t+1$	Both, t-wise	No	Yes
C2 in [86]	$(m-j+1)logp$	$(t+1)logp$	$2(t^2+t)$	Both, t-wise	No	Yes

nomial by using Lagrange formulation requires $2\{(t+1)^2-(t+1)\}=2(t^2+t)$ multiplication operations. Thus the computation overhead of *Scheme 2* in [87] is $2j((t^2+t))$ and that of *Construction 2* in [86] is $2((t^2+t))$. t-wise for forward and backward secrecy means these schemes can keep forward and backward secrecy for the coalition of at most t unauthorized users. The capability of resisting collusion means the scheme is secure even at most t newly joined users and revoked users collude to recover the session keys that they do not authorized to access. To date, the scheme in [85] has been the only self-healing key distribution scheme without the requirement of a secure channel between members and the group manager in the initialization stage.

Vector space secret sharing based self-healing key distribution schemes

Sáez in [90] considered applying vector space secret sharing instead of polynomial secret sharing schemes to a realize self-healing key distribution scheme. The scheme makes use of general monotone decreasing structures for the family of subsets of users that can be revoked, instead of a threshold one. Thus, the scheme achieves more flexible performance than do the polynomial secret sharing based schemes. Another advantage of the scheme is that the distance between the broadcasts used to recover the lost one is variable. The reason for this modification is to allow adjustment of the length of broadcasts according to the condition of networks. In the same year, Sáez in [91] considered the possibility that a coalition of users sponsor a user outside the group for one session. Firstly, the formal definition and some bounds on the required amount of information were given. Then a general construction of a family of self-healing key distribution schemes with sponsorization was presented. The sponsorization mechanism strengthens the robustness of the scheme. An authorized subset of users in the group has the ability to invite a new user to join the group without the help of the group manager. Both [90] and [91] are unconditionally secure schemes. The model in [91] not only includes *Definition 1* and *Definition 2*, but also includes *Definition 3*.

Definition 3. The scheme has the property of sponsorization. This means that the three following properties are satisfied:

1. Every user $U_l \in G_j$ can generate a proof of sponsorization P^j_{li} to sponsor a user $U_i \notin G_j$ for session j using its personal key. In other words:

$$H(P^j_{li}|S_l) = 0. \tag{2.8}$$

2. A user $U_i \notin G_j$ that receives enough sponsorization from a subset of users $A \subset G_j$ with $A \in \Gamma$ can compute the key K_j in the same conditions that users in G_j. That is: for $A \in \Gamma$, $A \subset G_j$, $i \notin G_j$ and $r \leq j \leq s$, there exists

$$H(K_j|P^j_{Ai}, B_r, B_s) = 0 \tag{2.9}$$

3. Suppose that a coalition of users $U_{i_1}, \dots, U_{i_u} \notin G_j$, not revoked before session j, have received sponsorization from subsets of users $C_1, \dots, C_u \notin \Gamma$ respectively, with $C_1 \cup \ldots \cup C_u = \{U_{l_1}, \dots, U_{l_v}\} \subset G_j$. This action is performed in such a way that users $U_{l_1}, \dots, U_{l_v}$ sponsor subsets of users $D_1, \dots, D_v \in \mathcal{S}$ respectively, with $D_1 \cup \ldots \cup D_v = \{U_{i_1}, \dots, U_{i_u}\} \subset U - G_j$; therefore $P^j_{C_1 i_1}, \dots, P^j_{C_u i_u} = P^j_{l_1 D_1}, \dots, P^j_{l_v D_v}$. In these conditions, such a coalition does not get any information about the value of key K_j. Formally, it holds that:

$$H(K_j|P^j_{C_1 i_1}, \dots, P^j_{C_u i_u}, B_r, B_s) = H(K_j) \tag{2.10}$$

for $C_1, \dots, C_u \notin \Gamma$, $D_1, \dots, D_v \in \mathcal{S}$, such that $P^j_{C_1 i_1}, \dots, P^j_{C_u i_u} = P^j_{l_1 D_1}, \dots$,

$P^j_{l_v D_v}$, $C_1 \cup \ldots \cup C_u = \{U_{l_1}, \ldots, U_{l_v}\} \subset G_j$, $D_1 \cup \ldots \cup D_v = \{U_{i_1}, \ldots, U_{i_u}\} \subset U - G_j$ and $r \leq j \leq s$.

In *Definition 3*, Condition (1) describes the property of sponsorization: the information used to sponsor is computed from the personal key. Condition (2) describes the fact that the information obtained from enough sponsorizations with the correspondent broadcast allows to compute the personal key of the session. Condition (3) describes the security requirement: a coalition of users outside G_j sponsored by an insufficient number of users cannot obtain any information about the value of the key K_j. The key remains secure even if every user receives sponsorization of a coalition in $\mathcal{S}$.

In fact, polynomial secret sharing based self-healing key distribution scheme is a particular case of vector space secret sharing based scheme. with polynomial secret sharing based schemes, the access structure is fixed to the degree of the underlying polynomial. Therefore, vector space secret sharing based schemes achieve more flexible properties.

The schemes [90, 91] use the same vector space secret sharing mechanism. The storage, communication and computation overheads are very similar. In terms of storage overhead, each user has to store a personal key of size $(m - j + 1)logp$ in [90, 91]. The storage overhead is optimal with respect to Theorem 1 in [90]. The communication overhead comes from the broadcast which depends on the particular function ψ used. According to [90], the broadcast can be divided two parts. The first part of broadcast is defined as (x_j, y_j). The total number of broadcast bits is $(1 + (t + 1)(m - 1 - 0.5t))logp$. According to [91], B^1_1 and B^1_m have $0.5tmlogp$ bits and B^1_j for $j \neq 1, m$ has $0.5t(m - 1)logp$ bits. Then the total number of broadcast bits is $0.5t(m^2 - m + 2)logp$. The second part of the broadcast in [90, 91] is composed of the identities of revoked users and its purpose is to perform the rejection action as well as the recovery of the session key. Its length depends on the history of rejected subsets. Since the user identities can be chosen from a small finite field, we can ignore the communication overhead for the broadcast of all those revoked sets. The computation overhead of [90, 91] is $2\sum_{l=1}^{j}(t_l^2 + t_l)$ because the user U_i has to compute $\psi(D)P_j = \sum_{k \in W_j \cup \{i\}} \lambda_k \psi(k) P_j$ for each lost key. Here we use Table 2.4 to highlight the features of the two schemes.

Table 2.4: Comparison of features of vector space secret sharing based self-healing key distribution schemes

Scheme	Storage overhead	Communication overhead	Computation overhead	Resisting collusion	Sponsorization
[90]	$(m-j+1)logp$	$(1+(t+1)$ $(m-1-t/2))logp$	$2\sum_{l=1}^{j}(t_l^2+t_l)$	Yes	No
[91]	$t(m-j+1)logp$	$0.5t(m^2+m+2)logp$	$2\sum_{l=1}^{j}(t_l^2+t_l)$	Yes	Yes

SDR-based self-healing key distribution schemes

SDR [103] is a stateless rekeying method. In this protocol, a key server maintains a logical key tree and every user is mapped to a leaf node of the key tree. In each rekeying operation, the key server partitions the current group into a minimal number of subsets, and then encrypts the new group key with the common key of each subset respectively. In this scheme, the communication overhead is independent of the group size; thus, the scheme is scalable. One of the advantages of the scheme is that there is no dependency between the keys used in different rekeying operations. In order to decode the current group key, a user only needs to receive the keys that are transmitted by the key server during the current rekey operation. This property makes the SDR scheme very attractive for secure multicast applications where users may go off-line frequently or experience burst packet losses.

The SDR method performs well in key recovery operations and is secure against the collusion of any number of revoked users. By contrast, the polynomial-based key distribution scheme proposed by Liu *et al.* in [81] has the similar message size and has a feature whereby group users can recover the session key on their own under certain conditions. However, this protocol has a constraint which may limit its application. That is, the maximum number of users that can be revoked during the life-long time of networks has to be pre-determined and must not be exceeded for the sake of security. The maximum number of users that can be revoked depends on the degree of the polynomial. The higher the degree of the polynomial, the larger is the number of users that can be revoked, and the higher is the communication and computation overhead and vice versa.

Zhu *et al.* first addressed the addition of a self-healing feature to SDR in [96].

The key idea is to bind the ability of a user to recover a lost group key to its membership duration. They used a one-way hash key chain such that revoked users and new users cannot collude to recover the keys that they should not know. The notions are different from those used in the general self-healing key distribution schemes. The scheme can be explained as follows:

1. In each group rekeying, the group manager generates a key chain of size $(m+1)$. Let the keys in the key chain generated for the rekeying at $T(i)$ be $K^m(i), \ldots, K^1(i), K^0(i)$, where $K^0(i) = H(K^1(i)) = H^2(K^2(i))) = \ldots = H^m(K^m(i)))$ and H is a one-way hash function such as SHA-1. Due to the one-way property of the hash function, a user who knows K_i^j can compute all the keys $K^{j-1}(i), \ldots, K^0(i)$ independently, but it can not compute all of the keys $K^{j+1}(i), \ldots, K^m(i)$. $K^0(i)$ is the group key that all the users should use for data encryption between $T(i)$ and $T(i+1)$.

2. The users in the group are considered to be partitioned into $m+1$ subgroups, and depending upon their membership duration, each subgroup is associated with a separate key from the one-way key chain generated in the first step. Specifically, $K^j(i)$ is the key intended for the users that joined the group at $T(i-j)$ for $0 \leq j < m$, and $K^m(i)$ is the key intended for users that joined at or before $T(i-m)$.

3. The group manager broadcasts m encrypted keys as shown below:

$$\{K^0(i-m)\}_{K^0(i-m-1) \oplus K^m(i)}, \ldots, \{K^0(i-1)\}_{K^0(i-2) \oplus K^1(i)}. \qquad (2.11)$$

The communication overhead of this protocol is very small. To allow an authorized user to recover the previous s group keys, the number of additional encryption keys to be transmitted is at most $3s$.

Bohio *et al.* in [97] considered incorporating the self-healing feature into the SDR rekeying method. Some optimization techniques that can be used to reduce the overhead caused by the self-healing capability were proposed in the paper. In addition, the idea of mutual-healing was discussed. One motivation behind mutual-healing is that, if a node has missed a key updating message, it does not have to wait until the next update broadcast to recover the previous session key; instead, it can acquire assistance from its neighboring nodes to recover that key instantly. However, the SDR scheme itself has a limitation. The positions of the revoked users in the tree can not be reused. Hence, deficiency discounts the feasibility of SDR-based self-healing key distribution schemes.

Hash chain based self-healing key distribution schemes

Jiang *et al.* proposed an efficient self-healing group key scheme with time-limited node revocation based on dual directional hash chains (DDHC) in [94]. The performance of the proposed scheme in a poor broadcast channel was evaluated by both theoretical analysis and numerical results. The result shows that the scheme tolerates high channel loss rate, and hence achieves a good balance between performance and security. Therefore, it is suitable for wireless networks.

As aforementioned, the schemes [86, 92] are based on a hash chain. The difference is that [86] utilizes a polynomial secret sharing masking mechanism thus the maximum number of the revoked users is constrained by the degree of polynomial while [92] adopts a general access structure that achieves flexibility. The proposed self-healing technique enables better performance over previous approaches in terms of storage, communication and computation complexity. They provided proof of the security of their scheme under an appropriate security framework. The proof shows that the scheme is computationally secure and achieves both forward and backward secrecy. Kausar *et al.* proposed a simpler self-healing key distribution scheme in [95]. In their scheme, all broadcast messages are masked with XOR operation. The scheme is efficient in terms of storage and computation overhead. However, the operation of dealing with a compromised node is too impractical to afford. If one compromised node is detected, all nodes are forced to be re-initialized.

Due to the efficiency of hash function, these schemes greatly reduce both communication and computation overheads. At the same time, forward and backward secrecy are achieved. However, there is a fatal defect in these constructions. The collusion between the newly joined users and the revoked users will recover all the session keys to which they are not entitled. Tian *et al.* in [93] dealt with the problem gracefully. They assigned each user U_i a pre-arranged life cycle (s_i, t_i).

Random numbers which are related to the users' membership are used in the procedure of *Key Recovery*. Compared with the scheme in [92], the scheme in [93] not only maintains forward and backward secrecy, but also resists collusion between the newly joined users and the revoked users. As far as we know, this is the first time that a scheme has been proposed to withstand a collusion attack against hash-chain-based self-healing key distribution. The schemes in [6, 7] slightly reduced storage and computation overheads of the scheme in [93] with a polynomial secret sharing masking mechanism and vector space secret sharing masking mechanism, respectively. In our recent research, we found that the schemes [6, 7, 93] can only resist collusion between the users whose life cycles have finished, and newly joined users. They in fact could not resist collusion attacks of the newly joined users and the revoked users whose life cycles have not yet expired. This problem remains unsolved.

The hash chain itself cannot be the only cryptographic primitive of a self-healing key distribution scheme. It forms a self-healing key distribution scheme together with other cryptographic primitives. For example, the scheme [94] is based on hash function and MAC, the scheme [95] is based on hash function and XOR operation, the scheme in [6, 86] is based on hash function and polynomial secret sharing, while the schemes in [7, 92, 93] are based on hash chain and vector space secret sharing. These schemes have some common characteristics while each has its own special features as well. We summarize the features of one-way hash function based self-healing key distribution schemes in Table 2.5 without [94]. In fact, the scheme in [94] is a totally different self-healing group key distribution mechanism. It supposes that there is a buffer at each sensor in order to recover the lost session keys. Each user has to send the *Request Key* message to explicitly request the current rekeying message for the current session. That is, this self-healing mechanism involves the interaction between the user and the group manager. The communication overhead varies at a different user end and it depends on how many rekeying messages the user fails to receives and how large the renewal interval t is. Because each message is encrypted with a Traffic Encryption Key (TEK), the computation overhead varies according to the encryption mechanism. The authors did not suggest which encryption mechanism is suitable for the scheme. Therefore, we exclude the scheme [94] from the Table 2.5. In Table 2.5, the storage overhead of a user U_i

Table 2.5: Summary of the one-way hash function based self-healing key distribution schemes

Scheme	Cryptographic primitives	Storage overhead	Communication overhead	Computation overhead	partial collusion resistance
C1 of [86]	hash function and polynomial secret sharing	$(m-j+1)logp$	$(t+1)logp$	$2t+1$	No
[92]	hash function and vector space secret sharing	$2(t_i-s_i+1)logp$	$(t_j+1)logp$	$2({t_j}^2+t_j)$	No
[95]	hash function and XOR operation	$(m-j+2)logp$	$jlogp$	constant	No
[93]	hash function, vector space secret sharing and XOR operation	$(2t_i-2s_i+3)logp$	$(t_j+1)logp$	$2({t_j}^2+t_j)$	Yes
C1 in [6]	hash function, polynomial secret sharing	$2(t_i-s_i+1)logp$	$(t+1)logp$	$2t(t+1)$	Yes
C2 in [6]	hash function, polynomial secret sharing	$2(t_i-s_i+1)logp$	$(t+1)logp$	$2(t+1)$	Yes
[7]	hash function, vector space secret sharing	$(t_i-s_i+t+2)logp$	$(t_j+1)logp$	$2({t_j}^2+t_j)$	Yes

is the size of its personal key which is closely related to its life cycle (s_i, t_i). The communication overhead is produced by the broadcast messages for Key Recovery and Self-healing. The communication overhead at the $j-$th session in [7, 92, 93] is $(t_j + 1)logp$ bits, where $t_j = |W_j \cup R_j|$, $R_j \notin \Gamma$ is the set of all revoked users for sessions in and before j and $W_j \subset U \backslash G_j$ with minimum cardinality such that $W_j \cup R_j \in \overline{\Gamma_0}$. The computation overhead is measured by the number of multiplications in the underlying field. In terms of computation overhead of the scheme in [95], it only includes hash and XOR operations, so the computation overhead at user end in [95] is a constant compared with that of other schemes such as those in [92] and [93]. The computation overhead of Key Recovery in [7, 92, 93] is $2(t_j^2 + t_j)$. This is the number of multiplication operations needed to recover $\psi(D)$ by using equation $\psi(D) = \sum_{k\in W_j\cup\{i\}} \lambda_k \psi(k)$ [7]. This is because self-healing mechanism only involves hash and addition operations. Compared with the computation overhead produced by multiplication operations, the computation overhead resulting from hash and addition operations can be ignored.

Bilinear pairing based self-healing key distribution schemes

Although a formal definition of ID-based cryptosystems has been known for a while [104], the first fully functional one fitting all the requirements of an ID-based cryptosystem appeared only quite recently in [105]. Inspired by the idea of [105], Du *et al.* proposed a broadcast encryption scheme for key distribution in [106]. We extended the broadcast encryption scheme for key distribution to a self-healing key distribution scheme in [98]. This scheme has some desirable properties. In terms of storage overhead, each user only stores the group manager's public key and its public/private key pair. The storage overhead for each user is a constant.

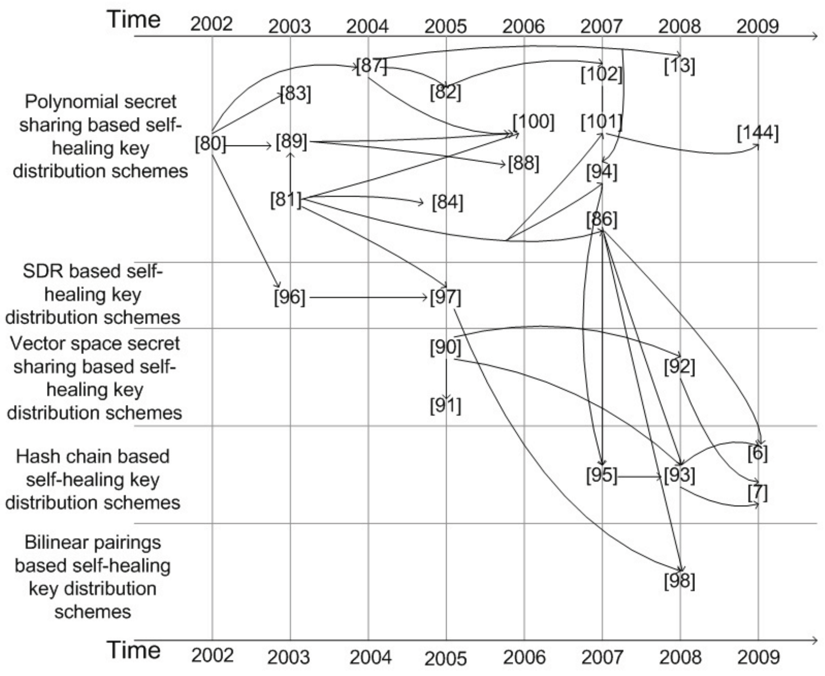
This scheme is collusion-free for any coalition of unauthorized users, including the revoked users and the newly joined users. In our scheme, if a user wants to obtain the session key K_i, it should compute $e(X_1, x_1 S_i)$ in the procedure of Key Recovery with personal key S_i. Therefore, only the authorized users can recover the session key. In addition, any coalition of non-authorized users cannot derive the private keys of authorized users from their public keys. The personal private key has nothing to do with the number of revoked users and can be reused as long as it is not disclosed. As far as we know, it is the first self-healing key distribution scheme using bilinear pairings.

2.5.5 Compositive Analysis of Self-healing Key Distribution Schemes

According to the state-of-the-art research, self-healing key distribution is the most suitable way to establish session keys for large and dynamic group communication over unreliable wireless networks. It is one branch of the key management schemes and has received much attention. In this section, we conduct a comprehensive analysis of existing schemes.

Although unconditionally secure self-healing key distribution schemes can fulfill strict security requirements, they should meet some lower bounds in terms of storage and communication overheads. Computationally secure self-healing key distribution schemes relax the security slightly and have some desirable features, such as constant storage overhead. As far as the cryptographic primitives are concerned, self-healing key distribution schemes are limited to polynomial secret sharing, vector space secret sharing, hash chain, and the SDR mechanism. We intended to design a bilinear pairing based self-healing key distribution scheme and have made a little progress.

Figure 2.11 provides a taxonomy of papers on self-healing key distribution schemes. The figure is a directed acyclic graph where nodes represent papers. Di-



Time
2002
2003
2004
2005
2006
2007
2008
2009
Polynomial secret sharing based self-healing key distribution schemes
SDR based self-healing key distribution schemes
Vector space secret sharing based self-healing key distribution schemes
Hash chain based self-healing key distribution schemes
Bilinear pairings based self-healing key distribution schemes
[87]
[13]
[102]
[83]
[82]
[100]
[101]
[144]
[80]
[89]
[88]
[94]
[81]
[84]
[86]
[96]
[97]
[90]
[92]
[91]
[6]
[95]
[93]
[7]
[98]

Figure 2.11: Taxonomy of the papers on self-healing key distribution schemes

Table 2.6: Summary of different classes of self-healing key distribution schemes

Class	Features
Polynomial secret sharing	Advantage: It performs easily; Disadvantage: High computation overhead; the maximum number of revoked users is constrained to the degree of the polynomial.
Vector space secret sharing	Advantage: A monotone decreasing family of rejected subset of users; Disadvantage: Some public mapping may incur large communication overhead.
Hash chain	Advantage: Efficient in computation; Disadvantage: can not resist collusion attack.
SDR	Advantage: Do not require re-distribution of any personal keys; can revoke any number of users; Disadvantage: The communication depends on the number of subsets, which increases with the number of joined and revoked users.
Bilinear pairings	Advantage: Efficient in computation; Disadvantage: Communication overhead is high in large-scale networks

rected edges show predecessor/successor relations among solutions provided by the papers. There is an edge from one paper to another one if the latter improves on the solution proposed by the former. Style of an edge represents the problem for which destination paper provides improvements. Existing papers are ordered over a horizontal time axis according to their publication dates. The vertical axis groups papers under five categories: (1) polynomial secret sharing, (2) vector space secret sharing, (3) SDR, (4) hash chain, and (5) bilinear pairing based self-healing key distribution schemes. The style of edge between two nodes represents the problem for which an improvement is provided. A paper may based on more than one cryptographic primitive; therefore, corresponding work may be reached from more than one origin paper, and there may be more than one edge with different styles in between two papers.

Table 2.6 summarizes the strengths and shortcomings of each class of self-healing key distribution schemes.

2.5.6 Security Issues of Self-healing Key Distribution Schemes

Every category has its own features, requirements and goals. In summary, polynomial secret sharing is the most common technique used to realize self-healing key distribution. It performs easily. However, the maximum number of revoked users is constrained to the degree of the polynomial. In addition, in the procedure of *Key Recovery*, Lagrange's interpolation formula should be used in order to recover the secret polynomial thus leading to much computation overhead.

Vector space secret sharing based self-healing key distribution schemes consider a monotone decreasing family of rejected subset of users instead of a monotone decreasing threshold structure. This general case makes the self-healing scheme more flexible and suitable for practical application. In addition, the constructions are general in the sense that they depend on the particular public mappings and for different choices of public mappings, different self-healing key distribution schemes can be attained. The length of broadcast messages also depends on the particular function.

A hash chain has many elegant features and can be used to design self-healing key distribution schemes. The constructions do not need to send the recorders of revoked subsets of users in order to perform self-healing, thereby reducing the communication cost. Both forward and backward secrecy and partial collusion resistance property can be assured. However, the collusion of the revoked nodes whose life cycles have not expired and the newly joined nodes, can recover the session keys to which they are not entitled.

One of the remarkable properties of SDR-based self-healing key distributions is that the personal keys of users are independent of the number of sessions. Because of this property, the SDR-based schemes do not require a redistribution of any personal keys after the number of sessions exceeds the estimation. Another remarkable property of SDR schemes is that they can revoke any number of users and remain secure against their collusion. As in the original SDR algorithm, the number of subsets increases when users join and leave. The communication complexity depends upon the number of additional subsets. The number of additional subsets depends on the group size, the number of the newly joined users and the revoked users in each rekeying period, and the value of session number.

Tian *et al.* presented a self-healing key distribution scheme by using bilinear pairings. The scheme achieved several nice features as aforementioned. However, the communication overhead increases with the number of authorized users in the communication group. How to reduce the communication overhead is an interesting problem yet to be solved.

2.6 Summary of Common Problems

The previous research investigations on key management for WSNs focus mainly on reducing overhead based on a hypothetical network model, rather than on a specific application. Some gaps in the area of key management for WSNs, are yet to be thoroughly investigated.

1. Lack of support for different types of communication. Most key management schemes for group communication in WSNs address only session key establishment and renewal. Almost all the key predistribution schemes consider only the establishment of pairwise keys between sensor nodes. There are still several unsolved problems in the key predistribution field. For example, path keys do not exclusively belong to two end nodes. All the intermediate nodes on the path know the pairwise keys forwarded by them; the compromise of a limited number of nodes may lead to the exposure of the whole key pool. Most importantly, there is no way to establish different types of keys for different kinds of communication requirements. A single key cannot meet the different communication requirements in WSNs, especially in hierarchical WSNs.

2. There is no feasible PKC-based/hybrid key management scheme for HWSNs. Although many researchers have demonstrated the feasibility of ECC-based public key generation from the hardware or software perspectives, rarely do current works propose a complete key management infrastructure using Public Key Cryptography. Traditional key management scheme for wired network cannot be directly transplanted to HWSNs given the unique attributes of the latter. As far as we know, SACK-P is the first trial of a PKC-based key management scheme for HWSNs. In fact, it cannot be seen as an original PKC-based key management scheme; it is a direct transformation of the basic SACK scheme which is based on Symmetric Key Cryptography. Public Key Cryptography has incomparable advantages over Symmetric Key Cryptography in key management and authentication. Nevertheless, great effort has to be put into making PKC-based/hybrid key management a reality in HWSNs.

3. Deficiencies of authentication. Existing authentication schemes have at least one or several of the following shortcomings: high computation or commu-

nication overhead, no resilience to nodes compromises, delayed authentication, loose or even strict time synchronization, and absence of scalability. Lightweight authentication and Public Key Cryptography based authentication are ideal alternative options for WSNs while the unique characteristics of WSNs should be given full consideration. Current literature is mainly concerned with authentication of communication content which happens after key establishment. In fact, authentication is also indispensable during the process of key establishment.

4. High-cost and limited capability of fault-tolerance. Due to the nature of WSNs, they are inherently susceptible to packet loss and node failure. Much current research on group management has been dedicated to minimizing storage, computational and the communication overhead to meet scalability. However, little attention has been paid to the robustness of these protocols. Most of the current key management protocols are not designed to cope with failures. However, such failures may block the normal run of key management protocols which are the building block of the whole security architecture. Self-healing key distribution is an ideal mechanism to deal with packet loss in session key distribution. However, problems such as robustness inconsistence remain unsolved. Existing countermeasures against node failure are either under stringent assumptions or are too costly to be feasible. To date, node failure tolerance property of pairwise key establishment protocols has not been addressed. How to maintain an acceptable trade-off between fault-tolerance and redundancy is still an open problem.

2.7 Conclusion

In this chapter, we carried out a survey of existing literature. We evaluated existing literature critically with a view to analyzing and assessing each category. We first surveyed existing key management schemes for WSNs security. We classified these approaches into different categories based on the key mechanism. We then talked about the necessity of and techniques for establishment of different types of keys in HWSNs. After that, we surveyed the authentication mechanism and self-healing mechanism. In the next chapter, we propose a problem definition and present a solution overview.

3

Problem Definition and Solution Overview

3.1 Introduction

As a result of significant advances in pervasive computing and wireless communication technology, WSNs have gained wide application. However, some unique features of sensor networks make them more vulnerable to security attacks than their wired counterparts. Security countermeasures should be taken to resist corresponding attacks. Key management works as the cornerstone of other security mechanisms as almost all of security mechanisms rely on, or are related to, encryption. Chapter 2 surveyed the literature and identified a series of weaknesses in current approaches aiming to address key management for sensor network security. We have found that, despite significant contributions have been made over the decades, very few practical approaches, especially in terms of key management solutions for HWSNs, have been proposed in the literature. In addition, as discussed in Chapter 2, some important auxiliary properties of key management in WSNs, such as authentication and self-healing, have not been addressed sufficiently.

To address the shortcomings discerned in the literature, in this chapter we will outline the problems that we intend to address in this thesis. Then the research methodology and solutions are proposed. In Section 3.2, we outline a set of definitions that will be used throughout the thesis. In Section 3.3, we define the main problems to be addressed in this thesis. In Section 3.4, we break down the main problems into research issues in order to better propose a solution. In Section 3.5, we introduce the research method that we will adopt in this thesis. In Section 3.6, we discuss the solutions to each of the research issues identified in Section 3.4. Finally, in Section 3.7, we conclude the chapter.

3.2 Key Concepts and Preliminaries

In this section, we explain the key concepts and preliminaries which we will be used in this chapter to formulate our problem, and subsequently throughout the thesis for proposing the solution to the defined problem.

3.2.1 Key Predistribution

Key Predistribution is a key distribution method. In large-scale WSNs whose physical topology is unknown prior to deployment, some keys or key materials would be installed in sensor nodes. Key predistribution alleviates the communication cost between group members and also provides secure connectivity between nodes.

3.2.2 Symmetric Key Cryptography

Symmetric Key Cryptography [107] concerns symmetric algorithms, also called secret key algorithms or single key algorithms, and are algorithms where the encryption key can be calculated from the decryption key and vice versa. In most symmetric algorithms, the encryption key and the decryption key are identical. Symmetric algorithms require that the sender and the receiver agree on a key before they can communicate securely. The security of a symmetric algorithm rests in the key. Symmetric key algorithms can be divided into stream ciphers and block ciphers. Some examples of popular and well-respected symmetric algorithms include AES (Rijndael), DES, 3DES, IDEA, RC4 and etc.

Encryption and decryption with a symmetric algorithm can be denoted as follows:

$$E_K(M) = C; \qquad D_K(C) = M.$$

3.2.3 Public Key Cryptography

Public Key Cryptography [107] concerns public key algorithms where the key used for encryption is different from the key used for decryption. Furthermore, the decryption key cannot be calculated from the encryption key. The encryption key is public and correspondingly is called the public key. Anyone can use the encryption key to encrypt a message, while only the person with the corresponding decryption key can decrypt the message. The decryption key is called a private key. RSA and ECC are two very popular public key algorithms.

Encryption and decryption with a public key algorithm is denoted as the same as the symmetric key algorithm even though the public key and the private key are different.

3.2.4 Hierarchical Wireless Sensor Network (HWSN)

Unlike flat WSNs where sensors act as routers and transfer data via multi-hop routing, in an *HWSN*, some more powerful fixed or mobile nodes are used to collect and transfer sensing data. These more powerful nodes also work as a cluster head by performing a management function. Commonly used HWSNs include three levels: the base station at the root level, cluster heads at the second level, and normal sensor nodes at the third level.

3.2.5 Mutual Authentication

Mutual authentication, also called two-way authentication, is a process in which both entities in a communication link authenticate each other. In general network environments, especially E-commence transactions, the client authenticates the server and vice versa. In sensor network environments, mutual authentication not only refers to authentication between the normal nodes and the base station, it can also refer to two counterparts that are assured of each other's identity.

3.2.6 Implicit Authentication

Implicit authentication is not performed as an independent process. Instead, it is the byproduct of other processes, such as key establishment. This authentication paradigm in wireless sensor networks can reduce operating complexity and minimizes power consumption.

3.2.7 Self-healing Key Distribution

Self-healing key distribution can be thought of as a branch of key distribution. The objective of self-healing key distribution is to enable group users to recover session keys by themselves, without requesting additional transmissions from the group manager, even if they miss some broadcast messages. The pioneering work on self-healing key distribution was proposed by Staddon *et al.* in [80]

3.2.8 Mutual-healing Key Distribution

Mutual-healing Key Distribution can be seen as complementary to the self-healing key distribution mechanism. In self-healing key distribution schemes, if a node has missed more than a fixed number of broadcast messages or the last broadcast message, it can get assistance from its neighboring nodes. The neighboring nodes in the same session group cooperate with each other, forwarding broadcast messages which the neighboring nodes missed. In this way, the nodes can receive the missed broadcast messages in a timely and efficient manner. Thus, the robustness of self-healing key distribution schemes is strengthened. The idea of mutual-healing was proposed by Bohio *et al.* in [97].

3.2.9 Session

In order to make key management convenient, the lifetime of a network can be divided into many time slots. Each time slot is called a *session*.

3.2.10 Access Structure

The terminology *Access Structure* is originally used in secret sharing. A secret is shared between users in $U = \{1, \ldots, n\}$, only qualified subsets of U can reconstruct the secret from their shares. The family of qualified subsets is called *access structure*, denoted by Γ. The subset $\Gamma \subseteq 2^U \setminus \phi$ must be monotone increasing, that is, $A_1 \in \Gamma$ and $A_1 \subseteq A_2 \subset U$ embodies $A_2 \subset \Gamma$. The family of authorized subsets Γ is the closure of minimal authorized subsets Γ_0 called the basis of the structure. The family of non-authorized subsets $\bar{\Gamma} = 2^U \setminus \Gamma$ is monotone decreasing. That is, if $A_1 \in \bar{\Gamma}$ and $A_2 \subseteq A_1$ imply $A_2 \in \bar{\Gamma}$. The family of non-authorized subsets $\bar{\Gamma}$ is determined by the set of maximal non-authorized subsets $\bar{\Gamma}_0$.

3.2.11 Shamir's Secret Sharing

Shamir's Secret Sharing [108] is based on a polynomial interpolation technique. It allows a dealer D to distribute a secret s to n players $P_1, \ldots, P_n$, such that at least $k \leq n$ players can reconstruct the secret, while any fewer than k players cannot obtain any information about the secret.

A (k, n) secret sharing protocol is as follows:

1. Share computation algorithm:
   - Dealer D creates a $k-1$ degree random polynomial $f(x) = a_0 + a_x x + a_2 x^2 + \ldots + a_{k-1} x^{k-1}$ which satisfies $a_0 = s$. Without loss of generality, we suppose $f(x)$ is constructed over a finite field.
   - Dealer D randomly chooses n distinct point $x_j \neq 0 (j = 1, \ldots, n)$, and secretly distributes each share $(x_j, f(x_j))$ to each player P_j
2. Share reconstruction algorithm: Given k distinct pairs of $(x_{t_i}, f(x_{t_i}))(1 \leq i \leq k)$, there is unique $k-1$ degree polynomial $f(x)$, passing through all the points. The polynomial can be reconstructed with Lagrange interpolation.

$$f(x) = \sum_{i=0}^{k-1} f(x_{t_i}) * L_i(x) \tag{3.1}$$

where $L_i(x)$ is the Lagrange polynomial $L_i(x) = \prod_{1\leq j\leq k, i\neq j} \frac{x-x_j}{x_i-x_j}$.

3.2.12 Vector Space Secret Sharing

Vector Space Secret Sharing was introduced by Brickell [109]. Suppose D wants to share a secret with the members of set U. It picks up a function

$$\psi : U \cup \{D\} \rightarrow GF(q)^l \tag{3.2}$$

where q is a prime power and $l \geq 2$ is an integer. This function satisfies the property: $A \in \Gamma$ if and only if the vector $\psi(D)$ can be expressed as a linear combination of the vectors in the set $\psi(A) = \{\psi(i)|i \in A\}$. An access structure Γ is said to be a vector space access structure if it can be defined in the above way.

A vector space secret sharing scheme for Γ with set of secrets $GF(q)$ is constructed as follows: (Please refer to [109] for a proof).

1. *Share Distribution.* To distribute a secret value $k \in GF(q)$, D takes at random an element $v \in GF(q)^l$, such that $k = v \cdot \psi(D)$. For $1 \leq i \leq n$, D sent the share $s_i = v \cdot \psi(i)$ to $i \in U$ over secure channel. Here the operation "$\cdot$" is the inner product modulo q.

2. *Key Recovery.* Let $A \in \Gamma$ be an authorized subset; then,

$$\psi(D) = \sum_{i\in A} \lambda_i \cdot \psi(i) \tag{3.3}$$

for some $\lambda_i = GF(q)$. In order to recover the secret k, members in A compute

$$\begin{aligned}\sum_{i\in A} \lambda_i s_i &= \sum_{i\in A} \lambda_i v \cdot \psi(i) \\ &= v \cdot \sum_{i\in A} \lambda_i \psi(i) = v \cdot \psi(D) \\ &= k.\end{aligned} \tag{3.4}$$

3.2.13 Pseudo-random Number Generator (PRNG)

A *Pseudo-random Number Generator (PRNG)* takes a seed of a certain length as input and outputs a string, which is of a greater length than that of the seed.

3.2.14 Cryptographically Secure Pseudo-random Number Generator (CSPRNG)

A *CSPRNG* [110] is a PRNG whose output string cannot be computationally distinguished from a truly random distribution. CSPRNG requirements fall into two groups:

1. Every *CSPRNG* should satisfy the next-bit test. The next-bit test is as follows: Given the first k bits of a random sequence, there is no polynomial-time algorithm that can predict the $(k + 1)$-th bit with probability of success higher than 50%.

2. Every *CSPRNG* should withstand "state compromise extensions". In the event that part or all of its state has been revealed (or guessed correctly), it should be impossible to reconstruct the stream of random numbers prior to the revelation. Additionally, if there is an entropy input while running, it should be impossible to use knowledge of the input's state to predict future conditions of the *CSPRNG* state.

3.2.15 One-way Hash Function

A hash function is the foundation of a hash chain. A hash function H takes a binary string M of arbitrary length as input, and outputs a binary string of fixed length, which is called hash value h: $h = H(M)$. A *one-way hash function* H satisfies the following three properties [107]:

1. *Computable property:* Given an input M, it is easy to compute h such that $h = H(M)$;

2. *One-way property:* Given a hash value h, it is computationally infeasible to find a second input M such that $H(M) = h$;

3. *Collision-free property:* Given a hash value h, it is computationally infeasible to find a second input M' such that $H(M') = h$, where $M' \neq M$.

3.2.16 One-way Hash Chain

The forward hash chain of length m can be derived based on a hash function as follows:

1. generate a random key seed K_0^F for forward hash chain;
2. iteratively apply the hash function H on the seed to produce forward hash key chain of length m, the forward hash chain is generated as:

$$\{H(K_0^F), \ldots, H^i(K_0^F), \ldots, H^m(K_0^F)\} \tag{3.5}$$

The backward hash chain can be derived based on a hash function as follows:

1. generate a random key seed K_0^B for backward hash chain;
2. iteratively apply the hash function H to the seed to produce a backward hash chain of length m. The backward hash chain is generated as:

$$\{H(K_0^B), \ldots, H^i(K_0^B), \ldots, H^m(K_0^B)\} \tag{3.6}$$

3.2.17 Bilinear Pairings

Let G_1 and G_2 be two cyclic groups of order q for a large prime q. G_1 is a cyclic additive group and G_2 is a cyclic multiplicative group. We assume that the discrete logarithm problems in both G_1 and G_2 are difficult. Let $e : G_1 \times G_1 \rightarrow G_2$ be a pairing which satisfies the following conditions:

- *Bilinearity*: $e(aP, bQ) = e(P, Q)^{ab}$, for $\forall\ P, Q \in G_1$ and $\forall\ a, b \in \mathbb{Z}_q^*$;
- *Non-degeneracy*: there exists $P \in G_1$ and $Q \in G_1$, such that $e(P, Q) \neq 1$; That is, for any point $P, Q \in G_1$, $e(P, Q) = 1$ iff $P = O$.
- *Computability*: there exists an efficient algorithm to compute $e(P, Q)$ for any $P, Q \in G_1$.

3.2.18 Bilinear Diffie-Hellman (BDH) Assumption

BDH Parameter Generator: A BDH parameter generator $\mathcal{IG}$ is a probabilistic algorithm that takes a security parameter $0 < k \in \mathbb{Z}$, runs in polynomial time, and outputs the description of two groups G_1 and G_2 of the same order q and the description of an admissible bilinear map $e : G_1 \times G_1 \rightarrow G_2$.

BDH Problem: Given $\langle P, aP, bP, cP \rangle$ for some $a, b, c \in \mathbb{Z}_q^*$, computes $e(P, P)^{abc} \in G_2$.

BDH Assumption: There is no polynomial time algorithm to solve the BDH problem.

3.2.19 Discrete Logarithm Problem (DLP)

Given two group elements P and Q, to find an integer $n \in \mathbb{Z}_q^*$, such that $Q = nP$ when such an integer exists.

3.2.20 ID-based PKI

ID-based PKI involves a trusted KGC and nodes. Nodes' private keys are calculated by KGC and send to the node via a secure channel. The basic operations consist of *Setup* and *Private Key Extraction*. When we use bilinear pairings to construct ID-based private/public keys, the operations can be implemented as follows: KGC runs BDH parameter generator to generate two groups G_1, G_2 and a bilinear pairing $e : G_1 \times G_1 \rightarrow G_2$. It chooses an arbitrary generator $P \in G_1$ and defines two cryptographic hash functions: $H_1 : \{0,1\}^* \rightarrow G_1$, $H_2 : G_2 \rightarrow \{0,1\}^*$.

- *Setup:* KGC chooses a random number $s \in \mathbb{Z}_q^*$ and set $P_{pub} = sP$. Then KGC publishes system parameters $params = \{G_1, G_2, q, P, P_{pub}, H_1, H_2\}$, and keeps s as a master-key, which is known only to himself.

- *Private Key Extraction:* A node submits its identity to KGC. KGC computes the node's public key $Q_{ID} = H_1(ID)$ and private key $S_{ID} = sQ_{ID}$, then privately returns $S_{ID} = sQ_{ID}$ to the node.

3.3 Problem Definition

As discussed in the previous chapter, significant advancement, documented in the literature, has been made in the area of key management for WSNs security. But most of existing schemes have drawbacks and there are some unsolved problems which really merit in-depth research. In this section, we give four fields of problem definition related to sensor network security, namely:

- Problems with key management schemes for WSNs
- Problems with establishment of different types of keys
- Problems with authentication mechanism
- Problems with self-healing mechanism

For each of these problems, the discussion is carried out from two perspectives: existing solutions, and the technical problems inherent in these solutions. These technical problems determine the research issues and are the key requirements of any new solution development.

3.3.1 Problems with Key Management Schemes for WSNs

Existing solutions of Symmetric Key Cryptography based key management

The key predistribution mechanism has been considered as the most suitable symmetric key management mechanism for wireless sensor networks. A standard key predistribution scheme has four stages as shown in Figure 3.1. In the key pool generation stage, a large pool of keys and key identities is generated. In the key predistribution stage, each sensor is preloaded a small number of keys which are selected from the key pool. In the shared key discovery stage, two neighbor nodes compare the list of identities of keys that they have. If two sensors have one or some keys in common, they can setup a secure link directly. Otherwise, the path key establishment stage is triggered to set up a link with the help of intermediate nodes.

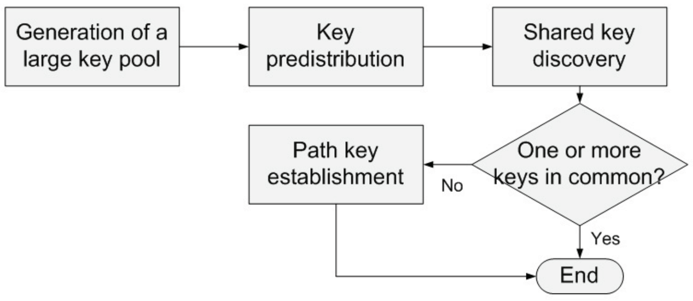


Figure 3.1: The general processes of a standard key predistribution paradigm

Technical problems of Symmetric Key Cryptography based key management

Almost all the Symmetric Key Cryptography based key management schemes for WSNs follow a key predistribution paradigm, or a variant. In the previous chapter, we have examined many different mathematical models that can be utilized to realize key predistribution in previous chapter. However, approaches of each model have one or several of the following drawbacks:

- All the intermediate nodes have knowledge of the established path key. In the path key establishment stage, any pair of nodes within wireless communication range that do not share a key can be assigned with a path key. The assigned path key is transferred by two or more links with multi-fold encryption. This method not only introduces heavy communication and computation overhead, but also brings vulnerability. The compromise of any intermediate node can lead to the disclosure of the path key.

- The shared key does not exclusively belong to two end nodes. Two or more different pairs of nodes may use the same shared key. The compromised link between one pair of nodes may introduce a security threat to links between other nodes.

- Many schemes endow nodes with the ability to establish pairwise keys with all nodes in the network. This ability requires a significant amount of overhead and places a heavy burden on nodes. We argue that it is impractical and unnecessary to establish links with all the nodes in the network. The main problem is to make sure that each node has the highest key share probability with nodes within its communication range rather than throughout the network.

- One of the main drawbacks of key predistribution schemes is weak resilience. A large number of keys, even the whole key pool, could be disclosed by intentionally compromising a few number of nodes.
- There are very few schemes that discuss key predistribution for HWSNs. We have discussed the advantages of HWSNs over DWSNs. It calls for the development of key predistribution schemes for HWSNs in order to keep up with the proliferation of the latter.

To sum up, current Symmetric Key Cryptography based key management schemes have the above technical issues and no solution to date has addressed all of the drawbacks. This situation motives us to carry out research into the development of new and practical key predistribution schemes for HWSNs.

Existing solutions and technical problems of PKC-based key management

Even though much of the literature has demonstrated the feasibility of Public Key Cryptography in WSNs, there have been few PKC-based key management schemes in this field. As far as we know, the first PKC-based key management scheme SACK-P [68] is directly transformed from the Symmetric Key Cryptography based key management scheme SACK. Several existing PKC-based key management schemes rely too much on some strict assumptions or incur heavy overhead.

Existing solutions and technical problems of hybrid key management

To the best of our knowledge, except for the SACK-H scheme [68] which is also directly transformed from the symmetric key based key management scheme SACK, there is no practical hybrid key management scheme for HWSNs. We considered the advantages of hybrid key management in hierarchical wireless sensor networks in the previous chapter. These motivate us to design a practical hybrid key management scheme for HWSNs.

3.3.2 Problems with the Establishment of Different Types of Keys

Existing solutions

As discussed in Chapter 2, LEAP [29] supports the establishment of individual keys, pairwise keys, cluster keys, and a global key. Individual key which is a unique key shared between the base station and each sensor node [29]. The key is pre-calculated and preloaded into each node's memory before being deployed. A pairwise key is shared between each pair of neighboring nodes. Each node U is preloaded with a key K_i. U drives the master key K_u using it. Correspondingly, a neighboring node V can derive its master key K_v too. The node U computes the pairwise key $K_{uv} = f_{K_v}(u)$. Node V can also compute K_{uv} in the same way. Cluster keys are built upon the pairwise keys. A node U encrypts the selected cluster key K_u^c with the pairwise keys of its neighboring nodes and sends the encrypted key to each neighboring node. The global key originates from the base station and recursively transits to all the nodes in a width priority manner. The time-based key management scheme [1] uses different initial keys for pairwise key establishment at different time slots. The establishment of other types of keys is the same as that in [29].

Technical problems

We cannot deny that LEAP presents an efficient way of establishing different types of keys. However, this scheme cannot be used in WSNs which have strict security requirements. This is because current schemes LEAP [29] and the time-based scheme [1] have the following technical issues:

- LEAP suffers from sinkhole attack. In a sinkhole attack, a compromised node attracts packets by advertising information such as high battery power, etc., then later drops all the packets [41].
- The initial key is reused again for node addition in LEAP. The initial key which is removed after the initialization phase is reused. However the new node could be captured before erasing the initial key. Various attacks can be launched based on the disclosed initial key.
- In scheme [1], the key connectivity is reduced even though the threat caused by the disclosure of the initial key is eliminated. The key connectivity is af-

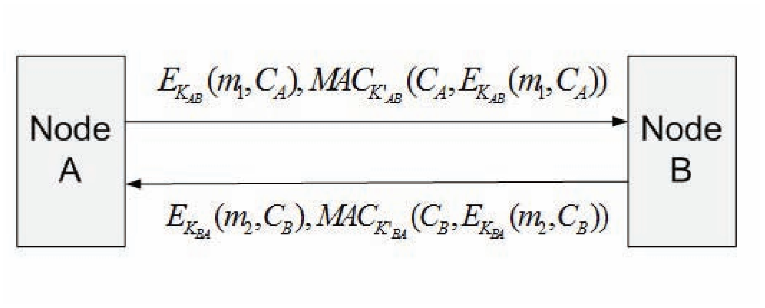

Node A
Node B
$E_{K_{AB}}(m_1,C_A), MAC_{K'_{AB}}(C_A, E_{K_{AB}}(m_1,C_A))$
$E_{K_{BA}}(m_2,C_B), MAC_{K'_{BA}}(C_B, E_{K_{BA}}(m_2,C_B))$

Figure 3.2: Plain SNEP protocol which offers weak freshness

fected by the number of preloaded master keys m and the order of the current time slot. A higher key connectivity can be achieved at the cost of heavy burden on storage.

- In scheme [1], the established pairwise key does not exclusively belong to the two end nodes. As we discussed in Chapter 2, nodes in one time slot may have knowledge of the pairwise keys established by nodes in other time slots. This drawback makes the scheme vulnerable to attacks against confidentiality and authentication.
- In scheme [1], some of preloaded master keys are useless. m randomly-chosen master keys are preloaded to the memory of each node before it is deployed to the network without consideration of lifetime of the node. Some preloaded master keys are unused because a node's lifetime is limited.

3.3.3 Problems with Authentication Mechanisms in Key Management

Existing solutions

Authentication is an important tool which is used to verify the transferred messages and communication counterparts. We have examined current research on authentication in wireless sensor networks: unicast authentication, broadcast authentication, filtering false data, PKC-based authentication, and lightweight authentication.

SNEP requires any two nodes keep a shared counter to keep data freshness. As

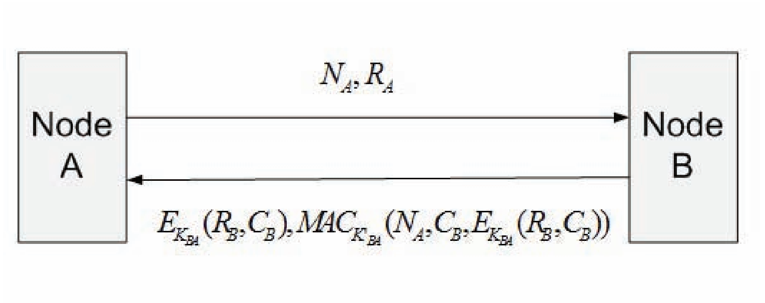

N_A, R_A
Node A
Node B
$E_{K_{BA}}(R_B, C_B), MAC_{K'_{BA}}(N_A, C_B, E_{K_{BA}}(R_B, C_B))$

Figure 3.3: Strong SNEP protocol which offers strong freshness

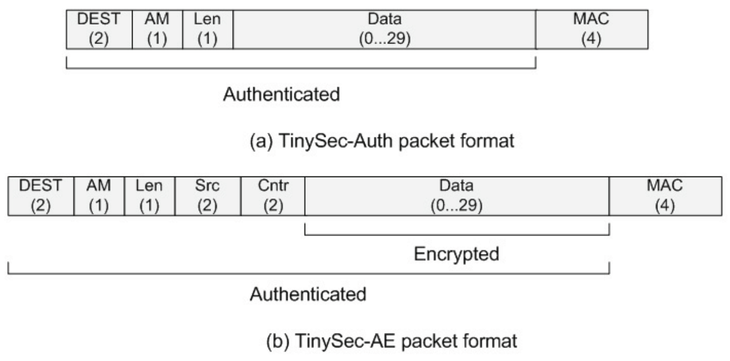


Figure 3.4: TinySec packet formats in TinySec-Auth mode and TinySec-AE mode

shown in Figure 3.2 and Figure 3.3, plain SNEP offers weak freshness and strong SNEP offers strong freshness. In Figure 3.2, K_{AB} and K_{BA} are encryption keys and for directional communication between node A and node B, respectively; K'_{AB} and K'_{BA} are authentication keys and for directional communication between node A and node B, respectively; C_A and C_B are counters maintained by node A and node B, respectively; M_1 is the message transferred from node A to node B and M_2 is the message transferred from node B to node A. In Figure 3.3, node A generates a nonce N_A and sends it along with a request message R_A to node B. Then B returns the nonce with the encrypted response message R_B, along with the MAC of encrypted message R_B, C_B, and N_A. If the MAC verifies correctly, node A knows that node B generated the response after it sent the request [111].

TinySec provides two different packet formats: TinySec-Auth for authentication only and TinySec-AE for authentication and encryption. The two packet formats are shown in Figure 3.4. In TinySec-Auth mode, the un-encrypted packet is

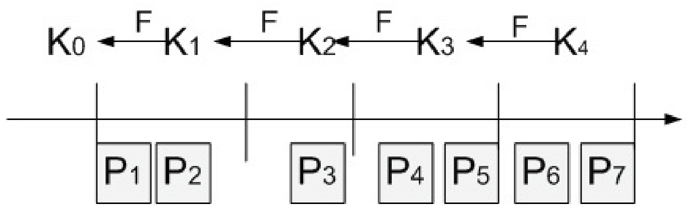


Figure 3.5: μTESLA one-way key chain [2].

authenticated with a MAC. In TinySec-AE mode, packet is encrypted first and then authenticated with a MAC.

μTESLA provides authentication for broadcast with a one-way key chain of authentication keys. Figure 3.5 displays the evolution of authentication keys. K_0 is the initial key.

MAC is the basic way used of filtering false data. Both SEF [75] and hop-by-hop authentication can be seen as variants of MAC. PKC-based authentication has incomparable advantage than symmetric key based authentication. Existing PKC-based authentication for WSN [77] cannot be seen as purely PKC-based authentication. The notion of "lightweight authentication" comes from authentication for RFID. Even though it was claimed as a "lightweight" mechanism, it has high computation and communication overhead.

Technical problems

- The feasibility of SNEP has not been fully specified and implemented. TinySec does not provide replay protection for the message, which is an important security requirement. It does not renew the key over time. TinySec requires extra energy, bandwidth, and incurs latency overhead with increased packet length [111].
- All μTESLA-based broadcast authentication schemes [2, 28, 29, 30] require loosely time synchronization. A delayed packet authentication mechanism may suffer from severe energy-depletion DoS attacks.

- It is not known how to keep the trade-off between the shared secrets between nodes and the probability of SEF [75]. The three hop-by-hop authentication schemes in [76] can only be applied to data collection rather than hop-by-hop data aggregation applications.
- To date, no feasible lightweight authentication and PKC-based authentication schemes have been proposed for WSNs.
- No unified standard can be used to evaluate whether a scheme is lightweight or not.

3.3.4 Problems with Self-healing Key Distribution Mechanisms

Existing solutions

The classification and the basic procedures of self-healing key distribution schemes have been discussed in Chapter 2. As shown in Figure 2.10, general self-healing key distribution schemes, either unconditionally secure or computationally secure, include five basic procedures: *Setup*, *Broadcast*, *Key Recovery*, *Adding/Revoking Users*, and *Self-healing*. Many different cryptographic primitives have been used to realize self-healing key distribution. Each category has different features and different limitations.

Technical problems

- For polynomial secret sharing based self-healing key distribution schemes, heavy computation overhead is introduced in the procedure of *Key Recovery*. Shamir's secret sharing inherent property determines that the maximum number of revoked users is constrained to the degree of the polynomial.
- Sponsorization can be seen as complementary to vector space secret sharing based self-healing key distribution. That is, a coalition of users sponsor a user outside the group for one session. Sáez discussed the technique details in [91]. However, a secure unicast channel between each sponsor and the sponsored user is required in order to fulfill the sponsorization. It is too rigorous to be feasible for WSNs. In addition, the operations introduce too much communication and computation overheads.

- Hash chain based self-healing key distribution schemes are more efficient than their counterparts based on other cryptographic primitives. However, these schemes are vulnerable to collusion attack. The collusion of the revoked nodes whose life cycles do not expire and the newly joined nodes can recover the session keys to which they are not authorized.

- For the bilinear pairing based self-healing key distribution schemes, the communication overhead increases with the number of authorized users in the communication group.

- Bihio *et al.* [97] proposed the general idea of mutual-healing without exploring the technical details. Tian *et al.* [98] made the security requirements more clear and proposed a rough idea for realization of mutual-healing. However, no formalized model definition has been proposed.

Hence, based on the above discussion, we can broadly define the problem that we intend to address in this thesis as:

To develop a key management framework for wireless sensor networks by which different types of keys can be established, among which network-wide key can be distributed in self-healing and mutual-healing manner; explicit or implicit authentication can be guaranteed according to the security requirements of expected applications.

3.4 Research Issues

In the previous section, we discussed four areas of problem definitions, including current available solutions, and existing technical problems arising from each category of solutions. These technical problems help in identifying research issues. In this section, we now explain in detail each research issue which needs to be addressed in order to provide the solution to the broadly defined problem.

3.4.1 Research Issues 1: Developing a hash chain based key predistribution algorithm which improves key sharing probability with reduced storage overhead

In Chapter 2, we reviewed existing key predistribution schemes which are regarded as the most suitable Symmetric Key Cryptography based key management methods for WSNs. In the third section of Chapter 3, we identified the main technical problems with existing key predistribution schemes. In summary, existing key predistribution schemes have one or several of the following weaknesses:

- All the intermediate nodes have knowledge of the established path key.
- The shared key does not exclusively belong to two end nodes.
- Many schemes endow nodes with the capability of establishing pairwise keys with all nodes in the network.
- Another drawback of key predistribution schemes is weak resilience.
- There are very few schemes that tackle key predistribution for HWSNs.

HWSNs have better scalability than do flat WSNs and the size of networks may scale to thousands of nodes. If we directly apply key predistribution schemes to HWSNs, each node should be preloaded with a great number of keys in order to reach expected key connectivity probability. In a large scale hierarchical sensor network, nodes are divided into different clusters. Node addition and revocation are performed by the cluster head. Also, each node communicates only with its cluster head and nodes in the same cluster head. Each cluster head can communicate with peer cluster heads and the base station. If we take advantage of hierarchical structure of networks with hierarchical key predistribution, some beneficial properties will be achieved. In HWSNs, each cluster works as a self-managed system. Hierarchical key predistribution allocates a small portion of the key pool to each cluster, so the storage overhead at each sensor nodes would be greatly reduced while the key connectivity probability within each cluster can be as high as 100%. The process of path key establishment can be removed. Another advantage of such key predistribution is that, if a node is compromised, only very few preloaded keys will be disclosed. In this way, better resilience is achieved.

The storage overhead can be further reduced and key connectivity can be further increased if we substitute a randomly generated key pool with keyed hash chains. Just as a coin has two sides, the one-way property of a hash chain has both an advantage and a disadvantage. Once an adversary obtains the key seed and hash function of a key chain, all the keys on the key chain can be derived. In order to resist node capture attack, it is prohibited to store the key seed at sensor nodes. Hence, we have to find an appropriate alternative predistribution method to eliminate the disadvantages of the one-way property.

In section 3.6, we will make the above considerations come true. We present a solution overview of the methodology of a hash chain based key predistribution scheme for hierarchical wireless sensor networks. The scheme improves key connectivity probability and can resist node capture attack. The concrete scheme will be presented in Chapter 5.

3.4.2 Research Issues 2: Developing an algorithm supporting the establishment of different types of keys whose security does not rely on the security of the initial key

LEAP [29] supports the establishment of multi-level keys. It has been served as the benchmark for localized encryption and authentication in sensor network since its emergence. However, LEAP [29] has some weakness. In Chapter 2, we reviewed LEAP [29]. In section 3.3, we identified its main technical problems.

- LEAP suffers from sinkhole attack.
- The initial key is reused again for node addition in LEAP.

Jang *et al.* proposed a time-based key management scheme which split the lifetime of a sensor network into multiple time slots and each time slot has its initial key. This scheme minimizes the effect of disclosure of the initial key. However, this scheme still has several technical problems as summarized in section 3.3:

- In [1], the key connectivity is reduced even though the threat caused by the disclosure of the initial key is eliminated.
- In [1], the established pairwise key does not exclusively belong to the two end nodes.
- In [1], some of preloaded master keys are useless.

As is well known, normal sensor nodes are powered by a non-chargeable battery so their lifetime is limited. Generally speaking, the lifetime of sensor nodes are far less than that of WSNs. New sensor nodes should be replenished over time to keep network connectivity. The main reason for the low key connectivity probability of Jang [1] is that the designer did not take the life-time of sensor nodes into consideration. Suppose a sensor node which is deployed at time slot j can survive at most t time slots, so the preloaded master keys for time slots after the $(j + t)$-th time slot are unused. Therefore, the preloaded master keys should match the lifetime of sensor nodes. That is, in order to make full use of a node's memory and increase key connectivity, only those master keys corresponding to the node's lifetime can be preloaded to the node's memory.

Resilience is an important evaluation factor of key management schemes. In order to achieve the resilience property, the established pairwise key should exclusively belong to the two end nodes. Once a node is compromised, the adversary can only get the pairwise key of the compromised node and its neighboring nodes, while the communication between the uncompromised nodes remain secure. If we can find a different way to derive master keys from the corresponding initial keys and a different way to establish pairwise keys, a higher key connectivity probability between nodes belonging to different time slots, and better resilience, can be achieved.

In section 3.6 we will solve the technical problems of [1]. We present a solution overview of the enhanced key management scheme whose security does not hold on that of the initial key and can resist sinkhole attack. The concrete scheme will be presented in Chapter 4.

3.4.3 Research Issues 3: Developing a fully anti-collusive hash chain based self-healing key distribution algorithm

Hash chain based self-healing key distribution schemes are much more efficient than those self-healing key distribution schemes based on other cryptographic primitives. However, the introduction of a hash chain improves efficiency but at the cost of possible collusion attack. As reviewed in the purely hash chain based self-healing key distribution scheme [95], even though forward and backward secrecy can be guaranteed, the collusion between the newly joined users and the revoked users will recover all the session keys to which they are not entitled. Tian *et al.* presented a scheme to resist collusion attack in [93]. Each user U_i is assigned a pre-arranged life cycle (s_i, t_i). Random numbers corresponding to its life cycle will be preloaded to the node as part of personal keys. However, we identified that [93] and two subsequent schemes [6, 7] could resist collusion only between the users whose life cycles have finished and the newly joined users. They could not resist collusion attacks of the newly joined users and revoked users whose life cycles have not yet expired.

In order to resist collusion attack, our first aim is to find the breaking point of the collusion attack. In Tian *et al.*'s scheme [93], the import of random numbers effectively against collusion attack between the users whose life cycles have finished, and the newly joined users. However, those users who are revoked before their life cycles finish, still have the random numbers. If they collude with the newly joined nodes, they can derive the several keys to which they are not authorized. In order to maintain security, a part of the key material should be related to the user's status. Once a user is revoked before its life cycle expires, at least part of the key material or broadcast message should be changed so that the revoked user cannot continue to access the subsequent session keys. At the same time, the operation of authorized users should continue. So the research issue for this technical problem is to find a suitable way to change key material or broadcast messages.

In section 3.6 we will solve the collusion problems of hash chain based self-healing key distribution schemes. We present a solution overview of the proposed self-healing key distribution scheme that is robust again collusion attack. The concrete scheme will be presented in Chapter 6.

3.4.4 Research Issues 4: Developing a secure HBT-based self-healing key distribution algorithm with implicit authentication

A Hash Binary Tree (HBT) is established by iteratively applying two different hash functions to the same seed to generate the left node or right node of any intermediate nodes, respectively. Jiang *et al.* proposed a novel HBT-based self-healing key distribution scheme in [112]. Like all the self-healing key distribution schemes, the life-time of a network is expressed by the number of sessions. The number of sessions is the same as the number of leaf nodes of the hash binary tree. Each leaf node in the hash binary tree is mapped to a forward key. This HBT-based technique is efficient because the generation of a hash binary tree involves only hash operation. This technique also helps to reduce storage overhead. However, HBT can be considered as a variant of a hash chain and this HBT-based scheme inherits all the vulnerabilities of a hash chain based self-healing schemes. Moreover, the mask method is based on Shamir's secret sharing; hence, the number of revoked users cannot exceed the threshold.

We can present several possible ways to improve the HBT-based self-healing key distribution scheme.

- If we can replace the threshold structure with a more flexible access structure, a more flexible self-healing key distribution scheme will be possible;
- If we can introduce random numbers as in [93], then the scheme can not only maintain forward and backward secrecy but resist collusion attack;
- Broadcasts over wireless channels are vulnerable to various attacks, such as replay attack and tamper attack. Authentication is an effective method to repel such attacks. However, explicit authentication has the further requirement of shared keys. If we can find an implicit authentication method which makes uses of a hash chain, the requirement of shared keys would be removed and computation cost of authentication and verification would be minimized.

In section 3.6 we consider how to add the collusion resistance property and implicit authentication to the HBT-based self-healing key distribution scheme. We present a solution overview of the proposed scheme. The concrete scheme will be presented in Chapter 7.

3.4.5 Research Issues 5: Exploring technical details of mutual-healing key distribution algorithm

Bohio *et al.* in [97] proposed the concept of mutual-healing to complement the self-healing key distribution schemes without presenting technical detail. In order to keep communication secure, Bohio *et al.* claimed in [97] that a mutual-healing scheme should meet two conditions. Firstly, in order to avoid attacks on limited resources, effective means of authentication of the requesting node should be developed to identify misbehaving nodes. Secondly, authentication of the requested keys should be developed to make sure that the requesting nodes are authorized to access the requested session keys. As reviewed in Chapter 2, the motivation of mutual-healing is that nodes can obtain assistance from neighboring nodes under certain special conditions. If the concept can be realized, it will complete the self-healing key distribution schemes.

Tian *et al.* in [98] pointed out that authentication of requested keys is unnecessary. This is because broadcast messages are broadcasted in a masked manner within the communication group. Any entities can receive the broadcast messages. If the requesting node is authorized for the session, it would be able to recover the session key. Otherwise, even if unauthorized nodes receive the broadcast message from their neighbors, they can not recover the session key. Hence, the only research issue related to mutual-healing is how to effectively authenticate requesting nodes.

In general self-healing key distribution schemes, each node receives a personal key from the group manager. Keys are distributed and renewed via broadcast communication. No pairwise keys between sensor nodes and no pairwise keys between nodes and the group manager are established. Originally, the self-healing key distribution scheme was intended to reduce the burden of the group manager so that the interaction between nodes and the group manager is removed. Hence, the establishment of pairwise keys between nodes and the group manager will countermand the self-healing key distribution. Mutual-healing through transfer of the group manager is infeasible. If there are pairwise keys between nodes, the authentication of the requesting nodes should be much easier.

As a complement to the self-healing mechanism, mutual-healing does not frequently occur. Hence, it is not necessary to establish pairwise keys for each pair of nodes in advance. Authentication of requesting nodes is executed on demand and a pairwise key can be established prior to authentication. If each node has an ID-based public/private key pair, it can perform node-to-node authentication when it is necessary.

In section 3.6, we discuss how to entitle each node with a public/private key pair to perform node-to-node authentication. We present a solution overview of the proposed mutual-healing scheme. The concrete scheme will be presented in Chapter 8.

3.4.6 Research Issues 6: Developing a practical hybrid key management algorithm for HWSNs

We have highlighted the advantages of HWSNs in Chapter 1. However, many existing key management schemes for HWSNs with a single key mechanism overlooked the architecture and individual resources of different tiers. The practical design of key management should take the heterogeneity into consideration. Otherwise, key management scheme itself will exhaust low cost bottom nodes.

As discussed in Chapter 2, the several existing PKC-based solutions for HWSNs have different technical problems. Du *et al.*'s routing-driven scheme [69] establishes a shared key for only neighbor nodes that communicate with each other. The scheme works under the strict assumption that all nodes must be deployed following the known tree structure and each node should be aware of the tree structure of the network. The security services of the proposed scheme [70] rely on the preloaded secret point and random number. While the renewal of random numbers brings large communication overhead and computation overhead at each cluster head. The scheme [71] also works under the strong assumption that each cluster head is equipped with an intrusion detection system to detect misbehavior and each node is aware of its location.

Hybrid solutions can exploit the advantages of Symmetric/Public Key Cryptography and heterogeneity of HWSNs. To date, no practical hybrid key management solution has been proposed for HWSNs. In fact, it is not easy to realize hybrid key management in wireless sensor networks. In a hierarchical sensor network, cluster heads act as a bridge between the base station and normal sensor nodes. If we use public key cryptosystem in the upper level and symmetric key cryptosystem in the lower lever, one of the main research issues is whether it is possible to install two cryptosystems on cluster heads. Even though the answer is definite, it would be better to find some lightweight or simplified cryptosystems in order to conserve the limited resources of cluster heads.

In section 3.6, we will discuss how to install two cryptosystems into cluster heads and how to establish different types of keys to meet different types of communication. We present a solution overview of the proposed hybrid key management scheme. The concrete scheme will be presented in Chapter 9.

3.5 Research Methodology

3.5.1 The Chosen of Science and Engineering Based Research Method

In addressing the stated technical problem, this thesis focuses on the development of an efficient and secure key management scheme for WSNs. In order to propose a solution for the research issues listed in the previous section, we need to follow a systematic scientific approach to ensure that the methodology development is scientifically-based. A science and engineering based research approach is adopted in this research project. Science and engineering research leads to the development of new techniques, architecture, methodologies, devices or a set of concepts, which can be combined together to form a new theoretical framework. This research approach commonly identifies problems and proposes solutions to these problems. Particularly in the engineering field, the spirit of "making something work" is essential [113, 114]. They decompose the science and engineering based research into three main levels:

- *Conceptual level (level one)*: creating new ideas and new concepts through analysis;
- *Perceptual level (level two)*: formulating new methods and approaches through designing and building the tools or environment or system through implementation.
- *Practical level (level three)*: carrying out testing and validation through experimentation with real world examples. The process of testing and validating a working system provides unique insights into the benefits of the proposed concepts, frameworks and alternatives.

3.5.2 Research Stages

We present the research stages based on the three levels of the science and engineering research approach.

- *Literature Review.* We reviewed survey papers on WSNs security so as to obtain a thorough understanding of the characteristics and security requirements of WSNs. Secondly, we reviewed existing key management schemes for WSNs. This review allowed us to identify open problems that are related to key management in WSNs. In addition, we reviewed some other lightweight cryptographic primitives, combinatorial mathematics and graph theory, simulation technique that will aid in the design and evaluation of key management schemes.
- *Conceptual Framework for Key Management in a Wireless Sensor Network.* Figure 1.3 shows that key management is not a separate component of security architecture. Key management provides security infrastructure to other security services. At the same time, it relies on other security services. Key management and other security services together make up the security architecture for sensor networks. Therefore, a comprehensive consideration of these factors is essential when designing a key management scheme for a wireless sensor network. Here we will propose a conceptual framework for

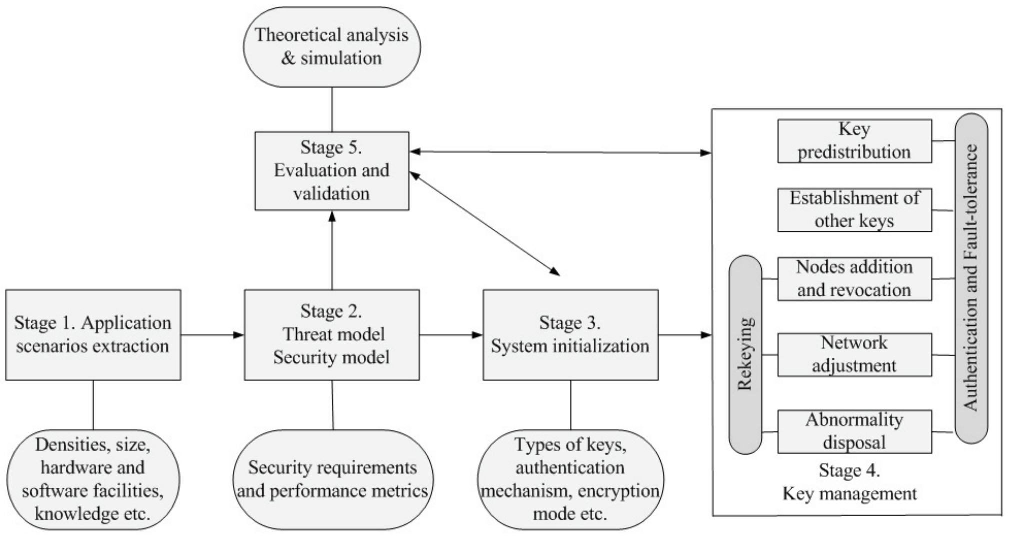

Theoretical analysis & simulation
Stage 5. Evaluation and validation
Key predistribution
Establishment of other keys
Nodes addition and revocation
Network adjustment
Abnormality disposal
Rekeying
Authentication and Fault-tolerance
Stage 4. Key management
Stage 1. Application scenarios extraction
Stage 2. Threat model Security model
Stage 3. System initialization
Densities, size, hardware and software facilities, knowledge etc.
Security requirements and performance metrics
Types of keys, authentication mechanism, encryption mode etc.

Figure 3.6: A conceptual framework of key management for wireless sensor networks security

key management for WSNs (Figure 3.6). The conceptual framework would provide a guideline to design key management schemes in the next stage.

1. We have mentioned that, WSNs are application-specific networks. Except for some common features, a sensor network for a specific application has some unique features and thus its own security requirements. Suppose a sensor network is deployed in the military surveillance environment and the other in an agricultural base. Both network's security requirements should be different based on the resources that nodes can use and the risks they face. Therefore, the first step is to fully understand the application background and extract a network model. The acquired information in this step includes the densities and the size of network, the available hardware and software resources, and some specialized knowledge that can be used in a particular real-time scenario, such as location information.
2. This step defines the security model. The security model is defined according to security requirements and will work as a performance metrics for proposed key management schemes. Here we will quantify the security with entropy. As far as we know, it is the first time that a se-

curity model has been defined with this terminology. The development of metrics, measurements, and evaluation of approaches are of great importance in order to establish a scientific methodology for the entire wireless network security research area.

3. This step initializes system parameters. The types of keys that need to be established according to different communication requirements, authentication mechanism, and encryption mode are fixed in this step. The basis includes the type of information to be encrypted (i.e. aggregated data, routing information, and key information), available memory and energy or any predefined policies or recommendations specified for the current network. A fully integrated view of the design factors is essential for the development of protocols for WSNs.
4. The outcome of this research is displayed in this step. Several hierarchical key management schemes are proposed for different applications. Information theory, combinatorial mathematics, graph theory, and various cryptographic primitives may be adopted in the development of these components but these have not been selected at this point in time. This stage corresponds to the perceptual level of the science and engineering research method.
5. Both theoretical analysis and simulation are good tools to test the designed schemes. Only by using these tools can we prove the validity of the schemes and, at the same time, discover their deficiencies.

The steps 3, 4, and 5 proceed in turn. They form a loop and perform numerous times before an efficient and secure key management scheme is obtained. This stage corresponds to the practical level of the science and engineering research method.

- *Development of Schemes and Protocols.* Based on the conceptual framework proposed in the first research stage, the main task in this stage would be to design several key management schemes for HWSNs. Each of these schemes has some specialties according to the application background. It is not our final goal and it is impossible to design a single protocol that will outperform all others for all possible models. We concentrate designing a key management scheme which matches the abstracted security model in Figure 3.6. A key management suite includes five modules: key predistribution, the establishment of other types of keys, nodes addition or revocation, network adjustment, and abnormity disposal. Rekeying, authentication, and fault-tolerance are not separated modules. Node addition or revocation, network adjustment, and abnormity disposal incur rekeying operations. Each module needs an authentication mechanism to ensure the origin of entities and the reality of data and a fault-tolerance mechanism to deal with some abnormities. The selection of a key predistribution mechanism is determined by the trade-off between security requirements and resource constraints. Both random, deterministic, and hybrid key predistribution mechanisms can be employed. Some theories, such as plane of projection, from combinatorial mathematics and graph theory can be used to improve the efficiency of key management. Lightweight cryptographic primitives, such as the ECC and Ntru cryptosystem, are ideal building blocks for public key management and lightweight authentication. Some novel techniques from information theory and coding, such as error correcting codes, can be adopted to realize the fault-tolerance capability.

- *Evaluation and Validation of Schemes and Protocols.* The evaluation here includes security evaluation and performance evaluation. Theoretical analysis is based on the effectiveness of cryptographic primitives used in the proposed schemes with the help of software Mathematica and Matlab. We will verify whether the schemes satisfy the security requirements with reasonable overheads. It would be useful to have comprehensive guidelines for evaluating a specific protocol and compare it with others. Based on the proposed security model, appropriate performance metrics would then be used to evaluate the strengths and weaknesses of each protocol. Due to the nature of WSNs, there is an inevitable coupling of the layers. Therefore, an evaluation framework would not be useful if it concentrates only on protocols of a specific layer. The framework should evaluate the goodness of the network as a whole and provide metrics to measure the effects of the design on the operation of the network (by evaluating, for example, energy efficiency, communication performance, etc). In order to quantify the effects of a compromise on the security architecture, Hwang *et al.* in [115] investigate the inherent tradeoffs involved between energy, memory, and security robustness in WSNs. A framework called the security-memory-energy (SME) is presented that is used to evalu-

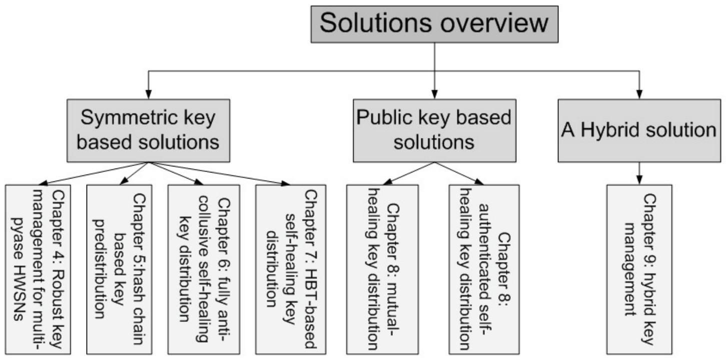

Solutions overview
Symmetric key based solutions
Public key based solutions
A Hybrid solution
Chapter 4: Robust key management for multi-pyase HWSNs
Chapter 5:hash chain based key predistribution
Chapter 6: fully anti-collusive self-healing key distribution
Chapter 7: HBT-based self-healing key distribution
Chapter 8: mutual-healing key distribution
Chapter 8: authenticated self-healing key distribution
Chapter 9: hybrid key management

Figure 3.7: Overview of the solutions proposed in this thesis

ate and quantify the multi-metric trade-offs involved in security design. The security of the lightweight key management schemes will be demonstrated by mathematical security proof.

3.6 Overview of Solutions

We have identified six research issues that are aimed at solving technical problems of existing key management schemes for HWSNs in the previous section. In this section, we propose an overview of the solution for each of the research issues. As shown in Figure 3.7, the solutions can be categorized into three categories: Symmetric Key Cryptography based solutions, PKC-based solutions, and a hybrid solution.

3.6.1 Overview of the solution for robust key management in multi-phase HWSNs

We have identified the main technical problems arising from LEAP [29] and the time-based key management scheme in section 3.3. In order to solve the identified technical problems, we should address several research issues outlined in section 3.4.

In the proposed solution, the lifetime of the network is denoted by the number of time slots. Each time slot has its initial key. The initial keys for different time slots are independent of each other. As reviewed in [1], a sensor node which is deployed in time slot j is preloaded with the initial key IK_j and m randomly chosen master keys. We have ascertained that a part of the master keys are useless due to the limited lifetime of the sensor. Here we take the nodes' lifetime into consideration. The lifetime time of sensor nodes is measured by the number of time slots. Suppose a node whose lifetime is G_w time slots joins the network at time slot j, it is preloaded with the initial key for time slot j and $(G_w - 1)$ masked initial keys. In this way, the initial key is used to establish pairwise keys with nodes deployed in the same time slot and the masked initial keys is used to establish pairwise keys with node deployed in different time slots.

The mask mechanism plays a vital role. On the one hand, it has to make sure that attackers cannot derive the actual initial keys from the masked initial keys. On the other hand, it has to ensure that nodes deployed at different time slots can establish pairwise keys. In the proposed solution, for a node deployed at time slot j, it will be preloaded with $(G_w - 1)$ masked initial keys for the subsequent generations $j + 1 \leq l < j + G_w - 1$. The masked initial keys are given as $\{K_{j,l} | K_{j,l} = H(IK_l \parallel j)\}$ where IK_l is the initial key for the l-th generation. With the masked initial keys, nodes can establish pairwise keys with nodes deployed at different time slots. The established pairwise keys belong exclusively to the two nodes. Hence, the proposed solution has a resilience property.

The current ZigBee standard does not provide forward secrecy. It does not specify how to deal with pairwise keys when nodes leave the network. The proposed solution can be used to fix these problems. In addition, we also provide authentication for node admission control. The specific solution is presented in Chapter 4.

3.6.2 Overview of the solution for hash chain based key predistribution

In this subsection, we present an overview of the solution for hash chain based key predistribution which should satisfy at least the following properties:

- Each intermediate node cannot access the path key passed-by it.

- The shared key should exclusively belong to two end nodes.
- The nodes only establish pairwise keys with nodes in its communication range or the same cluster.
- The proposed scheme should resist node capture attack.
- The proposed scheme should support the establishment of different types of keys.

As is usual with key predistribution schemes, the proposed scheme includes three stages: key pool generation, key ring assignment and common key discovery. While in the proposed scheme, the key pool generation is much different from that of the random key predistribution scheme. It is unnecessary to generate a great number of random keys. Instead, a small key pool with a very limited number of generation keys and their IDs as well as their commitments is generated. Each key chain is generated by a hash function with input of a seed and a specific generation key. Each sensor is preloaded with IDs and commitments of a small number of generation keys without actual generation keys. Each cluster head is preloaded with a large number actual generation keys, their IDs and commitments. Then both cluster heads and sensor nodes are uniformly and randomly deployed in the network.

We adopt a different clustering method in the proposed scheme to improve key connectivity probability. Each node establishes pairwise keys only with nodes within the same cluster. If any two nodes do not share any key chain, the cluster head will generate and unicast pairwise keys for them. We do not consider path key establishment as it is insecure and complex. We also talked about the establishment of other types of keys for the networks with preloaded master key or generation keys.

The proposed scheme reduces storage overhead greatly. Each node stores only the IDs and commitments of the corresponding generation keys. With the new clustering mechanism, each sensor node chooses the powerful nodes with which it has more generation keys in common as its cluster head. In this way, a higher key connectivity probability is achieved. The proposed scheme subtly overcomes the weakness of the one-way property of hash chain. The scheme can resist node compromise or node capture attack as sensor nodes do not store actual generation keys in their memory. In Chapter 5, we will introduce the proposed scheme in detail.

3.6.3 Overview of the solution for hash chain based self-healing key distribution

We have analyzed that Tian *et al.*'s scheme and its two improvements [6, 7] can resist collusion only between the users whose life cycles have finished and newly joined users. However, they are vulnerable to collusion attack of newly joined users and revoked users whose life cycles have not yet expired. Du *et al.* [116] proposed a self-healing method which aims to totally anti-collude in hash-based self-healing key distribution schemes. Before presentation of our solution, we present two attacks against Du *et al.*'s scheme [116]. Then we closely examine the breach of the collusion attack in self-healing key distribution schemes and propose two possible solutions. After ruling one of the solutions out, we prove that the only feasible solution is masking new random numbers in the corresponding broadcast messages.

We redefine the security model which provides more strict security requirements. As far as we know, this is the first time that the anti-collusive property and implicit authentication of self-healing key distribution schemes have been defined. In *Setup* phase of the proposed solution, each node not only be preloaded with random numbers and secret vectors corresponding to its life cycle, but also a commitment which is used to authenticate broadcast messages. Our technical innovation is embodied by the operations at the phase of *Broadcast*. If no user is revoked, the broadcast message is of the same format as that in [93]. If any user is revoked at session j, the group manager will select new random numbers and change the format of the broadcast message to maintain forward secrecy. The proof under an appropriate model and performance analysis demonstrated that the proposed scheme achieves fully anti-collusive property with acceptable overhead. The solution will be proposed in Chapter 6.

3.6.4 Overview of the solution for HBT-based self-healing key distribution with implicit authentication

We have reviewed the main technical problem of Jiang et al's HBT based self-healing key distribution scheme [112]. This scheme has all the weakness of hash chain based self-healing schemes. Moreover, the mask method is based on Shamir's secret sharing, hence the number of revoked users cannot go over the threshold. After that, we break the technical problem into several research issues. Here, we present a solution for each of these research issues.

In the proposed solution, the generation of a hash binary tree follows that in [112]. The number of leaf nodes of the hash binary tree is the same as the lifetime of the network which is denoted by the number of sessions m. The forward hash chain has a one-to-one correspondence relationship with leaf nodes of the hash binary tree. The backward hash chain is of length $(m + 1)$. The last element of the backward hash chain is used for implicit authentication. Even though the hash binary tree is generated based on a hash chain, its special construction has one good property. The forward keys themselves for each node match the node's prearranged life cycle. Hence, it is unnecessary to introduce random numbers as that in [93]. In the proposed solution, we replace Shamir's secret sharing with vector space secret sharing. The number of revoked nodes is not constrained by the threshold secret sharing scheme. The concrete scheme will be presented in Chapter 7.

3.6.5 Overview of the solution for mutual-healing key distribution and authenticated self-healing key distribution

In the previous section, we pointed out that the only research issue related to mutual-healing is to effectively authenticate requesting nodes. One possible solution which considers the establishment of pairwise keys between nodes and the group manager has been refuted after verification. The only feasible way is to endow each node with a public/private key pair so that it can perform node-to-node authentication on demand.

In the proposed solution, we bound each node with its identity and location rather merely one or the other of them. Before *Setup*, a localize method is used to determine each node's location. A public/private key pair can be calculated from its identity and its location. At the end of this process, each node stores its location and LBK pair.

The mutual-healing operation includes three stages: mutual-healing request, mutual-healing response, and verification. If a node u_i misses the broadcast message B_t and cannot recover session key SK_t in self-healing manner, u_i broadcasts a mutual-healing request including its identity ID_i, location l_i and the sequence number of the expected broadcast message t. The neighboring node u_j who receives the request will check whether the requesting node u_i is its neighboring node. If u_i cannot pass the check, u_j discards the request to avoid the unexpected attacks. If u_i passes the check, u_i will calculates a pairwise key with u_i and send mutual-healing response message to u_i. The requested broadcast message is encrypted the newly calculated pairwise key. u_i can drive the pairwise key from the mutual-healing response message and further obtain the broadcast message B_t.

We mentioned that the critical advantage of Public Key Cryptography is that it provides effective way of authentication. Hence, we implement authentication in bilinear pairings based self-healing key distribution scheme with short signature. The short signature is generated on the broadcast message by the group manager with its private key. Each sensor node verifies the signature with the group manager's public key. This authentication mechanism can effectively ensure the broadcast messages that authorized users receive are sent by the group manager.

The two solutions will be presented in further detail in Chapter 8.

3.6.6 Overview of the solution for practical hybrid key management in HWSNs

We have identified technical problems of existing PKC-based and hybrid solutions for HWSNs in the previous section. To date, no practical PKC-based or hybrid key management solution has been proposed for HWSNs.

The proposed hybrid solution takes advantage of the different capabilities of nodes at different tiers. It uses resource-consuming ECC-based Public Key Cryptography at the BS and cluster heads and efficient AES in low-cost nodes. The BS serves as a CA which generates ECC-based public/private key pairs, one pair for the BS itself and others for cluster heads. Each cluster head has a unique identity and has both cryptographic algorithms ECC/AES installed on it. Each cluster head is preloaded with the BS's public key and a unique ECC public/private key pair. Each low-cost node is preloaded with the node secret. After cluster formation, each cluster head stores a public key list of all of its neighboring cluster heads. Each L-node stores the public keys of its cluster head and the backup cluster head.

Each cluster equals a small communication group. The potential key management framework depends on the communication mode. For the broadcast communication dominated group, efficient group key establishment and renewal schemes are preferred. While for the peer communication dominated group, efficient and secure pairwise key establishment, such as key predistribution, is a better option.

3.7 Conclusion

In this chapter, we firstly outlined a set of definitions that will be used throughout the thesis. We then summarized the technical problems arising from the literature in terms of four key aspects, and formulated the problem definition that we intend to address in this thesis. We then decomposed the main defined problem into six research issues, which form the key requirements for the development of an individual new solution. Further, we discussed a science and engineering based research approach which will be utilized in this thesis for the proposed solution development. Finally in this chapter, we took the first step in proposing the solution for the problem that we intend to address in this thesis. We gave a brief overview of the solutions for each of the outlined research issues and showed how each of these solutions addresses these issues. This work leads us to solve the defined problem of the thesis.

From the next chapter, we will discuss in more detail each of these solutions for attaining the outlined research issues.

Part II

Symmetric Key Cryptography Based Solutions

4

Robust Key Management for Multi-phase Hierarchical Wireless Sensor Networks

4.1 Introduction

In the previous chapters, we mentioned that most of the key predistribution schemes consider a flat topology of homogeneous nodes and support only the establishment of pairwise keys. Even though flat networks are simple and efficient on a small network scale, such networks lack scalability due to "one-affects-n" effect in node addition and revocation. In a hierarchical WSN, the effect of node addition and revocation can be localized into a cluster; thus, scalability is achieved.

In an HWSN, various types of communication may occur. The BS broadcasts control commands to the whole network. The control node multicasts messages within the cluster. A node communicates with its neighboring nodes by unicasting. Therefore, network-wide key, cluster key, and pairwise key are required to satisfy different types of secure communication. One of the better known key management protocols, LEAP [29], supports the establishment of individual keys, pairwise keys, cluster keys, and a global key. Different keys are used to handle the different types of packets. The main problem of LEAP [29] is that the security of all types of keys relies upon that of the initial key. Jang *et al.* [1] improved LEAP by introducing a time-based key management protocol. The scheme strengthens the security with a new notion of probabilistic time intervals. However, the scheme solved the security of [29] at the cost of reduced key connectivity probability.

ZigBee is a protocol specification and industry standard for a type of low-rate wireless personal area networks (LR-WPAN) technology [117]. As depicted in Figure 4.1, the ZigBee standard is built upon the IEEE 802.15.4 standard [38] which defines Physical layer and Media Access Control layer. It defines higher-layer network and application services. Both ZigBee and IEEE 802.15.4 standards together enable the development of complete LR-WPAN systems. The 802.15.4 standard supports basic Media Access Control layer security services, such as access control and data encryption, leaving complex security services to the higher layers. Therefore, in addition to a standardized set of protocols and interfaces for hardware platform and software application, ZigBee provides a security model and a set of security services in order to provide a comprehensive network security infrastructure although security is not mandatory in ZigBee. The security model provides essential security services such as trust infrastructure, encryption, authentication, and admission control for nodes joining the network. After an in-depth review of the security model of ZigBee, we found it is inefficient in network key renewal. Neither forward secrecy nor backward secrecy is adequately addressed, thereby allowing the intrusion of a number of threats at the routing and the application layer [118]. The security of ZigBee should be given full consideration in order to keep up with increasingly wide application.

In this chapter, we propose a comprehensive key management solution. The solution is based on the time-based key management [1] which disperses the damage resulting from the disclosure of the initial key. We clearly handle key generation, distribution, delivery, and erasure processes. This solution can be used to enhance the security service of the ZigBee network. Compared with existing security mechanisms of ZigBee standard, the proposed solution provides sufficient security with acceptable overhead.

This chapter is organized as follows: We review the security problems of LEAP and the time-based key management scheme with focus on research issues to be

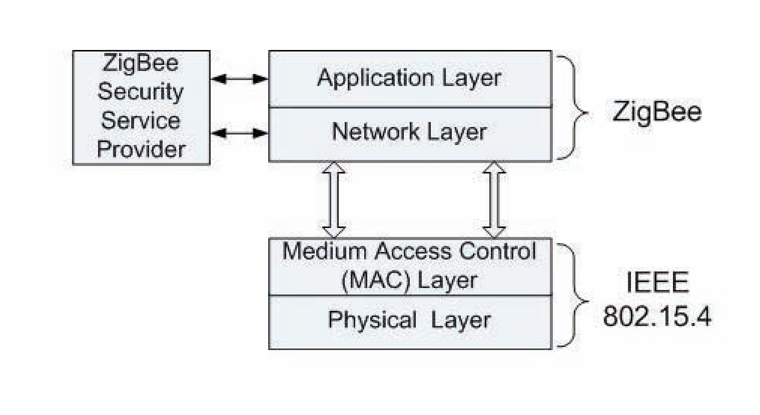

ZigBee Security Service Provider
Application Layer
Network Layer
ZigBee
Medium Access Control (MAC) Layer
Physical Layer
IEEE 802.15.4

Figure 4.1: The ZigBee layer model

addressed in this chapter in Section 4.2. We introduce the necessary requirements of multi-phase deployment of WSNs and outline the problems arising from it in Section 4.3. Then we propose a key management solution which is based on the time-based key management for multi-phase HWSNs in Section 4.4. We firstly give an overview of ZigBee key management and point out its vulnerabilities, then show how the proposed solution can be used to enhance the key management services of ZigBee protocol specification in Section 4.5. We analyze the performance of the proposed solution in Section 4.6 and security in Section 4.7. Finally, we conclude this chapter in Section 4.8.

4.2 The Security Problems of LEAP and the Time-based Key Management Scheme

As one of many existing key management protocols, LEAP assumes that the initial key is secure during the initialization phase and is erased from the memory of sensor nodes when the initialization phase finishes. The authors regard the scheme as secure under such an assumption. However, the same key IK should be used again for node addition and replacement after that phase while the new node can be captured before removing the initial key. According to the assumption, some new nodes may be captured at any time after the initialization phase. That is, the newly deployed nodes might be captured before removing the initial key. The security of the scheme is threatened by the attacks launched after the initialization phase.

Therefore, the initial key IK should never be used for node addition in LEAP after the initial time T_{min}.

Different initial keys are used for different time slots in the time-based key management scheme [1]. The threat caused by the disclosure of the initial key is eliminated. However, the key connectivity is constrained by the number of preloaded master keys m and the order of the current time slot. If m is far less than the lifetime P of the network, the key connectivity $\frac{m}{P-j}$ is far less than 1 at the time slots $j(j \leq P/2)$. On the contrary, if m is close to P, higher key connectivity can be achieved with a heavy burden on storage.

We consider the security problem of the established pairwise key between two nodes. The pairwise key does not exclusively belong to the two end nodes and threat against confidentiality and authentication may arise from it. As shown in Figure 2 in [1], Nodes of group N_1, N_2, and N_6 are preloaded with master key K_{u7}, the pairwise keys between any two groups of them are known by the other group. In addition, m master keys of randomly-chosen time slots are preloaded to the nodes when they are deployed to the network without taking the lifetime of nodes into consideration. Suppose a node which can survive at most G_w time slots is deployed at the j-th time slot with m master keys of randomly-chosen time slots. Those master keys of the time slots from $(j + G_w)$-th to P-th would never be used. They waste the scant memory of sensor nodes.

4.3 The Importance of Multi-phase Deployment of Wireless Sensor Networks

Most of the current key management systems neglect the relationship between the lifetime of WSNs and the limited battery of sensor nodes. Usually, WSNs should work for longer time than lifetime of ordinary sensor nodes. However, current literature about key management are designed for single-phase WSNs even though some of them allow dynamic node addition or revocation. WSNs are composed of low-cost sensor nodes which are powered by irreplaceable batteries. Therefore, new nodes should be deployed to replace erroneous nodes or power-off nodes. In this way, WSNs can retain their expected network connectivity and work well.

In order to make deployment easy, new nodes are periodically deployed into WSNs in batches. Then two new problems arise from multi-phase deployment. In some highly sensitive environments, how can key connectivity be maintained between nodes deployed at different periods? If an attacker can collect information from compromised nodes or power-off nodes and launch a lengthy attack against the networks, how can the networks be safeguarded from such a long-term attack or minimize the risk?

General key predistribution schemes assume a static key pool which does not renew over time. The replenished nodes also draw a subset of keys from the key pool. As a result, some already compromised keys might be in use again. It is possible to eventually discover the whole key pool if the attacker continues to capture nodes. A collapsed key pool may have a disastrous effect on the network. In view of these potential attacks, we have to find the right solution to renew the key pool and remove compromised or outdated keys. In addition, the nodes for a particular period should be preloaded with appropriate keys which enable them to sustain communication with nodes deployed at different periods.

To the best of our knowledge, RoK [119] is the first key predistribution scheme adapted to multi-phase WSNs. In this scheme, sensor nodes which run out of power will be removed from the network and new sensor nodes need to be periodically deployed to ensure network connectivity. Correspondingly, the predistributed keys have limited lifetimes and the key pool should be refreshed periodically. This scheme overcomes the drawback of the general key predistribution schemes [3, 42]. The security of the network does not degrade with time. Zo-RoK [120] takes advantage of prior deployment knowledge in order to reduce the size of the key ring. In this way, the resilience of the network against node capture attack increases with a smaller key ring of each node, yet deployment knowledge is not always available in WSNs. Recently, a random generation material (RGM) key predistribution scheme was proposed in [121]. In this scheme, the life-time of the whole network is divided into generations. Each generation has its own random keying material and pairwise keys shared by two nodes which are known only to the nodes in the two generations to which the two nodes belong. Nodes deployed in other generations have no access to the pairwise key. We extend this idea to fix security issues of the LEAP scheme [29] and the time-based key management scheme [1] in this chapter.

4.4 The Construction

We propose a novel key management protocol for multi-phase HWSNs. In this scheme, the time domain of the network is split into many time slots. Alternatively, we call the time slot generation as well. It is assumed that there are in total P generations. We assume that a BS handles key management and network management. The BS has numerous resources and is safe from the threat. The middle nodes are cluster heads which have the function of cluster management. The general sensors are usually powered by battery and have very limited resources. It is assumed that a node may live at most G_w generations. In the proposed solution, each generation has its initial key which is pre-installed to memory of nodes. There is no correlation between the initial keys for different time slots. This property prevents the attackers from concluding the previous and future initial keys. WSNs are set up for a longer life-time as compared to that of sensor nodes. Therefore, new nodes need to be replenished in some generations to provide continuity of the network in the case of node capture attack or depletion of battery. It is supposed that an attacker could obtain all the key materials stored in the captured node. In order to achieve a higher key connectivity probability between nodes belonging to different generations and resilience, the keys that are used to establish pairwise keys should evolve in a different way, independent of evolution of the initial keys. Table 4.1 presents the symbols used in our proposed key management protocol.

Table 4.1: Notations used in the proposed key management scheme

u, v	The low-cost devices
P	The initial key pool size and the whole life time of the network
KR_j	The key ring of nodes deployed at generation j
K_{uv}^{j}	Pairwise key between nodes u and v which deployed at generation j
K_{uv}^{gh}	Pairwise key between nodes u which deployed at generation g and v which deployed at generation v where $1 \leq g < h \leq g + G_w - 1$
$H(\cdot)$	Secure hash function
f_k	Pseudo-random function
$MAC(k, s)$	The MAC of message s using a symmetric key k

4.4.1 Precomputation of Key Materials

The BS computes key materials prior to deployment. The key materials include a Network key for network-wide broadcast communication, an individual key for each node, and initial keys and masked initial keys for nodes deployed at different generations.

- Network key. The Network key NK_1 is shared by all the entities in the network. It is used to encrypt broadcast messages. We call it NK_1 because it is an initial network key and will be renewed periodically or on each membership change.

- Individual key. An individual key is a unique key shared between the BS and each node. The individual key keys are computed from the Master key. For security reasons, the Master key K^m is only known to the BS and is not preloaded to ordinary nodes. Instead, each pre-computed individual key is loaded to the corresponding node prior to its deployment. The individual key is calculated as $K_u^m = f_{K^m}(ID_u)$.

- Initial keys and masked initial keys. As depicted in Figure 4.2, the group G_j deployed at generation T_j are assigned with an initial key IK_j and $(G_w - 1)$ masked initial keys in order to establish links with nodes deployed at the same or different generations. The initial key is reserved for the establishment of pairwise keys with nodes deployed in the same generation. The other $(G_w - 1)$ masked initial keys are used to establish pairwise keys with the groups deployed at subsequent generations. Different from the time-based key management scheme in [1], the masked initial keys in our scheme are constructed in an ingenious way. These masked initial keys are transformed from the initial keys for the generations within the node's generation window G_w. For a group of nodes deployed at generation j, their sub-keyring $S - KR_j$ containing the masked initial keys for the subsequent generations $j + 1 \leq l < j + G_w - 1$, is given as follows:

$$S - KR_j = \{K_{j,l} | K_{j,l} = H(IK_l \parallel j)\}, \tag{4.1}$$

  where IK_l is the initial key for the l-th generation.

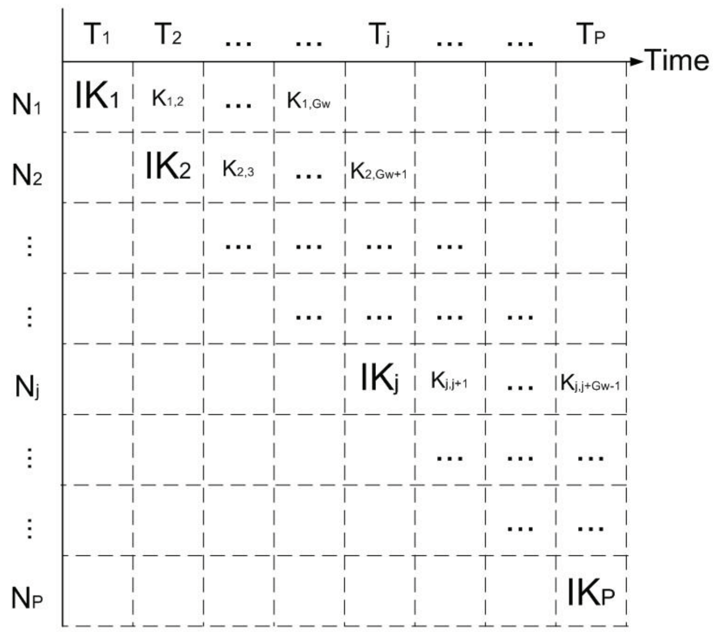


Figure 4.2: The key rings preloaded to nodes at different generations

This process can be detailed as follows. In order to form the sub-keyring $S - KR_j$, the BS first picks up $(G_w - 1)$ initial keys for the subsequent $(G_w - 1)$ generations. Each of these keys is appended with the generation number of the group, which is j, and hashed using a secure hash function like SHA-1 [122] or SHA-256 [123], depending on the key size. These hashed values are stored in the sub-keyring. In this way, we customize the keys belonging to a subsequent generation to be used in another generation without storing the actual initial keys, owing to the one-way property of the secure hash functions.

To sum up, the keyring of a sensor node contains two parts:

1. the initial key assigned to the current generation;
2. the masked initial keys for up to $(G_w - 1)$ subsequent generations.

More formally, for a node deployed at generation j, its keyring KR_j is shown as follows:

$$KR_j = \{IK_j, K_{j,j+1}, \dots, K_{j,j+G_w-1}\}. \tag{4.2}$$

4.4.2 Predistribution of Key Materials

Before deployment, a node u which belongs to the generation j is preloaded with the current Network key NK_j, an individual key K_u^m, and a keyring KR_j. The keyring consists of one initial key and ($G_w - 1$) masked initial keys for each upcoming generation. Because a sensor node may communicate with nodes at most ($G_w - 1$) generations, the maximum number of keys in the keyring of each node is G_w.

The BS is preloaded with the Network key NK_1 before it is deployed into the network. After deployment, the BS is responsible Network key updating. The Network key can be updated periodically or on each membership change in some security-critical environments. The BS always keeps the latest Network key. It is not necessary to store all the individual keys at the BS. Instead, it stores only the Master key K^m and generates the individual key with a specific node when it wants to unicast messages to the node. In addition, the BS stores P initial keys corresponding to the lifetime of the network.

4.4.3 Pairwise Key Establishment

After the keyring is created, the nodes are deployed at the sensor field generation by generation. Two situations exist for the establishment of pairwise keys.

1. Since all sensor nodes deployed at generation j contain the initial key IK_j, they can establish pairwise keys using IK_j. Suppose two nodes u and v belong to the same generation j, they compute their pairwise keys as follows:

    - Node u broadcasts a HELLO message including its ID and generation j and then waits for a response from the neighboring node v which is deployed in the same generation. Node v sends node u a response message including its ID and a MAC.

$$u \rightarrow * : ID_u, j, nonce;$$

$$v \to u : ID_v, MAC(K_v^j, ID_u \mid ID_v)$$

where $K_v^j = f_{IK_j}(ID_v)$.

- Both u and v can compute a pairwise key independently. In order to keep consistency, the pairwise key is computed by $K_{uv}^j = f_{K_u^j}(ID_v)$ if $ID_u < ID_v$ or $K_{vu}^j = f_{K_v^j}(ID_u)$ if $ID_u > ID_v$.

2. For any two nodes belonging to different generations, the establishment process of pairwise key is different. Let us suppose that node u deployed at generation g and another node v deployed at generation h ($1 \leq g < h \leq g + G_w - 1$). They compute their pairwise keys as follows:

$$K_{uv}^{gh} = f_{K_{gh}}(ID_u \mid ID_v).$$

Node u is already preloaded with the key K_{gh} before deployment. From the point of view of node v, K_{gh} is the masked initial key for group deployed at the previous generation g. Therefore, it is not in node v's keyring. Fortunately, node v can calculate K_{gh}. As discussed in the previous subsection, $K_{gh} = H(IK_h \parallel g)$, where H is a secure one-way hash function. Node v is deployed at generation h, and therefore it stores the initial key IK_h. By using IK_h, it calculates the key K_{gh} and then calculates K_{uv}^{gh}.

4.4.4 Pseudo-code of the Proposed Protocol

To sum up, we present a pseudo-code of the proposed protocol as follows:

Sub-program

```
1   Key generator $KG(l)$; '$l$ is the length of key'
2   MAC function $MAC(K, m)$;
3   Pseudo-random function $f_K(m)$;
4   Hash function $H(m)$.
```

Main Program

```
1    Initial parameters
```

```
  $p$, 'life-time of network, measured by number of generation'
  $G_w$, 'life-time of nodes'
  $n$, 'average number of one-hop neighbors of a node'
  $nonce$;

**Precomputation of key materials**
$NK_1 = GK(l)$ 'generate Network key';
$K^m = KG(l)$ 'generate Master key';
**For** each node $u$ to be deployed in the network
  $K_u^m = f_{K^m}(ID_u)$ 'generate the individual key'
**end**
**For** $i = 1 : p$
  $KG(l)$
**end** 'generate the initial keys for $p$ generations $\{IK_1, \ldots, IK_p\}$'
**For** $j = 1 : p$
    **For** $i = j + 1 : j + G_w + 1$
      $K_{j,i} = H(IK_i \mid j)$
    **end**
**end**'generate key ring for nodes deployed at the generation $j$'

**Predistribution of key materials**
preload $\{NK_1, K^m\} \cup \{IK_1, \ldots, IK_p\}$ to the BS
if a node $u$ is deployed to the network at generation $j$, preload $\{NK_j, K^m\} \cup$
$\{IK_j, K_{j,j+1}, \ldots, K_{j,j+G_w-1}\}$ to node $u$

**Pairwise key establishment**
**For** each node in the network, broadcast Hello message
  $u \to * : ID_u, j, nonce$
a node $v$ receives the Hello message
  $v \to u : ID_v, MAC(K_v^j, ID_u \mid ID_v)$
**If** $u$ and $v$ belongs to the same generation $j$
    **If** $ID_u < ID_v$,
      $K_{uv}^j = f_{K_u^j}(ID_v)$
    **else** $K_{vu}^j = f_{K_v^j}(ID_u)$
```

```
32      else if $v$ belongs to generation $h$
33          $K_{uv}^{jh} = f_{K_{jh}}(ID_u \mid ID_v)$.
```

4.4.5 Node Revocation and Rekeying

Rekeying occurs either periodically or after membership changes or node compromise is detected. Periodical rekeying and Network key rekeying on node addition are much easier than rekeying when the node leaves. In normal periodical rekeying, the BS encrypts the new Network key with the old Network key and broadcasts the encrypted message through the network. All authenticated nodes will receive the new Network key. Network key rekeying on node addition can be performed in the same way with an extra unicast step. The Base Station encrypts the new Network key with the individual key and unicasts it to the newly joined nodes.

Node revocation is incurred by energy depletion or node compromise. We classify it as active revocation when node is power off and passive revocation when node is compromised. When a node u's power is to be exhausted, node u encrypts a revocation message with the current Network key NK_j and broadcasts it as follows:

$$u \rightarrow * : ID_u \parallel timestamp \parallel E_{NK_j}(Rov \mid ID_u \mid timestamp).$$

At the same time, node u erases all the keys it has, including the Network key, the individual key, and pairwise keys shared with its neighboring nodes. The neighboring nodes which share a pairwise key will erase the pairwise key once they receive the revocation message. Network rekeying is unnecessary as the revoked node u's behavior is normal and no key materials are disclosed.

In the case of node compromise, the compromised node must be passively revoked by the Coordinator. A node revocation announcement must be authenticated to prevent an outsider adversary from impersonating the BS. In our proposed scheme, we follow the same rekeying method as that of LEAP. Once a node u is identified as compromised, the BS renews the group key and sends a message identifying the compromised node u and providing an authentication key $f_{K'_g}(0)$ for the new Network key. Each neighbor node that receives the message deletes the pairwise key shared with u. Then the BS encrypts the new Network key and sends it to its neighbor routers which in turn authenticate the new Network key and then send it to their neighbors and so on until all nodes receive the new Network key. In this recursive process, we assume that a breadth-first spanning tree is established by a routing protocol and the router has a basic management function. This manner of hop-by-hop flooding incurs significant overhead for sensor nodes. Fortunately, Network rekeying on node compromise is a relatively less frequent event.

4.5 ZigBee Key Management

4.5.1 ZigBee Network Topologies and Types of Keys

As depicted in Figure 4.3, the ZigBee network supports Star, Tree, and Mesh topologies. ZigBee network may comprise three types of devices: Coordinator, Router, and end device. In particular, a ZigBee network must have a Trust Center. The Trust Center is usually the Coordinator, which provides key management and other security services. In the Star topology, the network is controlled by the Coordinator, which is responsible for initiating and maintaining the devices on the network, while the end devices communicate with the Coordinator [124]. In Mesh and Tree topologies, the Coordinator is responsible for starting the network and for choosing certain key network parameters, but the network may also be extended through the use of ZigBee Routers [118]. Nowadays, the most important and promising ZigBee application profiles seem to be Home Automation [125] and Smart Energy [126]. Our research focuses on the latter since it considers security as a major issue and includes precise mechanisms for secure communications.

In terms of key types, ZigBee uses Master key, Link key, and Network key for authentication and encryption.

- Master key. The Master key is not used to encrypt data. Instead, it is used to authenticate the node when the node joins the network and as an initial shared secret between two devices when they generate Link keys.

- Link key. The Link key is unique for a pair of devices and is used for end-to-end encryption. In particular, the key shared by a device and the Trust Center is called Trust Center Link (TCL) key. The TCL key is established during

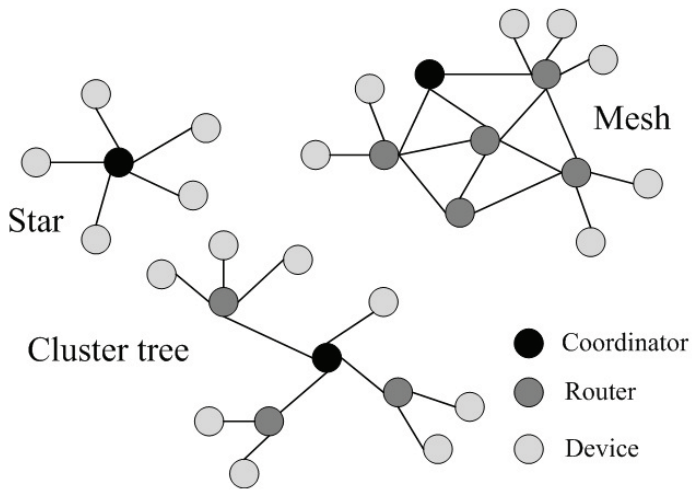


Figure 4.3: The topologies of ZigBee networks

the join procedure and is used to protect application level messages and stack commands.

- Network key. The Network key is shared by all devices and is used for broadcast management and to control communications. A high security Network key must always be sent encrypted over the air [127].

4.5.2 Security Problems of ZigBee Key Management

Two basic security requirements of network communication are forward secrecy which prevents a departing or expelled device from having continued access to further communication, and backward secrecy which prevents a newly joined device from accessing the previous communication. Usually, these two requirements are achieved by a timely key revocation and rekeying mechanism. Rekeying a group before the addition of a new member is simple, while rekeying the group after a member leaves is far more complicated [128]. The ZigBee specification provides a backward secrecy mechanism but leaves the issue of forward secrecy unresolved.

The ZigBee specification dictates that the Trust Center should refresh the Network key periodically, but it does not specify either the refresh period or any event that triggers the Network key refresh. It says nothing about key refreshment upon a device's departure. No effective measure is adopted to safeguard Network key and Link key on membership change. All the same, the revoked devices can access all communications encrypted by these keys. The adversary may compromise the device and exploit the Network key to spoof and inject bogus routing information. More seriously, the adversary may launch highly disruptive routing attacks such as sinkhole attack and selective forwarding attack [118].

As far as the two ways to refresh the Network key are concerned, none of them is effective. The overdue Network key is used to refresh the new Network key in the broadcast-based refresh method. While in the unicast-based refresh method, the new Network key is encrypted with each device's Transport key and delivered to each device in a one-to-one fashion. This solution is secure but has scalability limitations.

As to the Link between two devices, the ZigBee specification does specify how to deal with it when one of the two device leaves the network or how to inform its neighbors when a device leaves the network.

4.5.3 The Enhanced Key Management of the ZigBee Specification

The proposed solution can be directly used to enhance the key management services of the ZigBee tree architecture (As shown in Figure 4.3). We assume that the Coordinator has the function of the Trust Center and the Routers have cluster management function. The corresponding relationship of the entities in the proposed solution and the ZigBee specification is shown in the Table 4.2.

Table 4.2: The corresponding relationship of the entities in the proposed solution and the ZigBee specification

Entities in the proposed solution	Entities in the ZigBee specification
The BS	The Coordinator
Routers	Cluster heads
Ordinary sensor nodes	End devices

We mentioned that ZigBee uses three types of keys for authentication and encryption. The corresponding relationship of the keys in the proposed solution and the ZigBee specification is shown in the Table 4.3.

Table 4.3: The corresponding relationship of the keys in the proposed solution and the ZigBee specification

Keys in the proposed solution	Keys in the ZigBee specification
Master key	Master key
Link key	Pairwise key
TCL key	Individual key
Network key	Network key

4.5.4 Authentication for Node Admission Control in ZigBee Specification

Secure node admission avoids the situation where unauthorized nodes join the network. In the ZigBee specification, the Coordinator controls the policy of node admission. Each authorized node is preloaded with a common network security key prior to deployment. The Coordinator checks the validity of the common Network key. Only those nodes possessing the correct Network key should be permitted to join the network. This authentication method does not take the Network key updating into consideration. What this means, firstly, is that the Network key is renewed over the generations while the nodes still request admission with the original Network key in later generations. Secondly, the authentication is not secure if the Network key is disclosed.

In the enhanced key management services, we use the TCL key for authentication as it is the unique key shared by each node and the Coordinator. The Master key K^m is stored by the Coordinator and does not change over time. Suppose a node u is requesting for admission, the authentication process can be displayed as follows:

$$u \rightarrow \text{the Coordinator} : ID_u \parallel nonce \parallel MAC(K_u^m, ID_u \mid nonce).$$

The Coordinator generates the TCL key $K_u^m = f_{K^m}(ID_u)$ and uses it to check the correctness of the MAC value. If the MAC value is correct, node u passes the authentication and joins the network. Otherwise, node u is rejected from joining the network.

4.6 Security Evaluation

The advantage of the proposed protocol is that it embodies security improvement of LEAP [29], the time-based key management scheme [1], and the ZigBee security services. The goal of this section is to evaluate the security of our proposal and compare it with the refereed papers. We attempt to discuss the defences against node capture attack and the degree of resilience of the different schemes.

4.6.1 Advantages over LEAP and the Time-based Scheme

We assume that when a node is compromised, the key material stored in the node will be extracted by the adversary. The key material will be utilized to attack the rest of the network. In [1], the resilience of schemes is described as the additional portion of network that an adversary can compromise using the key material obtained from x compromised nodes. We still use this definition in this section. The security of the LEAP scheme depends on the security of the initial key IK. The whole network can be compromised once the initial key K_I is disclosed. The damage resulting from a disclosure of an initial key IK is localized by the time-based key management scheme [1]. The lifetime of the network is divided into P generations and each generation has its initial key. A compromised initial key IK_a at generation T_a affects only the nodes deployed at generation T_a rather than the whole network. In order to provide connectivity between nodes deployed at different generations, the preloaded master key for different generation is the same. However, as we mentioned in Section 4.2, the pairwise key does not exclusively belong to the two end nodes. If three nodes are preloaded with the same master key, the pairwise keys between any two groups of them are known by the other group. Once a node is captured, the pairwise keys shared by the other two nodes will be compromised as well. This property makes the scheme vulnerable to attacks against forward and backward secrecy.

In our scheme, either the pairwise key K_{uv}^{j} or K_{uv}^{gh} exclusively belongs to the two end nodes. For example, the pairwise key K_{uv}^{j} shared by node u and v which are deployed at generation j is shared only by them and the pairwise key K_{uv}^{gh} shared by node u deployed at generation g and v deployed at generation h is confined to the two end nodes deployed at these two generations. We take K_{uv}^{gh} as example. Nodes deployed at generations other than g and h have no right to access this key. This is because the masked initial key K_{gh} is preloaded to the nodes which is deployed at generation g and can be calculated by nodes if and only if these node are deployed at generation h and have the initial key IK_h in their key ring. As a result, a sensor node w, which is deployed at any other generation l (for $l \neq g$ & $l \neq h$) cannot calculate a masked initial key K_{gh}. Three conditions exist according to the value of l:

- $l < h$. The node w needs IK_h to calculate K_{gh}. Even though the node w has the masked initial key $K_{lh} = H(IK_h \parallel l)$, it cannot derive IK_h from K_{lh} due to the one-way property of the secure hash function H.

- $h < l \leq g + G_w - 1$. The node u is preloaded with the masked initial key $K_{gl} = H(IK_l \parallel g)$ and the node v is preloaded with the masked initial key $K_{hl} = H(IK_l \parallel l)$. Even the node w can calculate K_{gl} and K_{hl}, it cannot derive IK_h or K_{gh} due to the one-way property of the secure hash function H.

- $l > g + G_w - 1$. The node u has powered off at the generation l.

It is clear that the masked initial key K_{gh} is known only by the nodes deployed at generations g and h. No node deployed at any other generation can calculate the key that is unique to the generation g and h. Hence, an attacker has to spend extra effort if s/he wants to acquire the pairwise key between the nodes u and v that are deployed at the generations g and h, respectively. This has an advantage over the scheme in LEAP and the time-based key management scheme [1] in restricting the information that an attack acquires if s/he captures a node.

4.6.2 Advantages over ZigBee Specification

In the proposed solution for multi-phase ZigBee architecture, each node u which is to be powered off broadcasts a revocation message to its neighbors and erases all the keys it has, including the Network key, the TCL key, and Link keys shared with its neighboring nodes. The neighboring nodes which share a Link key will erase the Link key. In this case, Network rekeying is unnecessary as the leaving node has actively erased all the keys it has.

In the case where node compromise is detected, the Network key must be refreshed and node revocation announcement is authenticated to prevent possible attacks. Our solution follows the recursive rekeying method of LEAP.

In the case of node addition, the Network key must also be refreshed in order to maintain backward secrecy. The key refreshment in this case is as easy as periodically rekeying. In the proposed solution, nodes are periodically added to the network in batches. In practical applications, nodes perform a normal operation such as sensing and monitoring the environment during a normal phase. At the beginning of each new generation, a batch of new nodes is deployed to the network. There is a corresponding relationship between the generation and period for node addition. In the time-based model, the beginning of the new generation occurs with node addition so that the number of generations approximately equals the periods of node addition [1]. Usually, the main purpose of node addition is to complement the network and keep network connectivity so the frequency of node addition is not necessarily high. In addition, in our model, the size of the initial key pool is the same as the number of generations. Therefore, the size of key pool in our model is small.

4.7 Performance Evaluation

4.7.1 Key Connectivity.

The key connectivity of a group N_t which is deployed at generation t of the time-based key management scheme [1] is assessed from two aspects. One is N_t's probability of sharing keying materials with prospective sensor nodes, $p_{pros}(t)$ and the other is the probability of sharing keying materials with pre-deployed sensor nodes and nodes deployed at the current generation G_t, $p_{pre}(t)$. According to the scheme, when a sensor node is deployed at generation t, only the pre-deployed nodes which have the master key, which is derived from the initial key IK_t of the generation t, can establish a pairwise key with it. Because each node is preloaded with the initial key of the current generation and m master keys of randomly chosen generations after deployment, the probability of sharing key material with other prospective nodes is $\frac{m}{P-t}$ where P is the number of total generations (Please refer to [1] for the process of calculation). The variation $1 \leq t < P$. When $P - t \leq m$, the master keys of all remaining generations will be preloaded to the nodes deployed at generation t so that the key connectivity probability $p_{pros}(t)$ is 1 and keeps 1 for the subsequent generations. We make a comparison of the probabilities that N_t establishes the pairwise key with prospective groups in our scheme and the scheme in [1] in Figure 4.4. In order to facilitate the comparison, we keep the size of key pool (the same as the number of generations of a network) 500. The three curves in Figure 4.4 demonstrate the conditions of [1] when m equals 100, 150, and 200, respectively. As shown in the figure, the probability p_{pros} increases as the index the generation t increases. In our scheme, the group N_t can always share keying material with groups in its generation window $[t, t + G_w - 1]$ with 100% probability. The variation $1 \leq m \leq P - 1$, it is easy to reach the conclusion that the more master keys a sensor node has in [1], the higher is the probability of key connectivity. In our scheme, more than $(G_w - 1)$ preloaded master keys do not increase the key connectivity probability as $(G_w - 1)$ master keys are enough to reach 100% probability.

In terms of $p_{pre}(t)$, it can be calculated by using p_{pros}. If we assume that sensor nodes are uniformly distributed at each generation, p_{pre} can be calculated as:

$$p_{pre}(t) = (\sum_{i=1}^{t-1} p_{pros}(i) + 1)/t,$$

where 1 means that a sensor node can establish pairwise keys with nodes deployed at the same generation with 100% probability. In fact, it is the average value of the probabilities of key connectivity with each preloaded and the current group of sensor nodes. Figure 4.5 describes the change tendency of probability p_{pre} for

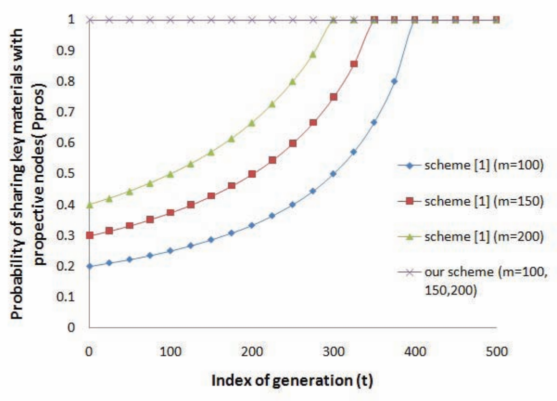


Figure 4.4: The comparison of the probabilities that N_t establishes pairwise keys with prospective groups in our scheme and the scheme in [1]

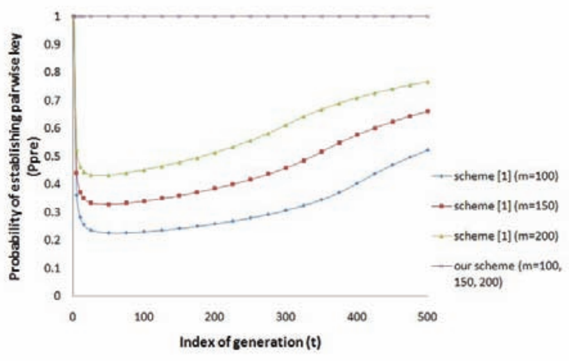


Figure 4.5: The comparison of the probabilities that N_i establishes pairwise key with pre-deployed sensors and the sensors being deployed in the same generation in our scheme and the scheme in [1]

various m when the number of generations P is fixed at 500.

Different from the scheme in [1], when a sensor node is deployed at generation t, it is preloaded with the initial key IK_t and $(G_w - 1)$ master keys for the subsequent $(G_w - 1)$ generations. The node can establish pairwise keys with nodes in at most $(2G_w - 1)$ generations with 100% probability as long as both of them are still viable. As described in Figure 4.6, a node deployed at generation j can establish pairwise keys with nodes deployed from generation $(j - G_w + 1)$ to generation j by using its initial key IK_j and establish pairwise keys with nodes deployed from generation $(j + 1)$ to generation $(j + G_w - 1)$ by using the preloaded master keys for each generation. In the time-based scheme [1], a node is preloaded with m master keys

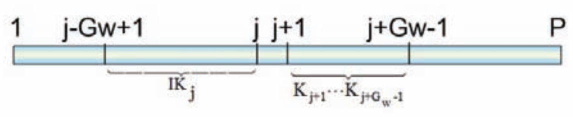


Figure 4.6: Nodes deployed at generation j can establish pairwise nodes with at most $2G_w - 1$ generations with 100% probability

of randomly- chosen generations. Some of these master keys might not be used before the node is power off. Suppose a node is deployed at generation j and is viable at most G_w generations, those master keys of the generations from $(j + G_w)$ to P are useless.

In the RGM key predistribution scheme [121], two conditions are discussed regarding key connectivity. One is the key sharing probability of the node pairs deployed at the same generations and the other is the key sharing probability of node pairs deployed at different generations. These two key sharing probabilities are different since different parts of keyring are utilized for connections at these two conditions.

In the RGM scheme, the key sharing probability of node pairs deployed at the same generation is denoted as P_{sg}. In fact, RGM scheme is an extension of the basic random key predistribution scheme [3]. The single-phase case of RGM scheme is totally the same as [3]. Suppose the size of key pool for each generation is P and each node stores m current generation keys in its keyring, the key sharing probability that a node pair shares at least one key is

$$P_{sg} = 1 - \frac{C(P-m, m)}{C(P, m)}. \tag{4.3}$$

The key sharing probability of node pairs deployed at the same generation is denoted as P_{dg}. A node can derive m past generation keys from its m current generation keys. Each node stores n future generation keys for each future generation in its generation window. This is equivalent to the case where a node has a sub-keyring with m keys and another node has a sub-keyring with n keys, which are chosen from the same key pool of P keys. Hence, the key sharing probability that a node pair has at least one key in common is

$$P_{sg} = 1 - \frac{C(P, m+n)C(m+n, m)}{C(P, m)C(P, n)}. \qquad (4.4)$$

It is supposed that $m \geq n$ in RGM scheme. It means $P_{sg} \geq P_{dg}$. The overall key connectivity of RGM scheme is a value within the range $[P_{dg}, P_{sg}]$. The maximum value of key connectivity of RGM scheme is P_{sg} which is the same as that of the basic random key predistribution scheme. It is obvious that the key connectivity probability of RGM scheme is lower than that of our solution, which is 1.

4.7.2 Complexity Analysis

Storage cost

The storage overhead is affected by the degree of the network. The degree of the network is usually defined as the average number of immediate neighboring nodes of a given node. It is proportional to the density of nodes in a network and may be a value from 10 to 20 for a reasonably dense network [129]. Here we use the same assumption that the degree of the networks in LEAP [29] and our protocol is d. The storage overhead of the time-based scheme [1] and RGM [121] are not related to the degree of the network.

In LEAP, each node stores three keys for each neighbor, including a pairwise key, a cluster key and a key chain with L values. In addition, each node stores a network key and a master key. The total number of keys that each node stores is $(3d + L + 2)$.

In the time-based key management scheme, each node is preloaded with an individual key shared with the BS and one initial key and m master keys. All of the other keys are established based on the preloaded keys. According to our analysis in Section 4.2, this scheme cannot guarantee 100% key connectivity.

In the RGM scheme, each node stores m current generation keys and n future generation keys for each future generations. A node whose generation window is G_w may establish communication with at most $(G_w - 1)$ next generations Hence, each node has to store $(G_w - 1) \times n + m$ keys.

In the proposed protocol, the storage is determined by the memory of a node and generations that a node can survive. The number of generations is approximately equal to the number of node additions. Therefore, the higher the frequency of node additions, the larger is the number of generations that a network has. According to our scheme, each node is preloaded with a Network key, an individual key, an initial key and $G_w - 1$ masked initial keys. So the total preloaded number of key materials is $G_w + 2$. After the phase of pairwise key establishment, each node stores $d + 2$ keys, including d pairwise keys and an individual key and a Network key. Suppose a sensor node surviving 100 generations has to store 100 keys including one initial key and 99 masked initial keys. These node can establish pairwise keys with nodes in 199 generations with 100% probability. The modern sensor nodes such as MICA-Z have 128KB program memory, 4KB runtime memory, and 512KB external memory [130]. Suppose the size of a key is 128bits, our scheme requires only 1.6KB memory. Our scheme has a reasonable storage requirement for modern sensor nodes.

Computation and communication cost

In LEAP [29], the time-based scheme [1], RGM [121], and the proposed solution, each node broadcasts a hello message to find its neighboring nodes when it joins the network. Both time or energy consumption are the same in this step for these four schemes. However, time and energy consumption varies in these four schemes for Link key establishment. In LEAP, all the nodes are deployed at the same time. We take nodes u and v as example, u and v establish Link key with algorithm $K^j_{uv} = f_{K^j_u}(ID_v)$ if $ID_u < ID_v$ or $K^j_{vu} = f_{K^j_v}(ID_u)$ if $ID_u > ID_v$. While in the proposed solution, nodes are deployed to network generation by generation. Neighboring nodes deployed in the same generation establish the pairwise key with the same algorithm as that in LEAP. While for the node u deployed at generation g and another node v deployed at generation $h(1 \leq g < h \leq g + G_w - 1)$, they calculate their pairwise keys with algorithm $K^{gh}_{uv} = f_{K_{gh}}(ID_u \mid ID_v)$. We have to mention that even though [1] adopts the same Link key establishment algorithm, it cannot

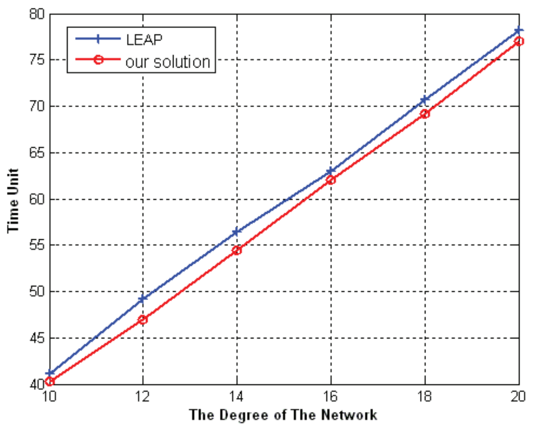


Figure 4.7: Comparison of time consumption of LEAP and the proposed solution

guarantee 100% key connectivity and the established pairwise key does not exclusively belong to the two generations. In RGM, all the shared generation keys contribute to the pairwise key. The pairwise key is the hash value of the concatenation of all the shared generation keys. The calculation does not increase computation overhead while strengthening the security. However, RGM does not guarantee that any two nodes share at least one generation key. That is, it cannot reach 100% key connectivity probability. It does not consider path key establishment as the basic random key predistribution scheme [3].

We suppose the degree of network varies from 10 to 20 with span 2 in LEAP and the proposed scheme. In the proposed scheme, we assume that half the neighboring nodes for a specific node are deployed in the same generation and the remaining neighboring nodes are deployed in the different generations. The length of ID of each node is 4 bytes and the key length is 128 bits. We run the pairwise key establishment algorithm with MATLAB. Figure 4.7 shows a comparison of the time consumption.

4.8 Conclusion

Although the LEAP key management mechanism is welcomed due to its multiple keying mechanism and apparently strengthens the security of ZigBee, the security of all types of keys is mainly dependent on that of an initial key. It is assumed that the initial deployment phase is secure and the key is erased from sensor nodes after the initialization phase. However, the same key should be used again for node addition after that phase while the new node can be captured before removing the initial key. A time-based key management scheme was proposed to eliminate the effect of disclosure of the initial keys. This scheme does disperse the damage resulting from the disclosure of the initial key with the cost of reduced key connectivity and other security vulnerabilities. We identified the security problems of LEAP and the time-based key management schemes and presented a robust key management scheme for multi-phase HWSNs. We identified security issues of the ZigBee specification in particular its key management mechanism and showed that the proposed robust key management solution can be directly used to enhance the key management services of the ZigBee specification. We demonstrated the advantages of the proposed solution with sound analysis, comparison, and simulation.

5

Keyed Hash Chain Based Key Predistribution Scheme

5.1 Introduction

Many previous researches focus on flat topology of wireless sensor networks which comprise only homogeneous nodes. This type of network is simple and efficient small-scale networks, but, it lacks scalability. In a homogeneous network, the sensor node which performs the data aggregation and forwarding function will run out of power in advance because it receives higher traffic volume. This in turn leads to the collapse of the whole network. An alternative way to extend the network life is to randomly and periodically rotate the responsibility of the data aggregation function to all nodes as the scheme proposed in [131]. However, this rotation raises the requirement of stronger hardware capabilities. Both [132] and [133] demonstrated the performance bottleneck of homogeneous networks.

HWSNs consist of heterogeneous nodes, including a large number of low-end nodes and a small number of high-end nodes. Low-end nodes have a tiny memory and very limited data processing capability; while high-end nodes have more storage space and stronger data processing capability. High-end nodes are deployed together with low-end nodes. They aggregate data and forward the aggregated data to the BS. Also, high-end nodes which work as cluster heads help the BS to manage the network. Addition and revocation operations can be performed within a cluster. In this way, the influence of a compromised node can be localized within a cluster rather than affecting the whole network. Therefore, HWSNs provide scalability and security benefits.

As applications gain more ground, security issues in wireless sensor networks have attracted more concern. There have been several attempts to secure WSNs, although all of them have their pros and cons and none gives a satisfactory performance. Security is a challenging problem in WSNs due to the constraint on memory and energy at low-end nodes. Asymmetric cryptographic algorithms, such as RSA public key techniques, while applicable to wired networks even general ad hoc networks, may not be suitable for sensor networks. It is well known that ECC can obtain the same security level as RSA with a shorter key length. A 160-bit ECC key has the same security level as a 1024-bit RSA key [134]. However, the research in [135] demonstrated that the Diffie-Hellman key agreement process using the Elliptic Curve asymmetric key algorithm in an ad hoc network is between one to two orders of magnitude larger than the key exchange process based on the AES symmetric key algorithm in a regular non ad hoc network. Therefore, we approach the problem using Symmetric Key Cryptography.

In this chapter, we focus on the design of a key predistribution solution for HWSNs. The proposed solution is more efficient and effective. This solution supports the establishment and renewal of five types of keys in the network because a single key can not satisfy different communication requirements. In the proposed solution, we utilize a keyed hash chain to reduce storage overhead. Unlike existing keyed hash chain based schemes, sensor nodes store only commitments for the corresponding key chains. Even though one or more nodes are compromised, no secret information is disclosed. We define a new cluster mechanism which improves the key sharing probability between the sensors and their cluster head. All keys involved in this solution are symmetric keys.

The rest of this chapter is organized as follows: In Section 5.2, we describe the HWSN model that is assumed throughout the chapter. In Section 5.3, we provide the full details of the solution. Then, we study the establishment of other types of keys in Section 5.4. We analyze the security and performance of the proposed

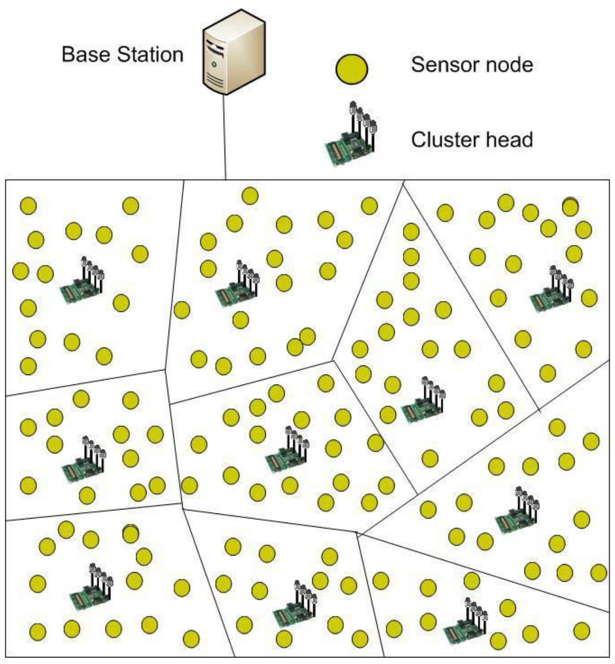


Figure 5.1: The network model of the proposed key predistribution scheme

solution in Sections 5.5 and 5.6 respectively. Finally, we summarize the work of this chapter in Section 5.7.

5.2 Network Model and Design Goal

5.2.1 Network Model

WSNs are application-specific networks. No scheme can be a one-fits-all security scheme. Therefore, it is important to define a network model according to a specific application. In this chapter, we focus on a military HWSN scenario. The entities involve a BS, a small number of cluster head (CH) nodes and a large number of ordinary sensor nodes. All of these entities form a heterogeneous network structure. The root level is the BS, the second level consists of powerful cluster heads, and the bottom lever includes a large number of low-cost sensor nodes. An example of such a network model is shown in Figure 5.1.

1. The BS is an infrastructure and assumed to be secure. Because it is usually located far from the wireless network, in our model, it takes charge of the

network by manipulating cluster heads. The interaction between the BS and low-end sensor nodes is not involved in this solution.

2. CHs are equipped with high powered batteries, large memory storages, strong data processing capabilities and wide radio communication range. More importantly, they are equipped with tamper-resistant hardware. Each CH can communicate with sensors within its cluster, the BS, and the peer cluster heads.

3. The ordinary sensor nodes have limited battery power, small memory space, weak data processing capability, and short radio range. Sensor nodes are mobile but with limited mobility. To reduce the energy consumption, a node communicates only with its CH or neighboring nodes within the same cluster.

We assume that the BS will not be compromised. A CH might be compromised but the adversaries cannot get the secret data stored on it. However, we assume that if an ordinary node is compromised, all the information it holds will be exposed to the attacker. Further, for simplicity, we assume cluster heads and sensor nodes are uniformly and randomly distributed. For scalability, both cluster head and sensor nodes can be added as needed. For security, the compromised cluster heads and nodes can be revoked.

5.2.2 Design Goal

Our goal is to increase security and key connectivity probability of the previous key predistribution schemes with reduced storage overhead at ordinary nodes. In view of the fact that no single keying mechanism is appropriate for all secure communications that are needed in WSNs, we also consider the establishment of different types of keys to secure communication at different levels. The proposed solution will meet the following security or performance requirements:

- The solution supports the establishment of five types of keys, including:

    1. a master key shared between CHs and the BS for sending aggregated data;

2. authentication keys shared by a CH and its cluster nodes for verifying messages shared by them;
3. pairwise keys shared by neighboring nodes within a cluster for intra-cluster node-to-node communication;
4. a cluster key for each cluster for broadcasting within a cluster; and
5. pairwise keys shared by cluster heads for inter-cluster communications.

- Authentication is required for all types of packets, whereas confidentiality may be required only for some sensitive data. Also, authentication of new nodes is also necessary as a new node might be an adversarial one.
- The scheme should reduce the storage overhead of ordinary sensor nodes and not introduce too much communication and computation overhead.
- The scheme should be resilient against node capture attacks. Once a node is compromised, it discloses only the pairwise keys that it stores. The revocation of some ordinary nodes does not affect the whole network operation.

5.3 The Proposed Solution

In this section, we present a key predistribution solution specifically for HWSNs. The essence of the suite of key management is a keyed hash chain based key predistribution scheme. Just like existing key predistribution schemes in [4] and [136], the solution includes three stages: key pool generation, key ring assignment and common key discovery. Here we introduce two definitions which will facilitate the understanding of the scheme.

Definition 5.1. The keyed hash function is the foundation of a keyed hash chain. A keyed hash function H takes a key K and a binary string M of arbitrary length as input, and outputs a binary string of fixed length, which is called hash value h: $h = H(K, M)$. A keyed hash function H satisfies the following two properties:

1. Given a key K and a string M, it is easy to compute h such that $h = H(K, M)$;

2. Given a hash value h and a string M, it is computationally infeasible to find a key K such that $H(K, M) = h$;

Definition 5.2. A keyed hash chain in this chapter can be derived as follows:

1. generating a random key seed value X and generation key K;
2. repeatedly applying the same keyed hash function to produce the hash chain. Assume that the expected length of the key chain C is L, then the j-th key of the key chain C is generated as follows:

$$k_j = \begin{cases} H(K, X), & \text{if } j = 0, \\ H(K, k_{j-1}), & \text{if } j = 1, \ldots, L-1. \end{cases} \tag{5.1}$$

5.3.1 Key Pool Generation

The size of a key pool should be determined before it is generated. The optimum size of the key pool is constrained by the expected probability of key sharing, the expected resilience capability, and the number of keys stored by each node. The number of keys on each node fully depends on the node's memory. We point out that the size of a key pool should maintain an acceptable key sharing probability and a reasonable resilience capability for the network.

Unlike a large key pool that is generated in the random key predistribution scheme [3], a small key pool which stores only generation keys is adopted in our solution. Suppose the expected number of keys is P and the length of each key chain is L. X is a publicly known seed. gk_i ($M = \lceil P/L \rceil$ and $1 \leq i \leq M$) are generation keys. H is a keyed hash function. Each generation key gk_i has unique ID IDG_i. Input the seed X and a generation key gk_i to the hash function H, the j-th key of the key chain C_i is generated as follows:

$$k_{i,j} = \begin{cases} H(gk_i, X), & \text{if } j = 0, \\ H(gk_i, k_{i,j-1}), & \text{if } j = 1, \ldots, L. \end{cases} \tag{5.2}$$

The last factor $k_{i,L}$ is called the commitment of the key chain C_i. We call it $Commit_i$ for simplicity. It is used for authentication and it does not belong to the key pool. Each commitment $Commit_i$ has a unique ID IDC_i. There is a corresponding relation between IDG_i and IDC_i. That is, we refer to the same key chain C_i when we talk about the key chain that corresponds to either IDG_i or IDC_i.

By iterating the above hash algorithm, we generate M key chains as well as the commitment for each key chain. It is assumed that no common keys exist between any two key chains. That is, $C_i \cap C_j = \phi$ for $1 \leq i, j \leq M$ and $i \neq j$. Table 5.1 displays the structure of a keyed hash chain based key pool. In addition, the BS also generates a master key K_M.

Table 5.1: The structure of a hash chain based key pool

Generation key	Generation Key ID	Hash chain	Keys	Commitment	Commitment ID
gk_1	IDG_1	C_1	$k_{1,0}, \ldots, k_{1,L-1}$	$k_{1,L}$	IDC_1
gk_1	IDG_2	C_2	$k_{2,0}, \ldots, k_{2,L-1}$	$k_{2,L}$	IDC_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
gk_M	IDG_M	C_M	$k_{M,0}, \ldots, k_{M,L-1}$	$k_{M,L}$	IDC_M

5.3.2 Key Ring Assignment

- Before deployment, each sensor S_i is preloaded with r randomly selected commitments $\{(i_1, Commit_{i_1}), \ldots, (i_r, Commit_{i_r})\}$. We denote the set of IDs of $\{i_1, \ldots, i_r\}$ as $[IDs]_{S_i}$. In addition, each node S_i is preloaded with an authentication key $K_{M,S_i} = H(K_M, ID_{S_i})$.

- Each CH_a is preloaded with T ($T \gg r$) randomly selected generation keys together the corresponding commitments $\{(a_1, gk_{a_1}, Commit_{a_1}), \ldots, (a_T, gk_{a_T}, Commit_{a_T})\}$ where $a_1, \ldots, a_T$ are IDs of commitments. We denote the set of IDs $(a_1, \ldots, a_T)$ as $[IDs]_{CH_a}$. Note that we refer to the same key chain C_i when we talk about the key chain that corresponds to either IDG_i or IDC_i, CH_a stores only IDs of commitments rather than IDs of both commitments and generation keys. In addition, each CH_a is preloaded with the master key K_M.

After the key ring assignment stage, both cluster heads and sensor nodes are uniformly and randomly deployed into the network.

5.3.3 Common Key Discovery

- *Cluster Formation*. Each cluster head CH_a broadcasts a Hello message to its neighboring nodes with a random delay in order to avoid the collision of the Hello message from neighboring CHs. The Hello message is $(ID_{CH_a}, [IDs]_{CH_a})$ where $[IDs]_{CH_a}$ is the set of IDs of the generation keys stored in CH_a. A node within the network can receive several Hello messages because of large communication range of CHs. Suppose a node S_i receives three Hello messages from CH_a, CH_b, and CH_c respectively. S_i makes a comparison between the intersections $[IDs]_{CH_a} \cap [IDs]_{S_i}$, $[IDs]_{CH_b} \cap [IDs]_{S_i}$, and $[IDs]_{CH_c} \cap [IDs]_{S_i}$. If $|[IDs]_{CH_b} \cap [IDs]_{S_i}| \geq |[IDs]_{CH_c} \cap [IDs]_{S_i}| > |[IDs]_{CH_a} \cap [IDs]_{S_i}|$, S_i will choose the CH_b as its cluster head. S_i also record CH_b as the backup cluster head. Other nodes also perform as the node S_i.

  In our scheme, a HWSN is partitioned into many distinct clusters after the deployment. Each cluster has a cluster head and a set of sensor nodes. To reduce the energy consumption and the redundant traffic loads in a network, sensor nodes communicate only with their cluster head and neighboring nodes. We assume that two nodes have no interaction if they are located in two clusters.

- *Neighborhood Discovery*. Each sensor node S_i broadcasts a Hello message within a short range. Suppose CH_a is the cluster head of the node S_i, the Hello message is $(ID_{S_i}, ID_{CH_a}, [IDs]_{S_i})$ where $[IDs]_{S_i}$ is the set of IDs of commitments. Suppose one of its neighboring node S_j receives the message, S_j first checks whether or not S_i belongs to the same cluster. If it does not belong to the same cluster, S_j will discard the message. Otherwise, S_j further compares $[IDs]_{S_i}$ and $[IDs]_{S_j}$. If $[IDs]_{S_i} \cap [IDs]_{S_j} = \phi$, S_j just acknowledges with a *Hello reply* message. Otherwise, S_j will reply with $[IDs]_{S_i} \cap [IDs]_{S_j}$. S_j adds S_i together with $[IDs]_{S_i} \cap [IDs]_{S_j}$ to its *neighboring nodes list*. After node S_i receives the reply message from neighboring node S_j, S_i will add S_j together with $[IDs]_{S_i} \cap [IDs]_{S_j}$ to its *neighboring nodes list*. Here we suppose *neighboring nodes list* is a list of two-tuples. Suppose S_i neighboring nodes are S_a, S_b andS_c. S_i will store $\{S_a, [IDs]_{S_i} \cap [IDs]_{S_a}\} \parallel \{S_b, [IDs]_{S_i} \cap [IDs]_{S_b}\} \parallel \{S_c, [IDs]_{S_i} \cap [IDs]_{S_c}\}$.

This process continues until all the sensor nodes have obtained the neighborhood information.

The procedures are as follows:

1. $S_i \Rightarrow * : ID_{S_i}, ID_{CH_a}, [IDs]_{S_i}$;
2. if $CH_a \neq CH_b$, S_j discards the message;
   if $CH_a = CH_b$ & $[IDs]_{S_i} \cap [IDs]_{S_j} = \phi$,
   $S_j \rightarrow S_i$: ID_{S_j};
   if $CH_a = CH_b$ & $[IDs]_{S_i} \cap [IDs]_{S_j} \neq \phi$,
   $S_j \rightarrow S_i$: ID_{S_j}, $[IDs]_{S_i} \cap [IDs]_{S_j}$;
   S_j adds ID_{S_i} and $[IDs]_{S_i} \cap [IDs]_{S_j}$ to its *neighboring nodes list*;
3. S_i adds ID_{S_j} and $[IDs]_{S_i} \cap [IDs]_{S_j}$ to its *neighboring nodes list*;

Here, $\Rightarrow$ denotes broadcast and $\rightarrow$ denotes unicast. No authentication mechanism is involved in this stage. It is a general assumption that adversaries do not launch active and explicit pinpoint attacks on the nodes during deployment and initialization which usually does not last too long.

- *Shared Pairwise Key Establishment*. The node S_i sends messages to its cluster head CH_a. The message contains the node's ID ID_{S_i}, nonce, *neighboring nodes list*, and MAC on all these values. Suppose S_j is a neighboring node of S_i, the cluster head CH_a determines the pairwise key for S_i and S_j by generating a random number k, where $0 \leq k \leq L-1$. k is used as an index in the key chain for selecting the pairwise key. After that CH_a disseminates the shared key information to S_i and S_j. In our solution, all the common keys contribute to the shared pairwise key. The shared-key information consists of the following parts: (1) IDs of neighboring nodes (S_i and S_j), (2) the result of $\oplus$ on all the shared generation keys and commitments. If $[IDs]_{S_i} \cap [IDs]_{S_j} = \phi$, CH_a will generate a generation key $gk_{i,j}$ for them. If $[IDs]_{S_i} \cap [IDs]_{S_j} = \{l\}$, CH_a will send $gk_l \oplus Commit_l$ to S_i and S_j. If $[IDs]_{S_i} \cap [IDs]_{S_j} = \{m, n\} (1 \leq m, n \leq M$ and $m \neq n)$, CH_a will send $gk_m \oplus Commit_m \oplus gk_n \oplus Commit_n$ to S_i and S_j. For the conditions where two neighboring nodes share more than two commitments, CH_a does the same operation. (3) k, (4) nonce, and (5) MAC which is calculated on all these values using corresponding authentication keys K_{M,S_i} and K_{M,S_j}.

The procedures are as follows:

1. $S_i \Rightarrow CH_a$: ID_{S_i}, nonce, *neighboring nodes list*, $MAC_{K_{M,S_i}}$;
2. If $[IDs]_{S_i} \cap [IDs]_{S_j} = \phi$,
   $CH_a \rightarrow S_i$: $ID_{S_i}, ID_{S_j}, E_{K_{M,S_i}}(gk_{i,j})$, n, nonce, $MAC_{K_{M,S_i}}$;
   $CH_a \rightarrow S_j$: $ID_{S_i}, ID_{S_j}, E_{K_{M,S_j}}(gk_{i,j})$, k, nonce, $MAC_{K_{M,S_j}}$;
   If $[IDs]_{S_i} \cap [IDs]_{S_j} = \{l\}$,
   $CH_a \rightarrow S_i$: ID_{S_i}, ID_{S_j}, $gk_l \oplus Commit_l$, k, nonce, $MAC_{K_{M,S_i}}$;
   $CH_a \rightarrow S_j$: ID_{S_i}, ID_{S_j}, $gk_l \oplus Commit_l$, k, nonce,$MAC_{K_{M,S_j}}$;
   If $[IDs]_{S_i} \cap [IDs]_{S_j} = \{m, n\}$,
   $CH_a \rightarrow S_i$: ID_{S_i}, ID_{S_j}, $gk_m \oplus Commit_m \oplus gk_n \oplus Commit_n$, n, nonce, $MAC_{K_{M,S_i}}$;
   $CH_a \rightarrow S_j$: ID_{S_i}, ID_{S_j}, $gk_m \oplus Commit_m \oplus gk_n \oplus Commit_n$, n, nonce, $MAC_{K_{M,S_j}}$;

CH_a iterates the above procedures to determine the shared pairwise keys for S_i with all the nodes in S_i's *neighboring nodes list*.

After S_i receives the message from CH_a, S_i will proceed as follows:

1. S_i authenticates the message by verifying the message with its authentication key K_{M,S_i}. If the verification fails, S_i will discard the message. Otherwise, S_i calculates $[IDs]_{S_i} \cap [IDs]_{S_j}$.
2. If $[IDs]_{S_i} \cap [IDs]_{S_j} = \phi$, S_i computes the key $H^k(gk_{i,j}, X)$ shared by it and S_j.
   If $[IDs]_{S_i} \cap [IDs]_{S_j} = \{l\}$, S_i first computes the generation key $gk_l = Commit_l \oplus gk_l \oplus Commit_l$. Then S_i computes $H^k(gk_l, X)$ shared by it and S_j.
   If $[IDs]_{S_i} \cap [IDs]_{S_j} = \{m, n\}$, S_i first computes the generation key $gk_m \oplus gk_n = Commit_m \oplus Commit_n \oplus gk_m \oplus Commit_m \oplus gk_n \oplus Commit_n$. Then S_i computes $H^k(gk_m \oplus gk_n, X)$ shared by it and S_j. For the conditions where S_i and S_j share more than two commitments, the key recovery operations are the same.

- *Special Condition*. There is a special condition where two nodes share a commitment, while their CH does not have the corresponding generation key.

For this condition, CH will turn for help to its neighboring cluster heads. The neighboring CH which has the corresponding generation key will encrypt and send it to the CH.

5.4 Establishment of Other Types of Keys

As discussed in previous chapters, no single keying mechanism can satisfy all types of secure communications in HWSNs. Therefore, we explore ways to establish other types of keys for the networks. From the aforementioned description, we know that CHs share the master key K_M with BS. Each node S_i shares a pairwise key with its cluster head $K_{M,S_i} = H(K_M, ID_{S_i})$. Also we have discussed how to establish pairwise keys between neighboring nodes within a cluster. In real applications, a cluster key CK shared by all nodes in a cluster is needed in order to broadcast a message within a cluster, and a network key NK which is shared by all the nodes in the network is also needed in order to share information throughout the network. Furthermore, the communication between CHs is also necessary. We now discuss the establishment of these keys in turn.

5.4.1 Establishment of Cluster Key within a Cluster

A specific cluster key for each cluster can facilitate secure broadcast within the cluster. In our solution, the cluster key CK_a is generated by CH_a. After its generation, CH_a will encrypt with the shared key between itself and each node. Finally, cluster head CH_a sends the following message to a cluster member S_i:

$$CH_a \rightarrow S_i : E_{K_{M,S_i}}(CK_a),$$

where K_{M,S_i} is the shared key between CH_a and node S_i.

The cluster key must be updated with each membership change. The updating of the session key is similar to the establishment of a new session key. CH_a generates a new session key and encrypts it with a key shared between itself and legal nodes. Finally, CH_a sends the encrypted session key to legal nodes in the cluster.

In practical applications, a node may broadcast messages to its neighboring nodes. We do not explore how to share a key between neighboring nodes as the aforementioned broadcasting happens infrequently. If a node wants to share a message with its neighboring nodes, it encrypts the message with the pairwise keys and then links up the encrypted messages with the IDs of the expected nodes. When a node receives the message, it checks whether its ID is included in the message. If so, it decrypts the corresponding part. Otherwise, it discards the message.

5.4.2 Establishment of Network Key within the Whole Network

The establishment of Network key NK is much easier than the establishment of other types of keys. The BS picks up a Network key NK and sends it to each cluster head. The cluster head encrypts the NK with its cluster key and broadcasts it to its cluster nodes. This level-by-level process distributes the cost of transmitting the Network key. This routine can be followed for the renewal of the Network key as well. However, membership change happens from time to time, and as a result, the Network key has to be renewed frequently. A better option is to remove the Network key if network-wide broadcasting does not occur frequently. When the BS wants to share information with all the nodes in the network, it broadcasts the message to cluster heads. Cluster heads then broadcasts messages within their clusters. In this way, the workload of Network key establishment and renewal is removed.

5.4.3 Establishment of Pairwise Keys between Clusters

In order to maintain secure communication between clusters, each cluster head has to agree on pairwise keys with other cluster heads. The pairwise key shared by two cluster heads CH_a and CH_b is generated as follows:

$$K_{CH_a,CH_b} = H(K_M, ID_{CH_a} \parallel ID_{CH_b}),$$

where H is a hash function, K_M is the master key, and $\parallel$ is the operation of concatenation. The pairwise keys established in this way guarantee that sensor nodes cannot decrypt messages shared by cluster heads. However, the communication between two cluster heads can be seen by other cluster heads.

So far, we have discussed the establishment of five types of keys. Please refer to Table 5.2 for details.

Table 5.2: Five types of keys in the proposed solution

Key	Entities	Function
Master key K_M	BS and CHs	Maintain secure communication between BS and CHs
Authentication key K_{M,S_i}	CH and S_i	Authenticate messages shared by CH and S_i
pairwise key K_{S_i,S_j}	S_i and S_j	Maintain secure communication between S_i and S_j
cluster key CK_a	CH_a and cluster nodes	Maintain secure broadcasts within the cluster
pairwise key K_{CH_a,CH_b}	CH_a and CH_b	Maintain secure communication between CH_a and CH_b

5.5 Security Evaluation

In this section, we discuss several security issues related HWSNs, including node revocation, addition of new nodes, and resilience against node capture attack. We demonstrate that our scheme satisfies these security requirements.

5.5.1 Node Revocation

In existing keyed hash chain based schemes [4, 136], the compromise of a node will result in the disclosure of all the key chains corresponding to generation keys which it stores. While in our scheme, a node is only preloaded with commitments rather than the specific generation keys or key chains. In the procedure of *shared pairwise key establishment*, all the common keys contribute to the shared pairwise key.

1. If $[IDs]_{S_i} \cap [IDs]_{S_j} = \phi$ or the set $[IDs]_{S_i} \cap [IDs]_{S_j}$ has at least two elements, the shared pairwise key is not in the key pool. In this condition, if either of nodes S_i and S_j is compromised, neither their shared key chains nor any other network-wide secret information is disclosed. Therefore, only neighboring nodes delete the pairwise keys they share with the compromised node once a sensor is compromised by an adversary. The cluster head will

send a revocation message to neighboring nodes of the compromised node. The message is authenticated by the key shared between the cluster head and its nodes. The cluster head will add the ID of the compromised node to the *Revoked Node List*.

2. If the set $[IDs]_{S_i} \cap [IDs]_{S_j} = \{l\}$ has only one element, the shared pairwise key is the k-th key of the key chain C_l. In this condition, if either of nodes S_i and S_j is compromised, the keys $k_{l,k}, \ldots, k_{l,L-1}$ of the key chain C_l will be disclosed. Even though this condition takes up very little proportion and the damage of a compromised node has been greatly reduced in the proposed scheme, it does produce damages in the highly sensitive environments. We can restrict the procedure of *shared pairwise key establishment*: if the set $[IDs]_{S_i} \cap [IDs]_{S_j} = \{l\}$ has only one element, their cluster head CH_a sends $gk_l \oplus Commit_l \oplus nonce_l$ rather than $gk_l \oplus Commit_l$, where $nonce_l$ is a nonce which ensures that the shared pairwise key of S_i and S_j is not a specific key in the key chain C_l.

5.5.2 Addition of New Nodes

The addition of new nodes is necessary for key management schemes in order to realize scalability property or substitute compromised nodes. However, it poses a challenge to security schemes because a newly deployed sensor could have been compromised. Therefore, an efficient way to authenticate new nodes is of great importance. Recall that cluster heads have the master key K_M, each new sensor S_{new} is preloaded with a authentication key which is $K_{M,S_{new}} = H(K_M, ID_{S_{new}})$. After the node S_{new} is deployed in the network, S_{new} sends join request to a cluster head CH_a. The request is given as:

$$S_{new} \rightarrow CH_a : ID_{S_{new}} \parallel nonce \parallel MAC_{K_{M,S_{new}}}(ID_{S_{new}}, nonce). \qquad (5.3)$$

CH_a firstly checks whether $ID_{S_{new}}$ is included in the *Revoked Node List*. If $ID_{S_{new}}$ is included in the *Revoked Node List*, CH_a discards the message. Otherwise, CH_a generates $K_{M,S_{new}}$ and then authenticates the node S_{new} by verifying the MAC. After that, S_{new} determines its neighboring nodes as described in the stage of Neighborhood Discovery. We do not discuss the attacks launched by a compromised node which has not been detected by the cluster head.

5.5.3 Resilience against Node Capture Attack

Because of the open nature of wireless communication, an adversary not only can eavesdrop on all traffic but also can inject packets or replay messages. Also, an adversary may inject new nodes into the network. In our scheme, all the sensitive data is authenticated by MAC or encrypted by corresponding keys. When a new node is added into the network, the first step is to check its validity. This authentication process prevent the captured node from rejoining the network.

It is infeasible to set up tamper-resistance hardware for each sensor node. Once node capture attack is launched, all the information the compromised node holds will be known to the attacker. Therefore, it is required that key predistribution schemes are resilient against node compromise attack. Usually, the resilience is measured by the expected number of compromised links due to key revealing of captured nodes [4].

In the scheme [4], there are M key chains in total. Each sensor node is preloaded with r key chains. The probability that a given key does not belongs to a sensor node is $1 - \frac{r}{M}$. If there are n compromised nodes, the probability that a given key is not compromised is $(1 - \frac{r}{M})^n$. The probability of total number of compromised keys is as follows:

$$\bar{p} = 1 - (1 - \frac{r}{M})^n. \tag{5.4}$$

In our scheme, a node stores only commitments rather than generation keys. Once a node is compromised, only commitments and the pairwise keys it shares with cluster head and its neighboring nodes are disclosed. Because of the one-way property of the hash function, the adversary cannot obtain any key on the key chain from the commitments. We discussed two conditions in subsection *Node Revocation*. In the second condition, if two nodes S_i and S_j have only one commitment $Commit_l$ in common, the shared pairwise key is a specific key of the key chain C_l. The subsequent keys of the key chain C_l can be calculated with the shared pairwise key. In the worst condition, the shared pairwise key is the first key of the key chain C_l. The compromise of either S_i or S_j will result in the disclosure of the whole key chain C_l. Hence, in the proposed solution, the probability of total number of compromised keys does not depend on the number of preloaded commitments. Instead, it depends on the probability that a node has only one common commitment with its neighbors. We assume that each node has d neighbors and the probability that a node has only one common commitment with its neighbors is μ, the probability that a given key is not disclosed by a compromised sensor node is $1 - \frac{\mu * d}{M}$. If there are n compromised nodes, the probability that a given key is not compromised is $(1 - \frac{\mu * d}{M})^n$. The probability of total number of compromised keys is as follows:

$$\bar{p} = 1 - (1 - \frac{\mu * d}{M})^n. \tag{5.5}$$

5.6 Performance Evaluation

The storage overhead of our scheme is the same as that of [4]. The major difference is that nodes store commitments rather than generation keys. In this section, we show that the proposed scheme can significantly reduce the storage load, while providing a higher key sharing probability among nodes than that of [4].

The memory of a sensor is fixed after the sensor is produced. The largest number r of commitments stored on each sensor is constrained by the memory of the sensor. Once r is set, the larger of key pool size P is, the smaller probability that two nodes share a key; the smaller the key pool size K is, the larger is the impact on other sensors' communication when a fixed number of sensor is compromised. We want to find the largest key pool size where the probability of a node and its cluster head sharing a key chain is no less than a threshold p.

Suppose the number of key chains is M. A sensor node has $C(M, r)$ different ways of selecting r commitments from a key pool with the size P.

$$C(M, r) = \frac{M!}{r!(M - r)!}. \tag{5.6}$$

A cluster head has $C(M, T)$ different ways of selecting T generation keys from a key pool with the size P.

$$C(M,T) = \frac{M!}{T!(M-T)!}. \tag{5.7}$$

Therefore, the total number of ways for a sensor node to pick up r commitments and a cluster head to pick up T generation keys respectively is $C(M,r)C(M,T)$.

$$C(M,r)C(M,T) = \frac{M!}{r!(M-r)!} \times \frac{M!}{T!(M-T)!}. \tag{5.8}$$

In [4], the probability that a sensor node and a cluster head node share a common key can be given as follows:

$$\begin{aligned} p_{[4]} &= 1 - Pr[\text{a node and a cluster head node do not share any key}] \\ &= 1 - \frac{(M-r)!(M-T)!}{M!(M-r-T)!}. \end{aligned} \tag{5.9}$$

In our scheme, we use a new cluster mechanism which can increase the probability of key sharing between sensor nodes and their clusters. In order to simplify the analysis, we assume that each node can receive broadcast messages from *three* cluster heads. The node chooses a cluster head with whom it shares the largest number of commitments as its cluster head. Therefore, the probability that a sensor node and its cluster head share a common key can be given as follows:

$$\begin{aligned} p_{our} &= 1 - Pr[\text{a node does not share any key with three cluster heads}] \\ &= 1 - (\frac{C(M,r)C(M-r,T)}{C(M,r)C(M,T)})^3. \end{aligned} \tag{5.10}$$

In order to highlight the advantage of our solution, we also repeat the probability of sharing at least one key between two nodes in basic schemes [3, 5].

$$p_{[3]} = 1 - \frac{C(P-m,m)}{C(P,m)}, \tag{5.11}$$

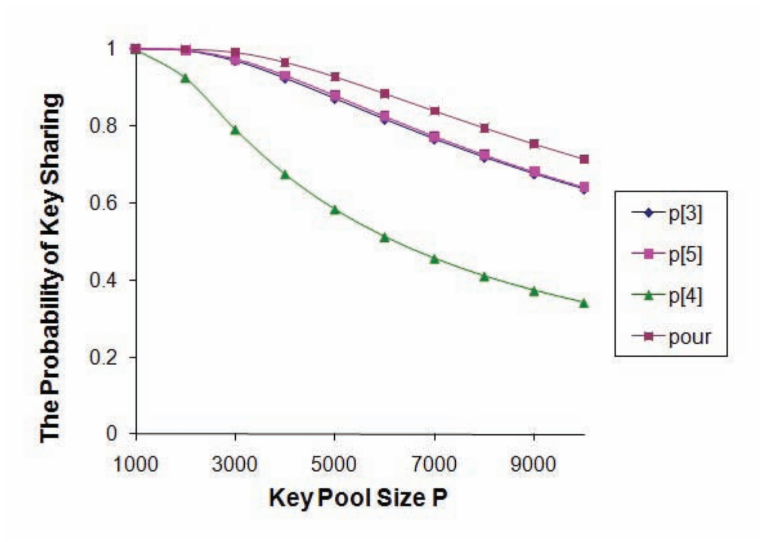


Figure 5.2: Comparison of key sharing probability for different key pool sizes in schemes [3, 4, 5] and the proposed solution

where m is the number of keys stored by each sensor node.

$$p_{[5]} = 1 - \frac{(P-l)!(P-L)!}{P!(P-l-L)!}, \tag{5.12}$$

where l is the number of keys picked by a low-end sensor node and L is number of keys picked by a high-end sensor node.

We derive the probability of sharing at least one key for schemes [3, 4, 5] and our scheme in Figure 5.2. In Figure 5.2, we assume the key pool size in schemes [3] and [5] ranges from 1000 to 10000, with an incremental of 1000. Suppose the length of each key chain is 10, the key pool size in scheme [4] and our scheme ranges from 100 to 1000, with an incremental of 100. The corresponding parameters $[T, r, L, l, m]$ are $[80, 5, 500, 20, 100]$. The relationship between these parameters is $L \times l = m^2$, $L > T$, and $l > r$. From Figure 5.2, we can see that the proposed scheme achieves a higher probability of key sharing while maintains low storage overhead.

It is easy to reach the conclusion that the probability of key sharing between a node and its cluster head increases with the number of commitments in sensor nodes. In Figure 5.3, we show our scheme always has a higher key sharing prob-

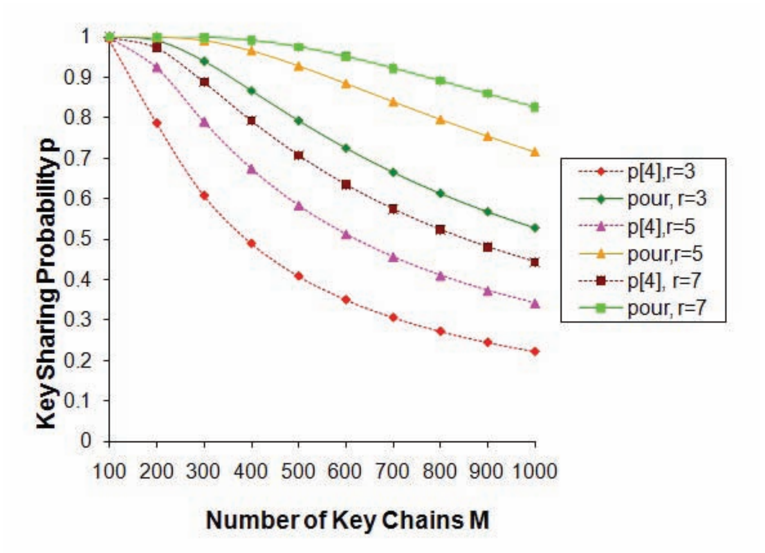


Figure 5.3: The relation between the key sharing probability and sensors' storage overhead in [4] and the proposed solution

ability than scheme [4] for different key pool sizes and different number of keys stored in sensors.

For keyed hash chain based random key predistribution scheme, one of the main problems is how to determine the length of the key chain. Once the key pool size is fixed, the longer the key chain, the smaller number is the number of key chains. A small number of key chains produces less storage overhead. However, in terms of security, a long key chain may weaken the security of the scheme. How to obtain the best key chain length which maintains an acceptable security level and storage overhead is an open problem.

5.7 Conclusion

We have presented a key management solution for hierarchical WSNs in this chapter. The essence of the key management scheme is a keyed hash chain based key predistribution scheme. The solution has the following properties:

1. It supports the establishment and renewal of five types of keys in the network because a single key cannot satisfy different communication requirements.

2. Each sensor node stores only several commitments instead of generation keys. On the one hand, it reduces storage overhead; on the other hand, the network is still secure even though a large number sensor nodes are compromised. However, in the previous schemes, once a node is compromised, all the key chains corresponding to the generation keys stored on the node will be disclosed.

3. A new cluster mechanism is used to increase the key sharing probability between the sensors and their cluster head. This mechanism also facilitates the management of clusters. Hence, the proposed solution is more suitable for HWSNs than existing schemes.

6

Fully Anti-collusive Hash Chain Based Self-healing Key Distribution

6.1 Introduction

Group communication is being widely used for various applications ranging from healthcare to warfare. The life of a group communication network is usually partitioned into short time-periods called ‘sessions’. One or several group managers generate and distribute a session key to group users at initialization stage. All data broadcasted within the group should be encrypted with the session key so that only authorized users with session key can access the secret messages. In order to maintain secure communication, the session key must be updated with sessions with each membership change. Therefore, key distribution as well as subsequent key updating are cornerstones of secure dynamic group communication. Existing key distribution schemes can be categorized as either key distribution over reliable channels and or key distribution over unreliable channels. Key distribution over reliable channels guarantees that data will be delivered to its destination punctually and therefore requires a strong infrastructure such as a wired network and substantial bandwidth. Key distribution messages can be delayed when they are delivered through unreliable channels such as wireless ad hoc networks or WSNs. In particular condition, some messages might never reach authorized users. Individual interactions between members and the group manager, such as requests and re-transmission, in a large group will incur much communication overhead and place a heavy burden on the group manager who might already be heavily burdened. In particular, many digital content and multi-media distribution systems are based on a uni-directional broadcast distribution channel, such as satellite or cable [137]. It is impossible to request or re-transmit via this communication channel. Therefore, key distribution via unreliable channel produces more constraints and challenges.

Key distribution over a reliable channel has attracted much research interest and many efficient schemes have been proposed [29, 138, 139]. However, these schemes are more appropriate for wired networks than wireless networks. Staddon *et al.* [80] in 2002 coined the term 'self-healing key distribution' which addresses the problem of how to distribute session keys over unreliable channels. Since then, self-healing key distribution has been a hot research topic.

As discussed in Chapter 2, hash chain based self-healing key distribution schemes [6, 7, 86, 92, 93, 94, 95] are more efficient than their counterparts which are based on other cryptographic primitives. Both forward and backward secrecy can be assured by dual directional hash chains. The collusion between users whose life cycles have finished and the newly joined users can be resisted by assigning XOR random number to session key recovery. However, how to resist collusion of revoked users whose life cycles have not finished and the newly joined users is an unsolved problem. Table 6.1 made an all-around comparison between the existing hash chain based self-healing key distribution schemes. In this table, $\surd$ represents that the scheme has the corresponding property and $\times$ represents that the scheme does not have the corresponding property. t wise represents that the scheme can revoke at most t users. 'Partial' represents that the schemes [6, 7, 93] can only resist the collusion of the users whose life cycles have finished and newly joined users.

Du *et al.* in [116] pointed out that the scheme [93] is invalid for resisting collusion of the newly joined users and revoked users whose lifetimes have not expired. After verification, we found that the schemes [6, 7] cannot resist such collusion attack, either. In the same paper, Du *et al.* proposed a new self-healing mechanism which aims to totally anti-collude in hash chain based self-healing key distribution schemes. However, we find Du's scheme cannot resist collusion of the newly joined users and revoked users whose life cycles have expired, to say nothing of collusion

Table 6.1: Performance comparison of hash chain based self-healing key distribution schemes

Scheme	Performance Measure		
	Forward and backward secrecy	Resisting collusion	number of revoked users
[85]	both, t-wise	t-wise	$\times$
[94]	both	$\times$	$\times$
[86]	both t-wise	$\times$	at most t
[92]	both	$\times$	according to the monotone decreasing structures
[95]	both	$\times$	$\times$
[93]	both	Partial	according to the monotone decreasing structures
[6]	both t-wise	Partial t-wise	at most t
[7]	both	Partial	according to the monotone decreasing structures

resistance of the newly joined users and revoked users whose life cycles have not expired.

The rest of this chapter is organized as follows: In Section 6.2, we summary the contribution of the proposed solution. In Section 6.3, we introduce evaluation indicators of self-healing key distribution schemes. In Section 6.4, we analyze the breach of collusion attack against hash chain based self-healing key distribution schemes and proposed an unified attack approach. In Section 6.5, we launch two attacks against Du *et al.*'s scheme with the proposed unified attack approach. In Section 6.6, we discuss the feasibility of the two possible solutions for collusion attack. In Section 6.7, we present a classification of attacks and define the security model. In Section 6.8, we describe a specific construction according to the security model defined in the previous section. In Section 6.9, we provide a proof of security of our proposed scheme under an appropriate framework and the defined model. In Section 6.10, we focus on performance analysis and compare several hash chain based schemes. We conclude the chapter in last section.

6.2 The Contribution of the Proposed Scheme

The contribution of this chapter is three-fold.

- We propose a unified approach of collusion attack against hash chain based self-healing key distribution schemes. And then we show the approach can be used to launch two attacks against the scheme in [116] which claimed that it can resist the collusion of revoked users, whose life cycle has not expired and the newly joined users.

- We propose a fully anti-collusive self-healing key distribution scheme with implicit authentication. Performance evaluation shows that our scheme can resist such collusion attack at an acceptable cost. As far as we know, this is first time that a solution has been proposed to address attacks by the revoked users whose life cycle have not expired and the newly joined users in hash chain based self-healing key distribution schemes.
- Implicit authentication is realized with acceptable storage and computation cost.

6.3 Evaluation Indicators of Self-healing Key Distribution Schemes

As can be seen in the review of related works, besides availability, integrity, confidentiality, authentication and non-reputation traditional security requirements, some particular evaluation indicators for self-healing key distribution should be highlighted due to a list of characteristics of WSNs [140]. The performance of a self-healing key distribution scheme can be evaluated by:

1. *Security.* The Security here refers to not only forward secrecy and backward secrecy but also collusion resistance. Vector space-secret-sharing-based schemes [90, 91] satisfy these security requirements. Shamir's secret-sharing-based schemes [80, 81, 87, 89] satisfy t-wise security requirements where t is the degree of the secret polynomial. While existing hash chain based schemes [86, 92, 95] only meet forward and backward secrecy. The improved schemes [6, 7, 93] can resist collusion only between the users whose life cycle have finished and the newly joined users. How to resist the collusion between the revoked uses, whose life cycle have not finished, and the new joined users is a problem to be solved.

2. *Efficiency.* The resource-limited property of sensor nodes must be given fully consideration when we design protocols for WSNs. In key management schemes, the energy required for computation is three orders of magnitude less than that required for communication [141]. In addition, the energy consumed for computation much depends on algorithm and hardware. Hence, the first concern is to reduce communication overhead.

3. *Scalability.* WSNs may scale up to thousands of sensors. Network management is more complex and available bandwidth decreases as the scale increases. Also, revocation should be localized within a short range rather than the whole network. Scalability also implies that operations are finished in a timely manner despite frequent changes to group node topology and density.

6.4 A Unified Approach of Collusion Attack against Hash Chain Based Self-healing Key Distribution Schemes

Hash chain based self-healing key distribution schemes reduce storage, computation, and communication overheads greatly. However, the one-way property of the hash function produces a new security problem. A session key is computed from the forward key and backward key in schemes [86, 92, 95]. If the forward key seed and backward key seed can be obtained, then all the session keys can be worked out. Therefore, the schemes [86, 92, 95] are vulnerable to collusion attack. There is an extreme occasion when the coalition of a user who is authorized only for the first session and the other user who is authorized only for the m-th session can calculate all the session keys $\{SK_2, \ldots, SK_{m-1}\}$ which they were not authorized to do. Tian *et al.* addressed the collusion problem in [93]. Random numbers corresponding to the life cycle are pre-assigned to each user when it joins the network. A session key is computed from three parts: forward key, backward key, and random number. This approach successfully blocks the collusion of the newly joined users and those users whose life cycle has expired. However, this approach cannot resist the collusion of the revoked users whose life cycle has not expired and the newly joined users. Suppose a user U_i is revoked at session t ($l < t < v$) before its life cycle (l, v) finishes.

U_i has random numbers $\{r_{t+1}, \ldots, r_v\}$ and can compute forward keys for sessions $(t+1, v)$. If a user U_i colludes with another user U_j who joins the network after session v, it can get backward keys for session $(t+1, v)$ and furthermore computes session keys $\{SK_{t+1}, \ldots, SK_v\}$. A similar attack can be launched against [6, 7].

From the collusion attack against the schemes in [6, 7, 93], we arrive at the conclusion that the forward key chain and the backward key chain themselves are not sufficient to resist collusion attack even though some masking mechanism are used to protect the backward hash key chain. The breach of collusion attack lies in the masking mechanism. As long as the masking mechanism is broken, the computation of the backward key chain does not relate to users' state.

6.5 Two Attacks with the Unified Approach

In this subsection, we consider two attacks against the scheme with the unified attack approach. One attack is launched by a revoked user itself whose life cycle has not expired, and the other attack is launched by the collusion of a revoked user, whose life cycle has finished, and a newly joined user. In order to facilitate understanding, we briefly introduce the approach in [116] (Please refer to [116] for detail). Du *et al.*'s scheme is as follows:

Setup. The group manager (GM) chooses m t-degree ($t < m$) polynomials $h_1(x)$, …, $h_m(x) \in F_q[x]$ and generates m random numbers $r_1, \ldots, r_m \in F_q$. Each user U_i whose life cycle is (l, v) receives the personal secret including $\{h_l(i), \ldots, h_v(i)\}$, $\{r_l, \ldots, r_v\}$, forward key K_l^F and backward key K_{m-v+1}^B over a secure channel. For session $1 \leq j \leq m$, the session key is computed as $SK_j = (K_j^F + K_{m-j+1}^B)c_j$ where K_j^F is the j-th forward key in the forward key chain, K_{m-j+1}^B is the j-th backward key in the backward key chain, and c_j is a random number corresponding to the session j.

Broadcast. In the $j-$th session, the GM chooses a random number c_j from a finite field F_q where q is a large prime. Then the GM computes the broadcast polynomial $W_j(x) = A_j(x)C_j + h_j(x)$, where $A_j(x)$ is the revocation polynomial which is computed from the revoked user set R_j. Thirdly, the GM computes the set

$C_j = \{c_j r_1(c_1 + c_2), c_j r_2(c_2 + c_3), \ldots, c_j r_{j-2}(c_{j-2} + c_{j-1}), c_j r_{j-1} c_{j-1}\}$. Finally, the GM broadcasts the message $B_j = \{W_j(x), C_j, R_j\}$.

Key Recovery. Any authorized user U_i receives the broadcast message B_j can recover $c_j = (W_j(i) - h_j(i))/A_j(i)$. Then, U_i computes K_j^F by hashing K_l^F and K_{m-j+1}^B by hashing K_{m-v+1}^B with hash function H. Finally, U_i gets the $j-$th session key $SK_j = (K_j^F + K_{m-j+1}^B)c_j$.

Self-healing. Suppose a user U_i whose life cycle is (l, v) receives broadcast message B_{j_1} in session j_1 but misses broadcast message B_j in session j, where $1 \le l < j < j_1 \le v \le m$. U_i can recover the lost session SK_j as follows:

1. U_i gets K_j^F by hashing K_l^F and K_{m-j+1}^B by hashing K_{m-v+1}^B with hash function H.

2. U_i computes c_{j_1} from $B_{j_1} = \{W_{j_1}(x), C_{j_1}, R_{j_1}\}$ as described in the procedure of *Key Recovery*. By dividing each item of set C_{j_1} by c_{j_1}, U_i gets a new set $C'_{j_1} = \{r_1(c_1 + c_2), r_2(c_2 + c_3), \ldots, r_{j_1-2}(c_{j_1-2} + c_{j_1-1}), r_{j_1-1}c_{j_1-1}\}$.

3. U_i has a private random number set $\{r_l, \ldots, r_v\}$ which includes $\{r_j, \ldots, r_{j_1-1}\}$. u_i divides respectively each item of the set $C''_{j_1} = \{r_j(c_j+c_{j+1}), \ldots, r_{j_1-2}(c_{j_1-2} +c_{j_1-1}), r_{j_1-1}c_{j_1-1}\}$ by corresponding item of the set $\{r_j, \ldots, r_{j_1-1}\}$. U_i works out the set $\{(c_j + c_{j+1}), \ldots, (c_{j_1-2} + c_{j_1-1}), c_{j_1-1}\}$. U_i finally works c_j out by a serial of substraction operation in reverse order. Therefore U_i can recover the lost key $SK_j = (K_j^F + K_{m-j+1}^B)c_j$.

6.5.1 Attack Launched by Revoked Nodes

We introduce an attack launched by a revoked user itself whose life cycle has not expired. Suppose a user U_i whose life cycle is (l, v). It is revoked at session $l < j < v$. If U_i is not removed from the network, it can receive broadcast message $B_j = \{W_j(x), C_j, R_j\}$ at session j among which $C_j = \{c_j r_1(c_1 + c_2), c_j r_2(c_2 + c_3), \ldots, c_j r_{j-2}(c_{j-2}+c_{j-1}), c_j r_{j-1} c_{j-1}\}$. Because U_i is an authorized user at session $j-1$, u_i gets $r_1(c_1+c_2), r_2(c_2+c_3), \ldots, r_{j-3}(c_{j-3}+c_{j-2}), r_{j-2}c_{j-2}\}$. By comparing the corresponding items in the two sets, u_i can get c_j easily. U_i gets forward key K_j^F and backward key K_{m-j+1}^B by hashing K_{j-1}^F and K_{m-j+2}^B respectively. Thus U_i

computes $SK_j = (K_j^F + K_{m-j+1}^B)c_j$. Furthermore, U_i can compute session keys $SK_{j+1}, \ldots, SK_v$ in the same way.

6.5.2 Attack Launched by Collusion

Here we introduce an attack launched by a collusion between a revoked user whose life cycle has expired and a newly joined user. Suppose a user U_i whose life cycle is (l, v) receives broadcast message $B_v = \{W_v(x), C_v, R_v\}$ at session v. U_i can recover c_v from $W_j(x)$ as described in the procedure of *Key Recovery*. By dividing each item in the set $C_v = \{c_v r_1(c_1 + c_2), c_v r_2(c_2 + c_3), \ldots, c_v r_{v-2}(c_{v-2} + c_{v-1}), c_v r_{v-1} c_{v-1}\}$ by c_v, U_i gets $\{r_1(c_1+c_2), r_2(c_2+c_3), \ldots, r_{v-2}(c_{v-2}+c_{v-1}), r_{v-1} c_{v-1}\}$.

Even if U_i's life cycle finishes at session v, U_i can receive broadcast message $B_{v+1} = \{W_{v+1}(x), C_{v+1}, R_{v+1}\}$, among which $C_{v+1} = \{c_{v+1} r_1(c_1+c_2), c_{v+1} r_2(c_2+c_3), \ldots, c_{v+1} r_{v-1}(c_{v-1}+c_v), c_{v+1} r_v c_v\}$. With the information obtained from session v, U_i can work out $c_{v+1} = (c_{v+1} r_1(c_1+c_2))/(r_1(c_1+c_2)) = \ldots = (c_{v+1} r_v c_v)/(r_v c_v)$. U_i can also compute forward key K_{v+1}^F by one hash operation on K_v^F. If u_i colludes with any newly joined user U_j whose life cycle starts after session $v + 2$, they can compute backward key K_{m-v}^B and further compute session key $SK_{v+1} = (K_{v+1}^F + K_{m-v}^B)c_v$ to which they are not authorized.

Here we consider an extreme occasion. Suppose a user U_i is only authorized for the first session and the other user U_j is only authorized for the last session m. U_i can work out $\{c_1, c_2, \ldots, c_{m-1}\}$ and $\{K_1^F, K_2^F, \ldots, K_{m-1}^F\}$ by using its personal key and public broadcast messages $\{B_1, B_2, \ldots, B_{m-1}\}$. U_j can work out $\{K_m^B, K_{m-1}^B, \ldots, K_2^B\}$. U_i and U_j collude can recover session keys $\{SK_2, SK_3, \ldots, SK_{m-1}\}$ which they are not authorized to access. In some commercial or military applications, such disclosure of session keys may lead to disastrous consequences.

6.6 Two Possible Solutions for Collusion Attack

On the one hand, in order to maintain security, a part of the key material should be related to the user's status. Once a user is revoked before its life cycle expires, the

GM should change the broadcast messages so that the revoked user cannot recover the session key from the broadcast message with the personal key. On the other hand, in order to keep the self-healing mechanism working, a non-revoked user can recover the lost session keys from the changed broadcast messages with its initial configuration. We do not consider reconfiguration of the non-revoked users as it is infeasible to configure nodes one by one in a large-scale network. The possibility of retransmission by unicast is also excluded as there is no pairwise keys for the GM and the users to secure the sensitive unicast messages. After ruling this out, we have two possible solutions. One solution is to select a new backward hash chain once a user is revoked before its life cycle has expired, and the other solution is to mask new random number in the corresponding broadcast message. We show in Figure 6.1 that the first possible solution does not work.

Suppose a user a whose life cycle is (t_0, t_2) is revoked at session t_1. The backward hash chain for life cycle (t_0, t_1) is generated by the seed S_0. A new hash chain which is generated by the seed S_1 is used start from session t_1. If there is any user is revoked at session t_3, the GM discards the hash chain generated by S_1 and utilizes a new one generated by the seed S_2. Suppose a user b joins the network between life cycle (t_2, t_3). b is pre-assigned several backward keys generated by S_1. By iterative hash operations on any of the pre-assigned backward key, b obtains all the backward keys for life cycle (t_1, t_2). If a and b collude, they can recover all the session keys for life cycle (t_1, t_2).

6.7 Security Model

6.7.1 A Classification of Attacks

We classify the attacks as outside attacks or inside attacks. The outside attacks refer to the attacks launched by users who are never involved in the communication group. Correspondingly, inside attacks are launched by users who are or will be authorized users of a communication group. Existing self-healing key distribution schemes are supposed to repel outside attacks under the assumption that the attackers cannot acquire a personal key from the compromised or captured node. That is, the outside attackers cannot obtain any information from the compromised

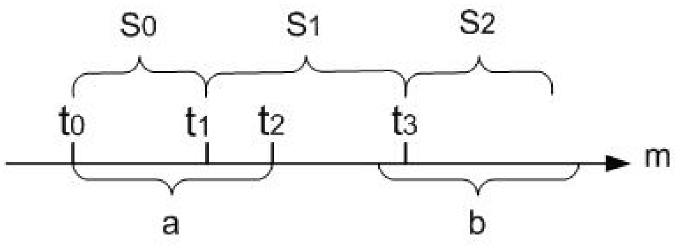


Figure 6.1: An example of re-selecting new hash chain once a user is revoked before its life cycle expires

nodes. This assumption can be achieved by some hardware techniques. Each node is installed with a tamper-resistance component. As to the inside attacks, we consider an attack scenario where an adversary can compromise one or more nodes. Specifically, we consider three different cases with differing degrees of damage.

- In the first scenario, we consider an attack launched by the revoked member subset and by the newly joined member subset separately.
- In the second scenario, we consider an attack is launched by the coalition of users whose life cycles have finished and the newly joined users. This attack incurs more serious damage.
- The third scenario cause the severest damage, the attack is launched by the coalition of users whose life cycles do not expire and the newly joined users.

This solution is designed to have a spectrum of broader impacts: it will

- focus on the fully anti-collusive property besides forward and backward secrecy; and
- achieve implicit authentication.

6.7.2 Definition of Security Model

Let $U = \{U_1, \ldots, U_n\}$ be the finite universe of the network. The communication group is a dynamic subset of U and $R \subset 2^U$ is a monotone decreasing access

Table 6.2: Notations used in the proposed scheme

U_i	the i-th user
n	the total number of users in the network
m	the total number of sessions
$GF(q)$	a field of order q
S_i	the personal secret of user U_i
SK_j	the session key for session j
B_j	broadcast message for session j
R_j	the set of the revoked users before and in session j
J_j	the set of the joined users after session j
G_j	the authorized users in session j
K_i^B	the i-th backward key

structure of subsets of users. Let $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$. The notations in Table 6.2 are used throughout the chapter.

To further clarify our goals and facilitate the later presentation, based on our analysis in Section 6.5, according to but not constrained by the security model of [6, 7, 93], we define the self-healing key distribution scheme from three aspects. *Definition 6.1* defines self-healing key distribution scheme with revocation capability. *Definition 6.2* defines forward and backward secrecy. *Definition 6.3* and *Definition 6.4* defines resisting collusion properties.

Definition 6.1. (Session key distribution with privacy and implicit authentication [6, 7, 93])

1. The scheme is a session key distribution scheme with privacy if

   (a) for any user $U_i \in G_j$, the session key SK_j is efficiently determined from B_j and S_i.

   (b) for any set $R_j \subseteq U$, where $R_j \in R$ and $U_i \notin R_j$, it is computationally infeasible for users in R_j to determine the personal key S_i.

   (c) it is computationally infeasible to compute session key SK_j from either m broadcast $B_1, \ldots, B_m$ alone or n personal keys $S_1, \ldots,\ S_n$.

2. The scheme has R-revocation capability if, given any $R_j \subseteq U$, where $R_j \in R$, the GM can generate a broadcast message B_j, such that for all $U_i \notin R_j$, U_i can *efficiently verify the truth of* B_j and recover the session key SK_j, but the revoked users cannot.

3. The scheme has self-healing capability if the following is true for any j $(1 \leq j_1 < j < j_2 \leq m)$ if each $U_i \in G_{j_1}$, who has not been revoked after session j_1 and before session j_2, can recover all session keys SK_j for $j = j_1, \ldots, j_2$, from broadcasts B_{j_1}, B_{j_2} and its personal key S_i.

Definition 6.2. [6, 7, 93] The scheme guarantees both R-wise forward and backward secrecy if

1. for any set $R_j \subseteq U$, where $R_j \in R$, it is computationally infeasible for the users $U_i \in R_j$ together to obtain any information about SK_j, even with the knowledge of group keys $SK_1, \ldots, SK_{j-1}$.

2. for any set $J_j \subseteq U$, where $J_j \in R$, it is computationally infeasible for the members $U_i \in J_j$ together to obtain any information about SK_j, even with the knowledge of group keys $SK_{j+1}, \ldots, SK_m$ after session j.

Definition 6.3. Let $B \subset R_l$ be a subset of users whose life cycles expire at session r while revoked at session $l(l < r)$. Let $C \subset J_s$ be a subset of users who join the group from session s with $1 \leq l < s \leq m$. That is, B and C are two disjoint subsets. Suppose $B \cup C \subset R$. The coalition $B \cup C$ doses not get any information about session keys SK_j, for any $l \leq j < s$. That is, the scheme is able to resist R-wise collusion of the revoked users whose life cycles have not finished and the newly joined users.

Definition 6.4. Let $B \subset R_r$ be a subset of users whose life cycles have expired before session r and let $C \subset J_s$ be a subset of users who join the group from session s with $r < s$. That is, B and C are two disjoint subset. Suppose $B \cup C \subset R$. The coalition $B \cup C$ does not obtain any information about session keys SK_j, for any $r \leq j < s$. That is, the scheme is able to resist R-wise collusion of the users whose life cycles have finished and the newly joined users.

6.8 The Construction of a Fully Anti-collusive Self-healing Key Distribution Solution

Following the idea of [6, 7], which slightly reduces storage and computation overhead of [93], we present a fully anti-collusive self-healing key distribution scheme with implicit authentication. The self-healing key distribution scheme comprises five procedures, each of which we describe one by one.

In our setting, the communication group is a dynamic subset of U. An unreliable broadcast channel is available, and time is defined by a global clock. The GM sets up and maintains the communication by performing addition and revocation operations. All operations take place in $GF(q)$, where q is a large prime number ($q > n$). Let $H : GF(q) \to GF(q)$ be a cryptographically secure one-way hash function. Let us consider a vector space secret sharing scheme realizing $\Gamma = 2^U - R$ over the set U. For simplicity, suppose that there exists a public map

$$\psi : U \cup \{GM\} \to GF(q)^l \tag{6.1}$$

which defines Γ as a vector space access structure. The use of a specific ψ fixes the properties of the scheme.

Setup. The GM uses a CSPRNG of large enough period to produce a sequence of m random numbers $r_1, \ldots, r_m \in GF(q)^l$. The GM randomly picks an initial backward key seed S^B from $GF(q)$. In the pre-processing time, the GM computes the backward hash chain of length $(m+1)$ by repeatedly applying the one-way hash function H on the backward key seed $K_0^B = S^B$. For $1 \leq i \leq m+1$, the hash sequences are generated as follows:

$$\{K_0^B = S^B, H(K_0^B), \ldots, H^m(K_0^B), H^{m+1}(K_0^B)\}$$

The last hash value $K_{m+1}^B = H^{m+1}(K_0^B)$ is used for implicit authentication. The key for session j is computed as $SK_j = K_{m-j+1}^B + r_j$.

Each user U_i is assigned a pre-arranged life cycle (s, t) where $1 \leq s < t \leq m$.

U_i will be involved in $k = t - s + 1$ number of sessions. The GM randomly chooses vectors $u_1, \dots, u_m \in GF(q)^l$ and computes secret vectors for each user U_i. The user U_i receives its private key S_i from the GM consisting of: (1) k random numbers corresponding to the sessions which the user U_i will participate in; (2) secret vectors $\{u_s \cdot \psi(i), \dots, u_t \cdot \psi(i))\}$; and (3) K^B_{m+1}. The GM sends the private key to user U_i via a secure channel between them, as shown below:

$$S_i = \{r_s, \dots, r_t || u_s \cdot \psi(i), \dots, u_t \cdot \psi(i) || K^B_{m+1}\} \tag{6.2}$$

In our setting, we allow a revoked user to rejoin the group at a later session with a new identity. However, in existing hash chain based self-healing key distribution schemes [86, 92, 95], rejoining is prohibited.

Broadcast. Without loss of generality, we take the broadcast B_j as example. In session $j = 1, \dots, m$, suppose $R_j \in R$ represent the set of revoked users in and before session j. The GM chooses a subset of users $W_j = U \setminus G_j$ with minimum cardinality such that $W_j \cup R_j \in \bar{\Gamma_0}$. The computation of broadcast B_j can be classified two conditions.

1. At least one user is revoked at session j. Suppose a user U_l whose life cycle is (s, t) is revoked at session j ($s < j < t$), the random numbers $\{r_j, r_{j+1}, \dots, r_t\}$ should be changed in order to keep backward secrecy. The GM generates new random numbers $\{r_{j'}, r_{(j+1)'}, \dots, r_{t'}\}$ with CSPRNG and computes $z_j = K^B_{m-j+1} + u_j \cdot \psi(GM) \in GF(q)$ and $z_{j'} = r_{j'} + u_j \cdot \psi(GM) \in GF(q)$ broadcasts the message $B_j = \{z_j \cup z_{j'}\} \cup \{(U_k, u_j \cdot \psi(U_k)) : U_k \in W_j \cup R_j\}$. The GM has to include the broadcast $B_{j'} = z_{j'} \cup \{(U_k, u_j \cdot \psi(U_k)) : U_k \in W_j \cup R_j\}$ to the subsequent broadcasts for self-healing reasons.

2. No user is revoked at session j. The GM computes $z_j = K^B_{m-j+1} + u_j \cdot \psi(GM) \in GF(q)$ and broadcasts the message $B_j = z_j \cup \{(U_k, u_j \cdot \psi(U_k)) : U_k \in W_j \cup R_j\}$. It is unnecessary to include B_j in the subsequent broadcasts.

In order to facilitate understanding, we use the example shown in Figure 6.2. Suppose a network life-time is divided into 10 sessions.

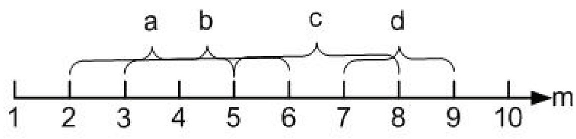


Figure 6.2: An example of different conditions for broadcast

1. Suppose a user a whose life cycle is $(2, 5)$ is revoked at session 4 and the other user b whose life cycle is $(3, 6)$ is revoked at session 5. The GM has to renew random numbers r_4, r_5, r_6 to $r_{4'}, r_{5'}, r_{6'}$. The GM calculates $z_4 = K_7^B + u_4 \cdot \psi(GM)$ and $z_{4'} = r_{4'} + u_4 \cdot \psi(GM)$. Then the GM broadcasts $B_4 = \{z_4 \cup z_{4'}\} \cup \{(U_k, u_4 \cdot \psi(U_k)) : U_k \in W_4 \cup a\}$. In sessions 5 and 6, the GM performs a similar computation and then broadcasts $B_{4'} \cup B_5$ at session 5 and $B_{4'} \cup B_{5'} \cup B_6$ at session 6.

2. No user is revoked at session 8. The GM computes $z_8 = K_3^B + u_8 \cdot \psi(GM)$ and broadcasts the message $B_8 = z_8 \cup \{(U_k, u_8 \cdot \psi(U_k)) : U_k \in W_8 \cup \{a, b\}\}$. Then the GM broadcasts $B_{4'} \cup B_{5'} \cup B_{6'} \cup B_8$. It is unnecessary to include B_7 in the broadcast message for session 8.

Key Recovery. When an authorized user U_i receives the broadcast message B_j at session j, since $U_i \in G_j$ has $\{(U_k, u_j \cdot \psi(U_k))\}_{U_k \in W_j \cup R_j}$ and its personal key, it computes $u_j \cdot \psi(GM)$ using $\{(U_k, u_j \cdot \psi(U_k))\}_{U_k \in W_j \cup R_j \cup \{U_i\}}$ because $W_j \cup R_j \cup \{U_i\} \in \Gamma$. In effect, as long as $W_j \cup R_j \cup \{U_i\} \in \Gamma$, the result $\psi(GM) = \sum_{U_k \in W_j \cup R_j \cup \{U_i\}} \lambda_k \psi(U_k)$ holds for some $\lambda_k \in GF(q)$. Therefore, $u_j \cdot \psi(GM) = \sum_{U_k \in W_j \cup R_j \cup \{U_i\}} \lambda_k u_j \cdot \psi(U_k)$. From the broadcast information B_j, U_i recovers the backward key $K_{m-j+1}^B = z_j - u_j \cdot \psi(GM)$, U_i verifies whether the equation $K_{m+1}^B = H^j(K_{m-j+1}^B)$. If the equation holds, it shows that B_j has not been tampered during transmission. U_i computes the j-th the current session key $SK_j = K_{m-j+1}^B + r_j$. Otherwise, U_i discards B_j.

If the GM renewed the random number r_j to $r_{j'}$, U_i further recovers $r_{j'} = z_j - u_j \cdot \psi(GM)$ after gets K_{m-j+1}^B. Finally, U_i computes the j-th session key $SK_j = K_{m-j+1}^B + r_{j'}$.

Adding/Revoking Users. When a new user wants to join the communication group, the user gets in touch with the GM. The GM assigns a life circle (s, t) and an unused identity $v \in GF(q)$ to the new user. GM computes the personal secret key $S_v = \{r_s, \ldots, r_t || u_s \cdot \psi(v), \ldots, u_t \cdot \psi(v)) || K^B_{m+1}\}$ to this new user via the secure communication channel between them. If any random number of $\{r_s, \ldots, r_t\}$ has been renewed, the GM uses the new random number to replace the old one. The revocation happens when the GM detects a user is compromised or a user's right should be limited in respect to some applications. The GM puts the revoked user's ID to R_j and generates new random numbers for the crossposting sessions and masks the new random numbers before broadcasting them.

Self-healing. Suppose U_i is a user who receives session broadcast messages B_{j_1} and B_{j_2} in session j_1 and j_2 respectively, where $1 \leq j_1 < j_2 \leq m$, but no broadcast message B_j for the session j, where $j_1 < j < j_2$. U_i can still recover all the lost session keys K_j $(j_1 < j < j_2)$ as follows.

1. U_i recovers $K^B_{m-j_2+1}$ from the broadcast message B_{j_2} in session j_2 and repeatedly applies the one-way hash function to it to obtain the backward keys for all j $(j_1 < j < j_2)$.

2. If none of the random numbers $\{r_{j_1+1}, \ldots, r_{j_2-1}\}$ was renewed, U_i recovers all the session keys $SK_j = K^B_{m-j+1} + r_j$, for $j_1 < j < j_2$. Otherwise, U_i has to firstly recover the renewed random numbers from the broadcast message and then compute $SK_j = K^B_{m-j+1} + r_{j'}$.

We use the same example as that described in Figure 6.2.

1. If a user c whose life cycle is $(5, 8)$ receives broadcast message $B_{4'} \cup B_5$ at session 5 and $B_{4'} \cup B_{5'} \cup B_{6'} \cup B_8$ session 8, but misses broadcast messages for session 6 and 7. c recovers backward key K^B_3 from B_8 and further recovers K^B_4 and K^B_5 by iterative hashing. c recovers the random number $r_{6'}$ from $B_{6'}$. c then recovers $SK_6 = K^B_5 + r_{6'}$ and $SK_7 = K^B_4 + r_7$.

2. The other user d whose life cycle is $(7, 9)$ receives broadcast message $B_{4'} \cup B_{5'} \cup B_{6'} \cup B_9$ at session 9, but misses broadcast messages for sessions 7 and 8. d recovers backward key K^B_2 from B_9 and further recovers K^B_3 and K^B_4 by iterative hashing. d then recovers $SK_7 = K^B_4 + r_7$ and $SK_8 = K^B_3 + r_8$.

6.9 Security Evaluation

In this section, we evaluation from two aspects. More specifically, we firstly prove the security of the proposed solution under an appropriate framework and then demonstrate that the proposed solution meets the security requirements of the defined security model.

6.9.1 Proof under an Appropriate Framework

We have discussed three cases of inside attacks in Section 6.5. Correspondingly, we define the attack goal in three ways. In the first scenario, the subset of revoked users and the subset of the newly joined users launch an attack respectively to jeopardize forward and backward secrecy. In the second scenario, the coalition of users whose life cycles have finished and the newly joined members launch an attack in order to obtain the session keys that they are not authorized to access. In the third scenario, the coalition of both revoked users whose life cycle have not expired and the newly joined users launch an attack to acquire all the session keys to which they had no authorization. We prove our scheme is secure against these three cases of inside attacks. More precisely, we can prove *Theorem 6.1* which states forward and backward secrecy, and *Theorem 6.2* which states the capability of collusion resistance of our Construction in an appropriate framework.

Theorem 6.1. Our construction is secure, self-healing session key distribution scheme with privacy and implicit authentication, R-revocation capability with respect to *Definition 6.1*, and achieves R-forward and backward secrecy with respect to *Definition 6.2*.

Proof: We first show that our construction is computationally secure with respect to revoked users under the difficulty of inverting a one-way hash function. That is, our construction satisfies forward secrecy. *i.e.* for any session j, it is computationally infeasible for any set of revoked users from R before and in session j to compute the session key SK_j with non-negligible probability, given the $View$ consisting of personal keys of revoked users, broadcast messages before, in and after session j and session keys before session j. In fact, forward secrecy implies R-revocation capability to some extent.

Consider a coalition of revoked users from R, say $R_j \in R$, who are revoked in and before session j. The users are not authorized to access the j-th session key SK_j. We can model this coalition of users from R as a polynomial-time algorithm $\mathcal{A}'$ that takes $View$ as input and outputs its guess for SK_j. We say that $\mathcal{A}'$ is successful in breaking the construction if it has a non-negligible advantage in determining the session key SK_j. Then using $\mathcal{A}'$, we can construct a polynomial time algorithm $\mathcal{A}$ for inverting one-way function H and have the following claim:

Claim: $\mathcal{A}$ inverts one-way hash function H with non-negligible probability if $\mathcal{A}'$ is successful.

Proof: Given any instance $y = H(x)$, $\mathcal{A}$ generates an instance $View$ for $\mathcal{A}'$ as follows: $\mathcal{A}$ constructs the following backward key chain by repeatedly applying H on y:

$$K_1^B = y, K_2^B = H(y), \ldots, K_m^B = H^{m-1}(y).$$

$\mathcal{A}$ sets the j-th session key $SK_j = K_{m-j+1}^B + r_j$.

$\mathcal{A}$ randomly chooses m vectors $u_1, \ldots, u_m \in GF(q)^l$. Each user $U_i \in U$ receives its personal secret keys $(u_{r_i} \cdot \psi(U_i), \ldots, u_{s_i} \cdot \psi(U_i))$ corresponding to its life cycle (r_i, s_i) from $\mathcal{A}$ via the secure communication channel between them. In this setting, $\Gamma = 2^U \setminus R$ is a monotone increasing access structure of authorized users over U and determined by the family of minimal qualified subsets Γ_0, which is called the basis of Γ. Now $R_j \subseteq R$ implies $R_j \notin \Gamma$.

Let G_j be the set of all non-revoked users in session j. At the j-th session, $\mathcal{A}$ chooses a subset of users $W_j \subset U \setminus G_j$ with minimal cardinality such that $W_j \cup R_j \in \bar{\Gamma}_0$. $\mathcal{A}$ then computes broadcast message B_j for $j = 1, \ldots, m$ as:

$$B_j = z_j \cup \{(U_k, u_j \cdot \psi(U_k)) : U_k \in W_j \cup R_j\} \tag{6.3}$$

where $z_j = K_{m-j+1}^B + u_j \cdot \psi(GM)$. $\mathcal{A}$ also choose $j-1$ random numbers $r_1, \ldots, r_{j-1} \in$

$GF(q)$. Then $\mathcal{A}$ sets $View$ as

$$y = \begin{cases} u_s \cdot \psi(U_k) & \text{for } U_k \in B \cup C \text{ and } 1 \leq s \leq m; \\ B_j & \text{for } j = 1, \ldots, m; \\ r_1, \ldots, r_{j-1}; & \\ SK_1, \ldots, SK_{j-1}. & \end{cases} \tag{6.4}$$

Remark: The $View$ here is different than that in the proof in [86, 92]. The $View$ in [86, 92] do not include the random numbers $r_1, \ldots, r_{j-1} \in GF(q)$.

$\mathcal{A}$ gives $View$ to $\mathcal{A}'$, which in turn selects $X \in GF(q)$ and $n_j \in GF(q)$ randomly, sets the j-th session key to be $SK'_j = X + n_j$ and returns $SK_{j'}$ to $\mathcal{A}$. $\mathcal{A}$ checks whether $SK_{j'} = SK_j$. If not, $\mathcal{A}$ chooses at random $x' \in GF(q)$ and outputs x'.

Note that from $View$, $\mathcal{A}'$ knows $\{u_j \cdot \psi(U_k) : U_k \in W_j \cup \ R_j, 1 \leq j \leq m\}$ and at most $j-1$ random numbers $r_1, \ldots, r_j$ and session keys $SK_1, \ldots, SK_{j-1}$. Consequently $\mathcal{A}'$ has knowledge of at most $j-1$ backward keys $K^B_m, \ldots, K^B_{m-j+2}$. Observe that $SK'_j = SK_j$ provided $\mathcal{A}'$ knows (1) the backward key K^B_{m-j+1} and (2) the random r_j or $r_{j'}$.

Condition (1) occurs if either of the following two holds:

1. $\mathcal{A}'$ is able to compute $u_j \cdot \psi(GM)$ from $View$. From View, $\mathcal{A}'$ knows $\{u_j \cdot \psi(U_k) : U_k \in W_j \cup \ R_j, 1 \leq j \leq m\}$, where $W_j \subset U \setminus G_j$ has minimal cardinality with $W_j \cup R_j \in \bar{\Gamma}_0$ and will not be able to compute $u_j \cdot \psi(GM)$ by the property of ψ. Observe that $u_j \cdot \psi(GM)$ is a linear combination of $\{u_j \cdot \psi(U_k) : U_k \in B\}$ if and only if $B \in \Gamma$. Consequently, $\mathcal{A}'$ will not be able to recover K^B_{m-j+1} from B_j as described above.

2. $\mathcal{A}'$ is able to choose $X \in GF(q)$ so that the following relations hold:

$$K^B_m = H^{j-1}(X), \ldots, K^B_{m-j+2} = H(X) \tag{6.5}$$

This occurs with a non-negligible probability only if $\mathcal{A}$ is able to invert the one-way hash function H. In that case, $\mathcal{A}$ returns $x = H^{-1}(y)$.

The condition (2) occurs if either of the following two holds:

1. $\mathcal{A}'$ is able to choose $n_j \in GF(q)$ so that $n_j = r_j$ or $n_j = r_{j'}$. This occurs with a negligible probability $1/q$.

2. $\mathcal{A}'$ is able to deduce $r_j \in GF(q)$ or $r_{j'} \in GF(q)$ from $View$. From $View$, $\mathcal{A}'$ knows $r_1, \ldots, r_{j-1} \in GF(q)$. Observe that $r_1, \ldots, r_{j-1}$ are generated by CSPRNG, if $\mathcal{A}'$ is able to reduce r_j or $r_{j'}$ from known random numbers $r_1, \ldots, r_{j-1}$, then the CSPRNG is insecure, thereby leading to a contradiction.

The above arguments show that if $\mathcal{A}'$ is successful in breaking the security of our construction, then $\mathcal{A}$ is able to invert the one-way hash function and break CSPRNG.

Hence, our construction is computationally secure for forward secrecy under the difficulty of inverting the one-way hash function and breaking CSPRNG. A quite similar proof can be conducted for backward secrecy of our construction. The only difference is that the coalition of new users knows the backward keys but they do not know the random numbers $r_1, \ldots, r_{j-1}$ and consequently are unable to compute the past session keys for which they had no authorization. We omit the proof here.

Theorem 6.2. Our Construction can resist collusion attack with respect to *Definition 6.3* and *Definition 6.4*.

Proof: We consider a coalition of $B \cup C \subset R$, where $B \subset R_l$ is a subset of users whose life cycles expire at session r while revoked at session $l (l < r)$ and $C \subset J_s$ is a set of users who join the group from session s with $l < s$. We show the users in $B \cup C$ together are not authorized to access the j-th session key SK_j for $l \leq j < s - 1$.

Also, we can model this coalition of users from $B \cup C$ as a polynomial-time algorithm $\mathcal{A}'$ that takes $View$ as input and outputs its guess for SK_j. We say that $\mathcal{A}'$ is successful in breaking the construction if it has a non-negligible advantage in determining the session key SK_j.

Here we only consider the condition that $r < s$. $\mathcal{A}$ first generates an instance View for $\mathcal{A}'$ as follows:

$$y = \begin{cases} u_s \cdot \psi(U_k) \quad \text{for } U_k \in B \cup C \text{ and } 1 \leq s \leq m; \\ B_j \quad \text{for } j = 1, \ldots, m; \\ B_{l'}, \ldots, B_{r'}; \\ S^B; \\ \{r_1, \ldots, r_{r-1}\} \cup \{r_s, \ldots, r_m\}; \\ \{SK_1, \ldots, SK_{r-1}\} \cup \{SK_s \ldots, SK_m\}. \end{cases}$$

$\mathcal{A}$ gives $View$ to $\mathcal{A}'$, which in turn selects random number $n_j \in GF(q)$ randomly, sets the j-th session key to be $SK_{j'} = K^B_{j-m+1} + n_j$ and returns $SK_{j'}$ to $\mathcal{A}$. $\mathcal{A}$ checks whether $SK_{j'} = SK_j$. If not, $\mathcal{A}$ chooses a random $x' \in GF(q)$ and outputs x'.

$\mathcal{A}'$ can compute $K^B_{m-j+1} = H^{m-j+1}(S^B)$ because it knows S^B from $View$. Note that from $View$, $\mathcal{A}'$ knows random numbers $\{r_1, \ldots, r_{r-1}\} \cup \{r_s, \ldots, r_m\}$ and session keys $\{SK_1, \ldots, SK_{r-1}\} \cup \{SK_s \ldots, SK_m\}$. Observe that $SK_{j'} = SK_j$ provided $\mathcal{A}'$ knows the random $r_{j'}$ for sessions (l, r) and r_j for sessions $(r+1, s-1)$.

For sessions (l, r), this condition occurs if any one of the following three holds:

1. $\mathcal{A}'$ is able to choose $n_j \in GF(q)$ so that $n_j = r_{j'}$. This occurs with a negligible probability $1/q$.

2. $\mathcal{A}'$ is able to deduce $r_{j'} \in GF(q)$ from $View$. From $View$, $\mathcal{A}'$ knows $\{r_1, \ldots, r_r\} \cup \{r_{s-1}, \ldots, r_m\}$ and $B_{l'}, \ldots, B_{r'}$. Observe that $\{r_1, \ldots, r_r\} \cup \{r_{s-1}, \ldots, r_m\}$ are are generated by CSPRNG, if $\mathcal{A}'$ is able to deduce $r_{j'}$ from known random numbers $\{r_1, \ldots, r_r\} \cup \{r_{s-1}, \ldots, r_m\}$, then the CSPRNG is insecure, and thus leads to a contradiction.

3. $\mathcal{A}'$ is able to reduce $r_{j'} \in GF(q)$ from $View$. From $View$, $\mathcal{A}'$ knows $B_{l'}, \ldots, B_{r'}$. Observe that the random number $r_{j'}$ is masked by $u_j \cdot \psi(GM)$. For each session j, the subset revoked users $R_j \in R$ can get vectors $\{(U_k, u_j \cdot \psi(U_k))\}_{U_k \in W_j \cup R_j}$ where $W_j \cup R_j \notin \Gamma$, and this does not obtain any information of $u_j \cdot \psi(GM)$ as it has minimal cardinality with $W_j \cup R_j \in \bar{\Gamma}_0$ and will not be able to compute $u_j \cdot \psi(GM)$ by the property of ψ. We arrive at this conclusion because for any scalar $v \in GF(q)$ there exists at least one vector $u \in GF(q)^l$ such

that
$$\begin{cases} u \cdot \psi(GM) = v \\ u \cdot \psi(U_k) = u_j \cdot \psi(U_k) \text{ for any } U_k \in W_j \cup R_j \end{cases}$$

because $W_j \cup R_j \notin \Gamma$. Notice that the number of vectors u satisfying this system of equations is independent of the value v. Therefore, $\mathcal{A}'$ is unable to deduce random numbers $r_{l'}, \ldots, r_{r'}$.

A quite similar proof can be conducted for sessions $(r+1, s-1)$. Similarly, we can prove that the *Definition 6.3* is correct for the condition that $r \geq s$. We can prove that our construction can resist collusion attack with respect to *Definition 6.4* in the same way. The only difference is that there is no the third condition as the broadcast messages $B_{r+1}, \ldots, B_{s-1}$ do not include any information about random numbers $r_{r+1}, \ldots, r_{s-1}$.

The above arguments show that if $\mathcal{A}'$ is successful in breaking the security of our construction, then $\mathcal{A}$ can break CSPRNG or ψ. Hence, our construction is computationally secure for resisting R-collusion under the hardness of breaking CSPRNG and ψ.

6.9.2 Proof under the Defined Security Model

We now demonstrate that our construction satisfies all the conditions required by *Definition 6.1.*

1. The scheme is a session key distribution scheme.

   (a) A non-revoked user $U_i \in G_j$ can recover the session key SK_j efficiently from broadcast message B_j and personal key S_i. As can be seen in the procedure of *Key Recovery*.

   (b) For any set $R_j \subseteq U$, where $R_j \in R$ and non-revoked user $U_i \notin R_j$ whose life cycle is (r_i, s_i), we show that the coalition R_j cannot compute personal secret $(u_{r_i} \cdot \psi(U_i), \ldots, u_{s_i} \cdot \psi(U_i))$ of U_i. For any session j, U_i uses its $u_j \cdot \psi(U_i)$ and r_j as its personal secret. Since the coalition $R_j \cup W_j \notin \Gamma$, the values $\{u_s \cdot \psi(U_k) : U_k \in R_j \cup W_j, 1 \leq s \leq m\}$ is not enough to compute $u_j \cdot \psi(U_i)$ by the property of ψ. So it is computationally infeasible for coalition in R_j to determine $u_j \cdot \psi(U_i)$ for $U_i \notin R_j$. Moreover, the random numbers

$r_1, \dots, r_m$ are generated by CSPRNG. The coalition R_j cannot guess r_j with knowledge of $r_1, \dots, r_{j-1}$ due to the security of CSPRNG.

(c) The session key SK_j for session j is computed from two parts: backward key K^B_{m-j+1}, and random number r_j or $r_{j'}$, where r_j is parts of personal key received from the GM before or when it joins the session group and $K^B_{m-j+1} = z_j - u_j \cdot \psi(GM)$ and $r_{j'} = z_{j'} - u_j \cdot \psi(GM)$ are recovered from the broadcast message B_j. So the personal secret keys alone do not give any information about any session key. Since backward seed S^B is chosen randomly, the backward key K^B_{m-j+1} is random. Furthermore, the broadcast $B_1, \dots, B_m$ determine $z_1, \dots, z_m$ and $z_{1'}, \dots, z_{m'}$ in some sessions. However, the scalar $u_j \cdot \psi(GM)$ perfectly masks the backward key and new random number $r_{j'}$. So it is computationally infeasible to determine session key SK_j from broadcast message B_j or personal key S_i alone.

2. The scheme has R-revocation capability. We have demonstrated that it is impossible in the proof of *Theorem 6.1*.

3. The scheme has self-healing capability as can be seen from the procedure of *Self-healing*.

We will show that our construction satisfies both the forward and backward secrecy required by *Definition 6.2*.

1. *Forward Secrecy*. Only those sets belongs to Γ can recover the $\psi(GM)$ and further recover the session key SK_j. Because of the fact that $R_j \cup W_j \subseteq R \notin \Gamma$, the coalition of R cannot recover $\psi(GM)$. Furthermore, because of the one-way property of H, it is computationally infeasible to compute K^B_s from K^B_t for $s < t$. That is, even R_j know the sequence of backward keys $K^B_m, \dots, K^B_{m-j+2}$, it cannot compute backward key K^B_{m-j+1} for session j and consequently SK_j from the sequence. In addition, the random numbers strengthen the forward secrecy of our scheme.

2. *Backward Secrecy*. When a new user U_l joins the group starting from session j, the GM gives U_l at most $r_j, \dots, r_m$ random numbers. Let $J_j \subseteq U$ is the set of users who join after the current session j. It is computationally infeasible

for J_j to recover random numbers $r_1, \ldots, r_{j-1}$ even with the knowledge of group keys after session j under the security of CSPRNG. Correspondingly, J_j cannot recover session keys $SK_1, \ldots, SK_{j-1}$.

We demonstrate that our construction can resist collusion attack required by *Definition 6.3*.

Let $B \subset R_l$ be a subset of users whose life cycles expire at session r while revoked at session $l(l < r)$. Let $C \subset J_s$ be a subset of users who join the group from session s with $l < s$. That is, B and C are two disjoint subset. $B \cup C \subset R$ can not recover SK_j for sessions $j = l, \ldots, s-1$ from broadcast B_j and their personal secrets.

Case 1: $r < s$: In order to recover session keys $SK_j(j = l, \ldots, s-1)$, $B \cup C$ require the knowledge $r_{l'}, \ldots, r_{r'}, r_{r+1}, \ldots, r_{s-1}$. $B \cup C$ know $\{r_1, \ldots, r_r\} \cup \{r_s, \ldots, r_m\}$ and $B_{l'}, \ldots, B_{r'}$. $\{r_1, \ldots, r_r\} \cup \{r_s, \ldots, r_m\}$ are are generated by CSPRNG. $B \cup C$ cannot deduce $r_{l'}, \ldots, r_{r'}$ from them. $B \cup C$ cannot deduce $r_{j'} \in GF(q)$ from $B_{l'}, \ldots, B_{r'}$, either. Observe that the random number $r_{j'}$ is masked by $u_j \cdot \psi(GM)$. For each session j, the subset revoked users $R_j \in R$ can get vectors $\{(k, u_j \cdot \psi(k))\}_{k \in W_j \cup R_j}$ where $W_j \cup R_j \notin \Gamma$, and this does disclose any information of $u_j \cdot \psi(GM)$ as it has minimal cardinality with $W_j \cup R_j \in \bar{\Gamma_0}$ and will not be able to compute $u_j \cdot \psi(GM)$ by the property of ψ. Therefore, $B \cup C$ are unable to deduce random numbers $r_{l'}, \ldots, r_{r'}$. For the same reason, $B \cup C$ cannot deduce $r_{r+1}, \ldots, r_{s-1}$ from broadcast messages $B_{r+1}, \ldots, B_{s-1}$.

Case 2: $r \geq s$: In order to recover session keys $SK_j(j = l, \ldots, s-1)$, $B \cup C$ require the knowledge $r_{l'}, \ldots, r_{(s-1)'}$. $B \cup C$ know $\{r_1, \ldots, r_r\} \cup \{r_r, \ldots, r_m\} \cup \{r_{s'}, \ldots, r_{(r-1)'}\}$ and $B_{l'}, \ldots, B_{r'}$. $r_{l'}, \ldots, r_{(s-1)'}$. $B \cup C$ know $\{r_1, \ldots, r_r\} \cup \{r_r, \ldots, r_m\} \cup \{r_{s'}, \ldots, r_{(r-1)'}\}$ are are generated by CSPRNG. $B \cup C$ cannot deduce $r_{l'}, \ldots, r_{(s-1)'}$ from them. $B \cup C$ cannot deduce $r_{j'} \in GF(q)$ from $B_{l'}, \ldots, B_{(s-1)'}$, either. Therefore, $B \cup C$ are unable to deduce random numbers $r_{l'}, \ldots, r_{(s-1)'}$.

We can demonstrate that our construction can resist the collusion required by *Definition 6.4* in the same way.

Table 6.3: Performance comparison in the j-th session of representative hash chain based self-healing key distribution schemes and the proposed scheme

Scheme	Performance measure			Security properties	
	Storage overhead	Communication overhead	Computation overhead	Forward & Backward secrecy	Collusion resistance
C1 in [86]	$(m-j+2)logq$	$(t+1)logq$	$2t+1$	both t-wise	$\times$
C2 in [86]	$(m-j+2)logq$	$(t+1)logq$	$2(t^2+t)$	both t-wise	$\times$
[92]	$(m-j+2)logq$	$(t_j+1)logq$	$2(t_j^2+t_j)$	both R-wise	$\times$
[93]	$(2k_i+1)logq$	$(t_j+1)logq$	$2(t_j^2+t_j)$	both R-wise	partial
[6]	$2k_ilogq$	$(t+1)logq$	$2t+1$	both t-wise	partial
[7]	$2k_ilogq$	$(t_j+1)logq$	$2(t_j^2+t_j)$	both R-wise	partial
Our scheme	$(2k_i+1)logq$	$>(t_j+1)logq$	$2(t_j^2+t_j)$	both R-wise	$\surd$

6.10 Performance Evaluation

We classify the schemes [6, 7, 86, 92, 93] and our scheme into two categories according to the masking mechanism. One category is Shamir's (t, n) threshold secret sharing masking mechanism based schemes [6, 86] and the other is vector space secret sharing masking mechanism based schemes [7, 92, 93] and our scheme. In fact, Shamir's (t, n) threshold scheme can be seen as a particular case of the vector space secret sharing scheme. The maximum number of revoked users in [6, 86] is fixed to t which is the degree of polynomial, while the vector space secret sharing mask mechanism based schemes realize a flexible access structure. Table 6.3 demonstrates the advantages that our construction has over other hash chain related schemes [6, 7, 86, 92, 93] regarding storage overhead, communication overhead, and computation overhead at user end rather than at the GM and security properties.

6.10.1 Storage Overhead

If we set $k_i = s_i - r_i + 1$, a user U_i whose life cycle is (r_i, s_i) has to store $(2k_i + 1)logq$ bits personal secret including k secret vectors, k random numbers and 1 hash value. Particularly, U_i requires $(2m + 1)logq$ bits memory if U_i is authorized to all the m sessions. If U_i joins in j-th session and belongs to all the subsequent sessions, it has to store $[2(m - j + 1) + 1]logq$ bits personal key. Compared with the storage overhead of schemes in [6, 7, 93], the $logq$ bits increase are used for implicit authentication. Dutta *et al.* claimed that the constructions in [86, 92] have the same storage overhead $(m-j+1)logq$ as the scheme 2 in [87]. We argue that authors overlooked storage overhead brought by the forward key seed. In Table 6.2, we correct the storage overhead for the two papers. If we denote the storage overhead of [86, 92] as $S_{[86,92]}$ and storage overhead of our construction as S_{new}, we have the following result:

$$\begin{cases} S_{new} < S_{[86,92]} \text{ if } k_i < \lceil \frac{m-j+1}{2} \rceil \\ S_{new} = S_{[86,92]} \text{ if } k_i = \lceil \frac{m-j+1}{2} \rceil \\ S_{new} > S_{[86,92]} \text{ if } k_i > \lceil \frac{m-j+1}{2} \rceil \end{cases} \tag{6.6}$$

Furthermore, the storage overhead $(m-j+2)logq$ of [86, 92] for users joining from session j is a fixed value. the storage overhead $2k_i logq$ of [6, 7] and $(2k_i + 1)logq$ of [93] and the proposed construction are fixed values as well. As can be seen easily, the storage overhead has nothing to do with the total number of users n within the communication group. That is, neither of them increases with the network expansion.

Here we present a concrete example. Suppose $m = 100$, $j = 25$, and $logq = 64$, so $(m-j+1)/2 = 38$. The storage overhead for users who join the session in j-th session is 4928 bits in [86, 92]. While in our construction, if a user is only involved in $k_i \leq 38$ session, it stores $(2k_i + 1) \times 64 \leq 4928$ bits personal key. When a user is involved in $k > 38$ sessions, our construction has more storage overhead. The extra storage overhead comes from random numbers. However, it is used to resist collusion which is the explicit difference between schemes in [86, 92] and our scheme.

6.10.2 Communication Overhead

The broadcast message B_j at the j-th session consists of a set of identities of revoked users and their vectors corresponding to session j. One can ignore the communication overhead for the set of identities because the user identities can be selected from a small finite field [81]. In our scheme, the length of vectors depends on the particular function ψ which is related to the set $W_j \cup R_j$. In Table 6.2, t_j denotes $|W_j \cup R_j|$, where $W_j \subset U \setminus G_j$ and $R_j \notin \Gamma$ is the set of all revoked users for sessions in and before j with minimum cardinality such that $W_j \cup R_j \in \bar{\Gamma_0}$.

If no user is revoked before its life cycle finishes, our construction has the same communication overhead as that of [7]. We notice that if $t_j = t$, both our construction and that of [7] have the same broadcast length [6]. If $t_j < t$, our construction and [7] have lower communication overhead than [6]. Furthermore, [6] constrained the revoked number to the threshold t. If more than t users were revoked, the security of [6] can not be guaranteed. Our construction removes the limitation and is therefore more practical. Let $|R_j|$ be the number of revoked users in and before session j. Similarly, we denote the communication overhead of the constructions in [6] as $C_{[6]}$ and communication overhead of our construction as $C_{new\&[7]}$, we have the following result:

$$\begin{cases} C_{new\&[7]} < C_{[6]} & \text{if } |R_j| < t \\ C_{new\&[7]} = C_{[6]} & \text{if } |R_j| = t \\ C_{new\&[7]} = (t_j + 1)logq & \\ \text{while [6] collapses} & \text{if } |R_j| > t \end{cases} \tag{6.7}$$

The communication comparison result is shown in Figure 6.3.

Our solution incurs more communication overhead if some users are revoked because the new random numbers should be masked when subsequently broadcasted. In the worst condition, at least one user is revoked at each session. The broadcast message at session j can be denoted as $\{B_{1'} \cup \ldots \cup B_{(j-1)'} \cup B_j\}$. Because the communication overhead for the set of identities can be ignored, the communication overhead of B_j is $(t_j + 2)$. For any $l < j$, the communication overhead of $B_{l'}$ is $(t_l + 1)$. Therefore, the communication overhead at session j is $(1 + \sum_{k=1}^{j}(t_k + 1))$ which is still much less than that of [90, 91]. In a real network environment, each user is assigned a pre-arranged life cycle which is a limited number of sessions. It takes time to compromise a user. The general condition is that there are only very few sessions to the end of its life cycle when a user is compromised. Only very few random numbers need to be renewed. Correspondingly, the communication

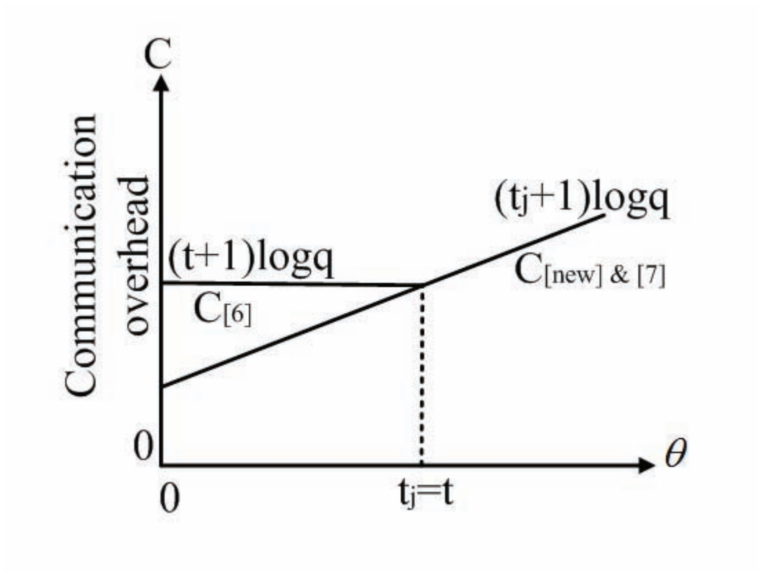


Figure 6.3: The comparison of communication overhead between [6, 7], and our construction especially when no user is revoked before its life cycle expires

overhead produced by broadcasting renewed random numbers is acceptable.

6.10.3 Computation Overhead

The computation in key recovery includes addition and multiplication over the finite field $GF(q)$ and hash operation. Here we measure the computation overhead by the number of multiplication operations. We ignore the computation cost for the addition and hash operations. We argue that the measurement will hold because computation costs for addition and hash operations can be ignored compared with that for multiplication. Even though our scheme achieves implicit authentication and is more secure, the computation overhead that it produces can be ignored. This is because implicit authentication is realized by the hash operation and new random number $r_{j'}$ is masked by $u_j \cdot \psi(GM)$ which has been worked out in backward key computation. The authorized user recovers $\psi(GM)$ from the set $W_j \cup R_j$ and the corresponding vectors as well as its personal identity and vector. We still use t_j to denote $|W_j \cup R_j|$, the computation overhead of our construction is $2(t_j^2 + t_j)$ multiplications which is the same as that of [7]. It can be easily concluded that our construction has the same computation overhead as [6] when $t_j = t$, and our construction has lower computation overhead than [6] when $t_j < t$. The computation

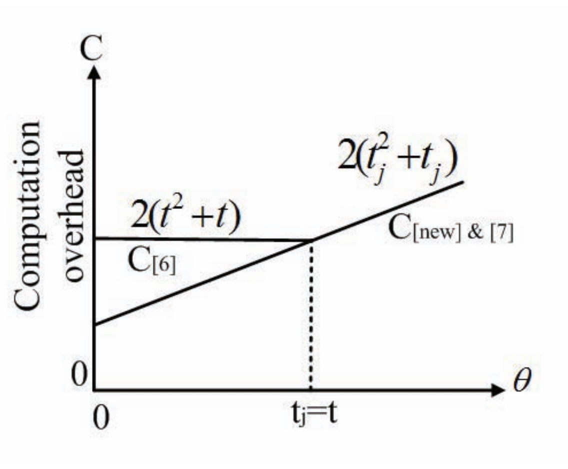


Figure 6.4: The comparison of computation overhead of [6, 7], and our construction

overhead of our construction continues to increase with t_j when $t_j > t$ while [6] collapse. If we denote the computation cost for [6] as $Comp_{[6]}$ and that for our construction and [7] as $Comp_{new}$, the following result holds.

$$\begin{cases} Comp_{new} < Comp_{[6]} & \text{if } |R_j| < t \\ Comp_{new} = Comp_{[6]} & \text{if } |R_j| = t \\ Comp_{new} = (2(t_j^2 + t_j)) & \\ \text{ while [6] collapses} & \text{if } |R_j| > t \end{cases} \tag{6.8}$$

This computation comparison has been shown in Figure 6.4. It is easy to see that our scheme is more computationally efficient than [6] if $|R_j| < t$ and both of them have the same computation overhead if $|R_j| = t$. The construction in [6] collapses if more than t users are revoked, while our construction is still secure.

6.11 Conclusion

We made an in-depth analysis of the breach of collusion attack against hash chain based self-healing key distribution schemes in this chapter and proposed an unified approach for collusion attack. Then we launched two attacks against Du's scheme [116] which claims that it can resist a collusion attack by a coalition of revoked users whose life cycle has not expired and the newly joined users with the proposed unified attack approach. We proposed two possible solutions for collusion attack and ruling one of the solutions out. We developed a self-healing key distribution solution with implicit authentication . We considered general access structures instead of threshold ones to provide more flexible performance for self-healing key distribution. This would suit various wireless network environments. The scheme has realized a general monotone decreasing structure for the family of subsets that can be revoked instead of a threshold one in polynomial-based schemes. Most important of all, the scheme can resist collusion between the newly joined users and revoked users whose life cycle has not expired, in addition to forward and backward secrecy. As far as we know, it is the first time that such a collusion attack has been repelled by hash chain based self-healing key distribution schemes. The schemes have been analyzed using an appropriate security model to prove that they are computationally secure and can be used for secure communications over unreliable wireless networks.

7

HBT-based Self-healing Key Distribution with Implicit Authentication

7.1 Introduction

We discussed the advantages of self-healing key distribution in previous chapters. Self-healing key distribution schemes enable large and dynamic groups of users over an unreliable network to establish group keys for secure communication over an unreliable channel in the manner that is resistant to packet loss. The goal of self-healing key distribution schemes is that even if in a certain session the broadcast is lost, the group users are still able to recover the session key from the broadcasts received before and after the session, without requesting additional transmissions from the group manager. Hence, non-interactive self-healing key distribution solutions are not only favorable but also necessary.

We devoted to solve the collusion attack problem in hash chain based self-healing key distribution schemes in [93]. Each user is assigned with a pre-arranged life cycle when it joins the network and forcefully revoked when its life cycle finishes. Different from previous schemes [86, 94, 95], we added XOR random numbers operation to *Broadcast*. The scheme not only keeps the forward and backward secrecy but also collusion resistance property. Dutta *et al.* addressed the problem of achieving collusion resistance following the same approach of [93] in [7] which further reduces the storage and computation overhead. Jiang *et al.* proposed a novel HBT-based self-healing key distribution scheme in [112] which has less computation overhead at user end. A HBT with the scale of maximum life cycle of network is generated by interactively hash operation on a seed. Each session key is mapped to a leaf node in the HBT. We propose a HBT-based self-healing key distribution scheme which has more flexible access structure than [112].

The rest of this chapter is organized as follows: In Section 7.2, we introduce the contribution of the proposed solution. In Section 7.3, we propose system parameters and security model. In Section 7.4, we present a construction of implicitly authenticated HBT-based self-healing key distribution scheme. In Section 7.5, we analyze the security of the scheme under security model. In Section 7.6, we make a comparison of our scheme with several representative schemes regarding the efficiency. Finally in Section 7.7, we conclude this chapter.

7.2 The Contribution of the Proposed HBT-based Self-healing Key Distribution Scheme

The contribution of the proposed solution is two-fold. Firstly, we propose an HBT-based self-healing key distribution scheme following approach of [112]. The scheme achieves more flexible access structure than that of [112]. The proposed scheme not only keeps forward and backward secrecy but also collusion resistance property. Secondly, implicit authentication rather than MAC is used to detect tamper attack. Such an authentication manner does not increase message size and computation cost. Performance analysis shows that the scheme is lightweight in storage and computation overhead without degrading security.

7.3 System Parameters and Security Model

Let $U = \{U_1, \ldots, U_n\}$ be the finite universe of the network. The communication group is a dynamic subset of U and $R \subset 2^U$ is a monotone decreasing access structure of subsets of users. Let $i \in \{1, \dots, n\}$ and $j \in \{1, \dots, m\}$ and G_j be the group in session j. In addition, the notations in Table 7.1 are used throughout the chapter.

Table 7.1: Notations used in the HBT-based self-healing key distribution scheme

GM	the group manager
U_i	the i-th user
n	the total number of users in the network
m	the maximum life cycle of the network
$GF(q)$	a field of order q
S_i	the personal secret of user U_i
SK_j	the session key for session j
B_j	broadcast message for session j
R_j	the set of the revoked users before and in session j
J_j	the set of the joined users after session j
$FK(h, j)$	the j-th forward key
BK_j	the j-th backward key

To further clarify our goals and facilitate the later presentation, according to the security model of [93], we define the self-healing key distribution scheme from three aspects. *Definition 7.1* defines self-healing key distribution scheme with revocation and implicit authentication capability. *Definition 7.2* defines forward and backward secrecy. *Definition 7.3* defines collusion resistance properties.

Definition 7.1. (Self-healing key distribution with revocation and implicit authentication capability.)

1. The scheme is a session key distribution scheme with privacy if

   (a) for any user $U_i \in G_j$, the session key SK_j is efficiently determined from B_j and S_i.

   (b) for any set $R_j \subseteq U$, where $R_j \subseteq R$ and $U_i \notin R_j$, it is computationally infeasible for users in R_j to determine the personal key S_i.

   (c) it is computationally infeasible to compute session key SK_j from either m broadcasts $B_1, \dots, B_m$ alone or n personal keys $S_1, \dots, S_n$.

2. The scheme has R-revocation and implicit authentication capability if given any $R_j \subseteq U$, where $R_j \subseteq R$, the GM can generate a broadcast message B_j, such that for all $U_i \notin R_j$, U_i can *efficiently verify the truth of* B_j and recover the session key SK_j, but the revoked users cannot.

3. The scheme has self-healing capability if the following is true for any session $j(1 \leq j_1 < j < j_2 \leq m)$ if each $U_i \in G_{j_1}$, who has not been revoked after session j_1 and before session j_2, can recover all session keys SK_j for $j = j_1, \ldots, j_2$, from broadcasts B_{j_1}, B_{j_2} and its personal key.

Definition 7.2. The scheme guarantees both R-wise forward and backward secrecy if

1. for any set $R_j \subset U$, where $R_j \subseteq R$, and all $U_i \in R_j$ are revoked in and before session j, it is computationally infeasible for the users in R_j together to get any information about SK_j, even with the knowledge of session keys $SK_1, \ldots, SK_{j-1}$.

2. for any set $J_j \subset U$, where $J_j \nsubseteq R$, and all users $U_i \in J_j$ join in and after session j, it is computationally infeasible for the users $U_i \in J_j$ together to get any information about SK_j, even with the knowledge of group keys $SK_{j+1}, \ldots, SK_m$.

Definition 7.3. Let $B \subset R_s$ be a subset of users whose life cycles expire at session s. Let $C \subset J_t$ be a subset of users who join the network from session t with $1 \leq s < t \leq m$. That is, B and C are two disjoint subsets. Suppose $B \cup C \subset R$. The coalition $B \cup C$ cannot get any information about session keys SK_j, for any session $s + 1 \leq j < t$. That is, the scheme is able to resist R-wise collusion of the revoked users and the newly joined users.

7.4 The Construction of the HBT-based Self-healing Key Distribution

Following the idea of the schemes in [93, 112], we present a HBT-based self-healing key distribution scheme.

In our setting, the communication group is a dynamic subset of users of U. An unreliable broadcast channel is available. The GM sets up and controls the group by performing addition and revocation operations. All of operations take place in $GF(q)$, where q is a large prime number ($q > n$). $H : GF(q) \rightarrow GF(q)$ is a cryptographically secure one-way hash function.

Let us consider a vector space secret sharing scheme realizing $\Gamma = 2^U - R$ over the set U. For simplicity, we suppose that there exists a public map

$$\psi : U \cup \{GM\} \rightarrow GF(q)^l \tag{7.1}$$

which defines Γ as a vector space access structure. The use of a specific ψ fixes the properties of the scheme.

The self-healing key distribution scheme is composed of five procedures. We describe the procedures one by one.

Setup. The GM randomly picks two initial key seeds, forward key seed $FK(0, 1)$ and backward key seed BK, from $GF(q)$. In the pre-processing time, the GM computes one hash binary tree with m leave nodes and one backward hash chain of length $m + 1$.

- The generation of HBT. For simplicity, we call compound operation of hash $H(\cdot)$ and left shift $LeftShift(\cdot)$ as $LH(\cdot)$ and compound operation of hash $H(\cdot)$ and right shift $RightShift(\cdot)$ as $RH(\cdot)$. The GM generates the left and right intermediate values in the first level by applying $LH(\cdot)$ and $RH(\cdot)$ to the initial seed $FK(0, 1)$, respectively. By the same way, the GM generates all intermediate and leaf nodes in the binary tree with depth $h = \lceil log_2 m \rceil$. Each leaf node in the HBT is related to a forward key. Figure 7.1 demonstrates a HBT for a network whose life cycle is m. The forward key for the first session is $FK(h, 1)$, the forward key for the second session is $FK(h, 2)$, and correspondingly the forward key for the last session is $FK(h, m)$. Figure 7.2 is a specific example with $m = 8$.

- The generation of backward hash chain. The GM computes the backward hash chain by repeatedly applying the same one-way hash function H on

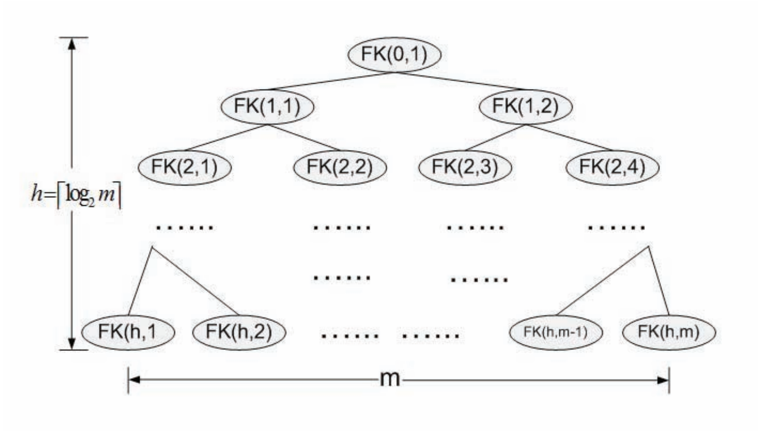


Figure 7.1: The hash binary tree for a network whose life cycle is m

backward key seed BK. The generated hash chain is $\{BK_1, \ldots, BK_m, BK_{m+1}\}$. The last element BK_{m+1} is used for implicit authentication.

The session key for session j is computed as:

$$SK_j = FK(h, j) + BK_{m-j+1}. \tag{7.2}$$

Each user U_i is assigned a pre-arranged life cycle (s_i, t_i) where $1 \leq s_i < t_i \leq m$. U_i is involved in $k_i = t_i - s_i + 1$ sessions. The GM chooses random vectors $v_1, \ldots, v_m \in GF(q)^l$ and computes secret vectors for each user U_i. The user U_i receives its private key S_i from the GM consisting of: (1) forward key or intermediate forward key seed corresponding to the life cycle (s_i, t_i); (2) secret vectors $(v_{s_i} \cdot \psi(U_i), \ldots, v_{t_i} \cdot \psi(U_i))$. The GM sends the private key to user U_i via the secure channel between them. We make an example for the selection of forward key or forward key seed in Figure 7.2. Figure 7.2 demonstrates a network whose life cycle is 8. For a user whose life cycle is $(1, 5)$, the GM assigns $\{FK(1, 1), FK(3, 5)\}$ to this node. It is unnecessary to distribute the set $\{FK(3, 1), FK(3, 2), FK(3, 3), FK(3, 4), FK(3, 5)\}$ to this user. The user can calculate the required forward key by applying $LH(\cdot)$ or $RH(\cdot)$ on the intermediate forward key seed. For an extreme scenario that a node's life cycle is $(1, 8)$, a forward key seed $FK(0, 1)$ is enough for it to calculate all forward keys.

Broadcast. In session $j = 1, \ldots, m$, suppose $R_j \in R$ represents the set of all

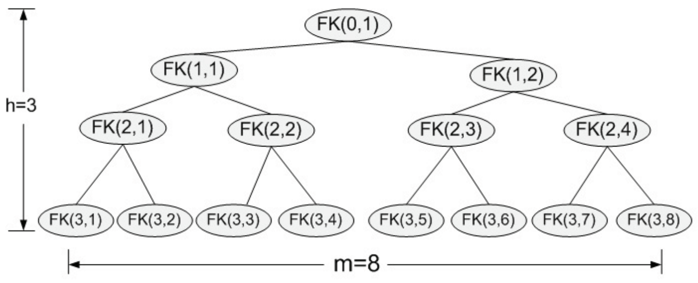


Figure 7.2: An example of a hash binary tree with m=8

revoked users in and before session j and G_j represents the set of all non-revoked users in session j. The GM chooses a subset of users $W_j \subset U \setminus G_j$ with minimum cardinality such that $W_j \cup R_j \in \bar{\Gamma}_0$. The GM computes $z_j = BK_{m-j+1} + v_j \cdot \psi(GM) \in GF(q)$ and broadcasts the message $B_j = z_j \cup \{(U_k, u_j \cdot \psi(U_k)) : U_k \in W_j \cup R_j\}$.

Key Recovery. When a non-revoked user U_i receives the key distribution message B_j for session j, since $U_i \in G_j$ has $\{(U_k, u_j \cdot \psi(U_k))\}_{U_k \in W_j \cup R_j}$ and its personal key, it computes $u_j \cdot \psi(GM)$ using $\{(U_k, u_j \cdot \psi(U_k))\}_{U_k \in W_j \cup R_j \cup \{U_i\}}$ because $W_j \cup R_j \cup \{U_i\} \in \Gamma$. In effect, as long as $W_j \cup R_j \cup \{U_i\} \in \Gamma$, the result $\psi(GM) = \sum_{U_k \in W_j \cup R_j \cup \{U_i\}} \lambda_k \psi(U_k)$ holds for some $\lambda_k \in GF(q)$. Therefore, $u_j \cdot \psi(GM) = \sum_{U_k \in W_j \cup R_j \cup \{U_i\}} \lambda_k u_j \cdot \psi(k)$. From the broadcast information B_j, U_i recovers the backward key

$$BK_{m-j+1} = z_j - u_j \cdot \psi(GM). \tag{7.3}$$

U_i can check the validity of the recovered BK_{m-j+1}. If $H^j(BK_{m-j+1}) \neq BK_{m+1}$, the broadcast message is tampered during transmission, U_i discards the message. Otherwise, U_i computes the j-th forward key $FK(h, j)$ by applying the hash function to the assigned intermediate forward key seed if it does not have a forward key. Finally U_i calculates the current session key

$$SK_j = FK(h, j) + BK_{m-j+1}. \tag{7.4}$$

Adding/Revoking Users. When a new user U_l wants to join the network, the user gets in touch with the GM. The GM assigns a life cycle (s_l, t_l) to the new user. The GM assigns a forward key or intermediate forward key seed corresponding to the life cycle (s_l, t_l) and secret vectors $\{u_{s_l} \cdot \psi(U_l), \ldots, u_{t_l} \cdot \psi(U_l)\}$ to this new user over the secure communication channel between them.

Our scheme supports implicit revocation. When a user U_i life cycles (s_i, t_i) finishes, its ID will be added to R_j by the GM. The GM will construct broadcasts for the subsequent sessions with new access structure. In addition, we allow a revoked user to rejoin the group in later sessions with a new identity. However, for existing hash chains based self-healing key distribution schemes [86, 95], rejoining is not permitted.

Self-healing. Suppose U_i is a user who receives session broadcast message B_{j_1} and B_{j_2} in session j_1 and j_2 respectively, where $1 \leq s_i \leq j_1 < j_2 \leq t_i \leq m$, but not broadcast message B_j for the session j, where $j_1 < j < j_2$. U_i can still recover all the lost session keys SK_j $(j_1 < j < j_2)$ as follows:

1. U_i recovers from the broadcast message B_{j_2} in session j_2, the backward key BK_{m-j+1} and repeatedly applies the one-way hash function to this and computes all the backward keys corresponding to sessions j $(j_1 < j < j_2)$.

2. U_i computes the forward keys $FK(h, j)$ for all j $(j_1 < j < j_2)$ by repeatedly applying hash functions $LH(\cdot)$ or $RH(\cdot)$ to the intermediate forward key seed.

3. U_i recovers all the session keys $K_j = FK(h, j) + BK_{m-j+1}$, for $j_1 < j < j_2$.

7.5 Security Evaluation

In this section, we evaluation from two aspects. We firstly prove the security of the HBT-based solution under an appropriate framework and then demonstrate that the HBT-based solution meets the security requirements of the defined security model.

7.5.1 Proof under an Appropriate Framework

We can prove Theorem 7.1 which states forward and backward secrecy, and Theorem 7.2 which states the capability of collusion resistance of our construction in an appropriate framework.

Theorem 7.1. Our construction is a secure, self-healing session key distribution scheme with privacy, R-revocation and implicit authentication capability with respect to *Definition 7.1*, and enjoys R-forward and backward secrecy with respect to *Definition 7.2*.

Proof: Our goal is R-wise forward and backward secrecy. We first show that our construction is computationally secure with respect to revoked users under the difficulty of inverting the one-way function. That is, our construction satisfies forward secrecy. For any session j, it is computationally infeasible for any set of revoked users from R before and on session j to compute with non-negligible probability the session key K_j, given the $View$ consisting of personal keys of revoked users, broadcast messages before, on and after session j and session keys of revoked users before session j. In fact, to some extent, forward secrecy implies R-revocation capability.

Consider a coalition of revoked users from R, say $R_j \in R$, who are revoked in and before session j. The users in R_j are not entitled to know the j-th session key K_j. We can model this coalition of users from R as a polynomial-time algorithm $\mathcal{A}'$ that takes $View$ as input and outputs its guess for K_j. We say that $\mathcal{A}'$ is successful in breaking the construction if it has a non-negligible advantage in determining the session key K_j. Then using $\mathcal{A}'$, we can construct a polynomial time algorithm $\mathcal{A}$ for inverting the one-way function H and have the following claim:

Claim: $\mathcal{A}$ inverts one-way function H with a non-negligible probability if $\mathcal{A}'$ is successful.

Proof: Given any instance $y = H(x)$ of one-way function H, $\mathcal{A}$ first generates an instance $View$ for $\mathcal{A}'$ as follows: $\mathcal{A}$ randomly selects a backward key seed $S^F \in$

F_q and constructs the following backward key chain by repeatedly applying H to y:

$$\begin{aligned} BK_1 = y, BK_2 = H(y), BK_3 = H^2(y), \dots, \\ BK_j = H^{j-1}(y), \dots, BK_m = H^{m-1}(y). \end{aligned} \tag{7.5}$$

$\mathcal{A}$ computes the j-th forward key $FK_{h,j}$ and sets the j-th session key

$$SK_j = FK(h, j) + BK_{m-j+1}. \tag{7.6}$$

$\mathcal{A}$ chooses at random m vectors $u_1, \dots, u_m \in GF(q)^l$. Each user $U_i \in U$ with life cycle (s_i, t_i) receives its personal secret keys $(u_{s_i} \cdot \psi(U_i), \dots, u_{t_i} \cdot \psi(U_i))$ and the intermediate forward key seed from $\mathcal{A}$ via the secure communication channel between them. In this setting, $\Gamma = 2^U \setminus R$ is a monotone increasing access structure of authorized users over U, Γ is determined by the family of minimal qualified subsets, Γ_0, which is called the basis of Γ. Now $R_j \subseteq R$ implies $R_j \notin \Gamma$.

Let G_j be the set of all non-revoked users in session j. At the j-th session, $\mathcal{A}$ chooses a subset of users $W_j \subset U \backslash G_j$ with minimal cardinality such that $W_j \cup R_j \in \bar{\Gamma}_0$. $\mathcal{A}$ then computes broadcast message B_j for $j = 1, \dots, m$ as:

$$B_j = z_j \cup \{(U_k, u_j \cdot \psi(U_k)) : U_k \in W_j \cup R_j\} \tag{7.7}$$

where $z_j = BK_{m-j+1} + u_j \cdot \psi(GM)$. Then $\mathcal{A}$ sets $View$ as

$$y = \begin{cases} u_s \cdot \psi(U_k) & \text{for all } U_k \in R_j \cup W_j \\ & \text{ and } s = 1, \dots, m; \\ B_j & \text{for } j = 1, \dots, m; \\ FK(0, 1); & \\ SK_1, \dots, SK_{j-1}. & \end{cases} \tag{7.8}$$

$\mathcal{A}$ gives $View$ to $\mathcal{A}'$, which in turn randomly selects $X \in GFq$, sets the j-th session key to be $SK_{j'} = FK(h, j) + X$ and returns $SK_{j'}$ to $\mathcal{A}$. $\mathcal{A}$ checks whether $SK_{j'} = SK_j$. If not, $\mathcal{A}$ chooses a random $x' \in GF(q)$ and outputs x'.

$\mathcal{A}'$ can compute the j-th forward key $FK(h, j)$ for $j = 1, \dots, m$ because it knows $FK(0,1)$ from $View$. Note that from $View$, $\mathcal{A}'$ knows $\{u_j \cdot \psi(U_k) : U_k \in W_j \cup \ R_j, 1 \leq j \leq m\}$ and session keys $SK_1, \ldots, SK_{j-1}$. Consequently, $\mathcal{A}'$ has knowledge of at most $j-1$ backward keys $BK_m, \ldots, BK_{m-j+2}$. There is $SK_{j'} = SK_j$ provided that $\mathcal{A}'$ knows the backward key BK_{m-j+1}.

The condition occurs if the following holds:

(a) $\mathcal{A}'$ is able to compute $u_j \cdot \psi(GM)$ from $View$. From View, $\mathcal{A}'$ knows $\{u_j \cdot \psi(U_k) : U_k \in W_j \cup \ R_j, 1 \leq j \leq m\}$, where $W_j \subset U \setminus G_j$ has minimal cardinality with $W_j \cup R_j \in \bar{\Gamma}_0$ and will not be able to compute $u_j \cdot \psi(GM)$ by the property of ψ. Observe that $u_j \cdot \psi(GM)$ is a linear combination of $\{u_j \cdot \psi(U_k) : U_k \in B\}$ if and only if $B \in \Gamma$. Consequently, $\mathcal{A}'$ is not able to recover BK_{m-j+1} from B_j.

(b) $\mathcal{A}'$ is able to choose $X \in GF_q$ so that the following relations hold:

$$BK_m = H^{j-1}(X), \ldots, BK_{m-j+2} = H(X). \tag{7.9}$$

This occurs with a non-negligible probability only if $\mathcal{A}$ is able to invert the one-way function H. In that case, $\mathcal{A}$ returns $x = H^{-1}(y)$.

The above arguments show that if $\mathcal{A}'$ is successful in breaking the security of our construction, then $\mathcal{A}$ is able to invert the one-way hash function.

Hence our construction is computationally secure for forward secrecy under the difficulty of inverting the one-way function. A quite similar proof can be conducted for the backward secrecy of our construction. The only difference is that the coalition of new users knows the backward keys but they don't know the forward keys. We omit the proof here.

Theorem 7.2. Our Construction can resist collusion attack with respect to *Definition 7.3*.

Proof: Consider a coalition from $B \cup C \subset R$, where $B \subseteq R_s$ is a set of users removed from the group before session r and $C \subseteq J_t$ is a set of users who join the group from session s with $s < t$. We show the users in $B \cup C$ together are not entitled to know the j-th session key K_j for any $s \leq j < t-1$.

Also, we can model this coalition of users from $B \cup C$ as a polynomial-time algorithm $\mathcal{A}'$ that takes $View$ as input and outputs its guess for K_j. We say that $\mathcal{A}'$ is successful in breaking the construction if it has a non-negligible advantage in determining the session key K_j.

$\mathcal{A}$ first generates an instance $View$ for $\mathcal{A}'$ as follows:

$$y = \begin{cases} u_j \cdot \psi(U_k) \quad \text{for all } U_k \in B \cup C \text{ and } j = 1, \ldots, m; \\ B_j \quad \text{for } j = 1, \ldots, m; \\ \{FK(h,1), \ldots, FK(h, s-1)\}; \\ \{FK(h,t) \ldots, FK(h,m)\}; \\ BK; \\ \{K_1, \ldots, K_{s-1}\} \cup \{K_t, \ldots, K_m\}. \end{cases} \tag{7.10}$$

$\mathcal{A}$ gives $View$ to $\mathcal{A}'$, which in turn selects $FK(h, j')$ randomly, sets the j-th session key to be $SK_{j'} = FK(h, j') + BK_{j-m+1}$ and returns SK'_j to $\mathcal{A}$. $\mathcal{A}$ checks whether $SK_{j'} = SK_j$. If not, $\mathcal{A}$ chooses a random $x' \in GF(q)$ and outputs x'.

$\mathcal{A}'$ can compute the j-th backward key $BK_{m-j+1} = H^{m-j+1}(BK)$ because it knows BK from $View$ for $j = 1, \ldots, m$. Note that from $View$, $\mathcal{A}'$ knows $\{FK(h,1), \ldots, FK(h, s{-}1)\} \cup \{FK(h,t) \ldots, FK(h,m)\}$ and session keys $\{SK_1, \ldots, SK_{s-1}\} \cup \{SK_t, \ldots, SK_m\}$. Observe that $SK_{j'} = SK_j$ provided $\mathcal{A}'$ knows the forward key $FK(h, j)$. This condition holds if and only if the $\mathcal{A}'$ is able to invert the one-way function.

The above arguments show that if $\mathcal{A}'$ is successful in breaking the security of our construction, then $\mathcal{A}$ can break the one-way hash function. Hence, our construction is computationally secure for resisting R-coalition under the difficulty of breaking one-way hash function.

7.5.2 Proof under the Defined Security Model

We now demonstrate that our construction satisfies all the conditions required by *Definition 7.1* to *Definition 7.3*. We first show that our construction realizes self-healing key distribution scheme with revocation capability. More precisely, we can prove our construction under the security model described in subsection 7.2.

We now show that our construction satisfies all the conditions required by *Definition 7.1*.

1. The scheme is a session key distribution scheme.

   (a) A non-revoked user $U_i \in G_j$ can recover the session key SK_j efficiently from broadcast message B_j and personal key S_i. As can be seen in procedure *Key Recovery*.

   (b) For any set $R_j \subseteq U$, where $R_j \in R$ and $U_i \notin R_j$, we show that the coalition R_j knows nothing about the personal secret $(u_1 \cdot \psi(U_i), \ldots, u_j \cdot \psi(U_i), \ldots, u_m \cdot \psi(U_i))$ of U_i. For any session j, U_i uses its $u_j \cdot \psi(U_i)$ as its personal secret. Since the coalition $R_j \notin \Gamma$, the values $\{u_s \cdot \psi(U_k) : U_k \in R_j, 1 \leq s \leq m\}$ is not enough to compute $u_j \cdot \psi(U_i)$ by the property of ψ. So it is computationally infeasible for coalition in R_j to determine $u_j \cdot \psi(U_i)$ for $U_i \notin R_j$.

   (c) The session key SK_j for session j is computed from two parts: forward key $FK(h, j)$ and backward key BK_{m-j+1} where $FK(h, j)$ is or can be computed from the parts of personal key received from the GM before or when it joins the session group and $BK_{m-j+1} = z_j - u_j \cdot \psi(GM)$ is recovered from the broadcast message B_j. So the personal secret keys alone do not give any information about any session key. Since backward seed BK is chosen randomly, the backward key BK_{m-j+1} is random as well. Furthermore, the broadcast $B_1, \ldots, B_m$ determine $z_1, \ldots, z_m$ but the scalar $u_j \cdot \psi(GM)$ perfectly hide the backward key sequence because $z_j = BK_{m-j+1} + u_j \cdot \psi(GM)$. So it is computationally infeasible to determine session key K_j from broadcast message B_j or personal key S_i alone.

2. The scheme has R-revocation capability. For each session j, let $R_j \in R$ be a collection of revoked users collude. It is infeasible for the coalition R_j to compute the j-th session key SK_j because knowledge of K_j implies the knowledge of backward key BK_{m-j+1} or disclosing of personal secret $u_j \cdot \psi(U_i)$ of $U_i \notin R_j$. A user $U_i \in R_j$ knows, from the broadcast B_j, vector $\{(U_k, u_j \cdot \psi(U_k))\}_{U_k \in W_j \cup R_j}$ and its personal key $u_j \cdot \psi(U_i) \in GF(q)$ where $U_i \in W_j \cup R_j \notin \Gamma$, and this does get any information on $u_j \cdot \psi(GM)$. We reach this conclusion because for any scalar $s \in GF(q)$, there exists at least

one vector $u \in GF(q)^l$ such that

$$\begin{cases} u \cdot \psi(GM) = s \\ u \cdot \psi(U_k) = u_j \cdot \psi(U_k) \text{ for any } U_k \in W_j \cup R_j \end{cases} \tag{7.11}$$

because $W_j \cup R_j \notin \Gamma$. Notice that the number of vectors u satisfying this system of equations is independent of the value s.

3. The scheme has self-healing capability as can be seen from the procedure of *Self-healing*.

We will show that our construction satisfies the forward and backward secrecy required by *Definition 7.2*.

1. Only those sets belonging to Γ can recover the $\psi(GM)$ and further recover the session key SK_j. Because of the fact that $R_j \subseteq R \notin \Gamma$, the coalition of R cannot recover $\psi(GM)$. Furthermore, because of the one-way property of H, it is computationally infeasible to compute BK_s from BK_t for $s < t$. That is, even though the users might know the sequence of backward keys $BK_m, \ldots, BK_{m-j+2}$, still cannot compute backward key BK_{m-j+1} for session j and consequently SK_j from the sequence. Therefore, our construction guarantees the forward secrecy.

2. The correctness is based on the one-way property of the hash function. In order to know SK_s $(s < j)$, $U_l \in J_j$ requires the knowledge of the s-th forward key $FK(h, s)$. However, when a new user U_l joins the group starting from session j, the GM gives j-th forward key $FK(h, j)$ and the intermediate forward key seed corresponding to U_l's life cycle. Therefore, it is computationally infeasible for the newly joined users to trace back previous forward key $FK(h, s)$ for $s < j$. Hence, we can claim that our scheme maintains backward secrecy. In fact, the backward secrecy is independent of secret vectors.

We show our construction has the collusion resistance property required by *Definition 7.3*.

The scheme has the capability of collusion resistance. Let $B \subset R_s$ be a coalition of users removed from the group before session s and let $C \subset J_t$ be a coalition of users who join the group from session t with $s < t$ such that $B \cup C \notin \Gamma$. The secret information held by users in $B \cup C$ and broadcast in all the sessions do not receive any information about SK_j for sessions $j = s, \ldots, t-1$. This is true because they know, in the worst case, the coalition knows $S_i = (u_t \cdot \psi(U_i), \ldots, u_m \cdot \psi(U_i)) \in GF(q)^{m-t+1}$ for $U_i \in C$, $S_i = (u_1 \cdot \psi(U_i), \ldots, u_{s-1} \cdot \psi(U_i)) \in GF(q)^{m-t+1}$ for $U_i \in B$, and $B_1, \ldots, B_m$. For each session $s \leq j \leq t-1$, the coalition can obtain backward key BK_j from C, However, the coalition cannot receive forward key $FK(h, j)$. The session key SK_j is computed from forward key and backward key. The coalition $B \cup C$ cannot obtain the forward key. it is easy to see that all the guesses for SK_j with $s \leq j \leq t-1$ have the same probability.

7.6 Performance Evaluation

7.6.1 Storage Overhead

For each user U_i whose life cycle is (s_i, t_i), the storage overhead comes from its personal key. We set $k_i = t_i - s_i + 1$ which is the number of sessions that U_i attends. The storage overhead for mask vectors is $k_i logq$ and the storage overhead for key seeds varies with different s_i and t_i. In the best circumstances, s_i and t_i are the starting point and the end point of a subtree, respectively. U_i has to store only one intermediate forward key seed. The worst condition occurs when s_i and t_i are separate nodes in the HBT. U_i has to store $2\lceil logk_i \rceil - 4$ intermediate key seeds and 2 forward keys. The total storage overhead for key seeds is $(2\lceil logk_i \rceil - 2)logq$ bits. Compared to schemes in [7, 93], our scheme reduces storage overhead for forward keys at U_i to logarithm level of U_i's life cycle. We take a HBT with 8 leaf nodes as examples (As shown in Figure 7.2). If U_i's life cycles is $(1, 4)$, U_i only stores intermediate key seed $FK(1, 1)$. In an extreme condition, U_i's life cycle is $(1, 8)$, the root seed $FK(0, 1)$ is enough for U_i to calculate all the forward keys. The worst condition occurs when U_i's life cycle is $(2, 7)$, U_i has to store $2(\lceil log[(7-2+1)/2] \rceil)logq = 2(\lceil log(7-2+1) \rceil - 1)logq = 4logq$ bits. It is the total length of 2 intermediate key seeds, $FK(2, 2)$ and $FK(2, 3)$, and two forward keys, $FK(3, 2)$ and $FK(3, 7)$.

7.6.2 Communication Overhead

The communication overhead is measured by the size of the message. The broadcast message B_j at the j-th session consists of two parts. The first part is a masked vector. The second part is a set of identities of revoked users and their vectors corresponding to session j. One can ignore the communication overhead for the set of identities, because the user identities can be selected from a small finite field [81]. In our scheme, the length of vectors depends on the particular function ψ which is related to the set $W_j \cup R_j$. We set $t_j = |W_j \cup R_j|$, where $W_j \subset U \setminus G_j$ and $R_j \notin \Gamma$ is the set of all revoked users for sessions in and before j with minimum cardinality such that $W_j \cup R_j \in \bar{\Gamma}_0$. The communication cost for session j is $(t_j + 1)logq$ bits.

Table 7.2: Performance comparison among the constructions in [7] and our construction in the j-th session

Scheme	Performance measure		
	Storage overhead	Communication overhead	Computation overhead
[93]	$(2k_i + 1)logq$	$(t_j + 1)logq$	$2(t_j^2 + t_j)$
[7]	$2k_i logq$	$(t_j + 1)logq$	$2(t_j^2 + t_j)$
Our construction	$(2\lceil logk_i \rceil - 2)logq$	$(t_j + 1)logq$	$2(t_j^2 + t_j)$

7.6.3 Computation Overhead

The computation cost comes from the calculation during *Key Recovery*. The calculation includes multiplication over the finite field $GF(q)$, hash operation, and addition. We ignore computation cost for the addition and hash operations and measure the computation overhead by the number of multiplication operations. The measurement is held because the computation cost produced by the addition and hash operation is negligible compared with that for multiplication. Even though our scheme achieves implicit authentication by the hash method and is more secure, its computation overhead can be ignored. The HBT-based scheme requires fewer hash operations for forward key computation, the minimum number of hash operation is 0, while the maximum number of hash operation is $\lceil log_2 m \rceil$ which is the depth of HBT. In general hash chain based schemes [86, 93, 95], the minimum number of hash operations for forward key recovery is 0, while the maximum number of hash operation is $(m-1)$. The authorized users recover $\psi(GM)$ from the set $W_j \cup R_j$ and the corresponding vectors as well as its personal identity and vector. The computation overhead of our construction is $2(t_j^2 + t_j)$ multiplications which is the same as that of [7].

In Table 7.2, we summarize a comparison of the proposed scheme with the schemes in [7, 93] which reach the same security level.

7.7 Conclusion

An implicitly authenticated self-healing key distribution scheme is proposed in this chapter. We adopt a HBT-based technique rather than a simple hash chain. The HBT-based technique helps to reduce storage overhead and computation overhead brought by *Key Recovery*. The scheme achieves more flexible access structure than that of [112] with vector space secret sharing based access structure. The proposed scheme not only maintains forward and backward secrecy, but it also has a collusion resistance property. Implicit authentication which is used to detect tamper attack against broadcast messages strengths the security of the scheme. Performance analysis shows that the scheme is lightweight in storage and computation overhead without degrading security.

Part III

PKC-based Solutions

8

Mutual-healing Key Distribution and Authentication on Broadcast

8.1 Introduction

Key management including key distribution and key update is significant for maintaining private communications in any dynamic networks [51, 62, 80, 81, 82, 94, 142, 143, 144, 145, 146, 147, 148, 149]. WSNs, being dynamic networks, therefore rely on a secure key management. This is because membership frequently changes in large dynamic group communications of WSNs. In order to maintain the security of communications, the session key has to be updated with each membership change. Therefore, one of the important questions is how to distribute and update session keys in a secure and efficient way for large dynamic WSNs. In recent years, many schemes for distributing session keys for large group communications have been proposed. According to the literature review in Chapter 2, we know there are different key updating mechanism. For example, LKH-based schemes [150] and OFT-based schemes [62, 142] aim to reduce the size of the rekeying message. Broadcast encryption addresses the problem of sending encrypted messages to a large node group so that the encrypted messages can only be decrypted by a dynamic changing privileged subset [103, 151, 152]. The EBS-based approach was proposed in [52], and then utilized for wireless sensor networks in [51, 153]. The members store fewer keys than does the LKH tree for the multicast group of the same size. All these works in the literature have assumed that the underlying networks are reliable. However, how to distribute session keys for unreliable wireless networks, in a manner that is resistant to packet loss, is an issue that has not been addressed adequately.

Packet loss happens frequently in WSNs. The key distribution broadcast for a particular session might never reach some nodes. A simple solution is to request re-transmission. On the one hand, both the requesting and re-transmission messages would incur more communication overhead. In a very large communication group, such individual interactions place a heavy burden on the group manager. On the other hand, nodes may reveal their current locations by sending messages within some high security environments. All these issues can be addressed by a self-healing key distribution mechanism. With the self-healing mechanism, nodes do not need to send any requesting message to the group manager and do not need to update their personal keys. In this regard, self-healing key distribution schemes are non-interactive and therefore can reduce the network traffic, decrease the workload on the group manager, and lower the risk of node exposure through traffic analysis. Therefore, self-healing key distribution schemes are desirable for reasons of both efficiency and security in WSNs.

However, a node in a WSN may miss its last broadcast message. This node is therefore not able to recover its session key for the last session in a self-healing key distribution scheme. This is because self-healing key distribution needs the broadcast messages the node received in the previous sessions and those in the subsequent sessions (assuming that the current session is the one where broadcast messages are lost). To tackle this scenario, a mutual-healing key distribution mechanism will be introduced. Mutual-healing key distribution can help a node which missed the last broadcast message to recover the session key for this last session. Recalling information given in previous chapters, broadcasts transferred over a wireless channel are vulnerable to various attacks. Hence, authentication is necessary to ensure the origin, integrity, and freshness of broadcasts.

The rest of this chapter is organized as follows: In Section 8.2, we present the motivation for the technical exploration of the mutual-healing mechanism and authenticated self-healing key distribution. In Section 8.3, we explain the contribution of the two schemes proposed in this chapter. In Section 8.4, we briefly introduce identity-based cryptography and digital signature based on pairings. This knowledge facilitates understanding of the proposed solutions. In Section 8.5, we present system parameters, a security model and a formal definition of mutual-healing key distribution. In Section 8.6, a construction for mutual-healing key distribution is proposed. In Sections 8.7 and 8.8, we provide security evaluation and performance evaluation, respectively. In Section 8.9, we propose a construction of authenticated self-healing key distribution. In Section 8.10 and Section 8.11, we evaluate the security and performance of the authenticated self-healing key distribution scheme, respectively. In Section 8.12, we conclude this chapter.

8.2 The Motivation of the Proposed Schemes

8.2.1 The Motivation of the Technical Exploration of Mutual-healing Mechanism

More *et al.* in [89] pointed out that the protocol in [80] suffers from inconsistent robustness in that some session keys cannot be recovered if the corresponding broadcast messages is lost, no matter how many other update messages are received. For example, if the broadcast message for the last session gets lost, nodes cannot recover the last session key by themselves even though they receive all the other broadcast messages. Subsequently, they used a sliding window to make error recovery consistently robust. That is, after the initial *Setup* procedure, any lost key can be recovered as long as two sufficiently close broadcast messages (one before it and the other after it) are received. A similar technique was used in [88]. The size of the window can be dynamically adjusted according to the condition of networks. Both [89] and [88] guarantee that authorized nodes can recover window size session keys as long as they receive corresponding broadcast messages. However, how to recover the session key if the last broadcast message is lost or more than the number of sliding window broadcast messages are lost, has never been addressed clearly. Obviously, it is impossible to make nodes completely self-healing according to existing self-healing key distribution mechanisms.

Moreover, there are concrete applications, such as live and pay-per-view TV, that have a strict requirement of freshness, where it is desirable that customers lose a minimal number of broadcast messages. In group communication, the last session is usually of great importance. The authorized nodes should never miss it. Therefore, it is important to design counteracting measures to deal with the aforementioned issues.

Bohio *et al.* in [97] considered the idea of mutual-healing so that if a node has missed more than a fixed number of broadcast messages, it does not have to keep on waiting. Instead, it can obtain assistance from its neighbors. With the help of its neighbors, the node can recover the missed session key and further recover the encrypted content before it becomes outdated. Similarly, if a node misses the last broadcast message, it cannot recover the last session key by performing self-healing. To provide a countermeasure for this situation, it can look for assistance from its neighboring nodes. However, the paper [97] dealt only with the feasibility of mutual-healing without providing any technical details.

8.2.2 The Motivation of Authentication on Broadcast

Broadcasts are vulnerable to various types of attacks due to the open nature of WSNs. Authentication ensures the origin of communication entities and integrity of communication content. Authentication is vital in WSNs. However, it is by no means a simple task to authenticate broadcasts with general self-healing key distribution schemes, excluding hash chain based schemes. Even if we can preload sensor nodes with a network-wide key for authentication, the renewal may happen frequently due to dynamic addition or revocation of nodes. The renewal of the network-wide key places a heavy burden on the network. We have proposed an implicit authentication solution to hash chain based self-healing key distribution schemes in Chapters 6 and 7. As is well known, authentication based on PKC has an advantage over that based on Symmetric Key Cryptography because no secret key has to be shared by the entities. Motivated by this advantage, we consider implementing authentication in bilinear pairings based self-healing key distribution.

8.3 The Contribution of the Proposed Schemes

We have narrowed the study of mutual-healing key distribution to effectively authenticate requesting nodes in Chapter 3. In this chapter, we formally address the mutual-healing key distribution. We define a security model of a computationally secure mutual-healing key distribution scheme. This security model outperforms those in [80, 81] in two aspects. The first one is that there is no threshold for the number of revoked nodes. The scheme is collusion-free for any coalition of non-authorized nodes. The second one is that a node can recover from a single broadcast message all keys associated with sessions in which it belongs to the session group. These two properties add flexibility to the self-healing key distribution scheme. We propose a self-healing key distribution scheme using bilinear pairings. The scheme is characterized by several desirable features. Firstly, it is collusion-free for any coalition of non-authorized nodes. Secondly, the private key has nothing to do with the number of revoked nodes and can be reused as long as it is not disclosed. Thirdly, the storage overhead for nodes is a constant. Subsequently, we discuss the requirements, provide a formal definition, and develop the technical details of the mutual-healing mechanism. To the best of our knowledge, it is the first time that a formal security model of mutual-healing has been presented. It is also the first time that a mutual-healing key distribution technique has been developed.

In this chapter, we implement bilinear pairings based authenticated self-healing key distribution scheme. The authentication method takes advantage of the public key pair of the group manager. It is the first time that the authentication on self-healing key distribution broadcasts using an short digital signature has been addressed. The security of our scheme relies on the hardness of the Diffie-Hellman problem in the random oracle model.

8.4 Identity-based Cryptography and Digital Signature Based on Pairings

8.4.1 Identity-based Cryptography

In identity-based cryptography, the public key of a user is some unique information about the identity of the user. Identity-based schemes can allow any party to generate a public key from a known identity value such as an ASCII string. A trusted third party, called PKG, generates the corresponding private keys. Therefore, any party without using certificates can verify the public key of a user. This eliminates the need for a public key distribution infrastructure.

Shamir [104] proposed the first identity-based cryptography to alleviate many of the problems inherent in managing certificates in 1985. Boneh and Franklin [105] proposed the first practical identity-based encryption scheme in 2001. Since then, many ID-based cryptographic schemes have been proposed using bilinear pairings. Inspired by the idea of [105], Du *et al.* proposed a broadcast encryption scheme for key distribution in [106]. By extending the broadcast encryption scheme, we propose a mutual-healing key distribution scheme for wireless sensor networks.

8.4.2 Digital Signature Based on Pairings

Digital signature is essential to ensure the authenticity of messages in low-bandwidth communication, low-storage and less computation networks. Compared with the traditional Public Key Cryptography, the extinct advantage of ID-based cryptography is the elimination of computation and memory requirements for certificate management. Many ID-based digital signature schemes based on bilinear pairings have been proposed so far. Cha *et al.* in [154] proposed an ID-based signature scheme using gap Diffie-Hellman groups. Paterson *et al.* in [155] proposed an ID-based signature scheme using the same computational primitives as the ID-based encryption scheme of [105]. This scheme is more efficient than [156], which is also claimed to be an efficient digital signature scheme. Hess in [157] designed an efficient ID-based signature scheme based on pairings which offered advantages over [154, 155, 156]. The security of the scheme relies on the hardness of the Diffie-Hellman problem based on bilinear pairings. Our authenticated self-healing key distribution scheme adopts this efficient digital signature.

8.4.3 An Efficient Digital Signature Scheme

Here we briefly review the efficient digital signature scheme of [157]. Suppose the system parameters are $params = \{G_1, G_2, q, P, P_{pub}, H_1, H_2\}$ (refer to Section 3.2.20 for parameters definition). The public/private key pair of the signer is (Q_{ID}, S_{ID}). All the parameters are defined as aforementioned in ***ID-based Public Key Infrastructure***. A hash function $H_3 : \{0,1\}^* \times G_2 \to (\mathbb{Z}/q\mathbb{Z})^\times$ is available for the signer and verifiers.

Sign: To sign a message m the signer chooses an arbitrary $P_1 \in G_1^*$ where $G_1^* = G_1 \setminus \{0\}$, picks a random integer $k \in (\mathbb{Z}/q\mathbb{Z})^\times$ and computes:

- $r = e(P_1, P)^k$;
- $v = H_3(m, r)$;
- $u = vS_{ID} + kP_1$.

The signature is the pair $(u, v) \in (G_1, (\mathbb{Z}/q\mathbb{Z})^\times)$.

Verify: On receiving a message m and signature (u, v), the verifier computes:

- $r = e(u, P) \cdot e(H_1(ID), -P_{pub})^v$;
- Accept the signature if and only if $v = H_3(m, r)$.

8.5 Security Model and Definition for Mutual-healing Key Distribution

In this section, we present system parameters, a security model and a formal definition for mutual-healing key distribution.

8.5.1 System Parameters

Let $U = \{u_1, \ldots, u_n\} (n > 2)$ be the finite universe of nodes. Each node u_i has a unique identifier ID_i. A broadcast unreliable channel is available, and time is defined by a global clock. A GM sets up and manages, by means of adding and revoking operations, a communication group which is a dynamic subset of U. m denotes the number of sessions. Let $G_j \subseteq U$ be the communication group established by the group manager in session j. Each node is preloaded with a public/private key pair (Q_i, S_i) before deployment. The public/private key pair is used to recover the session keys as long as node u_i is not removed by the GM from the group. Let $R_j \subseteq G_{j-1}$ denote the set of revoked group nodes in session j and $J_j \subset U \setminus G_{j-1}$ denote the set of nodes who join the group in session j with $R_j \cap J_j = \phi$. Hence, $G_j = (G_{j-1} \cup J_j) \setminus R_j$ for $j \geq 2$ and by definition $G_1 = U$. Moreover, for $j \in \{1, \ldots, m\}$, the session key K_j is randomly chosen by the GM according to the uniform distribution. For any non-revoked node $u_i \in G_j$, the j-th session key K_j is determined by the broadcast message B_j and the personal public/private key pair (Q_i, S_i).

8.5.2 Security Model

Just as self-healing means that nodes are capable of recovering lost group keys on their own, mutual-healing implies that nodes help each other to recover some lost group keys. The central concept of mutual-healing is that if a node has missed more than a fixed number of broadcast messages or the last broadcast message, it can obtain assistance from its neighboring nodes. The neighboring nodes within the same session group cooperate with each other by forwarding broadcast messages which their neighboring nodes have missed. In this way, the robustness of the self-healing key distribution mechanism is achieved.

To further clarify our design goal and facilitate understanding of the proposed solution, according to but not constrained by the security model of [86], we define the mutual-healing key distribution security model from four perspectives. *Definition 8.1* defines self-healing key distribution scheme with revocation capability. *Definition 8.2* defines mutual-healing key distribution scheme *Definition 8.3* defines forward secrecy and backward secrecy. *Definition 8.4* defines collusion resistance properties.

Definition 8.1. Let $U = \{u_1, \ldots, u_n\}$ and $j \in \{1, \ldots, m\}$.

1. The scheme is a session key distribution with privacy if

   (a) for any node u_i, the session key K_j is efficiently determined from B_j and the personal public/private key pair (Q_i, S_i).

(b) for any set $R \subseteq U$ and $u_i \notin R$, it is computationally infeasible for node in R to determine the personal private key S_i.

(c) what node $\{u_1, \ldots, u_n\}$ learn from B_j cannot be determined from broadcasts or personal key pairs alone. That is, if we consider separately either the set of m broadcasts $\{B_1, \ldots, B_m\}$ or the set of n personal key pairs $\{(Q_1, S_1), \ldots, (Q_n, S_n)\}$, then it is computationally infeasible to compute session key K_j from either set.

2. The scheme has revocation capability. In particular, there is no upper limitation of revocation. For each session j and $R_j \subset U$, the GM can generate a broadcast message B_j such that for all $u_i \notin R_j$ can efficiently recover the session key K_j, but the revoked nodes in R_j cannot even know all the information broadcast in sessions $1, \ldots, j$.

3. The scheme is self-healing if the following is true for any r, $1 \leq r < j \leq m$: For any node $u_i \in G_r$ who is also a member in session j, the session key K_r is efficiently determined by (Q_i, S_i) and B_j.

Definition 8.2. Let $U = \{u_1, \ldots, u_n\}$ and $j \in \{1, \ldots, m\}$. The scheme is mutual-healing if the following is true:

1. For any node $u_i \in U$, if it misses more than a fixed number of broadcast messages or the last broadcast message, it can generate and broadcast an effective requesting message to its neighbors.

2. For any u_i's neighboring node u_j, it can verify whether u_i is its qualified neighboring node or a malicious one. If u_i is a qualified neighboring node and u_j is an authorized node for the requested broadcast, u_j generates and sends a responsive message to u_i.

3. The requester u_i can verify whether or not the responser u_j is its neighbor. If u_j is its neighbor, u_i can decrypt the responsive message and thus obtain the requested broadcast message.

Definition 8.3. Let $U = \{u_1, \ldots, u_n\}$ and $j \in \{1, \ldots, m\}$. The scheme guarantees both forward secrecy and backward secrecy if:

1. for any set $R \subseteq U$, and all $u_i \in R$ are revoked before session j, it is computationally infeasible for the nodes in R together to obtain any information about K_j, even with the knowledge of group keys $K_1, \ldots, K_{j-1}$ before session j; and
2. for any set $J \subseteq U$, and all $u_i \in J$ join after session j, it is computationally infeasible for the nodes in J together to get any information about K_j, even with the knowledge of group keys $K_{j+1}, \ldots, K_m$ after session j.

Definition 8.4. Let $B \subset R_r \cup \ldots \cup R_2$ be a coalition of nodes who are revoked from the group before session r and let $C \subset J_s \cup \ldots \cup J_m$ be a coalition of nodes who join the group from session s with $r < s$.

The scheme is collusion-free for any coalition of non-authorized nodes if the coalition $B \cup C$ does not obtain any information about session keys K_j, for any $r \leq j < s$.

8.6 The Construction of Mutual-healing Key Distribution Scheme

In this section, we will present a mutual-healing key distribution scheme for WSNs. The proposed mutual-healing key distribution scheme has the following procedures:

- *Setup*;
- *Broadcast*;
- *Key Recovery*;
- *Self-healing*;
- *Mutual-healing*; and
- *Adding/Revoking Nodes*.

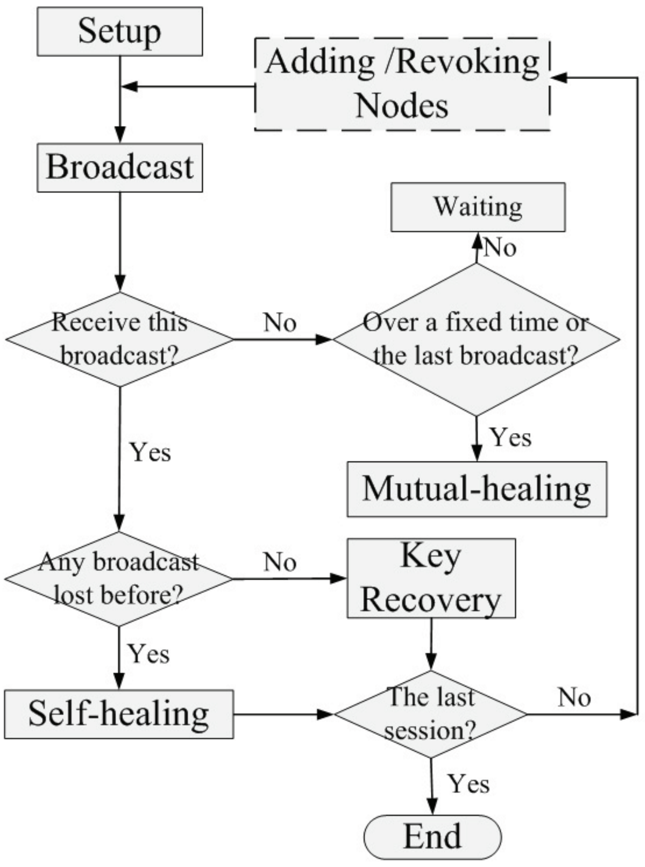


Figure 8.1: The process of the mutual-healing key distribution scheme.

The process of the proposed mutual-healing key distribution scheme is shown in Figure 8.1, where the panes represent operations which must be executed in each round, and the dashed panes represent the operations which may not be executed in every round.

The proposed mutual-healing scheme is computationally secure. All parameters and symbols used in this section have been defined in Sections 8.4 and 8.5. Otherwise, they will be defined in this section.

Setup. The GM obtains both public system parameters and all the public keys of possible nodes from the ID-based PKI. The GM chooses m session keys $K_1, \cdots, K_m$ from $\mathbb{Z}_q^*$. The session keys are independent of each other and are according to the uniform distribution. We also use the system parameters proposed in Section 8.5.

Broadcast. Suppose $|G_j|$ denotes the number of nodes in session j. For each session $1 \leq j \leq m$, according to the session group G_j, the GM computes $Q_{V_1} = \sum_{i=1}^{n} Q_i$ and a $(|G_j| - 1) \times |G_j|$ matrix which is defined as follows:

$$\begin{pmatrix} a_2 \\ a_3 \\ \vdots \\ a_{|G_j|} \end{pmatrix} = \begin{pmatrix} 1\ 1\ 0 \ldots 0 \\ 1\ 0\ 1 \ldots 0 \\ \vdots\ \vdots\ \vdots\ \ddots\ 0 \\ 1\ 0\ 0 \ldots 1 \end{pmatrix} \tag{8.1}$$

Let $a_i^{'}$ represents the transposition of a_i. The GM also constructs $|G_j| - 1$ auxiliary keys

$$Q_{V_i} = (Q_1, Q_2, \ldots, Q_{|G_j|}) \times a_i^{'} \quad 2 \leq i \leq |G_j|, \tag{8.2}$$

which means $Q_{V_2} = Q_1 + Q_2$, $Q_{V_3} = Q_1 + Q_3$, $\ldots$, $Q_{V_{|G_j|}} = Q_1 + Q_{|G_j|}$. The broadcast message is then formed by computing, for a random $r_j \in \mathbb{Z}_q^*$,

$$U_1 = r_j P, \quad U_i = r_j Q_{V_i} \quad 2 \leq i \leq |G_j|, \tag{8.3}$$

where P is a generator for a BDH group G_1 as shown in Chapter 3.

We define $V_j = K_j \oplus H_2(e(P_{pub}, r_j Q_{V_1}))$. Let $z_j = (U_i(1 \leq i \leq |G_j|), V_j)$. The GM broadcasts the ciphertext to the set of nodes G_j. The ciphertext for the j-th broadcast is in the following form: $B_j = \{z_1, \ldots, z_j\}$.

Key Recovery. When a node $u_i \in G_j$ receives the broadcast message B_j, it sets a vector $a_1 = (0, \ldots, 0, 1, 0, \ldots, 0)$ with $|G_j|$ elements, and only the i-th element is *1*. Then A_j is a $|G_j| \times |G_j|$ matrix

$$A_j = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{|G_j|} \end{pmatrix}. \tag{8.4}$$

The node u_i can solve the following system of equations using Cramer's Rule or other algebraic methods.

$$(x_1, x_2, \ldots, x_{|G_j|}) \times A_j = (1, 1, \ldots, 1). \tag{8.5}$$

With $(x_1, x_2, \ldots, x_{|G_j|})$, u_i gets

$$(x_1, x_2, \ldots, x_{|G_j|}) \times \begin{pmatrix} Q_i \\ Q_{V_2} \\ \vdots \\ Q_{V_{|G_j|}} \end{pmatrix} = Q_{V_1}. \tag{8.6}$$

In order to decrypt the ciphertext, u_i needs to compute $e(P_{pub}, rQ_{V_1})$. With the knowledge of the private key S_i, it can do so via:

$$\begin{aligned} & e(P_{pub}, r_j Q_{V_1}) \\ & = e(P_{pub}, r_j(x_1 Q_i + x_2 Q_{V_2} + \ldots + x_{|G_j|} Q_{V_{|G_j|}})) \\ & = e(P_{pub}, r_j x_1 Q_i) \cdot e(P_{pub}, r_j(x_2 Q_{V_2} + \ldots + x_{|G_j|} Q_{V_{|G_j|}})) \\ & = e(r_j P, x_1 s Q_i) \cdot e(P_{pub}, x_2 r_j Q_{V_2} + \ldots + x_{|G_j|} r_j Q_{V_{|G_j|}}) \\ & = e(U_1, x_1 S_i) \cdot e(P_{pub}, x_2 U_2 + \ldots + x_{|G_j|} U_{|G_j|}). \end{aligned} \tag{8.7}$$

Then, u_i can recover the session key

$$K_j = V_j \oplus H_2(e(U_1, x_1 S_i) \cdot e(P_{pub}, \sum_{i=2}^{|G_j|} x_i U_i)). \tag{8.8}$$

Self-healing. Without loss of generality, suppose u_i lost the broadcast message for a session $t < j$. As long as it belongs to the session group G_t, it can pick up the polynomial z_t from the broadcast message B_j and forms the $|G_t| \times |G_t|$ matrix A_t as operations in the procedure of *Key Recovery*. Then u_i solves the following system of equations:

$$(x_1, x_2, \ldots, x_{|G_t|}) \times A_t = (1, 1, \ldots, 1). \tag{8.9}$$

With $(x_1, x_2, \ldots, x_{|G_t|})$, u_i gets

$$(x_1, x_2, \ldots, x_{|G_t|}) \times \begin{pmatrix} Q_i \\ Q_{V_2} \\ \vdots \\ Q_{V_{|G_t|}} \end{pmatrix} = Q_{V_1}. \tag{8.10}$$

After that, with the knowledge of its private key S_i, u_i computes $e(P_{pub}, r_t Q_{V_1})$ as follows:

$$\begin{aligned} & e(P_{pub}, r_t Q_{V_1}) \\ &\quad = e(P_{pub}, r_t(x_1 Q_i + x_2 Q_{V_2} + \ldots + x_{|G_t|} Q_{V_{|G_t|}})) \\ &\quad = e(P_{pub}, r_t x_1 Q_i) \cdot e(P_{pub}, r_t(x_2 Q_{V_2} + \ldots + x_{|G_t|} Q_{V_{|G_t|}})) \\ &\quad = e(r_t P, x_1 s Q_i) \cdot e(P_{pub}, x_2 r_t Q_{V_2} + \ldots + x_{|G_t|} r_t Q_{V_{|G_t|}}) \\ &\quad = e(U_1, x_1 S_i) \cdot e(P_{pub}, x_2 U_2 + \ldots + x_{|G_t|} U_{|G_t|}). \end{aligned} \tag{8.11}$$

Finally, u_i recovers the lost session key

$$K_t = V_t \oplus H_2(e(U_1, x_1 S_i) \cdot e(P_{pub}, \sum_{i=2}^{|G_t|} x_i U_i)) \tag{8.12}$$

One may wonder how u_i obtains V_t if the *t*-th broadcast message is lost. In fact, u_i could get the broadcast message B_j. Then u_i retrieved the polynomial z_j from B_j. Finally, u_i could obtain V_t from z_j. If more than one broadcast message is lost, the operations of session key recovery are the same as those aforementioned.

Mutual-healing. Here we consider the mutual-healing between neighboring nodes in wireless communication networks and present a practical technique to realize it.

Many wireless networks have an intrinsic property nodes are stationary. Therefore, we can bind the Location-based Keys (LBKs) of nodes to both their identities and geographic locations rather merely their identities or locations as in conventional schemes. Based on their LBKs, two neighboring nodes can perform node-to-node neighborhood authentication. In order to reduce communication overhead, we impose a restriction whereby mutual-healing happens only between one-hop neighboring nodes. In order to realize mutual-healing capability, the scheme has to execute a range-based location operation after the *Setup* procedure.

There are many methods for localizing nodes. We adopt the first method in [21]. This step can be completed within several minutes after the deployment of networks. We assume that a group of mobile robots are dispatched to sweep across the whole network field along pre-planned routes. Mobile robots have capability of positioning, as well as more powerful computation and communication capacities than ordinary nodes have. The leading robot equipped with a master secret k. To localize a node, say u_i, mobile robots run the secure range-based localization protocol given in [158] or [159] to measure their respective absolute distance to node u_i and co-determine l_i, the location of u_i. Subsequently, the leading robot calculates $LK_i = kH(ID_i \parallel l_i)$, and sends $(\{LK_i \parallel l_i\}_{Q_i}, h_{Q_i}(LK_i \parallel l_i))$ to u_i. $\{M\}_k$ means encrypting message M with key k, and $h_k(M)$ refers to the Message Integrity Code (MIC) of message M under key k.

Upon receipt of the message, node u_i first uses its private key to decrypt LK_i and l_i and then regenerates the MIC. If the result matches what the robot sent, u_i saves LK_i and l_i for subsequent use. Following this process, all the nodes can be furnished with their respective locations and LBKs. After that, mobile robots leave the sensing field and the leading robot has to securely erase k from its memory. During subsequent network operations, the addition of nodes may be necessary in order to maintain good network connectivity. The localization of new nodes can be done in the same manner.

It is generally supposed that adversaries do not launch active and explicit pinpoint attack on nodes during deployment and initialization which usually does not last too long. According to [21], this assumption in range-based location operations is reasonable in that mobile robots are much fewer than ordinary nodes and can be equipped with tamper-proof hardware and be placed under a super monitor.

During the procedures of self-healing key distribution, if a node has missed more than a fixed number of broadcast messages or the last broadcast message, it looks for assistance from its neighboring nodes.The mutual-healing process includes three steps. We will introduce them one by one.

1. *Mutual-healing request.* Suppose a node u_i wishes to receive broadcast message B_t, u_i locally broadcasts an authentication request including its identity ID_i, location l_i and the sequence number of the expected broadcast message t.

2. *Mutual-healing response.* Upon receipt of a request, the neighboring node u_j first needs to ascertain that the claimed location l_i is within its one-hop communication range by verifying if the Euclidean distance $\parallel l_i - l_j \parallel \leq \mathcal{R}$, where $\mathcal{R}$ is one-hop communication distance.

   If the inequality does not hold, node u_j simply discards the request. Otherwise, u_j calculates a shared key as $K_{ji} = e(LK_j, H(ID_i \parallel l_i))$. Then it unicasts a reply to node u_i including its identity ID_j, location l_j and encrypted broadcast message $(B_t)_{K_{ji}}$.

   - $u_i \to * : ID_i, l_i, t$;
   - $u_j \to u_i : ID_j, l_j, (B_t)_{K_{ji}}$;

3. *Verification.* Upon receiving the response, node u_i also first checks if the inequality $\parallel l_i - l_j \parallel \leq \mathcal{R}$ holds. If the inequality does not hold, u_i directly discards the message received from u_j. Otherwise, u_i proceeds to derive a shared key as $K_{ij} = e(LK_i, H(ID_j \parallel l_j)) = K_{ji}$ between it and the node u_j whereby to decrypt the message $(B_t)_{K_{ji}}$ and obtain the broadcast message B_t. Using the broadcast message B_t and its public/private key pair, the authorized node u_i can recover the lost session keys.

Adding/Revoking Nodes. If a new node u_{new} applies for joining the session j, the GM firstly checks the validity of its identity. If it is an authorized node, in the procedure of *Broadcast*, the GM constructs a new $(|G_j| - 1) \times |G_j|$ matrix and computes new Q_{V_i} $(1 \leq i \leq |G_j|)$ which should include Q_{new}.

If a node u_{rov} is revoked from the session j, what the GM should do is construct a new $(|G_j| - 1) \times |G_j|$ matrix and computing new Q_{V_i} $(1 \leq i \leq |G_j|)$ which should exclude Q_{rov}.

The adding and revoking operations are very efficient in our scheme. If adding or revoking more than one node, the operations are the same as those aforementioned.

8.7 Security Evaluation of the Mutual-healing Key Distribution Scheme

Security analysis for the proposed mutual-healing key distribution scheme will be provided in this section. More precisely, we show that our construction satisfies all the security requirements in our security model described in Section 8.5.

8.7.1 Property 1: Self-healing Key Distribution

We show our construction satisfies the security requirements described in *Definition 4.1*.

1. The scheme is a session key distribution scheme. (a) Any node $u_i \in G_j$ can recover the session key K_j from B_j and the personal public/private key pair (Q_i, S_i). This is because when a node $u_i \in G_j$ receives the broadcast message B_j, it sets a vector $c_1 = (0, \ldots, 0, 1, 0, \ldots, 0)$ with $|G_j|$ elements, and only the i-th element is 1. Define a new matrix C_j using c_i with $i = 1, 2, ..., |G_j|$. Then C_j is a $|G_j| \times |G_j|$ matrix

$$C_j = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{|G_j|} \end{pmatrix}. \tag{8.13}$$

The node u_i can solve the following system of equations using Cramer's Rule or other algebraic methods.

$$(x_1, x_2, \ldots, x_{|G_j|}) \times C_j = (1, 1, \ldots, 1). \tag{8.14}$$

With $(x_1, x_2, \ldots, x_{|G_j|})$, u_i gets

$$(x_1, x_2, \ldots, x_{|G_j|}) \times \begin{pmatrix} Q_i \\ Q_{V_2} \\ \vdots \\ Q_{V_{|G_j|}} \end{pmatrix} = Q_{V_1}. \tag{8.15}$$

In order to decrypt the ciphertext, u_i needs to compute $e(P_{pub}, rQ_{V_1})$. With the knowledge of the private key S_i, u_i can do so via:

$$\begin{aligned} & e(P_{pub}, r_jQ_{V_1}) \\ & = e(P_{pub}, r_j(x_1Q_i + x_2Q_{V_2} + \ldots + x_{|G_j|}Q_{V_{|G_j|}})) \\ & = e(P_{pub}, r_jx_1Q_i) \cdot e(P_{pub}, r_j(x_2Q_{V_2} + \ldots + x_{|G_j|}Q_{V_{|G_j|}})) \\ & = e(r_jP, x_1sQ_i) \cdot e(P_{pub}, x_2r_jQ_{V_2} + \ldots + x_{|G_j|}r_jQ_{V_{|G_j|}}) \\ & = e(U_1, x_1S_i) \cdot e(P_{pub}, x_2U_2 + \ldots + x_{|G_j|}U_{|G_j|}). \end{aligned} \tag{8.16}$$

Then, u_i can recover the session key K_j by

$$K_j = V_j \oplus H_2(e(U_1, x_1S_i) \cdot e(P_{pub}, \sum_{i=2}^{|G_j|} x_iU_i)). \tag{8.17}$$

(b) Any coalition $R \subseteq U$ of non-authorized nodes cannot derive the private key S_i of any authorized node $u_i \notin R$. This is because each node is preloaded with a public/private key pair (Q_i, S_i) before deployment. The key pairs are computed by an ID-based PKI which can realize public and private keys without certificate management [160]. Because it is infeasible to solve the Discrete Logarithm Problem $S_{ID} = sQ_{ID}$, any coalition $R \subseteq U$ of non-authorized nodes cannot derive the private key S_i of any authorized node $u_i \notin R$.

(c) It is computationally infeasible to compute session key K_j from either broadcast messages or personal public/private key pairs. This is because the j-th session key is computed from V_j and $e(P_{pub}, r_jQ_{V_1})$. On the one hand, V_j is taken from the broadcast message while $e(P_{pub}, r_jQ_{V_1}) = e(U_1, x_1S_i) \cdot e(P_{pub}, \sum_{i=2}^{|G_j|} x_iU_i)$. Therefore, only the authorized nodes who holds corresponding private key S_i can recover the session key. They cannot get any session key only from the broadcast messages. On the other hand, the personal public/private key pairs of nodes are computed by the ID-based PKI. The session keys are chosen by the GM. They are independent of one another. Therefore, The coalition of nodes cannot obtain any session key except through their public/private key pairs $\{(Q_1, S_1), \ldots, (Q_n, S_n)\}$.

2. There is no threshold for the revocation in the proposed scheme. This means any number of nodes who are compromised or malicious can be revoked and, although they work together, they cannot work out the private key of any non-revoked authorized node. In our scheme, the broadcast messages are computationally related to the authorized nodes' public/private key pairs. On the one hand, only those nodes which hold the corresponding private keys can recover the session keys from the masked broadcast messages. On the other hand, even though all the revoked unauthorized nodes work together, they cannot obtain the private key of any non-revoked authorized node due to the difficulty of solving DLP.

3. The proposed scheme has a self-healing capability. It can also enable a node to recover from a single broadcast message all keys associated with sessions in which it belongs to the session group. It is a stronger self-healing key distribution scheme.

Without loss of generality, suppose $u_i \in G_t$ lost the broadcast message for a session t where $t < j$. It selects a polynomial z_t from the broadcast message

B_j and constructs a $|G_t| \times |G_t|$ matrix A_t as below:

$$A_t = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{|G_t|} \end{pmatrix}, \tag{8.18}$$

where $a_1 = (0, 0, ..., 0, 1, 0..., 0)$ (note: the i-th bit is 1 and all other bits are 0).

Then u_i solves the following system of equations.

$$(x_1, x_2, \ldots, x_{|G_t|}) \times A_t = (1, 1, \ldots, 1). \tag{8.19}$$

With $(x_1, x_2, \ldots, x_{|G_t|})$, u_i gets

$$(x_1, x_2, \ldots, x_{|G_t|}) \times \begin{pmatrix} Q_i \\ Q_{V_2} \\ \vdots \\ Q_{V_{|G_t|}} \end{pmatrix} = Q_{V_1}. \tag{8.20}$$

After that, with the knowledge of its private key S_i, u_i computes $e(P_{pub}, r_t Q_{V_1})$ as follows:

$$\begin{aligned} & e(P_{pub}, r_t Q_{V_1}) \\ & \quad = e(P_{pub}, r_t(x_1 Q_i + x_2 Q_{V_2} + \ldots + x_{|G_t|} Q_{V_{|G_t|}})) \\ & \quad = e(P_{pub}, r_t x_1 Q_i) \cdot e(P_{pub}, r_t(x_2 Q_{V_2} + \ldots + x_{|G_t|} Q_{V_{|G_t|}})) \\ & \quad = e(r_t P, x_1 s Q_i) \cdot e(P_{pub}, x_2 r_t Q_{V_2} + \ldots + x_{|G_t|} r_t Q_{V_{|G_t|}}) \\ & \quad = e(U_1, x_1 S_i) \cdot e(P_{pub}, x_2 U_2 + \ldots + x_{|G_t|} U_{|G_t|}). \end{aligned} \tag{8.21}$$

Finally, u_i recovers the lost session key

$$K_t = V_t \oplus H_2(e(U_1, x_1 S_i) \cdot e(P_{pub}, \sum_{k=2}^{|G_t|} x_k U_k)). \tag{8.22}$$

8.7.2 Property 2: Mutual-healing Key Distribution

We show that we securely establish mutual-healing property between neighboring nodes. More specifically, this satisfies the security requirements described in *Definition 8.2*.

1. It can be seen in the step for the *Mutual-healing request* that any node $u_i \in U$, if it misses more than a fixed number of broadcast messages or the last broadcast message, can generate and broadcast the requesting message to its neighbors. The broadcast message is composed of its identity ID_i, location l_i and the sequence number of the expected broadcast message t.

2. A false requesting node might send an request with a forged location within node u_j's range. Since the false requesting node does not hold the LBK corresponding to the forged location, even though it deceives u_j into believing it is in u_j's range, it can not recover the session key from the broadcast message that u_j sends. Therefore, it obtains any useful information from u_j. There are some false requesting nodes which might mount a DoS attack by continuously sending bogus mutual-healing requests to lure legitimate nodes into endlessly verifying such messages. Because the number of neighbors of any node is limited in reality, abnormally many mutual-healing requests are highly likely to be an indicator of malicious attacks. If this situation occurs, node u_j will discard the requesting message and refuse assistance.

3. Upon receiving the reply, node u_i also first checks whether u_j is its neighbor. This check is the baseline defence against an attack where adversaries surreptitiously tunnel authentication messages between u_i and a virtually non-neighboring node. Without the location check, u_i will falsely believe that the broadcast messages come from its neighbors. If the first check is true, then u_i checks whether the message received from u_j is effective. u_i can recover the requested broadcast message if the second check is true.

   If a requester receives too many replies, it decrypts only a fixed number of messages in order to conserve its limited resources and avoid an exhausting-resource attack. Furthermore, we assume that there are efficient mechanisms available for authorized nodes to report such an abnormality to the sink.

8.7.3 Property 3: Forward Secrecy and Backward Secrecy

We show that our construction satisfies forward secrecy and backward secrecy described in *Definition 8.3*.

In this scheme, due to the special construction of broadcast messages, only the current authorized nodes can recover the session keys by using their private keys. As described previously, any coalition of non-authorized nodes cannot derive the private key of any authorized node. Furthermore, session keys are independent of each other and are according to the uniform distribution. All of these features imply the forward secrecy and backward secrecy of our scheme.

8.7.4 Property 4: Collusion-free Property

We show that our construction is collusion-free for the new joined nodes and the revoked nodes described in *Definition 8.4*.

Our scheme is collusion-free for any coalition of non-authorized nodes, including the revoked nodes and the newly joined nodes. In our scheme, if one node wants to obtain the session key, it should compute $e(U_1, x_1 S_i)$ in the procedure of *Key Recovery*. Therefore, only the authorized nodes can recover the session key. In addition, due to the difficulty of solving DLP, any coalition of non-authorized nodes cannot derive the private keys of authorized nodes from their public keys.

8.7.5 The Limitation of the Proposed Scheme

The proposed mutual-healing scheme relies on identity and location-based keys.This implies that the proposed scheme can only be used over the wireless networks where nodes are stable. It is not trivial to realize mutual-healing in mobile wireless networks. In fact, the mutual-healing mechanism is more useful in mobile wireless networks because mobile WSNs have lower network connectivity than stable WSNs. Therefore, it is significant to investigate new methods to realize the mutual-healing feature in mobile wireless networks.

8.8 Performance Evaluation of the Mutual-healing Key Distribution Scheme

Unlike existing papers, we take advantage of a bilinear pairings based broadcast encryption to design the self-healing procedures in the proposed mutual-healing key distribution scheme. In this section, we first analyze the efficiency for self-healing procedures followed by the one for mutual-healing procedures.

8.8.1 Cost of Self-healing Procedures

Storage overhead

In terms of storage overhead, each node stores only its public/private key pair and the GM's public key P_{pub}. Therefore, the storage overhead for end nodes is a constant. In previous secret sharing-based self-healing key distribution schemes, the personal key could be reused on the condition that fewer than the threshold number of users were revoked. In the proposed scheme, the private key has nothing to do with the number of revoked nodes and can be reused as long as it is not disclosed. In addition, our scheme enables a node to recover from a single broadcast message all the keys associated with sessions in which it belongs to the session group.

Computation overhead

Generally speaking, the GM takes up more resources than end nodes and thus can perform more complex computation. We elaborate on analyzing the computation overhead at nodes. In the j-th session, all the computations in the procedure of *Key Recovery* are as follows:

1. Solving a set of linear equations with $|G_j|$ variables;
2. $|G_j|$ scalar multiplications in the cyclic additive group G_1;
3. $|G_j|$-1 additions in the cyclic additive group G_1;
4. Two pairings computation;
5. One hashing computation;

6. One XOR operation.

Generally speaking, the computation of the pairing is the most time-consuming in pairings-based cryptosystems. Although there have been many papers talking about the complexity of pairings and how to speed up the computation of bilinear pairings ([161] and [162]), the computation overhead of bilinear pairings are still larger than the scalar multiplication, let alone other types of computation. Therefore, the main computation overhead of the scheme comes from (4).

Communication overhead

The communication overhead comes from broadcast messages $B_j = \{z_1, \ldots, z_j\}$. z_j is composed of U_i $(1 \leq i \leq |G_j|)$ and V_j. Therefore, the size of z_j increases in direct proportion to the number of $|G_j|$. The length of broadcast is $(\sum_{i=1}^{j} |G_i| logq + jlogq)$ where $logq$ is the size of a session key.

8.8.2 Cost of the Mutual-healing Procedures

The mutual-healing procedures achieves all the security requirements and is efficient. Each node only has to store another pair of LBK and its location in order to achieve mutual-healing property. Therefore, the storage overhead is still a constant. In terms of communication overhead, only two interactions are involved. In the first step, the broadcast message is composed of the requester's identity, location and the sequence number of the expected broadcast message. Furthermore, the message is broadcasted within one-hop communication range. The communication overhead of this step is very small. The second interaction may include one or several unicasts. The unicast message is composed of the responser's identity, location, and encrypted the broadcast message. Therefore, the size of these unicast messages is a constant. The computation overhead at the responser includes generating a shared key and encrypting the requested message and the computation overhead at the requester includes generating a shared key and decrypting the requested message. Because mutual-healing does not occur very often, the computation overhead would not consume too much of the involved nodes' resources.

8.9 Authenticated Self-healing Key Distribution

8.9.1 System Parameters and Security Model

In the proposed authentication self-healing key distribution scheme, the same parameters as that in subsection 7.5.1 are used. In addition, the GM can acquire all the system parameters $params = \{G_1, G_2, q, P, P_{pub}, H_1, H_2\}$ and its private key s from an ID-based PKI. The GM keeps s as secret and makes $params$ and $H_4 : \{\{G_1\}^* \times \{0,1\}^*\}^* \times G_2 \to (\mathbb{Z}/q\mathbb{Z})^\times$ public (here $\{G_1\}^*$ represents a certain number of G_1). The GM is preloaded with a pair of public/private keys (Q_{GM}, S_{GM}) which is used for signature and its verification.

The security model in the proposed scheme is the same as that in Chapter 6. We do not repeat it here.

8.9.2 The Construction of Authenticated Self-healing Key Distribution

The authenticated self-healing key distribution scheme proposed in this section is a computational security scheme. It includes several procedures. In the procedure of *Broadcast*, the GM will sign the broadcast message which is used to recover session keys. Subsequently, at the beginning of *Key Recovery*, the nodes check the integrity and correctness of the broadcast message first. If the broadcast message is changed during the process of delivery, the group users will discard it.

Setup. The GM obtains both public system parameters and all the public keys of possible nodes from an ID-based PKI. The GM chooses m session keys $K_1, \cdots, K_m$ from $\mathbb{Z}_q^*$. The session keys are independent of each other and according to uniform distribution.

Broadcast. Suppose $|G_j|$ denotes the number of users in session j. For each session $1 \leq j \leq m$, according to the public keys of users in the session group G_j, the GM computes $Q_{V_1} = \sum_{i=1}^{n} Q_i$ and a $(|G_j| - 1) \times |G_j|$ matrix is defined as follows:

$$\begin{pmatrix} a_2 \\ a_3 \\ \vdots \\ a_{|G_j|} \end{pmatrix} = \begin{pmatrix} 1\ 1\ 0 \dots 0 \\ 1\ 0\ 1 \dots 0 \\ \vdots\ \vdots\ \vdots\ \ddots\ 0 \\ 1\ 0\ 0 \dots 1 \end{pmatrix} \tag{8.23}$$

Let $a_i^{'}$ represents the transpose of a_i. The GM also constructers $|G_j| - 1$ auxiliary keys

$$Q_{V_i} = (Q_1, Q_2, \dots, Q_{|G_j|}) \times a_i^{'} \quad 2 \leq i \leq |G_j|, \tag{8.24}$$

which means $Q_{V_2} = Q_1 + Q_2$, $Q_{V_3} = Q_1 + Q_3$, ..., $Q_{V_{|G_j|}} = Q_1 + Q_{|G_j|}$. The broadcast message is then formed by computing, for a random $r_j \in \mathbb{Z}_q^*$,

$$U_1 = r_j P, \quad U_i = r_j Q_{V_i} \quad 2 \leq i \leq |G_j|, \tag{8.25}$$

$$V_j = K_j \oplus H_2(e(P_{pub}, r_j Q_{V_1})) \tag{8.26}$$

Let $z_j = (U_i(1 \leq i \leq |G_j|), V_j)$. The broadcast message for j-th broadcast in the following form:

$$B_j = \{z_1, \dots, z_j\}. \tag{8.27}$$

The GM chooses an arbitrary $P_j \in G_1^*$, picks a random integer $k_j \in (\mathbb{Z}/q\mathbb{Z})^\times$ and computes:

- $t_j = e(P_j, P)^{k_j}$;
- $v_j = H_4(B_j, t_j)$;
- $w_j = v_j S_{GM} + k_j P_j$.

The signature for B_j is then the pair $(w_j, v_j) \in (G_1, (\mathbb{Z}/q\mathbb{Z})^\times)$. Finally, the GM broadcasts the message B_j together with the signature (w_j, v_j) to the set of users G_j.

Key Recovery. When a node $u_i \in G_j$ receives the message B_j and the signature (w_j, v_j), it computes $t_j = e(w_j, P) \cdot e(Q_{GM}, -P_{pub})^{v_j}$ and $H_4(B_j, t_j)$. If $v_j \neq H_4(B_j, t_j)$, it discards the incorrect message. Otherwise, it computes the session key from the broadcast message. The operation is the same as that in the *Key Recovery* procedure of mutual-healing key distribution scheme.

The operation of *Self-healing* and *Adding/Revoking Nodes* is the same as that in corresponding procedures of mutual-healing key distribution scheme, too.

8.10 Security Evaluation of the Authenticated Self-healing Key Distribution Scheme

According to the construction described in Section 8.8, we can say that our construction realizes an authenticated self-healing key distribution scheme using bilinear pairings. This scheme is simple without the deficiencies that occurred in [94] and [144]. According to the exact security proofs for the signature scheme in [157], the signature is secure under the random oracle model. In the procedure of *Key Recovery*, the receivers will first check the validity of the signature. If the verification fails, the broadcast message must be changed during the transmission. The receivers will not waste time on a useless broadcast message. While in a general self-healing key distrusting scheme without augmentation, the nodes do not know whether or not the recovered session keys are correct.

The scheme is collusion-free for any coalition of non-authorized nodes. In this scheme, if a node wants to obtain session keys, it should compute $e(U_1, x_1 S_i)$ in the procedure of *Key Recovery*. Therefore, only the authorized nodes can recover the session key. In addition, due to the hardness of solving DLP, any coalition of non-authorized nodes can not derive the private keys of authorized nodes from their public keys. Furthermore, session keys are independent of each other and according to uniform distribution, which subsumes forward secrecy and backward secrecy of the scheme.

8.11 Performance Evaluation of the Authenticated Self-healing Key Distribution Scheme

Different from the previous schemes, we utilize a short signature to authenticate broadcast of self-healing key distribution. There is a reasonable assumption that the GM takes up more resources than do the ordinary nodes and therefore can store more information and perform more complex computation as well as communication. This is the reason that in the process of efficiency analysis, we elaborate only on the analysis of the various overhead of ordinary nodes.

Storage overhead. Each node stores only its public/private key pair. Therefore, the storage overhead for each node is a constant. Because H_1 is a mapping from $\{0,1\}^*$ to G_1^* and the order of G_1 is q, so the length of all the public keys is $2logq$. The private keys are computed from the master key $s \in \mathbb{Z}_q^*$ and the public key, so the size of private keys is $2logq$. Therefore, the storage overhead for each group user is $4logq$, which is a constant number.

Communication overhead. The communication overhead comes from the broadcast message $B_j = \{z_1, \dots, z_j\}$ and the signature (w_j, v_j). z_j is composed of U_i ($1 \leq i \leq |G_j|$) and V_j. The size of U_i ($1 \leq i \leq |G_j|$) is $2logq$ and the size of V_j is $logq$ bits. Therefore, the length of z_j equals to $(2|G_j| + 1)logq$, which increases in direct proportion to $|G_j|$. The signature $(w_j, v_j) \in (G_1, (\mathbb{Z}/q\mathbb{Z})^\times)$. w_j is a point in G_1, so the bit size of w_j is $2logq$. v_j is a value in $(\mathbb{Z}/q\mathbb{Z})^\times$), so the bit size of v_j is $logq$. Consequently, the bit size of broadcast message is $[\sum_{i=1}^{j} 2|G_i| + (j+3)]logq$, which is related to the total number of users $\sum_{i=1}^{j} |G_i|$ in all the communication groups for different sessions and increases in direct proportion to session number j.

Computation overhead. All the computation in the procedure of *Key Recovery* is as follows:

1. Solving a set of linear equations with $|G_j|$ variables;

2. $|G_j| + 1$ scalar multiplications in the group G_1;

3. $|G_j|$ additions in the group G_1;

4. Four pairings computation. Two for the verification of signature and the other two for recovering the session key;

5. Two hashing computations;

6. One XOR operation.

In subsection 8.8.1, we mentioned that bilinear pairing computation is more time-consuming than scalar multiplication, let alone addition, hash and XOR operation. Therefore, the main computation overhead of the authenticated self-healing key distribution scheme is four pairing computation.

8.12 Conclusion

A mutual-healing key distribution scheme using bilinear pairings is proposed in this chapter. A security model and formal definition for mutual-healing key distribution were discussed. The proposed new scheme achieves several desirable features. The storage overhead for each node is a constant. The scheme is collusion-free for any coalition of non-authorized nodes. Each authorized node's private key has nothing to do with the number of revoked nodes and can be reused only if it is not disclosed. While in secret sharing-based self-healing key distribution schemes, the personal key can be reused on the condition that fewer than the threshold number nodes are revoked. In addition, our scheme enables a node to recover from a single broadcast message all keys associated with sessions in which it belongs to the session group. As far as we know, it is the first practical mutual-healing key distribution scheme.

Authenticated self-healing key distribution can confirm network users with the received broadcast actually originated from the group manager. An authenticated self-healing key distribution scheme based on bilinear pairings has been proposed in this chapter. The authentication is carried out with a short signature which is both computation and communication efficient. It enables the nodes to check the integrity and correctness of the broadcast messages before carrying out more complex key recovery operations. To the best of our knowledge, it is the first pairing-based authenticated self-healing key distribution scheme.

Part IV

A Hybrid Solution

9

A Hybrid Key Management Scheme for Hierarchical Wireless Sensor Networks

9.1 Introduction

Wireless Sensor Networks are fundamental building block of pervasive computing. Security issues become more important as application of WSNs becomes more mature. The robustness of security primitives mainly depends upon the strength of its key management mechanism. In WSNs, the required cost for key management is strongly subject to restriction by the node's available resources.

It is well known that Public Key Cryptography has some clear advantages over Symmetric Key Cryptography. It is not necessary to agree on a session key for the two communication parts like that in symmetric key algorithms. Key distribution is very easy in a PKC-based cryptosystem. Public key certificates may be freely distributed in advance or at the point of use. The keys are trusted on the basis of a CA. Another important advantage is that, unlike symmetric key algorithms, PKC-based cryptosystem can guarantee authentication, not only privacy. However, the main disadvantage of using public key systems is that they are not as efficient as symmetric key algorithms. It is generally acknowledged that PKC is infeasible in

WSNs due to its energy-consuming property.

In Chapter 2, we have reviewed some PKC-based solution for sensor networks. In the first trial of PKC-based key management for HWSNs [68], three key management schemes were proposed, among which SACK is based purely on Symmetric Key Cryptography, SACK-P is based purely on PKC, and SACK-H is the hybrid of both Symmetric Key Cryptography based and PKC-based approaches. However, both SACK-P and SACK-H are stiffly transformed from the basic SACK scheme which is based on Symmetric Key Cryptography. They cannot be seen as an original PKC-based key management scheme. Due to its design defect, the operations of scalability in these three schemes are trivial. Node addition and revocation involves interaction between cluster heads and the BS. If node addition and revocation happens frequently in the network, the authentication operations will collapse the BS. As for the other purely PKC-based schemes [69, 70, 73] and the hybrid scheme [71], they rely on too strong assumption or have unacceptable cost.

In this chapter, we will propose a hybrid key management solution for HWSNs. This chapter is organized as follows. In Section 9.2, we present the motivation of the design of the hybrid key management solution for HWSNs. In Section 9.3, we present the hybrid key management scheme for HWSNs in detail. In Section 9.4 and Section 9.5, we evaluate the performance and security of the proposed scheme and make a comparison with existing key management schemes for HWSNs from the perspective of performance and security. In Section 9.6, we draw the conclusion.

9.2 Motivation of the Proposed Hybrid Key Management Scheme

Recently deployed WSNs are increasingly following a hierarchical design. HWSNs have several advantages. As depicted in Figure 9.1, a HWSN typically consists of a BS, a few powerful cluster head (we alternatively call them H-nodes in this chapter), and a large number of low-cost sensor nodes (we represent it with L-nodes). It is assumed that the BS has numerous resources and is trustworthy. H-nodes have a large memory and strong data processing capability. They are responsible for

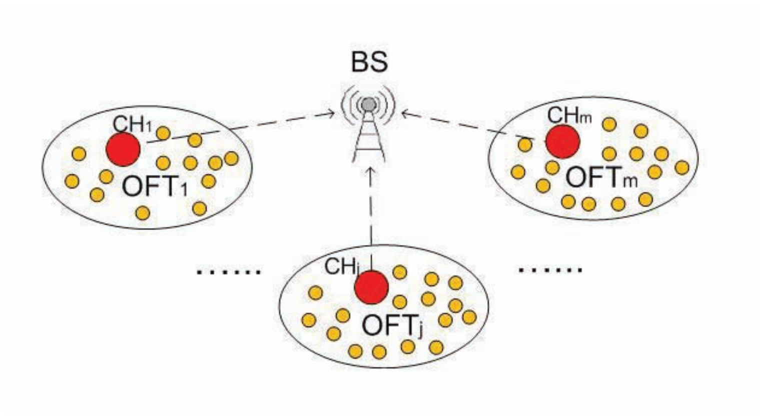


Figure 9.1: The hierarchical wireless sensor network topology

cluster management and data aggregation. The effect of node addition and revocation operations can be localized into one or several clusters. This property provides resilience and security benefits. In addition, because only H-nodes bear the responsibility of aggregating and transmitting data to the BS, L-nodes can be equipped with a tiny memory and very limited data processing capability. The cost of the network is reduced. Many popular key management schemes for HWSNs with single key mechanism overlooked the architecture and resource distinction of different tiers. A practical design of key management should take the heterogeneity into consideration. Otherwise, the key management scheme itself will exhaust the L-nodes.

We propose a key management scheme which takes advantage of the heterogeneous property. It puts the burden of energy-consuming algorithms on the H-nodes and lightweight algorithms on the L-nodes. It utilizes pairings-based Public Key Cryptography at upper level and efficient symmetric key cryptosystem in lower level. This is an advantage over general PKC-based key management which might quickly exhaust the energy of low-cost nodes in HWSNs. The hierarchical topology also provides flexibility. The lower level can choose an appropriate keying algorithm according to the dominated communication mode. For example, group key distribution schemes are suitable for group communication dominated environments, while key predistribution schemes are more appropriate for pairwise communication dominated environments. The proposed key management note support only the establishment of pairwise keys for peer communication but also cluster key for multicast and network key for broadcast.

9.3 The Proposed Framework

9.3.1 Network Model

We consider a HWSN scenario in this chapter. The entities involve a BS, a small number of H-nodes and a large number of L-nodes. All of these entities form a three-level network structure according to various clustering criteria. The root is the BS. It is natural to let H-nodes serve as cluster heads in the second level. The bottom level includes a large number of L-nodes. An example of such a network model has been shown in Figure 9.1.

1. The BS is an infrastructure and is assumed to be secure because it is usually located far from the wireless network. In our model, it takes charge of the network by manipulating cluster heads. Different from the SACK-P and SACK-H schemes in [68], interaction between BS and L-nodes is not involved.

2. Each cluster head is equipped with high power battery, large memory storage, strong data processing capabilities and good communication component. It can communicate with L-sensors within the cluster, peer cluster heads, and the BS. Cluster heads are responsible for aggregating sensing data and forward the date to the BS.

3. Each sensor node has limited battery power, small memory space, weak data processing capability, and short radio range. Sensor nodes are static but inherently support small scale mobility. To reduce energy consumption, a node communicates only with its cluster head or peer neighboring nodes within the same cluster. We assume that if a node is compromised, all the information it holds will be exposed to adversaries. Each L-node senses environment information around it and relays data to its cluster head.

9.3.2 The Proposed Hybrid Key Management Scheme

We present a hybrid key management scheme specifically designed for HWSNs in this section. The scheme takes advantage of the different capabilities at different levels. It uses resource-consuming ECC-based cryptography at the BS and cluster heads and efficient AES at L-nodes. The scheme combines the merits of both Public and Symmetric Key Cryptography.

The BS works as a CA which generates ECC-based public/private key pairs, one pair for the BS itself and others for cluster heads. The BS selects an elliptic curve over a prime field F_p which satisfies the equation $E : y^2 = (x^3 + ax + b)\text{mod}p$. Its parameters are $T = \{a, b, P, n, p\}$ where $a, b \in F_p$ are coefficients of the elliptic curve equation, which satisfies $4a^3 + 27b^2 \neq 0$, $P = (x_P, y_P)$ is the base point of the elliptic curve $E(F_p)$, n is the order of P and p is a prime which is the order of prime field F_p. The private key is an integer r and the corresponding public key is rP, denoted as (x_r, y_r), which is a point on $E(F_p)$. Here we refer to [163] for a description in detail of how the field and other parameters should be selected in practice for efficiency and security.

Each cluster head has a unique identity and has both cryptographic algorithms ECC/AES installed on it. A master key K_m and the basic keying materials, such as the graph $E : y^2 = (x^3 + ax + b)\text{mod}p$ and the basic point P for ECC are pre-installed on each cluster head as well. In addition, each cluster head is preloaded with the BS's public key and a unique ECC public/private key pair so that there is no requirement of tamper-resistant hardware to prevent the key pair from disclosure. Each L-node u is preloaded with the node secret x_u. For simplicity, we suppose that both cluster heads and L-sensors nodes are uniformly and randomly distributed in the deployed area. After deployment, clusters are formed in the network. Some efficient clustering algorithms with relevant attributes and objectives [164] can be used. At the end of cluster formation phase, each cluster head stores a public key list of all of its neighboring cluster heads. Each L-node stores the public keys of its cluster head and the back-up cluster head.

Each cluster equals a small communication group. The potential key management framework depends on the communication mode. For the broadcast communication dominated group, efficient group key establishment and renewal schemes are preferred. However, for peer communication dominated group, efficient and secure pairwise key establishment is the better option. OFT [56] is an efficient key management algorithm for a small group. However, the OFT algorithm can easily succumb to collusion attacks. Ku *et al.* [62] fixed the breach of collusion attacks in the OFT method with the increased communication overhead of $O((logn)^2)$ for group rekeying in one node revocation. Wang *et al.* presented a new method in [165], which is more efficient than the OFT method and can resist collusion attack. Simplicio *et al.* surveyed the pairwise key establishment schemes in [166]. Both arbitrated keying schemes and predistribution schemes can be used to efficiently establish pairwise keys. In the proposed scheme, we only consider broadcast communication dominated mode at the lower level. That is, the improved OFT scheme [165] is used at the lower level. For the peer communication dominated environment, only the lower level key establishment algorithm switches from the OFT scheme to an arbitrated keying scheme or key predistribution scheme.

There are four types of communication: the BS broadcasts to cluster heads, the BS unicasts to a cluster head, peer communication between cluster heads, cluster head broadcasts within its cluster. The corresponding keys should be established in order to keep communication secure. The process of key establishment is as follows:

1. In the proposed key management framework, the master key K_m shared by the BS and all the cluster heads can be used for broadcast communication from the BS to cluster heads. K_m must be renewed when a cluster head is compromised or node cluster head joins the network.

2. The BS has a public key list of all cluster heads. Each cluster head is preloaded with the BS's public key. The unicast between them is achieved with a public key algorithm. If unicast happens frequently, they can agree on a symmetric key with the protection of a public key algorithm.

3. Any two neighboring cluster heads CH_a with private key I_a and CH_b with private key I_b can establish a pairwise key exclusively shared to themselves. In fact, the process of key establishment can be combined with message exchange. The pairwise key is established during the first round of message exchange and both of them store the pairwise key for future communication.

   The process is presented as follows:

   - CH_a calculates $K_{a,b} = I_a I_b P$ and sends $\{ID_a, E_{K_{a,b}}(M, timestamp), timestamp\}$ to CH_b, where E is symmetric encryption algorithm.

- When CH_b receives the message sent by CH_a, CH_b firstly checks the freshness of the $timestamp$. If is outdated, CH_b discards the message. Otherwise, CH_b further checks whether it has a pairwise key shared with CH_a. If it already has the established pairwise key, CH_b decrypts the message with the pairwise key. Otherwise, CH_b searches CH_a's public key by indexing ID_a. And then CH_b calculates $K_{b,a} = K_{a,b} = I_b I_a P$ and decrypts $D_{K_{b,a}}(M)$ with a symmetric decryption algorithm. If $D_{K_{b,a}}(M)$ is readable, CH_b stores the pairwise key and continues secure communication with CH_a using the key $K_{a,b}$. The key $K_{a,b}$ is known only by cluster heads CH_a and CH_b. It is an authenticated key exchange scheme.

4. Each L-node establishes a shared key known only to itself and the cluster head with the preloaded node secret x_u as described in the [56]. After establishing the shared keys with all nodes in its cluster, the cluster head creates the OFT tree and numbers its nodes. A variety of methods can be used in this step. Some implementation issues of the OFT tree were discussed at the end of [56]. Each cluster equals a small communication group. The cluster head maintains a binary logical key tree. Each node is associated with a leaf node in the tree. The leaf node key is the shared key exclusively belonging to the node and the cluster head. Next, each node computes the cluster key after receiving crucial information from the cluster head. the cluster key is the root node key. We suppose that there is a separate secure channel between the cluster head and each node during this phase.

9.3.3 Rekeying

Rekeying issue is critical in determining the efficiency of any key management protocol. Node addition and revocation incur rekeying operations. Now we describe our rekeying algorithm for four possible circumstances.

1. When a cluster head CH_{new} joins the network;

   CH_{new} has been configured with related cryptographic algorithms and preloaded with related key materials, such as its public/private key pair.

   - $CH_{new} \rightarrow$ the BS: ID_{new}, its public key $I_{new}P$, $Join$ request;

- The BS checks the validity of CH_{new}'s public key $I_{new}P$. If it is not valid, The BS rejects the joining request. Otherwise, the BS renews the master key K_m to K'_m and sends it to the new cluster head CH_{new}.
  The BS $\rightarrow CH_{new}$: $\{E_{I_{new}P}(K'_m, timestamp), timestamp\}$;
- The BS broadcasts $\{E_{K_m}(K'_m, timestamp), timestamp\}$ to all the other cluster heads;
- Each cluster head decrypts $\{E_{K_m}(K'_m, timestamp), timestamp\}$ with K_m and gets the new master key K'_m.

2. When a cluster head CH_{rov} leaves the network.

   If a cluster head is forced to leave the network due to power depletion, it can send a $Leave$ request message to the BS. If a cluster head is compromised by an adversary, the misbehavior of the compromised cluster head can be detected by its neighboring cluster heads. The BS runs the revocation operation once it receives the report.

   - The BS creates a new network key K'_m, BS $\rightarrow CH_j$: $E_{I_jP}(K'_m)(j \neq rov), ID_{rov}$;
   - Each cluster head CH_j decrypts $E_{I_jP}(K'_m)$ with its private key I_j and gets the new master key K'_m;
   - Each CH_j which shares pairwise key with CH_{rov} deletes the shared pairwise key.

3. When a node joins the cluster CH_i.

   When the new node passes the authentication, it is assigned an individual key shared with the cluster head. Then the cluster head creates a new leaf node in the key tree for it. The new node is associated with a new leaf node in the key tree. Then the cluster head performs the rekeying operations as that in [165]. We assume a node m_8 joins a cluster as shown in Figure 9.2. m_8 is assigned with the individual key K_8. Then the cluster head performs the rekeying operations as follows:

   (a) generates a random number r and encrypts it with the current cluster key K, then broadcasts it to the nodes in the cluster.
   The cluster head $\rightarrow \{m_1, \ldots, m_7\} : E_K(r)$.

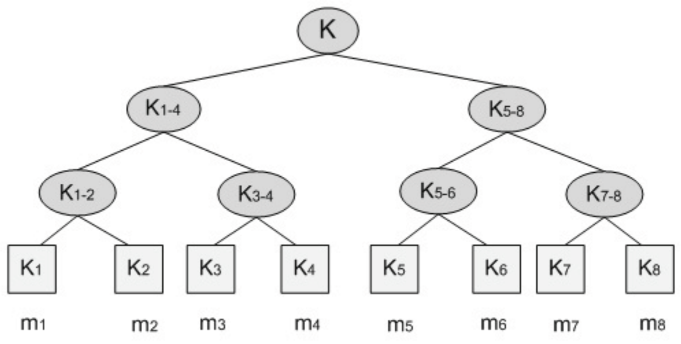


Figure 9.2: The One-way Function Tree of a cluster in the proposed scheme

(b) renews the keys on the path from the root to the new node m_8 using the hash function F with the random number r and the current cluster key as inputs.

$K' = F(K, r)$; $K'_{5-8} = F(K_{5-8}, r)$; $K'_{7-8} = F(K_{7-8}, r)$;

(c) encrypts these new keys K', K'_{5-8}, K'_{7-8} with the individual key K_8 and unicasts $E_{K_8}(K', K'_{5-8}, K'_{7-8})$ it to the new node m_8.

The nodes $\{m_1, \ldots, m_7\}$ can decrypt $E_K(r)$ and obtain the random number r. Then with r, each node in $\{m_1, \ldots, m_7\}$ can renew keys on the path from itself to the root. The new node m_8 can obtain the keys on the path from itself to the root with the individual key. At the end of this step, each node in the cluster can calculate a new cluster key K'.

4. When a node is evicted from the network.

When a node's power is off or compromised by an adversary, the node must be evicted from the network. The cluster head generates new key material and sends it to the remaining members encrypted individually with the appropriate subgroup keys unknown to the evicted node as depicted in [165]. Each remaining node decrypts the key material with the appropriate subgroup key and calculates the root key, that is, the renewed cluster key.

Here we assume a node m_3 is evicted from the cluster as shown in Figure 9.2. The cluster head performs the rekeying operations. The cluster head generates a random number r' and encrypts it with the sub-cluster keys K_{1-2}, K_4, K_{5-8} and then broadcasts them to appropriate sub-clusters.

The cluster head $\rightarrow \{m_1, m_2\} : E_{K_{1-2}}(r')$;

The cluster head $\rightarrow m_4 : E_{K_4}(r')$;

The cluster head $\rightarrow \{m_5 \ldots, m_8\} : E_{K_{5-8}}(r')$;

The remaining nodes $\{m_1, m_2, m_4, \ldots, m_8\}$ can obtain r' with the appropriate sub-cluster key. Then each node renews the keys on the path from itself to the root.

9.4 Performance Evaluation

In this section, we evaluate the performance of the proposed hybrid key management scheme and compare it with existing the key management schemes in [68, 71] and the schemes in [69, 70, 73] respectively. All of these schemes target HWSNs. More specifically, the schemes in [68, 71] are hybrid key management schemes for HWSNs and the schemes in [69, 70, 73] are complete PKC-based key management scheme for HWSNs. We discuss several important issues in HWSNs, including broadcast authentication, resilience, and flexibility.

9.4.1 Key Space Requirement

All of existing PKC-based and hybrid key management schemes and the proposed scheme guarantees 100% key connectivity with different storage cost. Both the schemes [69, 70] compute the total key space requirement without considering the heterogeneity of nodes. The analysis is done at cluster heads and L-nodes excluding the BS as it is assumed that the BS has no energy/memory constraints. Assume that the number of cluster heads and L-nodes in an HWSN is m and n, respectively. It is a practical assumption that $m \ll n$. When the nodes are uniformly deployed in the network, there are $\lceil n/m \rceil$ nodes in each cluster. Each cluster head can keep one-hop communication with at most $\lceil m/4 \rceil$ cluster heads and each L-node can keep one-hop communication with at most $\lceil n/(4m) \rceil$ L-nodes within the same cluster.

- In SACK-P in [68], each sensor node stores the public keys of sensor nodes and cluster head of its cluster except for its private key. There are $\lceil n/m \rceil$ nodes in each cluster on average. Thus, each node has to store $(\lceil n/m \rceil + 2)$ keys in total. Each cluster head has public keys of other cluster heads and the BS except for its private key. In addition, each cluster head stores the public keys of nodes in its cluster. There are $(m + \lceil n/m \rceil + 2)$ keys in all. SACK-H in [68] uses Public Key Cryptography for communication between the BS and cluster heads and Symmetric Key Cryptography for communication within a cluster. Each cluster head stores its public/private key pair and public keys of other cluster heads and the BS. So there are $(m + 2)$ keys stored by each cluster head. Each node stores only one symmetric key for cluster-wide communication. For the scheme in [71], each L-node must store $(\lambda + 1)$ elements for each chosen space to compute a key with the Blom key predistribution scheme. Each L-node carries k spaces and each cluster head carries m space. The total key space requirement at each L-node is $\lceil n/m \rceil(\lambda + 1)$ and that at each cluster head is $m(\lambda + 1)$. In addition, each cluster head stores its private/public key pair and the public key of the BS and other cluster heads.

- In the centralized ECC key management scheme in [69], each cluster head is preloaded with public keys of all L-nodes, plus a pair of public/private key for itself, and a key K_H for authentication purpose. Thus, a cluster head is preloaded with $(n + 3)$ keys. Each L-node is preloaded with its private key and the public key of the cluster heads. Given the protection from tamper-resistant hardware, the same public/private key pair can be used by all cluster heads. Thus, each node is preloaded with 2 keys. In the distributed ECC key management scheme, each cluster head is preloaded with its public/private key and the authentication key K_H. Each L-node is preloaded with its public/private key pair and public keys of all the cluster heads. In the distributed key establishment scheme, the assumption of having tamper-resistant hardware in cluster heads can be removed. Different ECC key pairs are used. So each L-node is preloaded with $m + 2$ keys. Many possible conditions are considered in [70]. We consider only the same condition that all nodes are uniformly spread throughout the network. Each L-node stores $(\lceil n/m \rceil + 1)$ number of keys and each cluster head stores $(\lceil n/m \rceil + m + 1)$ keys. The key space is not clearly addressed in [73]. We know that each L-node has

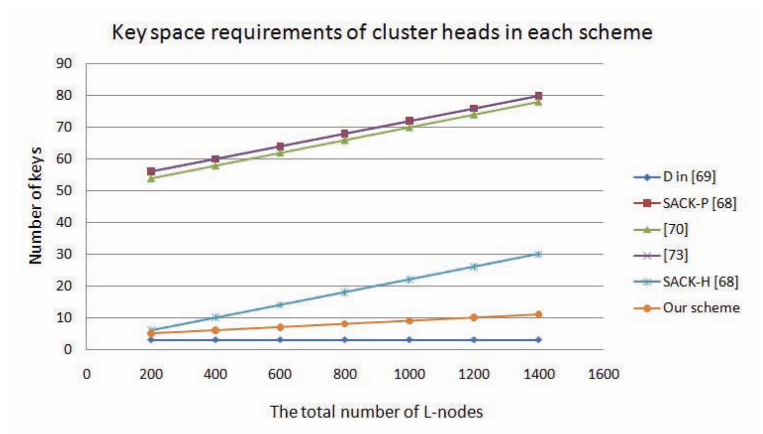


Figure 9.3: Comparison of key space requirements at cluster heads for different schemes for different sizes of networks

to know its neighbors' public key in order to compute pairwise keys. Each cluster head stores pairwise keys for its members and the public key of the BS and the other cluster heads.

- In our proposed scheme, each cluster head stores a symmetric master key K_m, a public/private key pair and public keys of the BS and the neighboring cluster heads. Suppose each cluster head neighbors with $m/4$ cluster heads, each cluster head has to store $(3+\lceil m/4\rceil+1(symm))$ keys. The neighboring cluster heads can securely establish a symmetric key with each other. Each L-node stores public keys for its cluster head and the back-up cluster head plus 2 symmetric keys for communication within the cluster.

Table 9.1: Key space requirements of existing key management schemes

Tiers	Purely PKC-based schemes					Hybrid schemes		
	C in [69]	D in [69]	SACK-P [68]	[70]	[73]	[71]	SACK-H [68]	Our Scheme
Cluster heads	$n+3$	3	$m+\lceil n/m\rceil$ $+2$	$m+\lceil n/m\rceil$	$m+\lceil n/m\rceil+2$	$\lceil n/m\rceil(\lambda+1)$	$m+2$	$\lceil m/4\rceil+3$ $+1(symm)$
L-nodes	2	$m+2$	$\lceil n/m\rceil+2$	$\lceil n/m\rceil$	$\lceil n/m\rceil+1$	$m(\lambda+1)+(m+2)$	1(symm)	$2+2(symm)$

We summaries the key space requirements of existing schemes in Table 9.1. C and D in [69] represent the centralized and distributed schemes in [69], respectively.

NIST guidelines state that ECC keys should be twice the length of equivalent strength symmetric key algorithms. So, for example, a 160-bit ECC key would

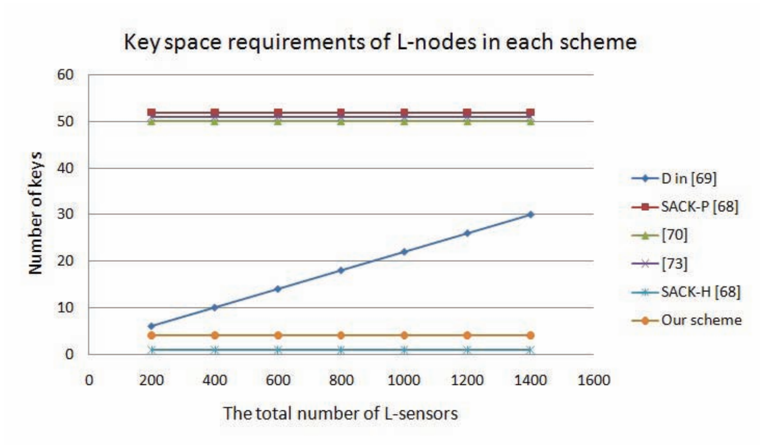


Figure 9.4: Comparison of key space requirements at L-nodes of different schemes for different sizes of networks

have roughly the same strength as a 64-bit symmetric key [69]. Suppose that the length of each public key is 160-bit and that of each symmetric key is 64-bit. We plot the key space requirements of different schemes for different sizes of WSNs in Figure 9.3 and Figure 9.4. The x-axis is composed of discrete points. It takes values of $(200, 400, 600, 800, 1000, 1200, 1400)$, the corresponding values of m are $(4, 8, 12, 16, 20, 24, 28)$. The y-axis is the required memory size for different schemes. We do not include the centralized schemes in [69, 71] in Figure 9.3 and Figure 9.4. The scheme in [69] has a clear disadvantage and the maximum number of compromised nodes of [71] is constrained by λ. As can be seen, our scheme has low key space requirements at both cluster heads end and L-nodes end. In the network of $m = 20$ cluster heads and $n = 1000$ sensor nodes. The memory requirement for cluster head is 1280 bits and 448 bits. It is reasonable for the popular sensor node MICA2 which has a 4KB primary memory.

9.4.2 Scalability

We have discussed the rekeying scheme incurred by membership change. When a new cluster head joins the network, the BS renews the master key K_m and sends it to the new cluster head with unicast and broadcast to existing cluster heads. When a cluster head leaves the network, the BS encrypts the renewed master key with the public key of each cluster head and sends it to the corresponding cluster head. Unlike the scheme in [68], the BS does not intervene in each node addition or revocation in our scheme. Each cluster manages the node addition and revocation operations. The effect of membership change is localized to a cluster. In addition, the storage overhead of sensor nodes does not increase with network scale. In this way, our scheme achieves good scalability. The scalability of the proposed key management scheme was not mentioned in [69]. The BS is responsible for random number renewal of the whole network [70]. It leads to poor scalability of the scheme.

9.5 Security Evaluation

In this section, we evaluate performance of the proposed hybrid key management scheme and compare it with existing the key management schemes in [68, 71] and the schemes in [69, 70, 73], respectively. We discuss several issues in HWSNs, including resilience, broadcast authentication, and flexibility.

9.5.1 Resilience

Network resilience is defined as its resilience against node compromise attack in [68]. Our scheme uses Public Key Cryptography for upper level communication and Symmetric Key Cryptography for lower level communication. Each cluster head stores the public keys of its neighboring cluster heads and each pair of neighboring cluster heads has a different shared key. A cluster head compromise does not reveal any keying information except its own private key and a few public keys. Each sensor is given a common symmetric key for cluster wide communication via the OFT scheme. Nodes compromise affects only the communication of that particular cluster. If compromised nodes can be found immediately, rekeying method can be applied for damage control. Therefore, appropriate methods which can detect node compromise as early as possible are required. SACK-H and [73] have the same degree of resilience as our scheme as they adopt the same rekeying mechanism. SACK-P and the scheme in [69] have better resilience since Public Key Cryptography is used in the lower level. The resilience of [71] is constrained by λ. More than λ compromised nodes within a cluster can work out all the pairwise keys for the specific cluster. The scheme in [70] has the best resilience with largest computation and communication cost.

9.5.2 Broadcast Authentication

Both SACK-H and SACK-P did not address the property of broadcast authentication. Authentication on broadcast messages was realized with the ECDSA algorithm [167] in [69]. When a cluster head broadcasts a message, such as the routing information, within the cluster, a signature over the message is generated by the cluster head and verified by nodes. However, the ECDSA algorithm computes kP during the signature generation phase and multiple scalar multiplications, such as $(kP + 1Q)$, during the signature verification phase. Vanstone verified that the signature verification of ECDSA takes twice the time it does for signature generation considering the same security level in [168]. Therefore, this authentication mechanism creates a bottleneck due to too much computation overhead at low-cost nodes. Lightweight authentication mechanisms are a better substitute for energy-consuming signature algorithms. In addition, the asymmetric requirement of energy for the authentication should be given full consideration. That is, the energy-consuming computation should be performed by powerful cluster heads and comparatively easy computation should be performed by sensors. We do not include the authentication property in the proposed key management scheme. If broadcast authentication is required, we recommend the broadcast authentication technique proposed in [30]. The technique supports a potentially large number of broadcast senders. It also provides two approaches to revoking the broadcast authentication capability from compromised senders.

9.5.3 Flexibility

The flexibility of the scheme is two-fold. Firstly, the key management framework of the lower level can be switched according to the communication mode. Group key management schemes such as [165] can be used for the broadcast communication dominated environment, while the pairwise key establishment scheme can be used for the peer communication dominated environment. Secondly, it is flexible in regards to the key selection. Each cluster is able to select the cluster key and renew it without the intervention of the BS. The purely PKC-based key management schemes in [69, 70, 73] are not flexible in these two aspects.

9.6 Conclusion

A hybrid key management scheme for hierarchical wireless sensor networks is proposed in this chapter. The scheme uses the best features of both Public Key Cryptography and Symmetric Key Cryptography and the difference in the computational capabilities of different nodes. Our scheme is proven to be secure and has the following advantages.

1. It is flexible in terms of the key selection. Each cluster is able to select its own cluster key for its own convenience, and can change its cluster key from CK to CK' for security reasons without the intervention of the BS.

2. The key space requirement of L-nodes is a small constant and is reasonable for the popular sensor node MICA2 which has a 4KB primary memory. The key space requirement of cluster heads is lower than most of existing schemes.

3. The rekeying approach is efficient. The scheme can be applied to large-scale networks. The scheme can also be applied to peer communication dominated environments if we replace the OFT rekeying algorithm with the key predistribution scheme. The upper level algorithm remains unchanged.

Part V

Conclusion and Future Work

10

Conclusion and Future Work

10.1 Introduction

Key management plays an important role in network security. It serves as the cornerstone of the security framework. Key management for group communication and traditional networks has been intensively studied. However, those techniques can not be applied to WSNs directly or even with minor revision due to the unique nature of the latter. The development of WSNs introduces revolutionary paradigms for network deployment and application. Almost all of the WSN security mechanisms are built upon solid key management infrastructures. Key management for sensor network security has attracted a lot of research interest in the past few years. However, it is evident from the literature that many unaddressed technical problems still exist in this field.

In our research, we focused on developing advanced solutions to address problems relating to:

- Key management for WSNs;
- Establishment of different types of keys;
- Implicit authentication or PKC-based authentication mechanism;
- Self-healing mechanism and mutual-healing key distribution mechanism.

In order to solve the technical problems arising from the four areas of key management in WSNs as discussed above, we break them down to six research issues. In the next section, we will recapitulate on the different issues that we have identified and addressed in this thesis. In Section 10.3, we highlight the contributions which have been made by the thesis to the literature as a result of our having successfully addressed the different issues. In Section 10.4, we identify some areas for future work and in Section 10.5 we conclude the chapter.

10.2 Recapitulation of Research Issues

As mentioned in Chapters 2 and 3, key predistribution has been considered as the most suitable key management model for sensor networks. However, almost all existing key predistribution schemes are designed for distributed WSNs and they have several weaknesses. HWSNs have incomparable advantages over distributed wireless sensor networks in terms of scalability and resilience against node capture. If we take advantage of the hierarchical structure of networks with hierarchical key predistribution, some desirable properties will be achieved. The storage overhead can be further reduced and key connectivity can be further increased if we substitute the randomly generated key pool with keyed hash chains. We have to eliminate the side effects of the one-way property of hash chain.

LEAP [29] is the first key management scheme which support establishment of multi-level keys in wireless sensor networks. Whereas both LEAP [29] and its improvement [1] have some technical deficiencies which have been outlined in Chapter 3. We noted that time-based key management scheme [1] eliminates the effect of disclosure of the initial key successfully with the cost of lower key connectivity probability. Hence, our research emphasis is on increasing the key connectivity probability in regards to this technical problem.

Hash chain based self-healing key distribution schemes are the most efficient self-healing key distribution schemes among the schemes in the same league. However, the types of self-healing key distribution schemes are vulnerable to collusion attack. Although one more effort has been made to resist collusion attack, none of them has quite assessed the breach or presented a perfect approach. Mutual-healing is a complementary mechanism which achieves self-healing key distribution. Although this notion is not new, to the best of our knowledge, the technical details have not been explored to date. We have mentioned that we do not propose an independent authentication approach in this thesis. Instead, we consider embedding implicit authentication or lightweight PKC-based authentication in self-healing key distribution.

Although HWSNs have many advantages over flat ones, many existing key management schemes have overlooked the heterogeneity of different levels. Hybrid solutions can make full use of heterogeneity of HWSNs. To date, no practical hybrid key management solution has been proposed for HWSNs.

In summary, the research issues addressed in this thesis are as follows:

- Developing an algorithm supporting the establishment of different types of keys whose security does not depend on the security of the initial key;
- Developing a hash chain based key predistribution algorithm which improves key connectivity probability and robust against node capture attack;
- Developing a hash chain based self-healing key distribution algorithm which is thoroughly robust against collusion attack;
- Developing a secure HBT-based self-healing key distribution algorithm with implicit authentication;
- Exploring the technical details of the mutual-healing key distribution and embedding a lightweight PKC-based authentication in bilinear pairings based self-healing key distribution;
- Developing a practical hybrid key management algorithm for HWSNs.

10.3 Contribution of the Thesis

The major contribution of this thesis to existing literature is that it proposes a conceptual framework for practical design of key management for HWSNs security. Following the conceptual framework, several solutions are developed for specific research issues. The solutions can be categorized according to three types of cryptosystems:

1. Symmetric Key Cryptography based solutions:
    - A Robust key management algorithm for multi-phase hierarchical wireless sensor networks;
    - A hash chain based key predistribution algorithm;
    - A fully anti-collusive hash chain based self-healing key distribution algorithm;
    - A HBT-based self-healing key distribution algorithm with implicit authentication.
2. PKC-based solutions:
    - A mutual-healing key distribution algorithm;
    - Authenticated bilinear pairings based self-healing key distribution algorithm.
3. A Hybrid solution:
    - A practical hybrid key management algorithm.

10.3.1 Contribution 1: The solution for robust key management for multi-phase hierarchical wireless sensor networks

The first major contribution of this thesis is the development of an robust key management protocol for multi-phase hierarchical wireless sensor networks. The work was presented in Chapter 4. Before the presentation of the solution, we rehearsed the technical problems arising from LEAP [29] and the time-based key management scheme [1] and the importance of multi-phase deployment of WSNs. The understanding of these deficiencies helps to pinpoint the security requirements and development of a robust key management protocol for multi-phase HWSNs.

The main features of the proposed solution are as follows:

- Following the idea of [1], we divided the lifetime of a sensor network into many generations. Each generation has its initial key. The initial keys for different time slots are independent of each other. However, in the proposed solution, we take the lifetime of nodes into the consideration, a node whose lifetime is G_w generations is preloaded with an initial key and $G_w - 1$ masked initial keys. The key predistribution method prevents memory waste and increases key connectivity probability.

- We utilized a unique masking mechanism which ensures that nodes deployed at different time slots can agree to pairwise keys and the pairwise keys belong exclusively to the two nodes.

- The proposed solution provides a perfect mechanism for maintaining forward and backward secrecy. We addressed the issue of key updating in the case of membership change.

- We identified the security weaknesses of key management services of the ZigBee specification and showed that the proposed key management protocol can be directly applied to enhance the key management services of the ZigBee specification.

The proposed solution successfully addressed the technical problems of LEAP [29] and the time-based key management scheme [1]. The solution can be seamlessly integrated with ZigBee networks for more comprehensive and effective key management services.

10.3.2 Contribution 2: The solution for hash chain based key predistribution

The second major contribution of this thesis is the development of hash chain based key predistribution algorithm for hierarchical WSNs. This work was presented in Chapter 5. The main features of this solution are as follows:

- The structure of the key pool is much different from that in random key predistribution schemes. Instead, generation of a large key pool with a great number of random keys, a small key pool with very limited number of generation keys and their IDs as well as their commitments, is generated.

- Each sensor is preloaded with IDs and commitments of a small number of generation keys without actual generation keys. Each cluster head is preloaded with a large number actual generation keys, their IDs and commitments. This key predistribution method can resist node capture attack and increase key connectivity probability with low memory requirement.

- We utilized a different clustering method in the proposed scheme to improve key connectivity probability.

The comparison with existing works shows that the proposed solution greatly reduced storage overhead at low-cost nodes, and at the same time, increased resilience against node capture attack. In addition, the introduction of a new clustering mechanism gives the scheme a higher key connectivity probability.

10.3.3 Contribution 3: The solution for fully anti-collusive hash chain based self-healing key distribution

The third major contribution of this thesis is the development of a solution for fully anti-collusive hash chain based self-healing key distribution. This work was presented in Chapter 6. The main features of this solution are as follows:

- We pinpointed the breach of collusion attack in hash chain based self-healing key distribution schemes and proposed a unified approach of collusion attack.

- We presented two attacks against Du *et al.*'s scheme [116] with the proposed unified attack approach. One attack is launched by revoked nodes and the other attack is launched by collusion between a revoked user, whose life cycle has finished, and a newly joined user.

- We proposed two possible solutions for collusion attack. After ruling one out, we prove that the only feasible solution is achieved by masking new random numbers in the corresponding broadcast messages.

- We redefined the security model which provides more strict security requirements and proposed a fully anti-collusive hash chain based self-healing key distribution scheme.

The security of the proposed solution is proved under an appropriate model. It shows that the proposed solution can fully resist collusion attack in hash chain based self-healing key distribution schemes. The performance analysis and the comparison with the same types of schemes demonstrated that the proposed solution achieves a fully anti-collusive property with acceptable overhead.

10.3.4 Contribution 4: The solution for HBT-based self-healing key distribution with implicit authentication

The fourth major contribution of this thesis is the development of a solution for HBT-based self-healing key distribution with implicit authentication. This work was presented in Chapter 7. The main features of this solution are as follows:

- We replaced Shamir's secret sharing with vector space secret sharing. The number of revoked nodes is not constrained by the threshold secret sharing scheme.
- We embedded implicit authentication in the HBT-based self-healing key distribution scheme with very little computation cost.

According to security proof, the HBT-based scheme not only maintains forward and backward secrecy, but it also has a collusion resistance property. The performance analysis shows that storage overhead of the proposed solution is almost a constant and the computation overhead produced by *Key Recovery* is also greatly reduced.

10.3.5 Contribution 5: The solution for mutual-healing key distribution and authenticated bilinear pairings based self-healing key distribution

The fifth major contribution of this thesis is the development of a solution for mutual-healing key distribution and a solution for authenticated bilinear pairing based self-healing key distribution. This work was presented in Chapter 8. The main features of these two solutions are as follows:

- We pointed out that the only requirement of mutual-healing is to effectively authenticate requesting nodes and the only feasible way is to endow each node with a public/private key pair so that they can perform node-to-node authentication on demand.
- We proposed a security model and a formal definition of mutual-healing key distribution.
- We bound each node both with its identity and location rather merely either one or the other of them. A public/private key pair can be calculated from each node's identity and its location. Each node stores its location and LBK pair which can be used for authentication on requesting nodes.
- We proposed an authenticated bilinear pairings based self-healing key distribution with short signature.

As far as we know, this is the first time that the technical details of mutual-healing key distribution has been explored. The storage overhead for each node is a constant and the key pair can be reused only if it is not disclosed. Hence, this is scheme can be used for long-life WSNs. It is also the first authenticated bilinear pairings based self-healing key distribution scheme to be developed. It enables the nodes to check the integrity and correctness of the broadcast messages before carrying out more complex key recovery operations.

10.3.6 Contribution 6: The solution for practical hybrid key management in HWSNs

The sixth major contribution of this thesis is the development of a solution for practical hybrid key management for HWSNs. This work was presented in Chapter 9. The main features of these two solutions are as follows:

- We utilized ECC-based Public Key Cryptography for upper level communication and efficient Symmetric Key Cryptography for lower level communication.

- The proposed solution provides flexible options. The potential key management framework depends on the communication mode. For the broadcast communication dominated group, efficient group key establishment and renewal schemes are preferred. While for a peer communication dominated group, efficient and secure pairwise key establishment, such as key predistribution, is a better option.

The proposed solution was proven to be secure and has several advantages. Firstly, each cluster head is fully autonomous. Secondly, the memory requirement of low-cost nodes is a small constant and the memory requirement of a cluster head is lower than that of most existing key management schemes. Thirdly, the rekeying approach is very efficient and scalability is achieved.

10.4 Future Work

We have demonstrated that this thesis has sufficiently achieved the research objectives for key management in hierarchical wireless sensor networks. However, there is some future work for further investigation in order to strengthen the proposed solutions.

- We have identified the security weaknesses of key management for ZigBee networks and in Chapter 4 proposed the solution for enhanced security. WirelessHART [169] and ISA100 [169] are two standards for WSNs and are attracting more and more attention. Both of them provide a stronger key management function than that of ZigBee standard. However, the key management function of WirelessHART and ISA100 is still inadequate, especially for certain environments with high security requirements. In future work, we intend to investigate the security weaknesses of WirelessHART and ISA100 standards and propose the solutions for them.

- The proposed mutual-healing scheme relies on identity and location based keys. This implies that the proposed scheme can be used only over the wireless networks where nodes are stable. It is significant to achieve mutual-healing in mobile wireless networks. In fact, the mutual-healing mechanism is more useful in mobile wireless networks because mobile wireless networks have lower network connectivity than stable wireless networks. Therefore, it is important to investigate new methods to achieve the mutual-healing feature in mobile wireless networks.

- In this thesis, we presented the procedures for developing general key management protocols. The performance of the proposed solutions has been verified based on analysis. Although the actual feasibility of solutions can probably only be verified through practical implementation, analysis results obtained in this thesis are a useful preliminary step to the implementation. In future, we will implement the proposed key management protocols in appropriate simulation environment in order to test the protocols' viability.

10.5 Conclusion

In this chapter, we have recapitulated the work that we have undertaken in this thesis. We highlighted research achievements according to the identified research issues. We then presented a brief description of the further work that we intend to undertake in order to strengthen the proposed solutions.

The work that we have undertaken in this thesis has been published extensively as a part of proceedings in peer-reviewed international journals and conferences. We have attached some selected publications in Appendix A. A complete list of all the publications arising as a result of the work documented in this thesis is attached at the beginning of the thesis.

References

[1] Jiyong Jang, Taekyoung Kwon, and Jooseok Song. A time-based key management protocol for wireless sensor networks. In *Proceedings of the 3rd International Conference on Information Security Practice and Experience*, ISPEC'07, pages 314–328, Berlin, Heidelberg, 2007. Springer-Verlag.

[2] Adrian Perrig, Robert Szewczyk, J. D. Tygar, Victor Wen, and David E. Culler. SPINS: security protocols for sensor networks. *Wireless Networks*, 8:521–534, September 2002.

[3] Laurent Eschenauer and Virgil D. Gligor. A key management scheme for distributed sensor networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS'02, pages 41–47, New York, USA, 2002. ACM.

[4] Firdous Kausar, Sajid Hussain, Laurence T. Yang, and Ashraf Masood. Scalable and efficient key management for heterogeneous sensor networks. *Journal of Supercomputing*, 45:44–65, July 2008.

[5] Xiaojiang Du, Yang Xiao, Mohsen Guizani, and Hsiao Hwa Chen. An effective key management scheme for heterogeneous sensor networks. *Ad Hoc Networks*, 5(1):24–34, 2007. Security Issues in Sensor and Ad Hoc Networks.

[6] Ratna Dutta, Sourav Mukhopadhyay, and Tom Dowling. Trade-off between collusion resistance and user life cycle in self-healing key distributions with t-revocation. In *Proceedings of the 2nd International Conference on the Applications of Digital Information and Web Technologies*, ICADIWT'09, pages 603–608, August 2009.

[7] Ratna Dutta, Sourav Mukhopadhyay, and Tom Dowling. Generalized self-healing key distribution in wireless ad hoc networks with trade-offs in user's pre-arranged life cycle and collusion resistance. In *Proceedings of the 5th ACM Symposium on QoS and Security for Wireless and Mobile Networks*, Q2SWinet'09, pages 80–87, New York, USA, 2009. ACM.

[8] Michael Dustin, Jagadish Shankarappa, Michael Petrowski, Hesiri Weerasinghe, and Huirong Fu. Analysis of key management in wireless sensor networks. In *IEEE International Conference on Electro/Information Technology*, pages 263–271, May 2007.

[9] Tian He, Sudha Krishnamurthy, John A. Stankovic, Tarek Abdelzaher, Liqian Luo, Radu Stoleru, Ting Yan, Lin Gu, Jonathan Hui, and Bruce Krogh. Energy-efficient surveillance system using wireless sensor networks. In *Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services*, MobiSys'04, pages 270–283, New York, USA, 2004. ACM.

[10] University of Washington. The assistive cognition project. http://www.cs.washington.edu/assistcog/, accessed on February 13, 2011.

[11] Tia Gao, Dan Greenspan, and Matt Welsh. Improving patient monitoring and tracking in emergency response. In *Proceedings of the International Conference on Information Communication Technologies in Health*, ICICTH'05, July 2005.

[12] David C. Steere, Antonio Baptista, Dylan McNamee, Calton Pu, and Jonathan Walpole. Research challenges in environmental observation and forecasting systems. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom'00, pages 292–299, New York, USA, August 2000. ACM.

[13] Paolo Baronti, Prashant Pillai, Vince W. C. Chook, Stefano Chessa, Alberto Gotta, and Y. Fun Hu. Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBee standards. *Computer Communications*, 30(7):1655–1695, 2007. Wired/Wireless Internet Communications.

[14] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102–114, August 2002.

[15] Vidyasagar Potdar, Atif Sharif, and Elizabeth Chang. Wireless sensor networks: A survey. *International Conference on Advanced Information Networking and Applications Workshops*, 0:636–641, 2009.

[16] Yun Zhou, Yuguang Fang, and Yanchao Zhang. Securing wireless sensor networks: A survey. *IEEE Communications Surveys Tutorials*, 10(3):6–28, 2008.

[17] Fang Liu, Xiuzhen Cheng, and Dechang Chen. Insider attacker detection in wireless sensor networks. In *Proceedings of the 26th IEEE International Conference on Computer Communications*, INFOCOM 2007, pages 1937–1945, May 2007.

[18] Bryan Parno, Adrian Perrig, and Virgil Gligor. Distributed detection of node replication attacks in sensor networks. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, pages 49–63, Washington, DC, USA, 2005. IEEE Computer Society.

[19] Ping Li, Yaping Lin, and Weini Zeng. Research on security in sensor networks. *Journal of Software*, 17(12):2577–2588, December 2006.

[20] John Paul Walters, Zhengqiang Liang, Weisong Shi, and Vipin Chaudhary. *Security in Distributed, Grid, and Pervasive Computing*, chapter 17: Wireless sensor network security: A survey, pages 1–51. CRC Press, 2007.

[21] Yanchao Zhang, Wei Liu, Wenjing Lou, and Yuguang Fang. Location-based compromise-tolerant security mechanisms for wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 24(2):247–260, February 2006.

[22] Zubair A. Baig. Pattern recognition for detecting distributed node exhaustion attacks in wireless sensor networks. *Computer Communications*, 34(3):468–484, 2011. Special Issue of Computer Communications on Information and Future Communication Security.

[23] Avinash Srinivasan, Joshua Teitelbaum, Jie Wu, Mihaela Cardei, and Huigang Liang. *Reputation-and-trust-based systems for ad hoc networks*, pages 375–403. John Wiley & Sons, Inc., 2008.

[24] Chris Karlof and David Wagner. Secure routing in wireless sensor networks: attacks and countermeasures. *Ad Hoc Networks*, 1(2-3):293–315, May 2003. Sensor Network Protocols and Applications.

[25] Anthony D. Wood and John A. Stankovic. Denial of service in sensor networks. *Computer*, 35:54–62, October 2002.

[26] Yong Wang, Garhan Attebury, and Byrav Ramamurthy. A survey of security issues in wireless sensor networks. *IEEE Communications Surveys Tutorials*, 8(2):2–23, 2nd Quarter 2006.

[27] Qiang Huang, Johnas Cukier, Hisashi Kobayashi, Bede Liu, and Jinyun Zhang. Fast authenticated key establishment protocols for self-organizing sensor networks. In *Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications*, WSNA'03, pages 141–150, New York, USA, 2003. ACM.

[28] Donggang Liu and Peng Ning. Multilevel μTESLA: Broadcast authentication for distributed sensor networks. *ACM Transaction on Embedded Computing Systems*, 3:800–836, November 2004.

[29] Sencun Zhu, Sanjeev Setia, and Sushil Jajodia. LEAP: efficient security mechanisms for large-scale distributed sensor networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, CCS'03, pages 62–72, New York, USA, 2003. ACM.

[30] Donggang Liu, Peng Ning, Sencun Zhu, and Sushil Jajodia. Practical broadcast authentication in sensor networks. In *Proceedings of the the 2nd Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, pages 118–132. IEEE Computer Society, 2005.

[31] Yoshinori Okazaki, Izuru Sato, and Shigeki Goto. A new intrusion detection method based on process profiling. In *Proceedings of the Symposium on*

Applications and the Internet, SAINT'02, pages 82–91, Washington, DC, USA, 2002. IEEE Computer Society.

[32] David Wagner. Resilient aggregation in sensor networks. In *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks*, SASN'04, pages 78–87, New York, USA, 2004. ACM.

[33] Avinash Srinivasan and Jie Wu. *A survey on secure localization in wireless sensor networks*, pages 1–26. Taylor and Francis Group, CRC Press, 2008.

[34] Jeremy Elson and Kay Römer. Wireless sensor networks: a new regime for time synchronization. *ACM SIGCOMM Computer Communication Review*, 33:149–154, January 2003.

[35] Saurabh Ganeriwal, Laura K. Balzano, and Mani B. Srivastava. Reputation-based framework for high integrity sensor networks. *ACM Transactions on Sensor Networks*, 4:15:1–15:37, June 2008.

[36] Sapon Tanachaiwiwat, Pinalkumar Dave, Rohan Bhindwale, and Ahmed Helmy. Location-centric isolation of misbehavior and trust routing in energy-constrained sensor networks. In *Proceedings of IEEE International Conference on Performance, Computing, and Communications*, IPCCC'04, pages 463–469, April 2004.

[37] Wikipedia. Key management. http://en.wikipedia.org/wiki/Key-management, accessed on February 24, 2011.

[38] Naveed Salman and Andrew Kemp. Overview of the IEEE 802.15.4 standards family for low rate wireless personal area networks. In *Proceedings of the 7th International Symposium on Wireless Communication Systems*, ISWCS'10, September 2010.

[39] Wenliang Du, Jing Deng, Yunghsiang S. Han, Pramod K. Varshney, Jonathan Katz, and Aram Khalili. A pairwise key predistribution scheme for wireless sensor networks. *ACM Transactions on Information System Security*, 8:228–258, May 2005.

[40] Seyit A. Camtepe and Bülent Yener. Key distribution mechanisms for wireless sensor networks: A survey (tr-05-07). Technical report, Rensselaer Polytechnic Institute, Computer Science Department, 2005.

[41] Yang Xiao, Venkata Krishna Rayi, Bo Sun, Xiaojiang Du, Fei Hu, and Michael Galloway. A survey of key management schemes in wireless sensor networks. *Computer Communications*, 30:2314–2341, September 2007.

[42] Haowen Chan, Adrian Perrig, and Dawn Song. Random key predistribution schemes for sensor networks. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, SP'03, pages 197–213, Washington, DC, USA, May 2003. IEEE Computer Society.

[43] Wenliang Du, Jing Deng, Yunghsiang Han, Shigang Chen, and Pramod K. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. In *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies*, INFOCOM'04, pages 586–597, March 2004.

[44] Carlo Blundo, Alfredo De Santis, Ugo Vaccaro, Amir Herzberg, Shay Kutten, and Moti Yong. Perfectly secure key distribution for dynamic conferences. *Information and Computation*, 146:1–23, October 1998.

[45] Donggang Liu and Peng Ning. Establishing pairwise keys in distributed sensor networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, CCS'03, pages 52–61, New York, USA, 2003. ACM.

[46] Donggang Liu, Peng Ning, and Rongfang Li. Establishing pairwise keys in distributed sensor networks. *ACM Transactions on Information System Security*, 8:41–77, February 2005.

[47] Donggang Liu and Peng Ning. Improving key predistribution with deployment knowledge in static sensor networks. *ACM Transactions on Senor Networks*, 1:204–239, November 2005.

[48] Rolf Blom. An optimal class of symmetric key generation systems. In *Proceeding of the Workshop on Advances in Cryptology: Theory and Application*

of Cryptographic Techniques, EUROCRYPT'84, pages 335–338, New York, USA, 1985. Springer-Verlag Inc.

[49] Zhen Yu and Yong Guan. A robust group-based key management scheme for wireless sensor networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference*, WCNC'05, pages 1915–1920, March 2005.

[50] Seyit A. Çamtepe and Bülent Yener. Combinatorial design of key distribution mechanisms for wireless sensor networks. *IEEE/ACM Transactions on Networking*, 15:346–358, April 2007.

[51] Mohamed Eltoweissy, Mohamed Younis, and Kajaldeep Ghumman. Lightweight key management for wireless sensor networks. In *Proceedings of the IEEE International Conference on Performance, Computing, and Communications*, pages 813–818, 2004.

[52] Mohamed Eltoweissy, Hossain Heydari, Linda Morales, and Hal Sudborough. Combinatorial optimization of group key management. *Journal of Network and System Management*, 12:33–50, March 2004.

[53] Mohamed Younis, Kajaldeep Ghumman, and Mohamed Eltoweissy. Location-aware combinatorial key management scheme for clustered sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 17(8):865 –882, August 2006.

[54] Mohamed Eltoweissy, Mohammed Moharrum, and Ravi Mukkamala. Dynamic key management in sensor networks. *IEEE Communications Magazine*, 44(4):122–130, April 2006.

[55] D. Wallner, E. Harder, and R. Agee. Key management for multicast: Issues and architectures. In *Internet Engineering Task Force*, volume 2627. IETF RFC Editor, United States, June 1999.

[56] Alan T. Sherman and David A. McGrew. Key establishment in large dynamic groups using one-way function trees. *IEEE Transactions on Software Engineering*, 29(5):444–458, May 2003.

[57] Roberto. Di Pietro, Luigi V. Mancini, Yee Wei Law, Sandro. Etalle, and Paul J. M. Havinga. LKHW: a directed diffusion-based secure multicast scheme for wireless sensor networks. In *Proceedings of the International Conference on Parallel Processing Workshops*, ICPPW'03, pages 397–406, October 2003.

[58] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom'00, pages 56–67, New York, USA, 2000. ACM.

[59] Ju Hyung Son, Jun Sik Lee, and Seung Woo Seo. Topological key hierarchy for energy-efficient group key management in wireless sensor networks. *Wirelless Personal Communications*, 52:359–382, January 2010.

[60] Loukas Lazos and Radha Poovendran. Secure broadcast in energy-aware wireless sensor networks. In *IEEE international symposiumon advances in wireless communications*, ISWC'02, 2002.

[61] Loukas Lazos and Radha Poovendran. Energy-aware secure multicast communication in ad-hoc networks using geographic location information. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, ICASSP'03, pages 201–204, April 2003.

[62] Wei Chi Ku and Shuai Min Chen. An improved key management scheme for large dynamic groups using one-way function trees. In *Proceedings of the International Conference on Parallel Processing Workshops*, ICPPW'03, pages 391–396, October 2003.

[63] Xuxin Xu, Lingyu Wang, Amr Youssef, and Bo Zhu. Preventing collusion attacks on the one-way function tree (OFT) scheme. In *Proceedings of the 5th International Conference on Applied Cryptography and Network Security*, ACNS'07, pages 177–193, Berlin, Heidelberg, 2007. Springer-Verlag.

[64] David J. Malan, Matt Welsh, and Michael D. Smith. A public key infrastructure for key distribution in tinyos based on elliptic curve cryptography. In *Proceedings of the First Annual IEEE Communications Society Conference*

on Sensor and Ad Hoc Communications and Networks, SECON'04, pages 71–80, October 2004.

[65] Ronald Watro, Derrick Kong, Sue-fen Cuti, Charles Gardiner, Charles Lynn, and Peter Kruus. TinyPK: securing sensor networks with public key technology. In *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks*, SASN'04, pages 59–64, New York, USA, 2004. ACM.

[66] Gunnar Gaubatz, Jens P. Kaps, Erdinc Ozturk, and Berk Sunar. State of the art in ultra-low power public key cryptography for wireless sensor networks. In *Proceeding of the 3rd IEEE International Conference on Pervasive Computing and Communications Workshops*, PerCom'05, pages 146–150, March 2005.

[67] Einar Mykletun, Joao Girao, and Dirk Westhoff. Public key based cryptoschemes for data concealment in wireless sensor networks. In *Proceedings of the IEEE International Conference on Communications*, ICC'06, pages 2288–2295, June 2006.

[68] Rabia Riaz, Ayesha Naureen, Attiya Akram, Ali Hammad Akbar, KiHyung Kim, and Farooq Ahmed. A unified security framework with three key management schemes for wireless sensor networks. *Computer Communications*, 31(18):4269–4280, 2008. Secure Multi-mode Systems and their Applications for Pervasive Computing.

[69] Xiaojiang Du, Mohsen Guizani, Yang Xiao, and HsiaoHwa Chen. A routing-driven elliptic curve cryptography based key management scheme for heterogeneous sensor networks. *IEEE Transactions on Wireless Communications*, 8(3):1223–1229, March 2009.

[70] Sk. Md. Mizanur Rahman and Khalil El. Khatib. Private key agreement and secure communication for heterogeneous sensor networks. *Journal of Parallel and Distributed Computing*, 70:858–870, August 2010.

[71] Manel Boujelben, Omar Cheikhrouhou, Mohamed Abid, and Habib Youssef. Establishing pairwise keys in heterogeneous two-tiered wireless sensor networks. In *Proceedings of the 3rd International Conference on Sensor Technologies and Applications*, SENSORCOMM'09, pages 442–448, June 2009.

[72] Liping Zhang, Guohua Cui, and Zhigang Yu. An ID-based key predistribution scheme for wireless sensor networks. In *Proceedings of the 4th International Conference on Wireless Communications, Networking and Mobile Computing*, WiCOM'08, pages 1–4, October 2008.

[73] Manel Boujelben, Omar Cheikhrouhou, Mohamed Abid, and Habib Youssef. A pairing identity based key management protocol for heterogeneous wireless sensor networks. In *the International Conference on Network and Service Security*, N2S'09, pages 1–5, June 2009.

[74] Chris Karlof, Naveen Sastry, and David Wagner. TinySec: a link layer security architecture for wireless sensor networks. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, SenSys'04, pages 162–175, New York, USA, 2004. ACM.

[75] Fan Ye, Haiyun Luo, Songwu Lu, and Lixia Zhang. Statistical en-route filtering of injected false data in sensor networks. *IEEE Journal on Selected Areas in Communications*, 23(4):839–850, April 2005.

[76] Sencun Zhu, Sanjeev Setia, Sushil Jajodia, and Peng Ning. Interleaved hop-by-hop authentication against false data injection attacks in sensor networks. *ACM Transactioins on Senor Networks*, 3:1–33, August 2007.

[77] Wenliang Du, Ronghua Wang, and Peng Ning. An efficient scheme for authenticating public keys in sensor networks. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc'05, pages 58–67, New York, USA, 2005. ACM.

[78] Tieyan Li, Guilin Wang, and Robert H. Deng. Security analysis on a family of ultra-lightweight rfid authentication protocols. *Journal of Software*, 3(3):1–10, March 2008.

[79] Claude. E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5:3–55, January 2001.

[80] Jessica Staddon, Sara K. Miner, Matthew K. Franklin, Dirk Balfanz, Michael Malkin, and Drew Dean. Self-healing key distribution with revocation. In

Proceedings of the IEEE Symposium on Security and Privacy, pages 241–257, 2002.

[81] Donggang Liu, Peng Ning, and Kun Sun. Efficient self-healing group key distribution with revocation capability. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, CCS'03, pages 231–240, New York, USA, 2003. ACM.

[82] Dowon Hong and Ju Sung Kang. An efficient key distribution scheme with self-healing property. *IEEE Communications Letters*, 9(8):759–761, August 2005.

[83] Carlo Blundo, Paolo D'Arco, and Massimilian Listo. A new self-healing key distribution scheme. In *Proceedings of the 8th IEEE International Symposium on Computers and Communication*, volume 2 of *ISCC'03*, pages 803–808, July 2003.

[84] Sun Haibo, Lin Dongdai, and Xue Rui. An improved efficient self-healing group key distribution. In *Proceedings of the IEEE International Symposium on Communications and Information Technology*, volume 1 of *ISCIT'05*, pages 192–196, October 2005.

[85] Biming Tian and Mingxing He. A self-healing key distribution scheme with novel properties. *International Journal of Network Security*, 7:147–152, September 2008.

[86] Ratna Dutta, Eechien Chang, and Sourav Mukhopadhyay. Efficient self-healing key distribution with revocation for wireless sensor networks using one way key chains. In *Applied Cryptography and Network Security*, volume 4521 of *Lecture Notes in Computer Science*, pages 385–400. Springer Berlin / Heidelberg, 2007.

[87] Carlo Blundo, Paolo D'arco, Alfredo De Santis, and Massimiliano Listo. Design of self-healing key distribution schemes. *Dessign, Codes, and Cryptography*, 32:15–44, May 2004.

[88] Xukai Zou and Yuan-Shun Dai. A robust and stateless self-healing group key management scheme. In *Proceeding of the International Conference on Communication Technology*, ICCT'06, pages 1–4, November 2006.

[89] Sara Miner More, Michael Malkin, Jessica Staddon, and Dirk Balfanz. Sliding window self-healing key distribution. In *Proceedings of the ACM workshop on Survivable and Self-regenerative Systems: in association with the 10th ACM Conference on Computer and Communications Security*, SSRS'03, pages 82–90, New York, USA, 2003. ACM.

[90] Germán Sáez. On threshold self-healing key distribution schemes. In *Cryptography and Coding*, volume 3796 of *Lecture Notes in Computer Science*, pages 340–354. Springer Berlin / Heidelberg, 2005.

[91] Germán Sáez. Self-healing key distribution schemes with sponsorization. In *Communications and Multimedia Security*, volume 3677 of *Lecture Notes in Computer Science*, pages 22–31. Springer Berlin / Heidelberg, 2005.

[92] Ratna Dutta, Sourav Mukhopadhyay, Amitabha Das, and Sabu Emmanuel. Generalized self-healing key distribution using vector space access structure. In *Proceedings of the 7th International IFIP-TC6 Networking Conference on Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet*, NETWORKING'08, pages 612–623, Berlin, Heidelberg, 2008. Springer-Verlag.

[93] Biming Tian, Song Han, Tharam S. Dillon, and Sajal Das. A self-healing key distribution scheme based on vector space secret sharing and one way hash chains. In *Proceedings of the International Symposium on a World of Wireless, Mobile and Multimedia Networks*, WoWMoM'08, pages 1–6, June 2008.

[94] Yixin Jiang, Chuang Lin, Minghui Shi, and Xuemin (Sherman) Shen. Self-healing group key distribution with time-limited node revocation for wireless sensor networks. *Ad Hoc Networks*, 5(1):14–23, 2007. Security Issues in Sensor and Ad Hoc Networks.

[95] Firdous Kausar, Sajid Hussain, Jong Hyuk Park, and Ashraf Masood. Secure group communication with self-healing and rekeying in wireless sensor networks. In *Proceedings of the 3rd International Conference on Mobile Ad Hoc and Sensor Networks*, MSN'07, pages 737–748, Berlin, Heidelberg, 2007. Springer-Verlag.

[96] Sencun Zhu, Sanjeev Setia, and Sushil Jajodia. Adding reliable and self-healing key distribution to the subset difference group rekeying method for secure multicast. In *Group Communications and Charges. Technology and Business Models*, volume 2816 of *Lecture Notes in Computer Science*, pages 107–118. Springer Berlin / Heidelberg, 2003.

[97] Muhammad Junaid Bohio and Ali Miri. Self-healing group key distribution. *International Journal of Network Security*, 1:110–117, September 2005.

[98] Biming Tian, Song Han, and Tharam S. Dillon. A self-healing and mutual-healing key distribution scheme using bilinear pairings for wireless networks. In *Proceedings of the IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, EUC'08, pages 208–215, December 2008.

[99] Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, USA, 1991.

[100] Carlo Blundo, Paolo D'Arco, and Alfredo De Santis. On self-healing key distribution schemes. *IEEE Transactions on Information Theory*, 52(12):5455–5467, December 2006.

[101] Rutta Dutta and Sourav Mukhopadhyay. Improved self-healing key distribution with revocation in wireless sensor network. In *Proceedings of the IEEE Wireless Communications and Networking Conference*, WCNC'07, pages 2963–2968, March 2007.

[102] Ratna Dutta and Sourav Mukhopadhyay. Designing scalable self-healing key distribution schemes with revocation capability. In *Parallel and Distributed Processing and Applications*, volume 4742 of *Lecture Notes in Computer Science*, pages 419–430. Springer Berlin/Heidelberg, 2007.

[103] Dalit Naor, Moni Naor, and Jeff Lotspiech. Revocation and tracing schemes for stateless receivers. In *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, volume 2139 of *CRYPTO'01*, pages 41–62, London, UK, 2001. Springer-Verlag.

[104] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of Advances in cryptology*, CRYPTO'84, pages 47–53, New York, USA, 1985. Springer-Verlag New York, Inc.

[105] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO'01, pages 213–229, London, UK, 2001. Springer-Verlag.

[106] Xinjun Du, Ying Wang, Jianhua Ge, and Yumin Wang. An id-based broadcast encryption scheme for key distribution. *IEEE Transactions on Broadcasting*, 51(2):264–266, June 2005.

[107] Bruce Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. John Wiley and Sons, 2nd edition, 1996.

[108] Adi Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, November 1979.

[109] Ernest F. Brickell. Some ideal secret sharing schemes. In *Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques on Advances in Cryptology*, pages 468–475, New York, USA, 1990. Springer-Verlag New York, Inc.

[110] Wikipedia. Cryptographically secure pseudorandom number generator. http://en.wikipedia.org/wiki/Cryptographically-secure-pseudorandom-number-generator, accessed on July 20, 2010.

[111] Levent Ertaul and Madhavi Ganta. Security in wireless sensor networks-a study. In *Proceedings of the International Conference on Wireless Network*, ICWN'09, pages 33–39, 2009.

[112] Yixin Jiang, Chuang Lin, Minghui Shi, and Xuemin (Sherman) Shen. Hash-Binary-Tree based group key distribution with time-limited node revocation. In *Security in Distributed and Networking Systems: Computer and Network Security*, 2006.

[113] Jay F. Nunamaker, Jr., Minder Chen, and Titus D. M. Purdin. Systems development in information systems research. *Journal of Management Information Systems*, 7:89–106, October 1990. Special issue on management support systems.

[114] Frada Burstein and Shirley Gregor. The systems development or engineering approach to research in information systems: An action research perspective. In *Proceedings of the 10th Australian Confenece on Information Systems*, ACIS'99, pages 122–134, Wellington, NZ, 1999.

[115] David D. Hwang, Bocheng Charles Lai, and Ingrid Verbauwhede. Energy-memory-security tradeoffs in distributed sensor networks. In *Ad-Hoc, Mobile, and Wireless Networks*, volume 3158 of *Lecture Notes in Computer Science*, pages 630–630. Springer Berlin / Heidelberg, 2004.

[116] Chunlai Du, Mingzeng Hu, Hongli Zhang, and Weizhe Zhang. Anti-collusive self-healing key distribution scheme with revocation capability. *Information Technology Journal*, 8(4):619–624, 2009.

[117] Ken Masica. Recommended practices guide securing ZigBee wireless networks in process control system environments (Draft), April 2007.

[118] Gianluca Dini and Marco Tiloca. Considerations on security in ZigBee networks. In *Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trust-worthy Computing*, SUTC'10, pages 58–65, Washington, DC, USA, 2010. IEEE Computer Society.

[119] Claude Castelluccia and Angelo Spognardi. Rok: A robust key predistribution protocol for multi-phase wireless sensor networks. In *Proceedings of the 3rd International Conference on Security and Privacy in Communications Networks and the Workshops*, SecureComm'07, pages 351–360, September 2007.

[120] Kubra Kalkan, Sinem Yilmaz, Omer Zekvan Yilmaz, and Albert Levi. A highly resilient and zone-based key predistribution protocol for multiphase wireless sensor networks. In *Proceedings of the 5th ACM Symposium on QoS and Security for Wireless and Mobile Networks*, Q2SWinet'09, pages 29–36, New York, USA, 2009. ACM.

[121] Murat Ergun, Albert Levi, and Erkay Savas. A resilient key predistribution scheme for multiphase wireless sensor networks. In *Proceeding of the 24th International Symposium on Computer and Information Sciences*, ISCIS'09, pages 375–380, September 2009.

[122] Federal Information. Fipspub180-1. http://www.itl.nist.gov/fipspubs/fip180-1.htm, accessed on July 12, 2010, 1995.

[123] Computer Security Division Computer Security Resource Center. Fips pub 180-2. http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf, accessed on July 12, 2010, 2002.

[124] Patrick Kinney. ZigBee technology: Wireless control that simply works. In *Communications Design Conference*, White Paper, pages 1–20. Kinney Consulting LLC, October 2003.

[125] ZigBee Alliance. Zigbee home automation public application profile. http://www.zigbee.org/, October 2007.

[126] ZigBee Alliance. Zigbee smart energy profile specification. http://www.zigbee.org/, December 2008.

[127] Jing Sun and Xiaofen Zhang. Study of ZigBee wireless mesh networks. In *Proceeding of the 9th International Conference on Hybrid Intelligent Systems*, volume 2 of *HIS'09*, pages 264–267, August 2009.

[128] Sandro Rafaeli and David Hutchison. A survey of key management for secure group communication. *ACM Computing Surveys*, 35:309–329, September 2003.

[129] Tim Landstra, Maciej Zawodniok, and Sarangapani Jagannathan. Energy-efficient hybrid key management protocol for wireless sensor networks. In *Proceedings of the 32nd IEEE Conference on Local Computer Networks*, pages 1009–1016, Washington, DC, USA, 2007. IEEE Computer Society.

[130] Crossbow Technology. Micaz datasheet (document part number: 6020-0060-04 rev a). http://www.openautomation.net/uploadsproductos/micaz-datasheet.pdf, accessed on July 20, 2010.

[131] Wendi Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications*, 1(4):660–670, October 2002.

[132] Piyush. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2):388–404, March 2000.

[133] Enrique J. Duarte-Melo and Mingyan Liu. Data-gathering wireless sensor networks: organization and capacity. *Computer Networks*, 43:519–537, November 2003. Special Issue: Wireless sensor networks.

[134] Nils Gura, Arun Patel, Arvinderpal Wander, Hans Eberle, and Sheueling Chang Shantz. Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs. In *Cryptographic Hardware and Embedded Systems*, pages 119–132, 2004.

[135] Alireza Hodjat and Ingrid Verbauwhede. The energy cost of secrets in ad-hoc networks (short paper). In *Proceedings of IEEE CAS Workshop on Wireless Communication and Networking*, September 2002.

[136] Sajid Hussain, Md Shafayat Rahman, and Laurence T. Yang. Key predistribution scheme using keyed-hash chain and multipath key reinforcement for wireless sensor networks. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications*, PerCom'09, pages 1–6, Washington, DC, USA, March 2009. IEEE Computer Society.

[137] Noam Kogan, Yuval Shavitt, and Avishai Wool. A practical revocation scheme for broadcast encryption using smartcards. *ACM Transactions on Information System Security*, 9:325–351, August 2006.

[138] Taejoon Park and Kang G. Shin. LiSP: A lightweight security protocol for wireless sensor networks. *ACM Transactions on Embedded Computing Systems*, 3:634–660, August 2004.

[139] Mohamed Eltoweissy, Ashraf Wadaa, Stephan Olariu, and Larry Wilson. Group key management scheme for large-scale sensor networks. *Ad Hoc Networks*, 3(5):668–688, 2005. Data Communication and Topology Control in Ad Hoc Networks.

[140] Song Han, Elizabeth Chang, Li Gao, and Tharam Dillon. Taxonomy of attacks on wireless sensor networks. In *EC2ND'05*, pages 97–105. Springer London, 2006.

[141] David W. Carman, Peter S. Kruus, and Brian J. Matt. Constraints and approaches for distributed sensor network security. Technical Report 010, NAI Labs, The Security Research Division Network Associates, Inc., September 2000.

[142] David M. Balenson, David Mcgrew, and Alan Sherman. Key management for large dynamic groups: one-way function trees and amortized initialization. *Internet-Draft*, August 25 2000.

[143] Fengling Han, Jiankun Hu, Xinghuo Yu, and Yi Wang. Fingerprint images encryption via multi-scroll chaotic attractors. *Applied Mathematics and Computation*, 185(2):931–939, 2007. Special Issue on Intelligent Computing Theory and Methodology.

[144] Song Han, Biming Tian, Mingxing He, and Elizabeth Chang. Efficient threshold self-healing key distribution with sponsorization for infrastructureless wireless networks. *IEEE Transactions on Wireless Communications*, 8(4):1876–1887, April 2009.

[145] Jiankun Hu and Fengling Han. A pixel-based scrambling scheme for digital medical images protection. *Journal of Network and Computer Applications*, 32(4):788–794, 2009.

[146] Jiankun Hu, Hsiao Hwa Chen, and Tingwei Hou. A hybrid public key infrastructure solution (HPKI) for HIPAA privacy/security regulations. *Computer Standards & Interfaces*, 32:274–280, October 2010.

[147] Kejie Lu, Yi Qian, and Jiankun Hu. A framework for distributed key management schemes in heterogeneous wireless sensor networks. In *the 25th IEEE International Performance, Computing, and Communications Conference*, IPCCC'06, pages 213–520, April 2006.

[148] Fahim Sufi, Fengling Han, Ibrahim Khalil, and Jiankun Hu. A chaos-based encryption technique to protect ecg packets for time critical telecardiology applications. *Security and Communication Networks*, 4(5):515–524, 2011.

[149] Kai Xi, Tohari Ahmad, Fengling Han, and Jiankun Hu. A fingerprint based bio-cryptographic security protocol designed for client/server authentication

in mobile computing environment. *Security and Communication Networks*, 4(5):487–499, 2011.

[150] Chung Kei Wong, Mohamed Gouda, and Simon S. Lam. Secure group communications using key graphs. *IEEE/ACM Transactions on Networking*, 8:16–30, February 2000.

[151] Amos Fiat and Tamir Tassa. Dynamic traitor tracing. In *Advances in Cryptology, CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 784–784. Springer Berlin / Heidelberg, 1999.

[152] Dani Halevy and Adi Shamir. The LSD broadcast encryption scheme. In *Advances in Cryptology, CRYPTO'02*, volume 2442 of *Lecture Notes in Computer Science*, pages 145–161. Springer Berlin / Heidelberg, 2002.

[153] Mohamed Moharrum, Mohamed Eltoweissy, and Ravi Mukkamala. Dynamic combinatorial key management scheme for sensor networks: Research articles. *Wireless Communication and Mobile Computing*, 6:1017–1035, November 2006. Wireless Ad Hoc Networks: Technologies and Challenges.

[154] Jae Choon Cha and Jung Hee Cheon. An identity-based signature from gap diffie-hellman groups. In *Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography*, PKC'03, pages 18–30, London, UK, 2003. Springer-Verlag.

[155] Kenneth G. Paterson. ID-based signatures from pairings on elliptic curves. *Electronics Letters*, 38(18):1025–1026, August 2002.

[156] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. In *Proceeding of the Symposium on Cryptography and Informatioin Security*, SCIS'00, January 2000.

[157] Florian Hess. Efficient identity based signature schemes based on pairings. In *Selected Areas in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 310–324. Springer Berlin / Heidelberg, 2003.

[158] Srdjan Capkun and Jean P. Hubaux. Secure positioning of wireless devices with application to sensor networks. In *Proceedings of the 24th Annual Joint*

Conference of the IEEE Computer and Communications Societies, INFO-COM'05, pages 1917–1928, March 2005.

[159] Yanchao Zhang, Wei Liu, and Yuguang Fang. Secure localization in wireless sensor networks. In *IEEE Military Communications Conference*, MIL-COM'05, pages 3169–3175, October 2005.

[160] Song Han, Jie Wang, and Wanquan Liu. An efficient identity-based group signature scheme over elliptic curves. In *Universal Multi-service Networks*, volume 3262 of *Lecture Notes in Computer Science*, pages 417–429. Springer Berlin / Heidelberg, 2004.

[161] Paulo Barreto, Hae Kim, Ben Lynn, and Michael Scott. Efficient algorithms for pairing-based cryptosystems. In *Advances in Cryptology, CRYPTO'02*, volume 2442 of *Lecture Notes in Computer Science*, pages 354–369. Springer Berlin / Heidelberg, 2002.

[162] Steven D. Galbraith, Keith Harrison, and David Soldera. Implementing the tate pairing. In *Proceedings of the 5th International Symposium on Algorithmic Number Theory*, ANTS-V, pages 324–337, London, UK, 2002. Springer-Verlag.

[163] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48:203–209, 1987.

[164] Ameer Ahmed Abbasi and Mohamed Younis. A survey on clustering algorithms for wireless sensor networks. *Computer Communications*, 30(14-15):2826–2841, 2007. Network Coverage and Routing Schemes for Wireless Sensor Networks.

[165] Yingjie Wang, Jianhua Li, Ling Tie, and Qiang Li. An efficient key management for large dynamic groups. In *Proceedings of the 2nd Annual Conference on Communication Networks and Services Research*, pages 131–136, Washington, DC, USA, 2004. IEEE Computer Society.

[166] Marcos A. Simplício, Jr., Paulo S. L. M. Barreto, Cintia B. Margi, and Tereza C. M. B. Carvalho. A survey on key management mechanisms for

distributed wireless sensor networks. *Computer Networks*, 54:2591–2612, October 2010.

[167] Ian F. Blake, Gadiel Seroussi, and Nigel P. Smart. *Elliptic Curves in Cryptography*. Lecture Note Series. Cambridge University Press, Cambridge, 1999.

[168] Scott Vanstone. Deployments of elliptic curve cryptography. In *Proceedings of the 9th workshop on Elliptic Curve Cryptography*, 2005.

[169] Delphine Christin, Parag S. Mogre, and Matthias Hollick. Survey on wireless sensor network technologies for industrial automation: The security and quality of service perspectives. *Future Internet*, 2:96–125, 2010.

Appendix A

The Selected Publications

A.1 A Mutual-healing Key Distribution Scheme in Wireless Sensor Networks

A.2 Self-Healing Key Distribution Schemes for Wireless Networks: A Survey

A.3 A Key Management Protocol for Multi-phase Hierarchical Wireless Sensor Networks

Journal of Network and Computer Applications 34 (2011) 80–88

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

A mutual-healing key distribution scheme in wireless sensor networks

Biming Tian[a], Song Han[a], Jiankun Hu[b,*], Tharam Dillon[a]

[a] DEBI Institute, Curtin University, Perth 6845, Australia
[b] School of Computer Science and IT, RMIT University, Melbourne 3001, Australia

A R T I C L E I N F O

Article history:
Received 22 January 2010
Received in revised form
1 September 2010
Accepted 5 September 2010

Keywords:
Sensor network
Security
Self-healing
Key distribution

A B S T R A C T

How to establish secure session keys is one of the central tasks for wireless sensor network communications. General key distribution schemes for traditional computer networks could not be directly shifted to wireless sensor network environments as broadcast messages may be lost due to sensor network internal factors or external attacks. Self-healing key distribution schemes, therefore, have been proposed to address packet loss issues since 2002. The essential issue that self-healing key distribution mechanism addressed is the fixed-number of broadcast messages (excluding the last broadcast message) loss. In other words, a node could not recover its new session keys if a node has missed more than a fixed number broadcast messages or the last broadcast message in a self-healing key distribution scheme for wireless sensor networks. This paper aims to address this emerged issue and provide a new key distribution scheme: mutual-healing key distribution scheme for wireless sensor networks. This mutual-healing key distribution can enable a node in a wireless sensor network to recover its new session key although its last broadcast message was lost. A formal definition for mutual-healing key distribution will also be proposed in this paper. The proposed mutual-healing key distribution scheme is based on bilinear pairings. The scheme is collusion-free for any coalition of non-authorized nodes. Each node's private key has nothing to do with the number of revoked nodes and can be reused as long as it is not disclosed. The storage overhead for each node is a constant.

1. Introduction

Key management including key distribution and key update is significant for maintaining private communications in any dynamic networks (Balenson et al., 1999; Ku and Chen, 2003; Staddon et al., 2002; Liu et al., 2003; Jiang et al., 2007; Han et al., 2007; Han et al., 2009; Hong and Kang, 2005; Hu and Han, 2009; Hu et al., 2010; Eltoweissy et al., 2004a; Lu et al., 2006; Sufi et al., 2010; Xi et al., 2010). Wireless sensor networks, especially mobile ad hoc networks, as one of dynamic networks, therefore rely on a secure key management. This is because membership frequently changes in large dynamic group communications of wireless sensor networks. In order to keep the security of communications, the session key has to be updated on each membership change. Therefore, one of the important questions is how to distribute and update session keys in a secure and efficient way for large dynamic wireless sensor networks. In recent years, many schemes on distributing session keys for large group communication have been proposed. These existing schemes focused on different key updating mechanism. For example, LKH (Logical Key Hierarchy)-based schemes (Wong et al., 2000) and OFT (One-way Function Tree) based schemes (Balenson et al., 1999; Ku and Chen, 2003) devote to reduce the size of the rekeying message. Broadcast encryption addresses the problem of sending encrypted messages to a large node group so that the encrypted messages can only be decrypted by a dynamic changing privileged subset (Fiat and Tessa, 2001; Halevy and Shamir, 2002; Naor and Pinkas, 2000). EBS (Exclusion Basis System) based approach was proposed in Eltoweissy et al. (2004b), and then be put into use for sensor networks in Eltoweissy et al. (2004a) and Moharrum et al. (2006). The members store less number of keys than LKH tree for the multicast group of the same size. All these literatures supposed that underlying networks are reliable. However, how to distribute session keys for unreliable wireless networks, in a manner that is resistant to packet loss, is an issue that has not been addressed deeply.

Packet loss happens frequently in wireless sensor networks. The key distribution broadcast for a particular session might never reach some nodes. A naive solution is requesting re-transmission. On the one hand, both requesting and re-transmission messages would incur more communication overhead. In a very large communication group, such individual interactions place a heavy burden on the group manager. On the other hand, nodes may reveal their current locations by sending messages in some high security environments. All these issues can be addressed by self-healing key distribution schemes (Staddon et al., 2002; Liu et al., 2003; Blundo et al., 2004; More et al., 2003;

* Corresponding author. Tel.: +61 3 99259793; fax: +61 3 9662 1617.
E-mail addresses: biming.tian@cbs.curtin.edu.au (B. Tian), song.han@cbs.curtin.edu.au (S. Han), jiankun@cs.rmit.edu.au (J. Hu), haram.dillon@cbs.curtin.edu.au (T. Dillon).

1084-8045/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jnca.2010.09.002

Sáez, 2005a, b; Dutta and Mukhopadhyay, 2007; Muhammad and Ali, 2005; Jiang et al., 2007; Zou and Dai, 2006; Dutta et al., 2007; Han et al., 2009; Hong and Kang, 2005; Kausar et al., 2007). Self-healing key distribution enables large and dynamic group nodes to establish group keys over an unreliable network for secure communications. The main property of self-healing key distribution schemes is that, even if at the beginning of certain sessions some broadcast packets get lost, group nodes are still capable of recovering the session key for those sessions simply by using the broadcasts they have received at a previous session and the packets they will receive at a subsequent one. In this kind of scheme, nodes do not need to send any requesting message to the group manager and do not need to update their personal keys. In this regard, self-healing key distribution schemes are noninteractive ones which can hence reduce the network traffic, decrease the workload on the group manager, and lower the risk of node exposure through traffic analysis. Therefore, self-healing key distribution schemes are desirable for both efficiency and security reasons in wireless sensor networks.

An emerged scenario is a node in a wireless sensor network may miss its last broadcast message. This node is therefore not able to recover its session key for the last session in a self-healing key distribution scheme. This is because self-healing key distribution needs the broadcast messages the node received in the previous sessions and those in the subsequent sessions (Suppose the current session is the one where broadcast messages are lost). To tackle this scenario, mutual-healing key distribution mechanism will be introduced. Mutual-healing key distribution can help a node who missed the last broadcast message to recover the session key for this last session.

Possible applications: Group communications over low-cost channels in different fields can benefit from mutual-healing key distribution mechanism, especially for those settings in which session keys need to be used for a short time-period, due to frequent adding or deleting nodes. For example, video conference for commercial content distribution or electronic services in which the contents are highly sensitive. Self-healing key distribution schemes are also good levers in military-oriented operations, scientific explorations and rescue missions. The nodes in these environment are powered by batteries. They may experience short-term off-line and rejoin the group once the power is on again. Also the nodes may move in and out of communication range frequently and experience burst packet losses.

Contribution. The contribution of this paper are the following. First of all we define a security model of computationally secure self-healing key distribution scheme. This definition outperforms those in Staddon et al. (2002) and Liu et al. (2003) in two aspects. The first one is that there is no threshold on the number of revoked nodes. The scheme is collusion-free for any coalition of non-authorized nodes. The second one is that a node can recover from a single broadcast message all keys associated with sessions in which it belongs to the session group. These two properties add flexibility to self-healing key distribution scheme. We propose a self-healing key distribution scheme using bilinear pairings. The scheme is highlighted by several desirable features. Firstly, it is collusion-free for any coalition of non-authorized nodes. Secondly, the private key has nothing to do with the number of revoked nodes and can be reused as long as it is not disclosed. Thirdly, the storage overhead for node is a constant. Subsequently, we discuss the requirement, state formal definition, and develop technical details of mutual-healing mechanism. It is the first time to realize mutual-healing key distribution technique.

Organization. The rest of the paper is organized as follows: In Section 2, we present an overview of earlier works in the area of self-healing key distribution, mutual-healing, and Identity-based cryptography. In Section 3, we briefly introduce the preliminaries to be used in the design of mutual-healing key distribution protocol. In Section 4, we give system parameters, security model and a formal definition. In Section 5, a concrete construction for mutual-healing key distribution is proposed. In Section 6, we provide security analysis and efficiency comparison. We conclude this paper and point out future research in Section 7.

2. Related works

2.1. Self-healing key distribution schemes

The first self-healing key distribution with revocation scheme was introduced by Staddon et al. (2002). They presented formal definitions, lower bounds on the resources as well as some constructions. However, the constructions given in this paper suffer from high storage overhead and communication overhead. Since then, self-healing key distribution has been one of the hot research topics. Subsequent works focus on improving performance or adding some new properties which make the self-healing key distribution mechanism more flexible or more robust. Liu et al. generalized the definitions in Staddon et al. (2002) and gave some constructions in Liu et al. (2003). The scheme reduces communication overhead and storage overhead by introducing a novel personal key distribution technique. Blundo et al. (2004) showed an attack that can be applied to the first construction in Staddon et al. (2002), developed a new mechanism under a slightly modified framework. More et al. (2003) used a sliding window to address three problems in Staddon et al. (2002). The three problems were inconsistent robustness, high overhead and expensive maintenance costs. Dutta et al. developed a new self-healing key distribution scheme in Dutta and Mukhopadhyay (2007). The scheme has significant improvement in terms of both storage overhead and communication overhead. The schemes (Staddon et al., 2002; Liu et al., 2003; Blundo et al., 2004; More et al., 2003; Dutta and Mukhopadhyay, 2007) are based on Shamir's secret sharing. They have a common property: the maximum number of revoked nodes is constraint to the degree of the polynomial. Sáez (2005a,b) considered applying vector space secret sharing instead of Shamir's secret sharing schemes to design self-healing key distribution scheme. He made use of general monotone decreasing structures for the family of subsets of nodes that can be revoked instead of a threshold one. All of the schemes (Staddon et al., 2002; Liu et al., 2003; Blundo et al., 2004; More et al., 2003; Sáez, 2005a, b; Dutta and Mukhopadhyay, 2007) are unconditionally secure.

Computationally secure self-healing key distribution schemes emerged recently. They achieved good properties at the cost of slight relax security requirements. Jiang et al. proposed an efficient self-healing group key scheme with time-limited node revocation based on DDHC (dual directional hash chains) in Jiang et al. (2007). The performance of the proposed scheme was evaluated by both theory analysis and experiment data. The results showed that the scheme made a good balance between performance and security. The scheme in Dutta et al. (2007) also based on hash function. It reduced communication overhead and computation overhead greatly without any increase in the storage overhead.

In general, according to the security level, the existing self-healing key distribution schemes can be classified as unconditionally secure self-healing key distribution schemes and computationally secure self-healing key distribution scheme. The former has more strict security while the latter is more flexible and efficient. According to the cryptographic primitives, the schemes can be classified as polynomial secret sharing based schemes, vector space secret sharing based schemes, and hash function based schemes. Polynomial secret sharing is the most common technique used to realize self-healing key distribution. It was used in the pioneering paper (Staddon et al., 2002) and was followed by several subsequent works. However, the maximum number of revoked nodes is constraint to the degree of the polynomial. Vector space secret sharing based self-healing key distribution schemes consider a monotone decreasing family of revoked subset of nodes instead of a threshold structure. This general case makes the self-healing scheme more flexible and suitable for practical application. Both forward and backward securities can be guaranteed by hash function based self-healing key distribution schemes. However, the feature of resisting collusion of revoked nodes and new joined nodes cannot be assured, due to the properties of one-way hash function.

2.2. *Motivation for mutual-healing key distribution*

We cannot deny the novel property of the self-healing idea. However, some improvements are still necessary for the original scheme (Staddon et al., 2002). For example, it may allow an authorized subset of nodes in the group to sponsor a new node without the help of the group manager. Sáez (2005b) considered the feature that a coalition of nodes sponsor a node outside the group for one session. This feature added a dynamic character to the self-healing key distribution scheme. There are still other schemes (Hong and Kang, 2005; Kausar et al., 2007) which focus on improving the efficiency.

More et al. (2003) pointed out that the protocol in Staddon et al. (2002) suffered from inconsistent robustness. The so-called inconsistent robustness is that some session keys cannot be recovered if the corresponding broadcast messages are lost, no matter how many other update messages are received. For example, if the broadcast message for the last session gets lost, nodes cannot recover the last session key by themeless even they receive all the other broadcast messages. Subsequently, they used a sliding window to make error recovery consistently robust. That is, after the initial SET-UP procedure, any lost key can be recovered as long as two sufficiently close broadcast messages (one before it and the other after it) are received. Similar technique was used in Zou and Dai (2006). The size of the window can be dynamic adjusted according to the condition of networks. Both More et al. (2003) and Zou and Dai (2006) guaranteed that authorized nodes can recover window size session keys as long as they receive corresponding broadcast messages. However, how to recover the session key if the last broadcast message gets lost or more than sliding window number broadcast messages get lost has never been addressed clearly. Obviously, it is impossible to make nodes completely self-healing according to existing self-healing key distribution mechanism.

In view of some concrete applications, such as live and pay-per-view TV, have strictly requirement of freshness. The customers would better lose only a limited number of broadcast messages. In the group communication, the last session usually is of great importance. The authorized nodes would never like to miss it. Therefore it is significant to detect counterpart measures to deal with the aforementioned issues.

Muhammad and Ali (2005) considered incorporating the self-healing feature to SD(Subset Difference) method, which was first proposed by Naor et al. (2001). Some optimization techniques that can be used to reduce the overhead caused by the self-healing capability were proposed in the paper. At last, the idea of mutual-healing was discussed. One motivation behind mutual-healing was that, if a node has missed more than a fixed number of broadcast messages, it does not have to keep on waiting. Instead it can get assistance from its neighbors. Similarly, if a node misses the last broadcast message, it cannot recover the last session key by performing self-healing. To provide a countermeasure for this situation, it can look for assistance from its neighboring nodes too. However, the paper (Muhammad and Ali, 2005) only talked about the feasibility of mutual-healing without exploring any technical detail.

2.3. *Identity-based cryptography*

In identity-based cryptography, the public key of a user is some unique information about the identity of the user. Identity-based schemes can allow any party to generate a public key from a known identity value such as an ASCII string. A trusted third party, called the private key generator (PKG), generates the corresponding private keys. Therefore, any party without using certificates can verify the public key of a user. This eliminates the need for a public key distribution infrastructure.

Shamir (1984) proposed the first identity-based cryptography to alleviate many of the problems inherent with managing certificates in 1984. Boneh and Franklin (2001) proposed the first practical identity-based encryption scheme in 2001. Since then, many ID-based cryptographic schemes have been proposed using bilinear pairings. Inspired by the idea of Boneh and Franklin (2001), Du et al. proposed a broadcast encryption scheme for key distribution in Du et al. (2005). By extending the broadcast encryption scheme, we will proposed a mutual-healing key distribution scheme for wireless sensor networks.

3. Preliminaries

In this section, we briefly describe bilinear pairings, BDH (bilinear Diffie–Hellman) assumption and ID-based PKI (public key infrastructure).

3.1. *Bilinear pairings and BDH assumption*

Let G_1 and G_2 be two cyclic groups of order q for a large prime q. G_1 is a cyclic additive group and G_2 is a cyclic multiplicative group. We assume that the discrete logarithm problems in both G_1 and G_2 are hard. Let $e: G_1 \times G_1 \to G_2$ be a pairing which satisfies the following conditions:

- Bilinearity: $e(aP,bQ)=e(P,Q)^{ab}$, for $\forall P,Q \in G_1$ and $\forall a,b \in \mathbb{Z}_q^*$;
- Non-degeneracy: there exists $P \in G_1$ and $Q \in G_1$, such that $e(P,Q) \neq 1$; That is, for any point $P,Q \in G_1$, $e(P,Q)=1$ iff $P=O$.
- Computability: there exists an efficient algorithm to compute $e(P,Q)$ for any $P,Q \in G_1$.

BDH parameter generator: A BDH parameter generator $\mathcal{IG}$ is a probabilistic algorithm that takes a security parameter $0 < k \in \mathbb{Z}$, runs in polynomial time, and output the description of two groups G_1 and G_2 of the same order q and the description of an admissible bilinear map $e: G_1 \times G_1 \to G_2$.

BDH problem: Given $\langle P,aP,bP,cP \rangle$ for some $a,b,c \in \mathbb{Z}_q^*$, computes $e(P,P)^{abc} \in G_2$.

BDH assumption: There is no polynomial time algorithm to solve the BDH problem.

3.2. *ID-based public key infrastructure*

DLP (Discrete Logarithm Problem). Given two group elements P and Q, to find an integer $n \in \mathbb{Z}_q^*$, such that $Q=nP$ when such an integer exists.

ID-based PKI involves a trusted KGC(key generation center) and nodes. Nodes' private keys are calculated by KGC and send to the node via a secure channel. The basic operations consist of *set up* and *private key extraction*. When we use bilinear pairings to construct ID-based private/public keys, the operations can be implemented as follows: KGC runs BDH parameter generator to generate two groups G_1, G_2 and a bilinear pairing $e : G_1 \times G_1 \rightarrow G_2$. It chooses an arbitrary generator $P \in G_1$ and defines two cryptographic hash functions: $H_1 : \{0,1\}^* \rightarrow G_1, H_2 : G_2 \rightarrow \{0,1\}^*$.

- *Set up*: KGC chooses a random number $s \in \mathbb{Z}_q^*$ and set $P_{pub}=sP$. Then KGC publishes system parameters *params* = $\{G_1, G_2, q, P, P_{pub}, H_1, H_2\}$, and keeps s as master-key, which is only known by him.
- *Private key extraction*: A node submits its identity to KGC. KGC computes the node's public key $Q_{ID}=H_1(ID)$ and private key $S_{ID}=sQ_{ID}$, then privately returns $S_{ID}=sQ_{ID}$ to the node.

4. Security model and definition for mutual-healing key distribution

In this section, we present system parameters, security model and formal definition for mutual-healing key distribution, and a new key distribution scheme.

4.1. System parameters

Let $U=\{u_1,\ldots,u_n\}$ be the finite universe of nodes. Each node u_i has a unique identifier ID_i. A broadcast unreliable channel is available, and time is defined by a global clock. GM (Group Manager) sets up and manages, by means of adding and revoking operations, a communication group which is a dynamic subset of U. m denotes the number of sessions. Let $G_j \subseteq U$ be the communication group established by the group manager in session j. Each node is preloaded with a public/private key pair (Q_i, S_i) before deployment. The public/private key pair is used to recover the session keys as long as node u_i is not removed by GM from the group. Let $R_j \subseteq G_{j-1}$ denote the set of revoked group nodes in session j and $J_j \subset U \backslash G_{j-1}$ denote the set of nodes who join the group in session j with $R_j \cap J_j = \phi$. Hence, $G_j = (G_{j-1} \cup J_j) \backslash R_j$ for $j \geq 2$ and by definition $G_1=U$. Moreover, for $j \in \{1,\ldots,m\}$, the session key K_j is randomly chosen by GM and according to the uniform distribution. For any nonrevoked node $u_i \in G_j$, the j-th session key K_j is determined by the broadcast message B_j and the personal public/private key pair (Q_i, S_i).

4.2. Mutual-healing key distribution security model

The idea of mutual-healing was discussed in Muhammad and Ali (2005) without exploring any technical detail. Just as self-healing means that nodes are capable of recovering lost group keys on their own, mutual-healing implies that nodes help each other in recovering some lost group keys. The central concept of mutual-healing is that if a node has missed more than a fixed number broadcast messages or the last broadcast message, it can get assistance from its neighboring nodes. The neighboring nodes in the same session group cooperate with each other forwarding broadcast messages which their neighboring nodes miss. By this way the robustness of self-healing key distribution scheme is achieved.

It was claimed in Muhammad and Ali (2005) that two requirements are necessary for mutual-healing. They are the authentication on the requesting node and the authorization on the requested session key. On the one hand, in order to avoid attacks on their limited resource, effective authentication on the requesting node must be developed to identify misbehaving nodes. On the other hand, any entities in this communication networks can access broadcast messages because broadcast messages are broadcasted in plain-text within the communication group. We argue that the second authentication is unnecessary. Instead, the neighboring nodes only need to forward the broadcast message which corresponds to the requested session key. If the requesting node is authorized for the session, it would be able to recover the session key. Otherwise, even unauthorized nodes receive the broadcast message from their neighbors, they can not recover the session key.

To further clarify our design goal and facilitate understanding of readers, according but not constraint to the security model of Dutta et al. (2007), we define the mutual-healing key distribution security model from four aspects. Definition 4.1 defines self-healing key distribution scheme with revocation capability. Definition 4.2 defines mutual-healing key distribution scheme Definition 4.3 defines forward secrecy and backward secrecy. Definition 4.4 defines resisting collusion properties.

Definition 4.1. Let $U=\{u_1,\ldots, u_n\}$ and $j \in \{1,\ldots,m\}$.

1. The scheme is a session key distribution with privacy if
   (a) for any node u_i, the session key K_j is efficiently determined from B_j and the personal public/private key pair (Q_i, S_i).
   (b) for any set $R \subseteq U$ and $u_i \notin R$, it is computationally infeasible for node in R to determine the personal private key S_i.
   (c) what node $u_1, \ldots, u_n$ learn from B_j cannot be determined from broadcasts or personal key pairs alone. That is, if we consider separately either the set of m broadcasts $\{B_1, \ldots, B_m\}$ or the set of n personal key pairs $\{(Q_1, S_1), \ldots, (Q_n, S_n)\}$, then it is computationally infeasible to compute session key K_j from either set.
2. The scheme has revocation capability. Particularly, there is no upper limitation of revocation. For each session j and $R_j \in U$, GM can generate a broadcast message B_j such that for all $u_i \notin R_j$ can efficiently recover the session key K_j, but the revoked nodes in R_j cannot even knowing all the information broadcast in sessions 1, …, j.
3. The scheme is self-healing if the following is true for any r, $1 \leq r < j \leq m$: For any node $u_i \in G_r$ who is also a member in session j, the session key K_r is efficiently determined by (Q_i, S_i) and B_j.

Definition 4.2. Let $U=\{u_1,\ldots,u_n\}$ and $j \in \{1,\ldots,m\}$. The scheme is mutual-healing if the following is true:

1. For any node $u_i \in U$, if it misses more than a fixed number of broadcast messages or the last broadcast message, it can generate and broadcast effective requesting message to its neighbors.
2. For any u_i's neighboring node u_j, it can verify whether u_i is its qualified neighboring node or a malicious one. If u_i is a qualified neighboring node and u_j is an authorized node for the requested broadcast, u_j generates and sends responsive message to u_i.
3. The requester u_i can verify whether the responser u_j is its neighbor or not. If u_j is its neighbor, u_i can decrypts the responsive message and thus get the requested broadcast message.

Definition 4.3. Let $U=\{u_1, \ldots, u_n\}$ and $j \in \{1,\ldots,m\}$. The scheme guarantees both forward security and backward security if

1. for any set $R \subseteq U$, and all $u_i \in R$ are revoked before session j, it is computationally infeasible for the nodes in R together to get

any information about K_j, even with the knowledge of group keys $K_1, \ldots, K_{j-1}$ before session j.
2. for any set $J \subseteq U$, and all $u_i \in J$ join after session j, it is computationally infeasible for the nodes in J together to get any information about K_j, even with the knowledge of group keys $K_{j+1}, \ldots, K_m$ after session j.

Definition 4.4. Let $B \subset R_r \cup \ldots \cup R_2$ be a coalition of nodes who are revoked from the group before session r and let $C \subset J_s \cup \ldots \cup J_m$ be a coalition of nodes who join the group from session s with $r < s$.

The scheme is collusion-free for any coalition of non-authorized nodes if the coalition $B \cup C$ doses not get any information about session keys K_j, for any $r \le j < s$.

5. The proposed mutual-healing key distribution scheme

In this section, we will present a mutual-healing key distribution scheme for wireless sensor networks. The proposed mutual-healing key distribution scheme has the following procedures:

(1) SYSTEM SET-UP;
(2) BROADCAST;
(3) KEY RECOVERY;
(4) SELF-HEALING;
(5) MUTUAL-HEALING; and
(6) ADDING AND REVOKING NODE.

We need to point out the scheme is computationally secure scheme. All parameters and symbols used in this section have been defined in Sections 3 and 4. Otherwise, they will be defined in this section.

5.1. System set-up

GM obtains both public system parameters and all the public keys of possible nodes from the ID-based PKI. GM chooses m session keys $K_1, \ldots, K_m$ from $\mathbb{Z}_q^*$. The session keys are independent to each other and according to the uniform distribution. We also use the system parameters proposed in Section 3.

5.2. Broadcast

Suppose $|G_j|$ denotes the number of nodes in session j. For each session $1 \le j \le m$, according to the session group G_j, GM computes $Q_{V_1} = \sum_{i=1}^{n} Q_i$ and a $(|G_j|-1) \times |G_j|$ matrix which is defined as follows:

$$\begin{pmatrix} a_2 \\ a_3 \\ \vdots \\ a_{|G_j|} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & \ldots & 0 \\ 1 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 1 & 0 & 0 & \ldots & 1 \end{pmatrix}$$

Let a_i' represents the transpose of a_i. GM also constructs $|G_j|-1$ auxiliary keys

$$Q_{V_i} = (Q_1, Q_2, \ldots, Q_{|G_j|}) \times a_i', \quad 2 \le i \le |G_j|,$$

which means $Q_{V_2} = Q_1 + Q_2, Q_{V_3} = Q_1 + Q_3, \ldots, Q_{V_{|G_j|}} = Q_1 + Q_{|G_j|}$. The broadcast message is then formed by computing, for a random $r_j \in \mathbb{Z}_q^*$,

$$U_1 = r_j P, \quad U_i = r_j Q_{V_i} (2 \le i \le |G_j|),$$

where P is a generator for a BDH group G_1 as shown in Section 3.2.

$$V_j = K_j \oplus H_2(e(P_{pub}, r_j Q_{V_1}))$$

Let $z_j = (U_i (1 \le i \le |G_j|), V_j)$. GM broadcasts the ciphertext to the set of nodes G_j. The ciphertext for the j-th broadcast is in the following form: $B_j = \{z_1, \ldots, z_j\}$.

5.3. Key recovery

When a node $u_i \in G_j$ receives the broadcast message B_j, it sets a vector $a_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ with $|G_j|$ elements, and only the i-th element is 1. Then A_j is a $|G_j| \times |G_j|$ matrix

$$A_j = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{|G_j|} \end{pmatrix}.$$

The node u_i can solve the following system of equations using Cramer's rule or other algebraic methods.

$$(x_1, x_2, \ldots, x_{|G_j|}) \times A_j = (1, 1, \ldots, 1).$$

With $(x_1, x_2, \ldots, x_{|G_j|})$, u_i gets

$$(x_1, x_2, \ldots, x_{|G_j|}) \times \begin{pmatrix} Q_i \\ Q_{V_2} \\ \vdots \\ Q_{V_{|G_j|}} \end{pmatrix} = Q_{V_1}.$$

In order to decrypt the ciphertext, u_i needs to compute $e(P_{pub}, rQ_{V_1})$. With the knowledge of the private key S_i, it can do via:

$$\begin{aligned} e(P_{pub}, r_j Q_{V_1}) &= e(P_{pub}, r_j(x_1 Q_i + x_2 Q_{V_2} + \ldots + x_{|G_j|} Q_{V_{|G_j|}})) \\ &= e(P_{pub}, r_j x_1 Q_i) \cdot e(P_{pub}, r_j(x_2 Q_{V_2} + \ldots + x_{|G_j|} Q_{V_{|G_j|}})) \\ &= e(r_j P, x_1 s Q_i) \cdot e(P_{pub}, x_2 r_j Q_{V_2} + \ldots + x_{|G_j|} r_j Q_{V_{|G_j|}}) \\ &= e(U_1, x_1 S_i) \cdot e(P_{pub}, x_2 U_2 + \ldots + x_{|G_j|} U_{|G_j|}). \end{aligned}$$

Then, u_i can recover the session key

$$K_j = V_j \oplus H_2\left(e(U_1, x_1 S_i) \cdot e\left(P_{pub}, \sum_{i=2}^{|G_j|} x_i U_i\right)\right).$$

5.4. Self-healing

Without loss of generality, suppose u_i lost the broadcast message for a session $t < j$. As far as it belongs to the session group G_t, it picks up the polynomial z_t from the broadcast message B_j and forms the $|G_t| \times |G_t|$ matrix A_t as operations in the procedure of KEY RECOVERY. Then u_i solves the following system of equations.

$$(x_1, x_2, \ldots, x_{|G_t|}) \times A_t = (1, 1, \ldots, 1).$$

With $(x_1, x_2, \ldots, x_{|G_t|})$, u_i gets

$$(x_1, x_2, \ldots, x_{|G_t|}) \times \begin{pmatrix} Q_i \\ Q_{V_2} \\ \vdots \\ Q_{V_{|G_t|}} \end{pmatrix} = Q_{V_1}.$$

After that, with the knowledge of its private key S_i, u_i computes $e(P_{pub}, r_t Q_{V_1})$ as follows:

$$\begin{aligned} e(P_{pub}, r_t Q_{V_1}) &= e(P_{pub}, r_t(x_1 Q_i + x_2 Q_{V_2} + \ldots + x_{|G_t|} Q_{V_{|G_t|}})) \\ &= e(P_{pub}, r_t x_1 Q_i) \cdot e(P_{pub}, r_t(x_2 Q_{V_2} + \ldots + x_{|G_t|} Q_{V_{|G_t|}})) \\ &= e(r_t P, x_1 s Q_i) \cdot e(P_{pub}, x_2 r_t Q_{V_2} + \ldots + x_{|G_t|} r_t Q_{V_{|G_t|}}) \\ &= e(U_1, x_1 S_i) \cdot e(P_{pub}, x_2 U_2 + \ldots + x_{|G_t|} U_{|G_t|}). \end{aligned}$$

Finally, u_i recovers the lost session key

$$K_t = V_t \oplus H_2\left(e(U_1,x_1S_i)\cdot e\left(P_{pub},\sum_{i=2}^{|G_t|}x_iU_i\right)\right)$$

One may wonder how u_i got V_t if the t-th broadcast message was lost. In fact, u_i could get the broadcast message B_j. Then u_i retrieved the polynomial z_j from B_j. Finally, u_i could obtain V_t from z_j. If more than one broadcast messages get lost, the operations of session key recovery are the same as aforementioned.

5.5. *Mutual-healing*

Here we consider the mutual-healing between neighboring nodes in wireless communication networks and present a practical technique to realize it.

Many wireless networks have an intrinsic property that nodes are stationary. Therefore, we can bound LBKs (the location-based keys) of nodes to both their identities and geographic locations rather merely their identities or locations as in conventional schemes. Based on their LBKs, two neighboring nodes can perform node-to-node neighborhood authentication. In order to reduce communication overhead, we restrict that mutual-healing only happens between one-hop neighboring nodes. In order to realize mutual-healing capability, The scheme has to execute range-based location operation after SET-UP procedure.

There are many methods to localize nodes. We adopt the first method in Zhang et al. (2006). This step may complete within several minutes after the deployment of networks. We assume that a group of mobile robots are dispatched to sweep across the whole network field along preplanned routes. Mobile robots have GPS (Global Positioning System) capability, as well as more powerful computation and communication capacities than ordinary nodes have. The leading robot equipped with the a master secret k. To localize a node, say u_i, mobile robots run the secure range-based localization protocol given in Capkun and Hubaux (2005) or Zhang et al. (2005) to measure their respective absolute distance to node u_i and co-determine l_i, the location of u_i. Subsequently, the leading robot calculates $LK_i = kH(ID_i \| l_i)$, and sends $\langle \{LK_i \| l_i\}_{Q_i}, h_{Q_i}(LK_i \| l_i)\rangle$ to u_i. $\{M\}_k$ means encrypting message M with key k, and $h_k(M)$ refers to the MIC (message integrity code) of message M under key k.

Upon receipt of the message, node u_i first uses its private key to decrypt LK_i and l_i and then regenerates the MIC. If the result matches with what the robot sent, u_i saves LK_i and l_i for subsequent use. Following this process, all the nodes can be furnished with their respective locations and LBKs. After that, mobile robots leave the sensor field and the leading robot have to securely erase k from its memory. During subsequent network operations, Adding nodes may be necessary in order to maintain good network connectivity. The localization of new nodes can be done in the same manner.

It is general supposed that adversaries do not launch active and explicit pinpoint attack on nodes during deployment and initialization which usually dose not last too long. According to Zhang et al. (2006), this assumption in range-based location operation is reasonable in that mobile robots are much fewer than ordinary nodes and can be equipped with tamper-proof hardware and putting them under super monitor.

During the procedures of self-healing key distribution, if a node has missed more than a fixed number broadcast messages or the last broadcast message, it looks for assistance from its neighboring nodes. The realization of mutual-healing includes three steps. We will introduce them one by one.

5.5.1. *Mutual-healing request*

Suppose node u_i wishes to receive broadcast message B_t, u_i locally broadcasts an authentication request including its identity ID_i, location l_i and the sequence number of the expected broadcast message t.

5.5.2. *Mutual-healing response*

Upon receipt of a request, the neighboring node u_j first needs to ascertain that the claimed location l_i is within its one-hop communication range by verifying if the Euclidean distance $\|l_i - l_j\| \le \mathcal{R}$, where $\mathcal{R}$ is one-hop communication distance.

If the inequality does not hold, node u_j simply discards the request. Otherwise, u_j calculates a shared key as $K_{ji} = e(LK_j, H(ID_i \| l_i))$. Then it unicasts a reply to node u_i including its identity ID_j, location l_j and encrypted broadcast message $(B_t)_{K_{ji}}$.

- $u_i \to * : ID_i, l_i, t$;
- $u_j \to u_i : ID_j, l_j, (B_t)_{K_{ji}}$;

5.5.3. *Verification*

Upon receiving the response, node u_i also first checks if the inequality $\|l_i - l_j\| \le \mathcal{R}$ holds. If the inequality does not hold, u_i directly discards the message received from u_j. Otherwise, u_i proceeds to derive a shared key as $K_{ij} = e(LK_i, H(ID_j \| l_j)) = K_{ji}$ between it and the node u_j whereby to decrypt the message $(B_t)_{K_{ji}}$ and get the broadcast message B_t. Using the broadcast message B_t and its public/private key pair, the authorized node u_i can recover the lost session keys.

5.5.4. *Adding and revoking node*

If a new node u_{new} applies for joining the session j, GM checks the validity of its identity firstly. If it is an authorized node, in the procedure of BROADCAST, GM constructs a new $(|G_j|-1) \times |G_j|$ matrix and computes new $Q_{V_i} (1 \le i \le |G_j|)$ which should include Q_{new}.

If a node u_{rov} is revoked from the session j, what GM should do is constructing a new $(|G_j|-1) \times |G_j|$ matrix and computing new $Q_{V_i} (1 \le i \le |G_j|)$ which should exclude Q_{rov}.

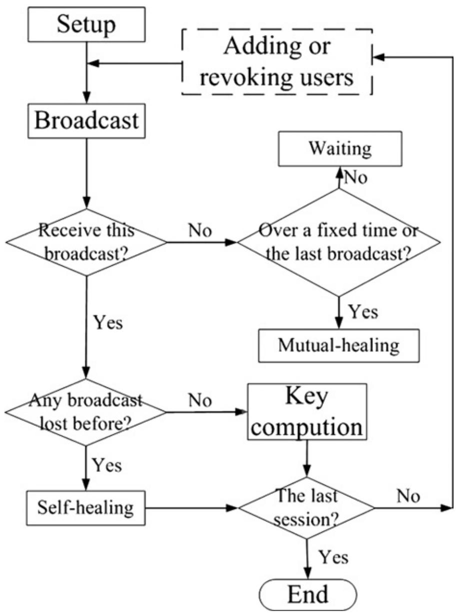


Fig. 1. The process of the mutual-healing key distribution scheme, where the panes are operation which must be executed in each round, the dashed panes represent the operations which may not be executed in every round.

The adding and revoking operations are very efficient in our scheme. For the condition that adding or revoking more than one node, the operations are the same as aforementioned.

Figure 1 shows all the procedures involved in a mutual-healing key distribution scheme.

6. Security and efficiency

Security analysis for the proposed mutual-healing key distribution scheme will be provided in this section. Performance discussion for the scheme will also be presented.

6.1. Security of the proposed scheme

In this subsection, we analyze the security of the proposed scheme. More precisely, we show that our construction satisfies all the security requirements in our security model described in Section 4.2.

6.1.1. Property 1: Self-healing key distribution

We show our construction satisfies the security requirements described in Definition 4.1.

1. The scheme is a session key distribution scheme.
   (a) Any node $u_i \in G_j$ can recover the session key K_j from B_j and the personal public/private key pair (Q_i, S_i). This is because: When a node $u_i \in G_j$ receives the broadcast message B_j, it sets a vector c_i=(0,…,0,1,0,…,0) with $|G_j|$ elements, and only the i-th element is 1. Define a new matrix C_j using c_i with $i = 1,2,\ldots,|G_j|$. Then C_j is a $|G_j| \times |G_j|$ matrix

   $$C_j = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{|G_j|} \end{pmatrix}.$$

   The node u_i can solve the following system of equations using Cramer's rule or other algebraic methods.

   $$(x_1,x_2,\ldots,x_{|G_j|}) \times C_j = (1,1,\ldots,1).$$

   With $(x_1,x_2,\ldots,x_{|G_j|})$, u_i gets

   $$(x_1,x_2,\ldots,x_{|G_j|}) \times \begin{pmatrix} Q_i \\ Q_{V_2} \\ \vdots \\ Q_{V_{|G_j|}} \end{pmatrix} = Q_{V_1}.$$

   In order to decrypt the ciphertext, u_i needs to compute $e(P_{pub},rQ_{V_1})$. With the knowledge of the private key S_i, u_i can do so via:

   $$\begin{aligned} e(P_{pub},r_jQ_{V_1}) &= e(P_{pub},r_j(x_1Q_i+x_2Q_{V_2}+\ldots+x_{|G_j|}Q_{V_{|G_j|}})) \\ &= e(P_{pub},r_jx_1Q_i)\cdot e(P_{pub},r_j(x_2Q_{V_2}+\ldots+x_{|G_j|}Q_{V_{|G_j|}})) \\ &= e(r_jP,x_1sQ_i)\cdot e(P_{pub},x_2r_jQ_{V_2}+\ldots+x_{|G_j|}r_jQ_{V_{|G_j|}}) \\ &= e(U_1,x_1S_i)\cdot e(P_{pub},x_2U_2+\ldots+x_{|G_j|}U_{|G_j|}). \end{aligned}$$

   Then, u_i can recover the session key K_j by

   $$K_j = V_j \oplus H_2\left(e(U_1,x_1S_i)\cdot e\left(P_{pub},\sum_{i=2}^{|G_j|}x_iU_i\right)\right).$$

   (b) Any coalition $R \subseteq U$ of non-authorized nodes cannot derive the private key S_i of any authorized node $u_i \notin R$. This is because each node is preloaded with a public/private key pair (Q_i, S_i) before deployment. The key pairs are computed by an ID-based PKI which can realize public and private keys without certificate management (Han et al., 2004). Because it is infeasible to solve the discrete logarithm problem $S_{ID}=sQ_{ID}$, any coalition $R \subseteq U$ of non-authorized nodes cannot derive the private key S_i of any authorized node $u_i \notin R$.
   (c) It is computationally infeasible to compute session key K_j from either broadcast messages or personal public/private key pairs. This is because the j-th session key is computed from V_j and $e(P_{pub},r_jQ_{V_1})$. On the one hand, V_j is taken from the broadcast message while

   $$e(P_{pub},r_jQ_{V_1}) = e(U_1,x_1S_i)\cdot e\left(P_{pub},\sum_{i=2}^{|G_j|}x_iU_i\right).$$

   Therefore, only the authorized nodes who holds corresponding private key S_i can recover the session key. They cannot get any session key only from the broadcast messages. On the other hand, the personal public/private key pairs of nodes are computed by the ID-based PKI. The session keys are chosen by the GM. They are independent of one another. Therefore, The coalition of nodes cannot get any session key only by their public/private key pairs $\{(Q_1, S_1), \ldots, (Q_n, S_n)\}$.
2. There is no threshold for the revocation in the proposed scheme. This means any number of nodes who are compromised or malicious can be revoked and although they work together they cannot work out the private key of any nonrevoked authorized node. In our scheme, the broadcast messages are computationally related to the authorized nodes' public/private key pairs. On the one hand, only nodes who hold the corresponding private keys can recover the session keys from the masked broadcast messages. On the other hand, even all the revoked unauthorized node work together they cannot get the private key of any nonrevoked authorized node due to the difficulty of solving the discrete logarithm problem.
3. The proposed scheme has the self-healing capability. It can also enable a node to recover from a single broadcast message all keys associated with sessions in which it belongs to the session group. It is a stronger self-healing key distribution scheme.
   Without loss of generality, suppose $u_i \in G_t$ lost the broadcast message for a session t where $t < j$ and . It selects polynomial z_t from the broadcast message B_j and constructs a $|G_t| \times |G_t|$ matrix A_t as below:

   $$A_t = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{|G_t|} \end{pmatrix},$$

   where a_k=(0,0,…,1,0…,0) (note: the kth bit is 1 and all other bits are 0) and $k = 1,2,\ldots,|G_t|$.
   Then u_i solves the following system of equations.

   $$(x_1,x_2,\ldots,x_{|G_t|}) \times A_t = (1,1,\ldots,1).$$

   With $(x_1,x_2,\ldots,x_{|G_t|})$, u_i gets

   $$(x_1,x_2,\ldots,x_{|G_t|}) \times \begin{pmatrix} Q_i \\ Q_{V_2} \\ \vdots \\ Q_{V_{|G_t|}} \end{pmatrix} = Q_{V_1}.$$

After that, with the knowledge of its private key S_i, u_i computes $e(P_{pub},r_tQ_{V_1})$ as follows:

$$\begin{aligned} e(P_{pub},r_tQ_{V_1}) &= e(P_{pub},r_t(x_1Q_i+x_2Q_{V_2}+\ldots+x_{|G_t|}Q_{V_{|G_t|}})) \\ &= e(P_{pub},r_tx_1Q_i)\cdot e(P_{pub},r_t(x_2Q_{V_2}+\ldots+x_{|G_t|}Q_{V_{|G_t|}})) \\ &= e(r_tP,x_1sQ_i)\cdot e(P_{pub},x_2r_tQ_{V_2}+\ldots+x_{|G_t|}r_tQ_{V_{|G_t|}}) \\ &= e(U_1,x_1S_i)\cdot e(P_{pub},x_2U_2+\ldots+x_{|G_t|}U_{|G_t|}). \end{aligned}$$

Finally, u_i recovers the lost session key

$$K_t = V_t \oplus H_2\left(e(U_1,x_1S_i)\cdot e\left(P_{pub},\sum_{k=2}^{|G_t|}x_kU_k\right)\right).$$

6.1.2. Property 2: Forward security and backward security

We show our construction satisfies forward security and backward security described in Definition 4.2.

In this scheme, due to the special construction of broadcast messages, only the current authorized nodes can recover the session keys by using their private keys. As described before, any coalition of non-authorized nodes cannot derive the private key of any authorized node. Furthermore, session keys are independent to each other and according to the uniform distribution. All of these features imply forward security and backward security of our scheme.

6.1.3. Property 3: Collusion-free property

We show our construction is collusion-free for the new joined nodes and the revoked nodes described in Definition 4.3.

Our scheme is collusion-free for any coalition of non-authorized nodes, including the revoked nodes and new joined nodes. In our scheme, if one node wants to obtain session key, it should compute $e(U_1,x_1S_i)$ in the procedure of KEY RECOVERY. Therefore, only the authorized nodes can recover the session key. In addition, due to the difficulty of solving DLP, any coalition of non-authorized nodes cannot derive the private keys of authorized nodes from their public keys.

6.1.4. Property 4: Mutual-healing key distribution

We show that we securely realize mutual-healing property between neighboring nodes. More specifically, the realization satisfies the security requirements described in Definition 4.4.

1. It can be seen in the step of MUTUAL-HEALING REQUEST that any node $u_i \in U$, if it misses more than a fixed number of broadcast messages or the last broadcast message, it can generate and broadcast the requesting message to its neighbors. The broadcast message is composed of its identity ID_i, location l_i and the sequence number of the expected broadcast message t.
2. A false requesting node might send an request with a forged location within node u_j's range. Since the false requesting node does not hold the LBK corresponding to the forged location, even it deceive u_j into believing it is in u_j's range, it cannot recover the session key from the broadcast message that u_j sends. Therefore, it get any useful information from u_j. There are some false requesting nodes who might mount DoS (denial of service) attack by continuously sending bogus mutual-healing requests to allure legitimate nodes into endless verifying of such messages. Because the number of neighbors of any node is limited in reality, abnormally many mutual-healing requests are highly like an indicator of malicious attacks. If this situation happens, node u_j will discard the requesting message and stop assistance.
3. Upon receiving the reply, node u_i also first checks whether u_j is its neighbor. This check is the baseline defense against the attack that adversaries surreptitiously tunnel authentication messages between u_i and a virtually non-neighboring node. Without the location check, u_i will falsely believe that the broadcast messages come from its neighbors. If the first check is true, then u_i checks whether the message received from u_j is effective. u_i can recover the requested broadcast message if the second check is true.
If a requester receives too many replies, it only decrypts a fixed number messages in order to save its limited resource and avoid exhausting-resource attack. Furthermore, we assume that there are efficient mechanisms available for authorized nodes to report such an abnormality to the sink.

6.2. The analysis of efficiency

Different from the existing papers, we take advantage of a bilinear pairings-based broadcast encryption to design the self-healing procedures in the proposed mutual-healing key distribution scheme. In this section, we first analyze the efficiency for the self-healing procedures followed by the one for mutual-healing procedures.

In terms of storage overhead, each node only stores its public/private key pair and GM's public key P_{pub}. Therefore, the storage overhead for end nodes is a constant. Furthermore, the private key has nothing to do with the number of revoked nodes and can be reused as long as it is not disclosed. In addition, our scheme enables a node to recover from a single broadcast message all keys associated with sessions in which it belongs to the session group.

Generally speaking, GM takes up more resources than end nodes and thus can perform more complex computation. We elaborate on analyzing the computation overhead at nodes. In the j-th session, all the computations in the procedure of KEY RECOVERY are as follows: (1) Solving a set of linear equations with $|G_j|$ variables; (2) $|G_j|$ scalar multiplications in the cyclic additive group G_1; (3) $|G_j|-1$ additions in the cyclic additive group G_1; (4) Two pairings computation; (5) One hashing computation; (6) One XOR operation. Generally speaking, the computation of the pairing is the most time-consuming in pairings-based cryptosystems. Although there have been many papers talking about the complexity of pairings and how to speed up the computation of bilinear pairings (Barreto et al., 2002; Galbraith et al., 2002), the computation overhead of bilinear pairings are still larger than the scalar multiplication, let alone other types of computation. Therefore, the main computation overhead of the scheme comes from (4).

The communication overhead comes from broadcast messages $B_j=\{z_1, \ldots,z_j\}$. z_j is composed of $U_i(1 \le i \le |G_j|)$ and V_j. Therefore, the size of z_j increases in direct proportion to the number of $|G_j|$. The length of broadcast is $\sum_{i=1}^{j}|G_i|\log q + j\log q$. $\log q$ is the size of session key.

The mutual-healing procedures achieves all the security requirements and is efficient. Each node only has to store another pair of LBK and its location in order to achieve mutual-healing property. Therefore, the storage overhead is still a constant. In terms of communication overhead, only two interactions are involved. In the first step, the broadcast message is composed of the requester's identity, location and the sequence number of the expected broadcast message. Furthermore, the message is broadcasted within one-hop communication range. The communication overhead of this step is very small. The second interaction may include one or several unicasts. The unicast message is composed of the responser's identity, location, and encrypted the broadcast message. Therefore, the size of these unicast messages is a constant. The computation overhead at the responser includes generating a shared key and encrypting the requested message and the computation overhead at the requester includes generating a shared key and decrypting the requested message. Because mutual-healing happens not very often, the computation overhead would not consume too much resource of the involved nodes.

7. Conclusion and future research

A mutual-healing key distribution scheme using bilinear pairings is proposed in this paper. Security model and formal definition for mutual-healing key distribution were discussed. The proposed new scheme achieves several desirable features. The storage overhead for each node is a constant. The scheme is collusion-free for any coalition of non-authorized nodes. Each authorized node's private key has nothing to do with the number of revoked nodes and can be reused only if it is not disclosed. While in secret sharing-based self-healing key distribution schemes, the personal key can be reused on the condition that less than threshold number nodes are revoked. In addition, our scheme enables a node to recover from a single broadcast message all keys associated with sessions in which it belongs to the session group.

The proposed mutual-healing scheme relies on identity and location-based keys. This implies that the proposed scheme can only be used over the wireless networks where nodes are stable. It is not trivial to realize mutual-healing in mobile wireless networks. In fact, the mutual-healing mechanism is more useful in mobile wireless networks because mobile wireless networks have lower network connectivity than stable wireless networks. Therefore, it is significant to investigate new methods to realize the mutual-healing feature in mobile wireless networks.

Acknowledgement

The work is financially supported by the Australia Research Council (ARC) Projects with project IDs LP100200693, LP100200538, LP1000100404 and DP0985838.

References

Balenson D, McGrew D, Sherman A. Key management for large dynamic groups: one-way function trees and amortized initialization, IRTF SMUG Meeting, March 15, 1999.

Barreto PSLM, Kim HY, Lynn B, Scott M. Efficient algorithms for pairing-based cryptosystems. In: Advances in cryptology—crypto'02. Lecture notes in computer science, vol. 2442; 2002. p. 354–68.

Blundo C, D'Arco P, Santis A, Listo M. Design of self-healing key distribution schemes. Design Codes and Cryptography 2004;32:15–44.

Boneh D, Franklin M. Identity based encryption from the weil pairing. In: Advanced in cryptology—CRYPTO'01; 2001. p. 213–29.

Capkun S, Hubaux J-P. Secure positioning of wireless devices with application to sensor networks. In: Proceedings of IEEE INFOCOM. Miami, Florida; March 2005. p. 1917–28.

Du X, Wang Y, Ge J, Wang Y. An ID-based broadcast encryption scheme for key distribution. IEEE Transactions on Broadcasting 2005;51(2):264–6.

Dutta R, Mukhopadhyay S. Improved self-healing key distribution with revocation in wireless sensor network. In: Wireless communications and networking conference; 2007. p. 2963–8.

Dutta R, Chang EC, Mukhopadhyay S. Efficient self-healing key distribution with revocation for wireless sensor networks using one way key chains. In: ACNS 2007, Lecture notes in computer science, vol. 4521; 2007. p. 385–400.

Eltoweissy M, Younis M, Ghumman K. Lightweight key management for wireless sensor networks. In: IEEE international conference on performance, computating, and communications; April 2004. p. 813–8.

Eltoweissy M, Heydari M, Morales L, Sudborough H. Combinatorial optimization for group key management. Journal of Network and System Management 2004b;12(1):33–50.

Fiat A, Tessa T. Dynamic traitor tracing. Journal of Cryptology 2001;14:211–23.

Galbraith SD, Harrison K, Soldera D. Implementing the tate pairing. In: Proceedings of the 5th international symposium on algorithmic number theory. Lecture notes in computer science, vol. 2369; 2002. p. 324–37.

Halevy D, Shamir A. The LSD broadcast encryption scheme. In: Advances in cryptology—crypto' 02. Lecture notes in computer science, vol. 2442; 2002. p. 47–60.

Han F, Hu J, Yu X, Wang Y. Fingerprint images encryption via multi-scroll chaotic attractors. Applied Mathematics and Computation 2007;185(2):931–9.

Han S, Tian B, He M, Chang E. Efficient threshold self-healing key distribution with sponsorization for infrastructureless wireless networks. IEEE Transactions on Wireless Communications 2009;8(4).

Han S, Wang J, Liu W. An efficient identity-based group signature scheme over elliptic curves. In: European conference on universal multiservice and networks. Lecture notes in computer science, vol. 3262. Springer-Verlag; 2004. p. 417–29.

Hong D, Kang JS. An efficient key distribution scheme with self-healing property. IEEE Communication Letters 2005;9(8).

Hu J, Chen HH, Hou TW. A hybrid public key infrastructure solution (HPKI) for HIPAA privacy/security regulations. Computer Standards & Interfaces 2010;32(5–6):274–80.

Hu J, Han F. A pixel-based scrambling scheme for digital medical images protection. Journal of Network and Computer Applications 2009;32(4): 788–94.

Jiang Y, Lin C, Shi M, Shen X. Self-healing group key distribution with time-limited node revocation for wireless sensor networks. Ad hoc Networks 2007;5:14–23.

Kausar F, Hussain S, Park JH, Masood A. Secure group communication with self-healing and rekeying in wireless sensor networks. In: Proceedings of the third international conference, MSN 2007; 2007. p. 737–48.

Ku W, Chen S. An improved key management scheme for large dynamic groups using one-way function trees. In: International conference on parallel processing workshops '03. October 2003. p. 391–6.

Liu D, Ning P, Sun K. Efficient self-healing key distribution with revocation capability. In: Proceeding of the 10th ACM conference on computer, communications and security; 2003.

Lu K, Qian Y, Hu J. A framework for distributed key management schemes in heterogeneous wireless sensor networks. In: The 25th IEEE international conference on performance, computing, and communications, USA; April 2006. p. 513–9.

Moharrum M, Eltoweissy M, Mukkamala R. Dynamic combinatorial key management scheme for sensor networks. Wireless Communications and Mobile Computing 2006;6(7):1017–35.

More SM, Malkin M, Staddon J, Balfanz D. Sliding window self-healing key distribution with revocation. In: ACM workshop on survivable and self-regenerative systems; 2003. p. 82–90.

Muhammad JB, Ali M. Self-healing group key distribution. International Journal of Network Security 2005;1(2):110–7.

Naor D, Naor M, Lotspiech J. Revocation and tracing schemes for stateless receivers. In: Advances in cryptology—crypto '01. Lecture notes in computer science, vol. 2139; 2001. p. 41–62.

Naor M, Pinkas B. Efficient trace and revoke schemes. In: Financial cryptography' 2000. Lecture notes in computer science, vol. 1962; 2000. p. 1–21.

Sáez G. On threshold self-healing key distribution schemes. In: Cryptography and coding, Lecture notes in computer science, vol. 3796; 2005a. p 340–54.

Sáez G. Self-healing key distribution schemes with sponsorization. In: International federation for information processing, IFIP'05, Lecture notes in computer science, vol. 3677; 2005b. p. 22–31.

Shamir A. Identity-based cryptosystems and signature scheme. Proceedings of Crypto 1984;84:47–53.

Staddon J, Miner S, Franklin M, Balfanz D, Malkin M, Dean D. Self-healing key distribution with revocation. In: Proceedings of IEEE symposium on security and privacy; 2002. p. 224–40.

Sufi F, Han F, Khail I, Hu J. A chaos based encryption technique to protect ECG packets for time critical telecardiology applications. Journal of Security and Communication Networks (2010), doi:10.1002/sec.226.

Wong CK, Gouda MG, Lan SS. Secure group communication using key graphs. IEEE/ACM Transactions on Networking 2000;8(1):16–30.

Xi K, Ahmad T, Han F, Hu J. A Fingerprint based bio-cryptographic security protocol designed for client/server authentication in mobile computing environment. Journal of Security and Communication Networks (2010), doi:10.1002/sec.225.

Zhang Y, Liu W, Fang Y. Secure localization in wireless sensor networks. In: Proceedings of IEEE MILCOM; 2005.

Zhang Y, Liu W, Lou W, Fang Y. Location-based compromise-tolerant security mechanisms for wireless sensor networks. IEEE Journal on Selected Areas in Communications 2006;24(2):247–60.

Zou X, Dai Y. Robust and stateless self-healing group key management scheme. International Conference on Communication Technology ICCT '06 2006:1–4.

Advance Access publication on March 20, 2011
doi:10.1093/comjnl/bxr022

Self-Healing Key Distribution Schemes for Wireless Networks: A Survey

Biming Tian[1], Song Han[1]*, Sazia Parvin[1], Jiankun Hu[2], Sajal Das[3]

[1]*DEBII Institute, Curtin University,*
[2]*School of Engineering and Information Technology, UNSW@ADFA, Canberra, Australia*
[3]*Center for Research in Wireless Mobility and Networking, University Texas at Arlington*
**Corresponding author: hansongau@gmail.com*

The objective of self-healing key distribution is to enable group users to recover session keys by themselves, without requesting additional transmissions from the group manager (GM), even when they miss some broadcast messages. One major benefit of the self-healing key distribution mechanism is the reduction of energy consumption due to the elimination of such additional transmission. Also in some applications, e.g., uni-directional broadcast channel from the GM, the self-healing key distribution mechanism seems to be the ideal solution. Desired features of self-healing key distribution schemes include energy awareness, short broadcast message, efficient users addition, revocation and so on. A primary challenge is managing the trade-off between providing an acceptable level of security and conserving scarce resources in particular energy which is critical for wireless network operations. Over a decade, a great number of self-healing key distribution schemes have been proposed for establishing a group key amongst a dynamic group of users over an unreliable, or lossy, network. In this paper a comprehensive survey is conducted on the state-of-the-art in the field of self-healing key distribution. First, we clarify the security requirements of self-healing key distribution scheme for their special application environment. Then, we present a classification of self-healing key distribution schemes according to different cryptographic primitives, and give an insight to their features and goals. Furthermore, we consider several problems, namely authentication on broadcast messages, sponsorization and mutual-healing, related to the robustness of self-healing key distribution schemes. At last, we delineate their similarities and differences and outline several future research directions.

Keywords: wireless security; self-healing key distribution; one way hash function; subset difference rekeying; wireless networks

Received 20 January 2011; revised 20 January 2011
Handling editor: Erol Gelenbe

1. INTRODUCTION

Group communication can enjoy the benefit of communication efficiency from broadcast in distributing secret digital content. However, there is a challenge of effectively controlling access to the transmitted data. The broadcast by itself does not provide any mechanisms for preventing non-group members from the group communication. One common method for enabling secure broadcast communication is distributing a session key to group users and updating it on each operation of adding or revoking users. All messages broadcasted within the group during a certain sessions are communicated securely through encryption under the session key. Although the sensitivity of the broadcasted messages makes session keys essential to secure group communication, distributing session keys become an issue, especially distributing session keys for large dynamic communication groups. Researchers have proposed many different key distribution schemes. These schemes can be divided into two main classes according to the underlying networks. One is key distribution over reliable networks and the other is key distribution over unreliable networks. The problem of distributing keys over a reliable channel has received much attention [1–4]. The research on self-healing key distribution for establishing keys over an unreliable network began in 2002. Since then, self-healing key distribution, as one of the techniques for wireless security, has been one of the hot research topics.

The target of key distribution is to establish and maintain secure channels between the group manager (GM) and multiple group users. Session keys need to be updated as long as membership changes in order to maintain security and resilience to attacks [5]. The updated session key prevents a new user from decoding messages broadcasted before it joins the group and a revoked user from accessing the group communication after it leaves the group. However, distributing the session keys to authorized users is a complex problem. Although rekeying a group before the join of a new user is trivial (send the new group key to the old group users encrypted with the old group key), rekeying the group after a user leaves is far more complicated. The old key cannot be used to distribute a new one, because the leaving user knows the old key. Therefore, the GM must provide another scalable mechanism to update the session keys.

A naive scheme for key updating within a group with n users is to have the GM assign a pair-wise key shared between the group manger and the intended user during the phase of initiation. In order to update a session key, the GM encrypts it with the pair-wise keys shared between them and then unicasts the encrypted messages. On receiving the messages, the intended user can recover the session key using the shared key. In fact, this method is not simple or scalable. In each round of key updating, the group manager has to encrypt the group key and unicast the encrypted key n times. Therefore, both the computation and communication overhead brought by this operation are $O(n)$. Even though the operation is simple, the overheads of the scheme in large dynamic groups are too high to afford.

Mobile wireless networks are often highly volatile [5, 6]. Wireless nodes may move in and out of range frequently, and there is usually no infrastructure support to guarantee reliable delivery of messages. Thus, a message sent to a group may or may not reach all the group users. The problem of packet loss should be highlighted in an unreliable network. The key distribution broadcast for a particular session might never reach all users. Individual interactions, such as requesting and re-transmission, in large group will induce more communication overhead and place a heavy burden on the GM. In addition, users may reveal their current location by sending messages in some high-security environments. Particularly, many digital content and multi-media distribution systems are based on a uni-directional broadcast distribution channel, such as satellite or cable [7]. It is impossible to request or re-transmit under these communication channel. The main property of self-healing key distribution is that, even if at the beginning of certain sessions some broadcast packets get lost, group users are still capable of recovering the session key for those sessions simply by using the broadcasts they have received at a previous session and the packets they will receive at a subsequent one. This noninteractive key distribution scheme reduces the network traffic, decreases the work load on the GM and lowers the risk of user exposure through traffic analysis. Therefore, self-healing key distribution schemes are desirable for both efficiency and security reasons.

1.1. Application

This survey is motivated by an investigation of the security in several settings in which session keys need to be used for a short time-period or need to be updated frequently, due to frequent changes in the membership. Self-healing is a good property for key distribution in wireless sensor networks, where the nodes/devices are powered by batteries and have the unique feature of moving in and out of range frequently. Also there might be situations where some users are not constantly on-line or experience burst packet losses. It can rejoin the group once the power is on again. All these considerations can take great advantage from self-healing key distribution schemes with revocation capability. Military-oriented applications as well as rescue missions and scientific explorations, where the adversary may intercept, modify and/or partially interrupt the communication, are few important examples which can benefit from self-healing key distribution schemes. Self-healing key distribution schemes have also found applications in broadcast communication, pay-per-view TV, information service delivering sensitive content/information to authorized recipients over low-cost and uni-directional communication channel.

1.2. Contribution

The objective of this paper is to highlight the features and performance/security attributes of self-healing key distribution schemes. We describe the procedures of self-healing key distribution and present a classification of existing self-healing key distribution schemes according to the different cryptographic primitives that they are based on. The classification enables identification of common architectural elements and features, and to expose common vulnerabilities of each class of the existing schemes. In order to delineate their similarities and differences, we make a thorough comparison of their security and performance. Subsequently, we discuss three considerations to strengthen the robustness of self-healing key distribution schemes. Finally, we outline some problems to be solved and several future research directions. We expect that our survey provides a guideline and certain criteria to fairly evaluate the performance of self-healing key distribution schemes.

1.3. Organization

The rest of the paper is organized as follows: in Section 2, we introduce the desirable features of self-healing key distribution schemes. In Section 3, we classify the self-healing key distribution schemes from different aspects. In Section 4, we present the classic model and general procedures of self-healing key distribution schemes. In Section 5, we give an overview of existing works in the area of self-healing key distribution according to their classification. Then, in Section 6, we discuss three aspects, namely authentication on broadcast messages, sponsorization and mutual-healing, which can be used to strengthen the robustness of self-healing key distribution schemes. A compositive analysis of the existing schemes was proposed in Section 7. We outline the problems to be solved and future works in this field in Section 7. At last, conclusions are provided in Section 8.

2. DESIRABLE FEATURES OF SELF-HEALING KEY DISTRIBUTION SCHEMES

Self-healing key distribution can be considered as a branch of key distribution. Key distribution schemes should meet availability, integrity, confidentiality, authentication and non-reputation traditional security requirements. The same is true of self-healing key distribution schemes. In addition, according to the feature and application environment of self-healing key distribution, some particular evaluation measurements should be highlighted. The first one is security. The security here includes not only forward and backward secrecy but also collusion resistance between the newly joined nodes and the revoked nodes. Considering the wireless application environment, storage and bandwidth are regarded as constrained resources as well as computational power. Therefore, storage, communication and computation overheads should be as low as possible. Furthermore, the group may become too large to be managed by a single party, thus raising the issue of scalability. A self-healing key distribution scheme used in a resource-limited environment seeks to minimize the workloads of both GM and end users in order to augment the scalability. To sum up, the performance of self-healing key distribution scheme can be evaluated by:

(i) Forward and backward secrecy. Forward secrecy is used to prevent a revoked user from continued accessing the session key even if it keeps receiving the broadcast messages. Backward secrecy is used to prevent a new user from decoding messages broadcasted before it joins the group. When a group requires forward and backward secrecy, the session key must be changed for every membership change [8].

(ii) Collusion resistance. The collaboration of the newly joined users and the revoked users must not be able to recover the session keys which they are not entitled to. This is a stronger and practical security requirement.

(iii) Efficiency. The condition of limited storage, communication and computation ability must be given full consideration. In self-healing key distribution schemes, storage overhead refers to the number of private keys that group users store in their memory; computation overhead means the necessary computation load to generate session key by the GM and the necessary computation cost to recover session key by end users; and communication overhead is measured by the size of messages delivered by the GM in the process of distributing session keys.

(iv) Scalability. Self-healing key distribution operations should be finished in a timely manner despite a varying number of users and node densities. The fraction of the available bandwidth occupied by network management traffic should be kept as low as possible. Any increase in management traffic will reduce the available bandwidth for payload data accordingly. Hence, scalability of self-healing key distribution schemes is crucial.

TABLE 1. Classification of self-healing key distribution schemes.

	Unconditionally secure	Computationally secure
Theory basis	Information theory	One or several open hard problems
Polynomial secret sharing	[10]–[11], [12]	[13], [14], [15]
Vector space secret sharing	[16], [17]	[18], [19], [20]
Hash chain	None	[21]–[19], [15, 20]
SDR	None	[22], [23]
Bilinear pairing	None	[24]

3. CLASSIFICATION OF SELF-HEALING KEY DISTRIBUTION SCHEME

The success of a self-healing key distribution scheme is determined in part by its ability to securely and efficiently recover session key in low-cost wireless networks. Self-healing key distribution schemes can be broadly classified into unconditionally secure and computationally secure schemes. Unconditionally, secure schemes based on information theory, which is generally considered to have been founded in 1948 by Claude Shannon in his seminal work [9] and are defined by entropy function $H(\cdot)$. While computationally secure schemes are based on one or several open hard problems. The first and several subsequent self-healing key distribution schemes are unconditionally secure. More and more research works are computationally secured in recent years. The existing self-healing key distribution schemes can be classified into more classes with regard to different cryptographic primitives that they based on. They are polynomial secret sharing, vector space secret sharing, hash chain, subset difference rekeying (SDR) and bilinear pairings based self-healing key distribution schemes.

Table 1 shows a classification based on the two criteria: unconditionally secure vs. computationally secure and different cryptographic primitives they based on.

4. THE MODEL AND PROCEDURES OF SELF-HEALING KEY DISTRIBUTION SCHEMES

The life span of a network is partitioned into time intervals called sessions. m denotes the number of sessions, which should be determined in advance in some schemes. Let $U = \{U_1, \ldots, U_n\}$

be the finite universe of users. Each user U_i has a unique identifier ID_i. A broadcast unreliable channel is available, and time is defined by a global clock. The GM sets up and manages, by means of joining and revoking operations, a communication group which is a dynamic subset of U. All the operations take place in a fixed range (here we suppose the range is F_p, where p is a sufficiently large prime number). Let $G_j \subseteq U$ be the communication group established by the GM in session $j(j = 1, \ldots, m)$. Each user $U_i \in G_j$ holds a personal key S_i, received from the GM before or when joining G_j. The personal key is used to recover the session keys as long as U_i is not removed by the GM from the group. Figure 1 depicts a general communication model.

Let $R_j \subseteq G_{j-1}$ denote the set of revoked group users in session j and $J_j \subset U \setminus G_{j-1}$ denotes the set of users who join the group in session j with $R_j \cap J_j = \phi$. Hence, $G_j = (G_{j-1} \cup J_j) \setminus R_j$ for $j \geq 2$ and by definition $G_1 = U$. Moreover, for session $j \in \{1, \ldots, m\}$, the session key K_j is chosen independently and according to the uniform distribution on F_p. For any non-revoked user $U_i \in G_j$, the j-th session key K_j is determined by broadcast information B_j and personal key S_i. All the aforementioned notions are used throughout the paper.

4.1. The general model of self-healing key distribution scheme

Self-healing key distribution schemes are defined under security models. The pioneering self-healing key distribution scheme [10] is unconditionally secure and defined by entropy function $H(\cdot)$ (see [25] for more details). The subsequent models are slightly modified versions of the basic one.

To clarify the performance of the schemes and facilitate the later analysis, we mention the following formal security model of the self-healing key distribution scheme with revocation capability. Here we consider the original communication model. Suppose there are n users and a GM. The life-time of the network is divided into m sessions. t is the maximum number of users that can be revoked in the life-time of the network. We define the general self-healing key distribution scheme from two aspects. *Definition 1* defines self-healing key distribution scheme with revocation capability. *Definition 2* defines the properties of security including forward secrecy, backward secrecy and the capability of collusion resistance.

Definition 1. *Let U be the universe of users of a network, m be the maximum number of sessions and t be the maximum number of users that can be revoked by the GM.*

(i) *The scheme is a session key distribution scheme if the following conditions are true:*

(a) *For any user $U_i \in G_j$, the key K_j is determined by B_j and S_i. Formally, it holds that:*

$$H(\mathbf{K}_j|\mathbf{B}_j, \mathbf{S}_i) = 0. \tag{1}$$

(b) *What users learn from the broadcast B_j and their own personal key cannot be determined from the broadcasts or personal keys alone. That is:*

$$\begin{aligned} &H(\mathbf{K}_1, \ldots, \mathbf{K}_m|\mathbf{B}_1, \ldots \mathbf{B}_m) \\ &\quad = H(\mathbf{K}_1, \ldots, \mathbf{K}_m|\mathbf{S}_{G_1 \cup \ldots \cup G_m}) \\ &\quad = 0. \end{aligned} \tag{2}$$

(ii) *The scheme has t-revocation capability if, for each session j, let $R = R_j \cup \ldots \cup R_1$, such that $|R| \leq t$, the GM can generate a broadcast B_j such that the collusion of all revoked users in R cannot recover K_j. Formally, it holds that:*

$$H(\mathbf{K}_j|\mathbf{B}_1, \ldots \mathbf{B}_j, \mathbf{S}_R) = H(\mathbf{K}_j) \tag{3}$$

where $\mathbf{S}_R$ denotes the set of personal keys of all users in R.

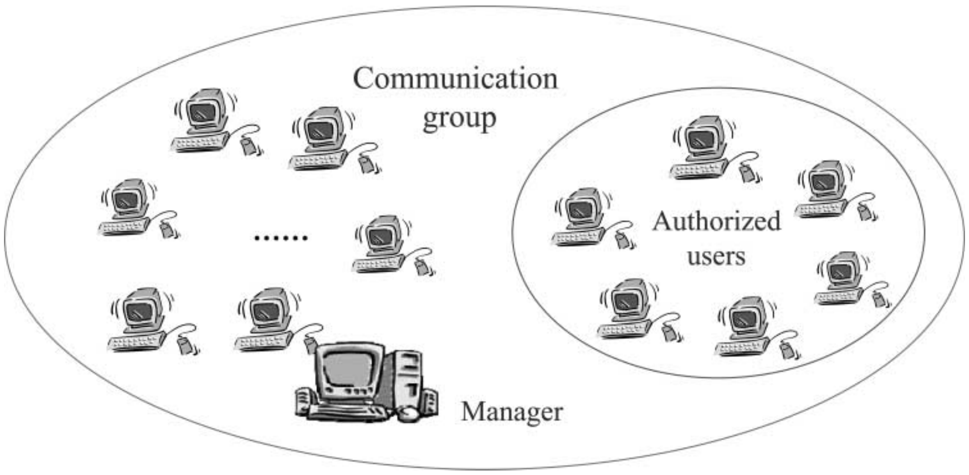


FIGURE 1. The general communication model of self-healing key distribution schemes.

(iii) *The scheme is self-healing if the following property is satisfied:*

Every $U_i \in G_r$, who has not been revoked after session r and before session s, from broadcast messages B_r and B_s, where $1 \leq r < s \leq m$, can recover all keys K_l, for $l = r, \ldots, s$. Formally, it holds that:

$$H(\mathbf{K}_r, \ldots, \mathbf{K}_s | \mathbf{B}_r, \mathbf{B}_s, \mathbf{S}_i) = 0. \tag{4}$$

Definition 2. *Let $U = \{U_1, \ldots, U_n\}$ and $j \in \{1, \ldots, m\}$. The scheme guarantees both forward and backward secrecy if the following properties are satisfied:*

(i) *For any set $R \subseteq U$, and all the users in R are revoked before session j, it is computationally infeasible for the $U_i \in R$ together to get any information about K_j, even with the knowledge of group keys $K_1, \ldots, K_{j-1}$ before session j. Formally, it holds that:*

$$H(\mathbf{K}_j | \mathbf{B}_1, \ldots, \mathbf{B}_m, \{S_i\}_{U_i \in R}, K_1, \ldots, K_{j-1}) = H(K_j) \tag{5}$$

(ii) *For any set $J \subseteq U$, and all $U_l \in U$ join after session j, it is computationally infeasible for the users in J together to get any information about K_j, even with the knowledge of group keys $K_{j+1}, \ldots, K_m$ after session j. Formally, it holds that:*

$$H(\mathbf{K}_j | \mathbf{B}_1, \ldots, \mathbf{B}_m, \{S_i\}_{U_i \in R}, K_{j+1}, \ldots, K_m) = H(K_j) \tag{6}$$

(iii) *Let $C \subseteq R_r \cup, \ldots, \cup R_1$ be a coalition of users removed before session r and let $D \subseteq J_s \cup, \ldots, \cup J_m$ be a coalition of users who join the group from session s. Let $|C \cup D| \leq t$. Then, such a coalition does not get any information about keys K_j, for any $r \leq j < s$. Formally, it holds that:*

$$H(\mathbf{K}_r, \ldots, \mathbf{K}_{s-1} | \mathbf{B}_1, \ldots, \mathbf{B}_m, \mathbf{S}_C, \mathbf{S}_D) = H(\mathbf{K}_r, \ldots, \mathbf{K}_{s-1}) \tag{7}$$

In *Definition 1*, Condition (i) states that every user U_i, from the broadcast B_j and its own personal key S_i, recovers the current session key K_j; while, personal keys and broadcasts alone, do not give any information about any session key. Condition (ii) means the GM is able to revoke at most t users from the group. Condition (iii) characterizes the self-healing property: any two broadcasts are enough to recover all lost session keys for the ‘sandwich’ sessions. In *Definition 2*, Conditions (i) and (ii) describe the forward secrecy and backward secrecy, respectively. Condition (iii) defines the feature of resisting collusion. It states that a coalition of or less t revoked users and the newly joined users cannot obtain any information about the current session key.

4.2. A lower bound on personal keys

Here we emphasize a lower bound on personal keys, which can be neglected easily, for unconditionally self-healing key distribution scheme.

Theorem 1. *In any unconditionally secure self-healing key distribution scheme with key space of size p, for any user U_i belonging to the group since session j, it holds that*

$$H(S_i) \geq (m - j + 1) \log p$$

Therefore, each user who joins group in session j must store a personal key of at least $(m - j + 1) logp$ bits. (see [10, 16, 26] for proof.)

4.3. The procedures of self-healing key distribution schemes

Self-healing key distribution entails five basic procedures, namely *Setup*, *Broadcast*, *Key Recovery*, *Adding or Revoking Users* and *Self-healing*. The basic procedures in a general self-healing key distribution are described in Fig. 2. The entities in self-healing key distribution scheme includes a GM and a great number of end users. The GM is responsible for the procedures of *Setup*, *Broadcast* and *Adding or Revoking Users*. The end

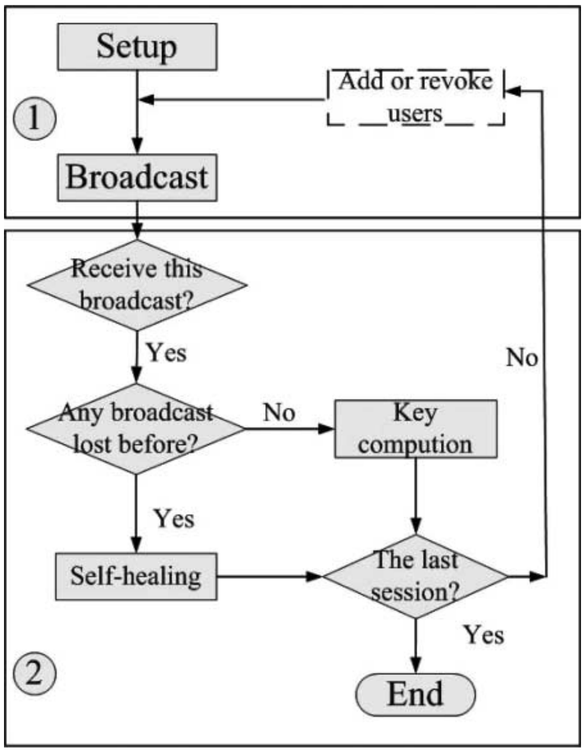


FIGURE 2. The basic procedures in a general self-healing key distribution, where the small panes are operation which must be executed in each round, the small dashed frames represent the operations which may not be executed in some round of the scheme.

users perform *Key Recovery* and *Self-healing*. In Fig. 2, The large pane 1 includes the operations performed by the GM, while the large pane 2 includes all the operations performed by the end users.

4.3.1. Setup

The GM chooses and releases system parameters. It also generates and privately distributes personal keys to current group users. It is supposed that the personal keys are distributed in a off-line manner or there is a secure channel between the group manager and each user in this procedure. At last, the GM chooses m session keys $K_1, \cdots, K_m$ from a fixed key space (for example F_p). These session keys are independent to each other and according to uniform distribution.

4.3.2. Broadcast

Let $|G_j|$ denote the number of users in session j. For each session $1 \leq j \leq m$, according to the session group G_j, the GM computes and broadcasts message B_j so that only authorized users can recover the session key from the broadcast message and their personal keys. At the same time, the construction of broadcast message should meet the properties of forward and backward secrecy as well as the property of collusion resistance.

4.3.3. Key recovery

When an authorized user $U_i \in G_j$ receives the broadcast message B_j, it can recover the session key for the jth session using its personal key and the received broadcast message.

4.3.4. Adding or revoking users

When the GM wants to add a new user from session j, it will generate and privately distribute personal key to the new user so that the new user can recover session keys in subsequent sessions. Similarly, if the GM wants to revoke a user from session j, it will add or delete some information related to the revoked user in the procedure of *Broadcast* so that the revoked user cannot recover the session keys in subsequent sessions. Both forward and backward secrecy as well as the property of collusion resistance should be guaranteed in this procedure.

4.3.5. Self-healing

If a user misses some broadcast messages before session j, when it receives the broadcast message in session j, it can recover all the lost session keys using its personal key and the broadcast B_j. The only requirement that must be satisfied, in order for the user to recover the lost keys, is the membership in the group both before and after the sessions in which the broadcast messages containing the keys are sent.

Among these procedures, *Setup* only happens at the period of initialization. *Broadcast* and *Key Recovery* are performed in each session. While *Adding or Revoking Users* is not necessarily happen in each session. *Self-healing* is triggered by the loss of broadcast messages. It happens only in the sessions before which the broadcast messages get lost.

5. TYPICAL SELF-HEALING KEY DISTRIBUTION SCHEMES

In this section, we present an overview of existing works in the area of self-healing key distribution according to their classification. Here we omit the proof of security of each scheme because the security of each scheme was proved in the corresponding referenced paper. Nevertheless, we make a thorough analysis and comparison on performance of schemes in the same classification.

5.1. Polynomial secret-sharing-based self-healing key distribution schemes

The idea of self-healing key distribution scheme was introduced by Staddon *et al.* in the pioneering work [10]. Formal definitions, lower bounds on the resources as well as some constructions of self-healing key distribution scheme were proposed in it. The GM, at the beginning of each session, sends packets over an unreliable broadcast channel in order to provide a key to each user of the group. Combined with the pre-distributed secret key, every authorized user can recover the session key from the packets. By this way, the GM can launch multiple sessions during a certain time interval, by adding/removing users to/from the initial group. The self-healing feature in key distribution scheme enables a GM to distribute session key for a dynamic group over an unreliable network. If, at the beginning of a certain session, some broadcasted packets get lost, then users are still capable of recovering the session key for that session simply by using the packets they have received at the beginning of one of a previous session and the packets they will receive at the beginning of a subsequent one, without requesting additional transmission from the GM. The only requirement that a user must satisfy in order to recover the lost keys through self-healing is its membership in the group both before and after the sessions in which the broadcast messages containing the keys are sent. Such approach reduces network traffic and the work load on the GM as well as the risk of user exposure through traffic analysis. Since then, self-healing key distribution has been one of hot research topics.

This scheme explored the way to extend the lifetime of the basic self-healing key distribution scheme. After a set of m session has expired, the operation of rekeying is necessary before distributing new session keys due to the changed membership and released personal keys of the revoked users. A straightforward method is to redistribute a new set of personal keys to each user, and proceeds as before. Another method is to do polynomial interpolation in the exponent. This is accomplished through the broadcast of random values from the GM. This operation allows each user to evolve their personal keys from one set of m sessions to the next, thus making the scheme long-lived without any unicasts from the GM. The second method in significant bandwidth saving over the first one.

Liu et al. generalized the definitions in [10] and presented several constructions in [27]. The scheme introduced a novel key distribution scheme first by using both revocation polynomial and mask polynomial. Then they added the self-haling feature to the basic key distribution scheme, thus the scheme reduces communication and storage overhead greatly. Furthermore, they developed two techniques that allow a trade-off between the broadcast message size and the recoverability of lost session keys. The first technique deals with possibly frequent but short-term communication failures and the second technique aims at situations where there are relatively long term but infrequent communication failures. The two methods further reduced the size of broadcast messages in these two conditions, respectively. An unreasonable requirement of [27] is that the set of revoked group users must change monotonically. That is, $R_{j_1} \subseteq R_{j_2}$ for sessions j_1 and j_2 $(1 < j_1 < j_2 < m)$. Otherwise, a group user who is revoked in session j_1, rejoins the group in a later session j_2, can recover the key for session j_1. Therefore, the scheme is prone to a rejoin–collusion attack and does not allow temporary revocation.

More *et al.* [11] addressed the three problems discussed in [10] using a sliding window mechanism. The three problems are inconsistent robustness, high overhead and expensive maintenance cost. The scheme achieved good performance. Above all, the usage of a sliding window mechanism makes error recovery consistently robust. In addition, the GM is entitled to spread the cost of personal key distribution over multiple sessions, rather than having to distribute new personal keys to all users at the same time. Furthermore, reusing masking polynomial reduces personal key storage and broadcast size dramatically.

Blundo *et al.* [28] proposed a new mechanism for implementing self-healing key distribution. Moreover, they described a secure and efficient construction which has optimal memory storage and communication complexity. Shortly after that, they in [26] presented an attack on the first construction discussed in [10]. Then, they proposed a new self-healing key distribution scheme, which requires low storage and communication overhead. Finally, they slightly modified the security model, in order to extend the self-healing key distribution model, and proposed a scheme which enables a user to recover from a single broadcast message all keys associated with sessions in which it is a user of the communication group. They pointed out two problems of the long-lived ‘Construction 5’ in [10]. In [10], the values that are used for computing the instances of the personal key are sent by means of a single broadcast message at the beginning of each new set of m sessions. Since the network is not reliable, if some user does not receive such a message, She will get out from the corresponding m sessions. This problem is solved by sending with each broadcast the values that are used for evolving the personal key in each session. For the solution of the first problem, the involved cost of modular exponentiation operations is too high to afford for low-cost wireless networks. This solution is turned out to be infeasible. The second problem lies in the join operation in the presence of new users. It cannot be solved by slight modifying ‘Construction 5’ given in [10] and ‘Scheme 4’ given in [26] in order to enable a secure join. It is still an open problem.

Hong *et al.* proposed a new self-healing key distribution scheme in [29]. It is one of efficient unconditionally secure self-healing key distribution scheme. As the original self-healing key distribution scheme, it includes five procedures:

(i) Setup: The GM randomly chooses m t-degree polynomials $s_1(x), \ldots, s_m(x) \in F_p[x]$ and m session keys $K_1, \ldots, K_m \in F_p$. Both polynomials and session keys are independent on each other and according to uniform distribution. Then, the GM privately sends personal key $S_i = \{s_1(i), \ldots, s_m(i)\}$ to $U_i (i = 1, \ldots, n)$.

(ii) Broadcast: Let $W_j = \{r_1^j, \ldots, r_{w_j}^j\}$ be the set of revoked user IDs for sessions in and before j such that $|W_j| = w_j \le t$, and let $r_j(x) = (x - r_1^j), \ldots, (x - r_{w_j}^j)$. The broadcast message for the j-th $(j = 1, \ldots, m)$ session is in this form:

$$B_j = \{P_1(x), \ldots, P_j(x)\} \cup \{W_i\}_{i=1,\ldots,j},$$

where $P_j(x) = r_j(x)K_j + s_j(x)$.

(iii) Key Recovery: When a user $U_i \in G_j$ receives the j-th broadcast message, it evaluates the polynomial $r_j(x)$ at point i, and computes current session key by computing

$$K_j = \frac{P_j(i) - s_j(i)}{r_j(i)}.$$

(iv) Adding or Revoking Users: When the GM wants to add a new user starting from the r-th session, it gives a unused unique identity $i' \in F_p$, computes personal keys corresponding to the current and future sessions $\{s_k(i')\}(k = r, \ldots, m)$ and privately sends the keys to this new user. When the GM wants to revoke a user $U_{i'}$ starting from session s, it adds the identity $i' \in F_p$ to W_j for $j = s, \ldots, m$.

(v) Self-healing: For any U_i that is a user in session r and $s(1 \le r < s \le m)$, it can recover $\{P_r(i), \ldots, P_s(i)\}$. By the method described in Key Recovery, U_i can subsequently recover the whole sequence of session keys $K_r, \ldots, K_s$.

Compared with previous schemes, this scheme has good combined performance. It is optimal in terms of user memory storage and more efficient in terms of communication complexity.

The papers [10, 11] focus on unconditionally secure self-healing key distribution. The definitions and constructions were stated in terms of the entropy function. Blundo *et al.* [30] analyzed current definitions of self-healing key distribution. They showed that no protocol can achieve the security requirements stated in [10] and [27] and identified where the proposed schemes fail. They also showed that a previously derived lower bound on the size of the broadcast messages that the GM has to send in order to establish session keys, proved in [10] and also used in [27], does not hold. After analysis, they proposed a new definition of self-healing key distribution and showed that it can be achieved by concrete schemes. Some lower bounds on the resources, such as user memory storage and communication complexity, required for implementing such schemes were given finally and showed that these lower bounds are tight through simple constructions.

Dutta claimed that they realized unconditionally secure self-healing key distribution schemes in [31] and [32]. The storage overhead in these two papers is $(t + 1) \log p$, where t is the maximum number of compromised users and p is the key space. However, this claim cannot hold. Indeed, in [10], [16] and [26], it has been proved that a lower bound on the size of the storage overhead is equal to $(m - j + 1) \log p$, where m is the life-time of the network, j is order of the current session and *logp* is the size of the distributed key. According to the above-mentioned bound, one cannot design an unconditional secure self-healing scheme for m sessions by giving to each user a $(t+1) \log p$-size secret key which is updated by a broadcast message. The papers [10], [16] and [26] are referenced by both [31] and [32], but it seems that the authors missed this lower bound.

Zou and Dai [33] proposed a new self-healing scheme based on a novel concept of access polynomial. It overcomes some shortcomings of the existing schemes yet still possesses all the advantages of them. We should point that the communication overhead in [33] comes from broadcast B_j which is composed by $2m + 2$ polynomials. As far as the polynomial $P_j(x)$ is concerned, $P_j(x) = A_j(x) \cdot S_j(x) + H(x)$. Both $S_j(x)$ and $H(x)$ are t degree polynomials, and the degree of $A_j(x)$ amounts to $(|G_j| + 1)$ where $|G_j|$ is the number of users in current communication group. Generally speaking, $|G_j|$ is larger than t. Therefore, the claim that communication overhead is $O(mt)$ is incorrect.

Tian *et al.* introduced and improved a secret-sharing scheme in [13]. Then, they applied the secret-sharing scheme to the design of a self-healing key scheme. The new scheme achieves several nice properties. First, the scheme reduces the storage overhead of personal key to a constant. Secondly, the scheme cancels the requirement of secure channel in setup phase. In addition, the long-lived scheme is much more efficient than those in [10] and [26]. However, the efficiency improvements are obtained by relaxing the security slightly. The scheme is a computationally secure scheme. By introducing a one-way key chain, Dutta *et al.* proposed two constructions of scalable self-healing key distribution with t revocation capability in [14]. The schemes reduce both communication and computation overheads greatly without increasing the storage overhead. At the same time, forward and backward secrecy are achieved. However, there is a fatal defect in their constructions. The collusion between the newly joined users and the revoked users can recover all the session keys which they were not authorized to.

In terms of storage overhead, the schemes require that each user stores a personal key for each session. It comes from the procedure of *Setup* and after receiving the session key distribution broadcast. Communication overhead comes from the procedure of *Broadcast*. Generally speaking, the broadcast message for the j-th session consists of identifiers of a set of revoked users. Since the user identities can be selected from a small finite field , one can ignore the communication overhead brought by the identifies of all the revoked users set [29]. Computation overhead is introduced by the procedures of *Key Recovery* and *Self-healing*. It is supposed that the GM has more resource to operate computation while end users have limited computational ability. Therefore, we talk only about the computation cost at users' end. For different cryptographic primitives the schemes based on, the computation overhead varies.

To clarify the performance of the proposed schemes, we present a thorough comparison of them. Table 2 summarizes the comparison in terms of storage, communication, computation overheads and several security parameters. We use C to denote Construction and S to denote Scheme, for example, $C3$ in [10]

TABLE 2. Performance comparison among different polynomial secret-sharing based self-healing key distribution schemes.

Method	Storage overhead	Communication overhead	Computation overhead	Forward and backward secrecy	Collusion resistance	Secure channel in initialization
C3 in [10]	$(m-j+1)^2 \log p$	$(mt^2+2mt+m+t)\log p$	$2mt^2+3mt-t$	Both, t-wise	At most t	Yes
S3 in [27]	$2(m-j+1)\log p$	$[(m-j+1)t+(m+1)]\log p$	$mt+t+2tj+j$	Both, t-wise	At most t	Yes
S2 in [26]	$(m-j+1)\log p$	$(2tj+j)\log p$	$2j(t^2+t)$	Both, t-wise	At most t	Yes
C1 in [29]	$(m-j+1)\log p$	$(tj+j-t-1)\log p$	$j(2t+1)$	Both, t-wise	At most t	Yes
[28]	$(m-j+1)\log p$	$(t+1)tj\log p$	$j(2t^2+3t)$	Both, t-wise	At most t	Yes
[13]	*logp*	$(t+1)j\log p$	$j(\log p+2t^2+2t)$	Both, t-wise	At most t	No
C1 in [14]	$(m-j+1)\log p$	$(t+1)\log p$	$2t+1$	Both, t-wise	No	Yes
C2 in [14]	$(m-j+1)\log p$	$(t+1)\log p$	$2(t^2+t)$	Both, t-wise	No	Yes

denote the ‘Construction 3’ in [10]. We suppose the life-span of schemes is m sessions. j represents the order of a session, t represents the maximum number of users that can be revoked and $logp$ represents the length of private and session keys. The storage overhead is measured by the length of personal key. The communication overhead comes from broadcast messages. For a user U_i at the j-th session, the computation overhead is incurred by recovering all previous session keys up to the j-session (worst case) by self-healing mechanism [14]. The key recovery operations include computing one or more points on polynomials (such as operations in [27, 29]) or recovering a t degree polynomial by using Lagrange formulation (such as operations in [13, 14, 26, 28]) or both (such as operations in [10]). Computing a point on a t degree polynomial requires at most t multiplication operations and division can be regarded as multiplication when we calculate computation overhead. Compared with the overhead brought by multiplication, the overhead brought by addition and hash operations can be neglected. Therefore, the computation cost for each user is $2t+1$ of ‘Construction 1’ in [14], whereas the computation complexity of the scheme of ‘Construction 1’ in [29] is $j(2t+1)$. Recovery of a t degree polynomial by using Lagrange formulation requires $2\{(t+1)^2-(t+1)\} = 2(t^2+t)$ multiplication operations. Thus the computation overhead of ‘Scheme 2’ in [26] is $2j((t^2+t))$ and that of ‘Construction 2’ in [14] is $2((t^2+t))$. t-wise for forward and backward secrecy means these schemes can keep forward and backward secrecy for the coalition of at most t unauthorized users. The capability of resisting collusion means the scheme is secure even at most t newly joined users and revoked users collude to recover the session keys that they do not authorized to access. Up to now, the scheme in [13] has been the only self-healing key distribution scheme without the requirement of secure channel between members and the GM in initialization stage.

In most of the previous schemes, storage and communication overhead increases with the order of a session. Computation overhead for the current session key is fixed in each session while computation overhead for self-healing mechanism increases with the order of session too. Without loss generality, we set $j = m/2$. In addition, we assume that p is a 64-bit integer. Suppose the maximum number of sessions $m = 100$ and the number of revoked users vary from 0 to 100. We further use figures to illustrate the performance of the existing schemes. For simplicity, we compare only several representative schemes. They are ‘Construction 1’ in [13, 26, 29], and ‘Construction 1’ in [14].

Figures 3, 4, and 5 illustrate the increase in tendency in terms of storage, communication and computation overhead for the four self-healing key distribution schemes, respectively. As can

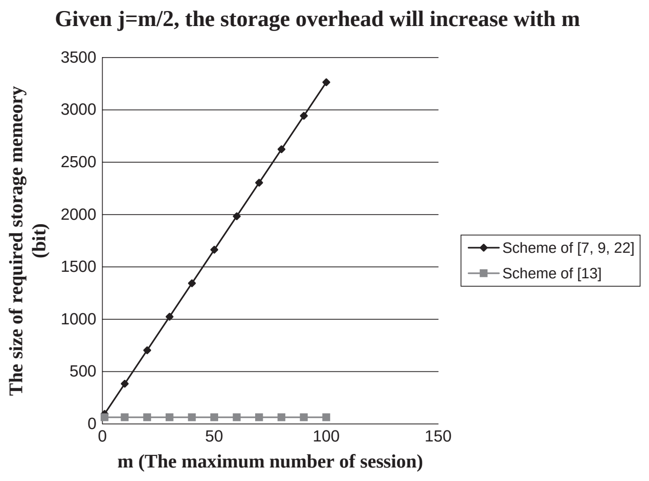


FIGURE 3. This figure demonstrates storage overhead of four representative schemes when m varies from 0 to 100 and $j = m/2$.

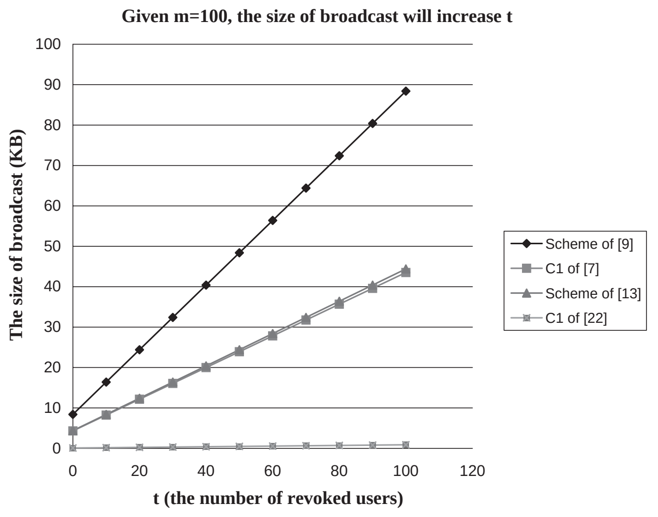


FIGURE 4. Communication overhead of four representative schemes when t varies from 0 to 100, $m = 100$ and $j = m/2 = 50$.

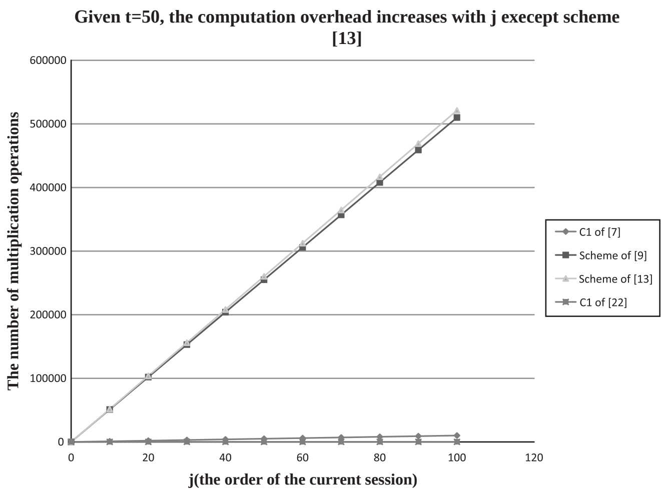


FIGURE 5. Computation overhead of four representative schemes when j varies from 1 to 100 given $t = 50$.

be seen from the figures, the scheme in [13] has lowest storage overhead, while the Construction 2 in [14] is the best scheme in terms of communication and computation overhead.

5.2. Vector space secret-sharing-based self-healing key distribution schemes

Sáez [16] considered applying vector space secret-sharing instead of polynomial secret-sharing schemes to realize self-healing key distribution scheme. The scheme makes use of general monotone decreasing structures for the family of subsets of users that can be revoked instead of a threshold one. Thus the scheme achieves more flexible performance than polynomial secret-sharing-based schemes. Another advantage of the scheme is that the distance between the broadcasts used to recover the lost one is variable. The reason for this modification is to allow adjusting the length of broadcasts according to the condition of networks. In the same year, Sáez [17] considered the possibility that a coalition of users sponsors a user outside the group for one session. First, the formal definition and some bounds on the required amount of information were given. Then a general construction of a family of self-healing key distribution schemes with sponsorization was presented. The sponsorization mechanism strengthens the robustness of the scheme. An authorized subset of users in the group has the ability to invite a new user to join the group without the help of the GM. Both [16] and [17] are unconditionally secure schemes. The model in [17] not only includes *Definitions 1* and *2* but also includes *Definition 3*.

DEFINITION 3. *The scheme has the property of sponsorization. This means that the three following properties are satisfied:*

(i) *Every user $U_l \in G_j$ can generate a proof of sponsorization P_{li}^j to sponsor a user $U_i \notin G_j$ for session j using his personal key. In other words:*

$$H(P_{li}^j | S_l) = 0. \tag{8}$$

(ii) *A user $U_i \notin G_j$ that receives enough sponsorization from a subset of users $A \subset G_j$ with $A \in \Gamma$ can compute the key K_j in the same conditions that users in G_j. That is: for $A \in \Gamma$, $A \subset G_j$, $i \notin G_j$ and $r \le j \le s$, there exists*

$$H(K_j | P_{Ai}^j, B_r, B_s) = 0 \tag{9}$$

(iii) *Suppose that a coalition of users $U_{i_1}, \ldots, U_{i_u} \notin G_j$, not revoked before session j, have received sponsorization from subsets of users $C_1, \ldots, C_u \notin \Gamma$, respectively, with $C_1 \cup \ldots \cup C_u = \{U_{l_1}, \ldots, U_{l_v}\} \subset G_j$. This action is performed in such a way that users $U_{l_1}, \ldots, U_{l_v}$ sponsor subsets of users $D_1, \ldots, D_v \in \mathcal{S}$, respectively, with $D_1 \cup \ldots \cup D_v = \{U_{i_1}, \ldots, U_{i_u}\} \subset U - G_j$; therefore, $P_{C_1 i_1}^j, \ldots, P_{C_u i_u}^j = P_{l_1 D_1}^j, \ldots, P_{l_v D_v}^j$. In these conditions, such a coalition does not get any information about the value of key K_j. Formally, it holds that:*

$$H(K_j | P_{C_1 i_1}^j, \ldots, P_{C_u i_u}^j, B_r, B_s) = H(K_j) \tag{10}$$

for $C_1, \ldots, C_u \notin \Gamma$, $D_1, \ldots, D_v \in \mathcal{S}$, such that $P_{C_1 i_1}^j, \ldots, P_{C_u i_u}^j = P_{l_1 D_1}^j, \ldots, P_{l_v D_v}^j$, $C_1 \cup \ldots \cup C_u = \{U_{l_1}, \ldots, U_{l_v}\} \subset G_j$, $D_1 \cup \ldots \cup D_v = \{U_{i_1}, \ldots, U_{i_u}\} \subset U - G_j$ and $r \le j \le s$.

In *Definition 3*, Condition (i) describes the property of sponsorization: the information used to sponsor is computed from the personal key. Condition (ii) describes the fact that the information obtained from enough sponsorizations with the correspondent broadcast allows to compute the personal key of the session. Condition (iii) describes the security requirement: a coalition of users outside G_j sponsored by not enough users cannot obtain any information about the value of the key K_j. The key remains secure even if every user receives sponsorization of a coalitions in $\mathcal{S}$.

The self-healing key distribution scheme with sponsorization includes six procedures. The process was described in Fig. 2 of [34].

(i) Setup: The GM randomly chooses $t \times t$ matrices $P_1, \ldots, P_m$ and session keys $K_1, \ldots, K_m \in GF(q)$. For each session $j = 1, \ldots, m$, the group manager computes the vector $z_j = K_j + \psi(D)^\top P_j \in GF(q)^T$. For each $U_i \in G_1$, the GM computes the person key $S_i = (\psi(i)^\top P_1, \ldots, \psi(i)^\top P_m) \in GF(q)^{tm}$. The use of specific function ψ fixes the properties of the scheme.

(ii) Broadcast: In jth session, Suppose $R_j \subset G_{j-1}$ with $R_1 \cup \ldots \cup R_j \in \mathcal{R}$ if $j \ge 2$. By definition, we have $R_1 = \phi$. The GM chooses a maximal non-authorized subset of users $W_j \in \mathcal{R}_0 = \overline{\Gamma}_0$ such that $R_1 \cup \ldots \cup R_j \subset W_j$ and $W_j \cap G_j = \psi$ with minimum cardinality. The broadcast B_j in session $j = 1, \ldots, m$ is given in the form of $B_j = B_j^1 \cup B_j^2$. The first part B_j^1 is composed of vectors z_j and each z_j is divided into two parts (x_j, y_j) where the x_j is the first part of the binary representation of every component of z_j and y_j is the second part. Then $B_j^1 = (X_j, Y_j)$, where

$$X_j = \begin{cases} x_j & \text{if } j = 1, 2 \\ x_1 + x_2, \ldots, x_1 + x_{j-1}, x_j & \text{if } j = 3, \ldots, m \end{cases}$$

$$Y_j = \begin{cases} y_j, y_m + y_{j+1}, \ldots, y_m + y_{m-1} & \text{if } j = 1, 2 \\ y_j & \text{if } j = m-1, m \end{cases}$$

The second part of the broadcast is defined as follows:

$$B_j^2 = \begin{cases} \{(k, \psi(k)^\top P_j)\}_{k \in W_j} & j = 1, 2 \\ B_{j-1}^2 \cup \{(k, \psi(k)^\top P_j)\}_{k \in W_j} & j \ge 3 \end{cases}$$

(iii) Key recovery: When a non-revoked user U_i receives the key distribution message B_j for session j, since

$U_i \in G_j$ has $\{(k, \psi(k)^\top P_j)\}_{k\in W_j}$ and its personal key, it computes $\psi(D)P_j$ using $\{(k, \psi(k)^\top P_j)\}_{k\in W_j\cup\{i\}}$ because $W_j \cup \{i\} \in \Gamma$. In fact, as far as $W_j \cup \{i\} \in \Gamma$, the result $\psi(D) = \sum_{k\in W_j\cup\{i\}} \lambda_k \psi(k)$ holds for some $\lambda_k \in GF(q)$. Therefore, $\psi(D)P_j = \sum_{k\in W_j\cup\{i\}} \lambda_k \psi(k)P_j$. From the broadcast information B_j, then U_i recovers the jth session key

$$K_j = z_j - \psi(D)^\top P_j.$$

(iv) Self-healing: The phase can be seen as an iterative algorithm of Key recovery. If a user U_i receives two broadcast message B_r and B_s, it can recover the polynomials $z_r, z_{r+1}, \ldots, z_s$. If U_i is an authorized user for session $j(j = r, \ldots, s)$, it performs the exact operation on each z_j and $\{(k, \psi(k)^\top P_j)\}_{k\in W_j}$ as it does in the procedure of Key recovery. Then it can recover the session keys $K_r, \ldots, K_s$.

(v) Adding or revoking users: If the GM wants to add users $J_j \subset U$ in session j, the GM computes $S_i = (\psi(i)^\top P_{j+1}, \ldots, \psi(i)^\top P_m) \in GF(q)^{t(m-j+1)}$ and send it to each user $U_i \in J_j$ as its personal key through the secure channel between them.

If the GM want to revoke users $R_j \subset G_{j-1}$ in session j, what the GM should do is adding the identifiers of users in R_j to W_j and constructing a new broadcast message. The revoked users cannot decode the broadcast message B_j.

(vi) Sponsored addition of users: If a user $U_l \in G_j$ wants to sponsor a user $U_i \notin G_j$ for session j, then it computes $(l, \psi(l)^\top P_j \psi(i))$ from its personal key $(l, \psi(l)^\top P_j)$ and privately sends it to U_i. Then the sponsored user can compute the current session key. See [17] for detail of key computation by the sponsored user.

In fact, polynomial secret-sharing-based self-healing key distribution scheme is a particular case of vector space secret-sharing-based scheme. In the class of polynomial secret-sharing-based schemes, the access structure is fixed to the degree of the underlying polynomial. Therefore, vector space secret-sharing-based schemes achieve more flexible properties.

The schemes [16, 17] use the same vector space secret-sharing mechanism. The storage, communication and computation overheads are much similar. In terms of storage overhead, each user has to store a personal key of size $(m - j + 1)\log p$ in [16] and [17]. The storage overhead is optimal with respect to Theorem 1 in [16]. The communication overhead comes from the broadcast which depends on the particular function ψ used. According to [16], the broadcast can be divided into two parts. The first part of broadcast is defined as (x_j, y_j). The total number of broadcast bits is $(1 + (t + 1)(m - 1 - t/2))\log p$. According to [17], B_1^1 and B_m^1 have $1/2tm \log p$ bits and B_j^1 for $j \neq 1, m$ has $1/2t(m - 1)\log p$ bits. Then the total number of broadcast bits is $1/2t(m^2 - m + 2)\log p$. The second part of broadcast in both [16] and [17] is composed of the identities of revoked users and its purpose is to perform the rejection capability as well as the recovery of the session key. Its length depends on the history of rejected subsets. Since the user identities can be chosen from a small finite field, we can ignore the communication overhead for the broadcast of all those revoked sets. The computation overhead of [16] and [17] is $2\sum_{l=1}^{j}(t_l^2 + t_l)$ because the user U_i has to compute $\psi(D)P_j = \sum_{k\in W_j\cup\{i\}} \lambda_k \psi(k)P_j$ for each lost key. Here we use Table 3 to highlight the features of the two schemes. The schemes [18–20] are hash chain-based self-healing key distribution schemes with vector space secret-sharing-based masking mechanism. The performance of them will be compared in Table 4.

5.3. SDR-based self-healing key distribution schemes

SDR [35] is a stateless rekeying method. In this protocol, a key server maintains a logical key tree and every user is mapped to a leaf node of the key tree. In each rekeying operation, the key server partitions the current group into a minimal number of subsets, and then encrypts the new group key with the common key of each subset, respectively. In this scheme, the communication overhead is independent of the group size, thus the scheme is scalable. One of the advantage of the scheme is that there is no dependency between the keys used in different rekeying operations. In order to decode the current group key, a user only needs to receive the keys that are transmitted by the key server during the current rekey operation. This property makes the SDR scheme very attractive for secure multicast applications where users may go offline frequently or experience burst packet losses.

The SDR method has good performance in key recovery operation and is secure against the collusion of any number of revoked users. In contrast, polynomial-based key distribution

TABLE 3. Comparison of features of vector space secret-sharing-based self-healing key distribution schemes.

Scheme	Storage overhead	Communication overhead	Computation overhead	Resisting collusion	Sponsorization
[16]	$(m - j + 1)\log p$	$(1 + (t + 1)$ $(m - 1 - t/2))\log p$	$2\sum_{l=1}^{j}(t_l^2 + t_l)$	Yes	No
[17]	$t(m - j + 1)\log p$	$0.5t(m^2 + m + 2)\log p$	$2\sum_{l=1}^{j}(t_l^2 + t_l)$	Yes	Yes

TABLE 4. Summary on the one-way hash function based self-healing key distribution schemes.

Scheme	Cryptographic primitives	Storage overhead	Communication overhead	Computation overhead	Partial collusion resistance
C1 of [14]	hash function and polynomial secret sharing	$(m-j+1)\log p$	$(t+1)\log p$	$2t+1$	No
[18]	hash function and vector space secret sharing	$2(t_i-s_i+1)\log p$	$(t_j+1)\log p$	$2({t_j}^2+t_j)$	No
[36]	hash function and XOR operation	$(m-j+2)\log p$	$j\log p$	constant	No
[19]	hash function, vector space secret sharing and XOR operation	$(2t_i-2s_i+3)\log p$	$(t_j+1)\log p$	$2({t_j}^2+t_j)$	Yes
C1 in [15]	hash function, polynomial secret sharing	$2(t_i-s_i+1)\log p$	$(t+1)\log p$	$2t(t+1)$	Yes
C2 in [15]	hash function, polynomial secret sharing	$2(t_i-s_i+1)\log p$	$(t+1)\log p$	$2(t+1)$	Yes
[20]	hash function, vector space secret sharing	$(t_i-s_i+t+2)\log p$	$(t_j+1)\log p$	$2({t_j}^2+t_j)$	Yes

proposed by Liu *et al.* [27] has the similar message size and has the feature that group users can recover group on its own under some conditions. However, this protocol has a constraint which may limit its application. That is, the maximum number of users that can be revoked during the life-long time of networks has to be pre-determined and must not be exceeded for the sake of security. The maximum number of users that can be revoked is depend on the degree of the polynomial. The higher the degree of the polynomial is, the larger the number of the users that can be revoked, and the larger the communication and computation overhead is and vice versa.

Zhu *et al.* first addressed adding self-healing feature to SDR in [22]. The key idea is to bind the ability of a user to recover a lost group key to its membership duration. They used a one-way hash key chain such that revoked users and new users cannot collude to recover the keys that they should not know. The notions are different from those used in the general self-healing key distribution schemes. The scheme can be displayed as follows:

(i) In each group rekeying, the GM generates a key chain of size $m+1$. Let the keys in the key chain generated for the rekeying at $T(i)$ be $K^m(i), \ldots, K^1(i), K^0(i)$, where $K^0(i) = H(K^1(i)) = H^2(K^2(i))) = \ldots = H^m(K^m(i)))$ and H is a one-way hash function such as SHA-1. Due to the one-wayness of the hash function, a user who knows K_i^j can compute all the keys $K^{j-1}(i), \ldots, K^0(i)$ independently, but it cannot compute all of the keys $K^{j+1}(i), \ldots, K^m(i)$. $K^0(i)$ is the group key that all the users should use for data encryption between $T(i)$ and $T(i+1)$.

(ii) The users in the group are considered to be partitioned into $m+1$ subgroups, depending upon their membership duration, each subgroup is associated with a separate key from the one-way key chain generated in the first step. Specifically, $K^j(i)$ is the key intended for the users that joined the group at $T(i-j)$ for $0 \leq j < m$, and $K^m(i)$ is the key intended for users that joined at or before $T(i-m)$.

(iii) The GM broadcasts m encrypted keys as shown below:

$$\{K^0(i-m)\}_{K^0(i-m-1)\oplus K^m(i)}, \ldots,$$
$$\{K^0(i-1)\}_{K^0(i-2)\oplus K^1(i)}.$$

The communication overhead of this protocol is very small. To allow an authorized user to recover the previous s group keys, the number of additional encryption keys to be transmitted is at most $3s$.

Bohio *et al.* [23] considered incorporating the self-healing feature to SDR rekeying method too. Some optimization techniques that can be used to reduce the overhead caused by the self-healing capability are proposed in the paper. In addition, the idea of mutual-healing was discussed. One motivation behind mutual-healing is that, if a node has missed a key updating message, it does not have to wait until the next update broadcast to recover the previous session key, instead it can acquire assistance from its neighboring nodes to recover that key instantly. The procedures of self-healing and mutual-healing are displayed in Fig. 6.

However, the SDR scheme itself has a limitation. The positions of the revoked users in the tree cannot be reused. This deficiency discounts the feasibility of SDR-based self-healing key distribution schemes accordingly.

5.4. Hash chain-based self-healing key distribution schemes

Jiang *et al.* [21] proposed an efficient self-healing group key scheme with time-limited node revocation based on dual directional hash chains (DDHCs). The performance of the proposed scheme under poor broadcast channel was evaluated by both theoretical analysis and numerical results. The result shows that the scheme tolerates high channel loss rate, and hence makes a good balance between performance and security. Therefore, it is suitable for wireless networks.

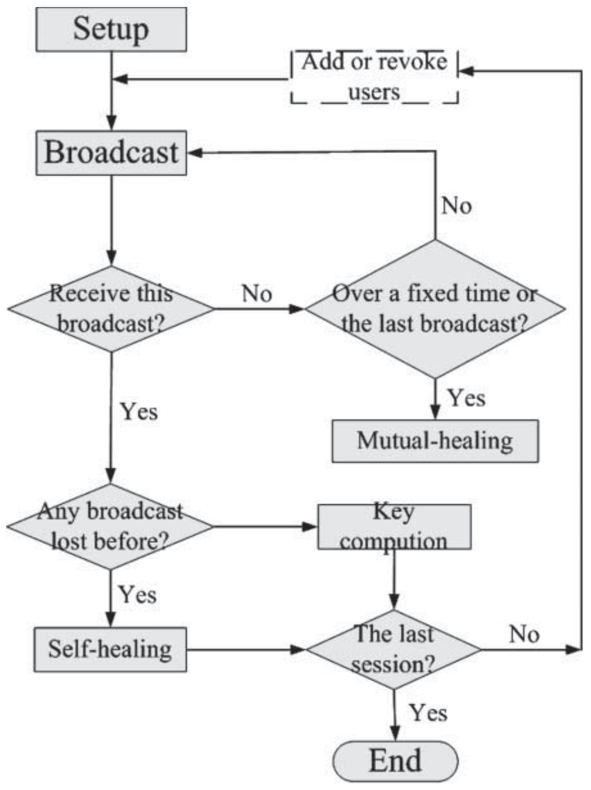


FIGURE 6. The basic procedures in a self-healing and mutual-healing key distribution scheme, where the small panes are operation which must be executed in each round, the small dashed frame represents the operations which may not be executed in some round of the scheme.

As aforementioned, the schemes [14, 18] are based on hash chain. The difference is that [14] utilizes polynomial secret-sharing masking mechanism, thus the maximum number of the revoked users is constrained by the degree of polynomial while [18] adopts general access structure thus achieves flexible property. The proposed self-healing technique enables better performance over previous approaches in terms of storage, communication and computation complexity. They provided proof of the security of the proposed scheme under an appropriate security framework. The proof shows that the scheme is computationally secure and achieves both forward and backward secrecy. Kausar *et al.* [36] proposed a simpler self-healing key distribution scheme. In their scheme, all broadcast messages are masked with XOR operation. The scheme is efficient in terms of storage and computation overhead. However, the operation of dealing with compromised node are too cockamamie to afford. If one compromised node is detected, all nodes are forced to be re-initialized.

Due to the efficiency of hash function, these schemes reduce both communication and computation overheads greatly. At the same time, forward and backward secrecy were achieved. However, there is a fatal defect in these constructions. The collusion between the newly joined users and the revoked users will recover all the session keys which they are not entitled to. Tian *et al.* in [19] dealt with the problem gracefully. They assigned each user U_i a pre-arranged life cycle (s_i, t_i). Random numbers which is related to the users' membership are used in the procedure of *Key recovery*. Compared with the scheme in [18], the scheme in [19] not only keeps forward and backward secrecy but also resists collusion between the newly joined users and the revoked users. As far as we know, this is the first time to defend collusion attack against hash chain based self-healing key distribution schemes. The schemes in [15, 20] slightly reduced storage and computation overheads of the scheme [19] with polynomial secret-sharing masking mechanism and vector space secret-sharing masking mechanism, respectively. In our recent research, we found that the schemes [15, 19, 20] can only resist collusion between the users whose life cycles have finished and newly joined users. They in fact could not resist collusion attacks of newly joined users and revoked users whose life cycles have not yet expired. How to resist collusion of newly joined users and revoked users whose life cycles have not yet expired is an unsolved problem.

Similarly, the self-healing key distribution scheme [19] is composed of five procedures.

(i) Setup: The GM produces a sequence of m random numbers $r_1, \ldots, r_m$ by using a pseudo-random number generator of large enough period. The GM randomly picks two initial key seeds, S^F and S^B, from $GF(q)$. In the pre-processing time, it computes two hash chains of equal length m by repeatedly applying the same one-way hash function H on each seed $K_0^F = S^F$ and $K_0^B = S^B$. For $1 \leq j \leq m$, the hash sequences are generated as follows:

$$\{K_0^F = S^F, H(K_0^F), \ldots, H^j(K_0^F), \ldots, H^m(K_0^F)\}$$
$$\{K_0^B = S^B, H(K_0^B), \ldots, H^j(K_0^B), \ldots, H^m(K_0^B)\}$$

The session key of the jth session is computed as:

$$SK_j = K_j^F + (K_{m-j+1}^B \oplus r_j)$$

The GM randomly chooses vectors $P_1, \ldots, P_m \in GF(q)^l$. For each session $j = 1, \ldots, m$, the GM pre-computes the scalar $z_j = K_{m-j+1}^B + \psi(D)^\top P_j \in GF(q)$. Each user U_i is first assigned a prearranged life cycle (s, t) where $1 \leq s < t \leq m$, thus U_i will be involved in $k = t - s + 1$ number sessions. The GM privately send personal key S_i to U_i. It is in the following form:

$$(K_s^F \| r_s, \ldots, r_t \| (\psi(i)^\top P_s, \ldots, \psi(i)^\top P_t))$$

It is composed of a forward key K_s^F for session s, k random numbers and secret vectors $(\psi(i)^\top P_s, \ldots, \psi(i)^\top P_t)$

(ii) Broadcast: Suppose $R_j \subseteq G_{j-1}$ with $R_1 \cup \ldots \cup R_j \in \mathcal{R}$ for session j. By definition $R_1 = \phi$. The GM chooses a maximal non-authorized subset of users $W_j \in \mathcal{R}_0 = \overline{\Gamma}_0$ such that $R_1 \cup \ldots \cup R_j \subset W_j$ and with minimum cardinality. The broadcast $B_j = B_j^1 \cup B_j^2$ in session $j = 1, \ldots, m$ is given as follows:

$$B_j^1 = z_j$$

$$B_j^2 = \begin{cases} \{(k, \psi(k)^\top P_j)\}_{k \in W_j} & j = 1, 2 \\ B_{j-1}^2 \cup \{(k, \psi(k)^\top P_j)\}_{k \in W_j} & j \geq 3 \end{cases}$$

(iii) Key recovery: When an authorized user U_i received the key distribution message B_j for session j, since $U_i \in G_j$ has $\{(k, \psi(k)^\top P_j)\}_{k \in W_j}$ and its personal key, it computes $\psi(D)P_j$ using $\{(k, \psi(k)^\top P_j)\}_{k \in W_j \cup \{i\}}$ because $W_j \cup \{i\} \in \Gamma$. Therefore, $\psi(D)P_j = \sum_{k \in W_j \cup \{i\}} \lambda_k \psi(k) P_j$. From the broadcast information B_j, U_i can recover the backward key

$$K_{m-j+1}^B = z_j - \psi(D)^\top P_j.$$

At last, U_i computes the jth forward key $K_j^F = H^{j-1}(S^F)$ and evaluates the current session key

$$SK_j = K_j^F + (K_{m-j+1}^B \oplus r_j).$$

(iv) Adding or revoking users: When the GM wants to add a new user the communication group, the GM assigns a life circle (s, t) and an unused identity $v \in GF(q)$ to the new user. The group manager computes the personal secret key $S_v = (K_s^F \| r_s, \ldots, r_t || \psi(v)^\top P_s, \ldots, \psi(v)^\top P_t))$ and send it to this new user via the secure communication channel between them.
When the GM wants to revoke a user from the communication group, the GM adds the revoked user to the non-authorized subset of users such that $R_1 \cup \ldots \cup R_j \subset W_j$ and with minimum cardinality. Then the GM constructs the broadcast message based on W_j. Only authorized users can recover session keys from the broadcast message due to the special construction of the broadcast message.
The broadcast $B_j = B_j^1 \cup B_j^2$ in session $j = 1, \ldots, m$ is given as follows.

(v) Self-healing: Suppose U_i is a user who receives broadcast messages B_{j_1} and B_{j_2} in session j_1 and j_2, respectively, where $1 \leq j_1 < j_2 \leq m$, but misses broadcast message B_j for the session j, where $j_1 < j < j_2$, U_i, can still recover all the lost session keys K_j $(j_1 < j < j_2)$ as follows:
   (a) U_i recovers the backward key K_{m-j+1}^B from the broadcast message B_{j_2} in session j_2. Then it repeatedly apply the one-way hash function on K_{m-j+1}^B to compute the backward hash sequence for the session j $(j_1 < j < j_2)$.
   (b) U_i computes the forward keys K_j^F for all $j\,(j_1 \leq j \leq j_2)$ by repeatedly applying the one-way hash function H on the forward key $K_{j_1}^F$.
   (c) U_i recovers all the session keys $SK_j = K_j^F + (K_{m-j+1}^B \oplus r_j)$, for $j_1 < j < j_2$.

The hash chain itself cannot be the only cryptographic primitive of a self-healing key distribution scheme. It forms a self-healing key distribution scheme together with other cryptographic primitives. For example, the scheme [21] is base on hash function and MAC, the scheme [36] is base on hash function and XOR operation, the scheme in [14, 15] is based on hash function and polynomial secret sharing, while the schemes in [18–20] are based on hash chain and vector space secret sharing. These schemes share some commonness and each has its special features as well. We summarize the features of one-way hash function-based self-healing key distribution schemes in Table 4 without [21]. In fact, the scheme in [21] is a totally different self-healing group key distribution mechanism. It supposes that there is a buffer at each sensor in order to recover the lost session keys. Each user has to send the *Request Key* message to explicitly request the current rekeying message for the current session. That is, this self-healing mechanism involves interaction between the user and the GM. The communication overhead varies at different user end and it depends on how many rekeying messages the user fails to receives and how large the renewal interval t is. Because each message is encrypted with a Traffic Encryption key (TEK), the computation overhead varies according to the encryption mechanism. The authors did not suggest which encryption mechanism is suitable for the scheme. Therefore, we exclude the scheme [21] from the Table 4. In Table 4, the storage overhead of a user U_i is the size of its personal key which is closely related to its life cycle (s_i, t_i). The communication overhead comes from the broadcast messages for Key recovery and Self-healing. The communication overhead at the jth session in [18–20] is $(t_j + 1) \log p$ bits, where $t_j = |W_j \cup R_j|$, $R_j \not\in \Gamma$ is the set of all revoked users for sessions in and before j and $W_j \subset \mathcal{U} \backslash G_j$ with minimum cardinality such that $W_j \cup R_j \in \overline{\Gamma}_0$. The computation overhead is measured by the number of multiplication in the underlying field. In terms of computation overhead of the scheme in [36], it only includes hash and XOR operations, so the computation overhead at user end in [36] is a constant compared with that of other schemes such as [18] and [19]. The computation overhead of Key recovery in [18–20] is $2(t_j^2 + t_j)$. This is the number of multiplication operations needed to recover $\psi(D)$ by using equation $\psi(D) = \sum_{k \in W_j \cup \{i\}} \lambda_k \psi(k)$ [18]. This is because self-healing mechanism involves only hash and addition operations. Compared with the computation overhead brought by multiplication operations, the computation overhead brought by hash and addition operations can be neglected.

5.5. Bilinear pairing-based self-healing key distribution schemes

Although a formal definition of ID-based cryptosystems have been known for a while [37], the first fully functional fitting all the requirements ID-based cryptosystem appeared only quite recently in [38]. Inspired by the idea of [38], Du *et al.* proposed a broadcast encryption scheme for key distribution in [39]. We extended the broadcast encryption scheme for key distribution to a self-healing key distribution scheme in [24] and further proposed formal definition and security model in [40]. The whole process is displayed as follows:

(i) Setup: It is supposed that the GM obtains both public system parameters $params = \{G_1, G_2, q, P, P_{pub}, H_1, H_2\}$ and its private key s and all the public key of possible users from an ID-based public key infrastructure (PKI). The GM makes $params$ public and keeps s secret. In $params$, where G_1 is a cyclic additive group and G_2 is a cyclic multiplicative group and both of them of the same large prime order q; $P \in G_1$ and $s \in \mathbb{Z}_q^*$ and $P_{pub} = sP$; H_1 and H_2 are two cryptographic hash functions: $H_1 : \{0,1\}^* \rightarrow G_1$, $H_2 : G_2 \rightarrow \{0,1\}^*$. Each node is preloaded with a public/private key pair (Q_i, S_i) where $Q_i = H_1(ID_i)$ and $S_i = sQ_i$ before the deployment of the network. Here we suppose each user U_i has a unique identifier ID_i. The GM chooses m session keys $K_1, \ldots, K_m$ from $\mathbb{Z}_q^*$. The session keys are independent to each other and according to uniform distribution.

(ii) Broadcast: Suppose $|G_j|$ denotes the number of users in session j. For each session $1 \leq j \leq m$, according to the session group G_j, the GM computes $Q_{Y_1} = \sum_{i=1}^n Q_i$ and defines a $(|G_j| - 1) \times |G_j|$ matrix. The matrix is defined as follows:

$$\begin{pmatrix} a_2 \\ a_3 \\ \vdots \\ a_{|G_j|} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & \ldots & 0 \\ 1 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 1 & 0 & 0 & \ldots & 1 \end{pmatrix}$$

Let a_i' represents the transpose of a_i. The group manager also constructers $|G_j| - 1$ auxiliary keys

$$Q_{Y_i} = (Q_1, Q_2, \ldots, Q_{|G_j|}) \times a_i' \quad 2 \leq i \leq |G_j|$$

which means $Q_{Y_2} = Q_1 + Q_2$, $Q_{Y_3} = Q_1 + Q_3, \ldots, Q_{Y_{|G_j|}} = Q_1 + Q_{|G_j|}$. The broadcast message is then formed by computing, for a random $r_j \in \mathbb{Z}_q^*$,

$$X_1 = r_j P \quad X_i = r_j Q_{Y_i} \quad 2 \leq i \leq |G_j|,$$
$$Y_j = K_j \oplus H_2(e(P_{pub}, r_j Q_{Y_1}))$$

Let $z_j = (X_i(1 \leq i \leq |G_j|), Y_j)$. The j-th broadcast is in the following form:

$$B_j = \{z_1, \ldots, z_j\}.$$

(iii) Key recovery: When a user $U_i \in G_j$ receives the broadcast message B_j, it sets a vector $a_1 = (0, \ldots, 0, 1, 0, \ldots, 0)$ with $|G_j|$ elements, and only the ith element is 1. A_j is a $|G_j| \times |G_j|$ matrix which is defined as follows:

$$A_j = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{|G_j|} \end{pmatrix}.$$

U_i can solve the following system of equations using Cramer's rule or other algebraic methods.

$$(x_1, x_2, \ldots, x_{|G_j|}) \times A_j = (1, 1, \ldots, 1).$$

With $(x_1, x_2, \ldots, x_{|G_j|})$, U_i gets

$$(x_1, x_2, \ldots, x_{|G_j|}) \times \begin{pmatrix} Q_i \\ Q_{Y_2} \\ \vdots \\ Q_{Y_{|G_j|}} \end{pmatrix} = Q_{Y_1}.$$

In order to decrypt the ciphertext, user U_i needs to compute $e(P_{pub}, r_j Q_{Y_1})$, which with knowledge the private key S_i it can do via:

$$\begin{aligned} & e(P_{pub}, r_j Q_{Y_1}) \\ & \quad = e(P_{pub}, r_j(x_1 Q_i + x_2 Q_{Y_2} + \ldots + x_{|G_j|} Q_{Y_{|G_j|}})) \\ & \quad = e(P_{pub}, r_j x_1 Q_i) \cdot e(P_{pub}, r_j(x_2 Q_{Y_2} + \ldots \\ & \quad + x_{|G_j|} Q_{Y_{|G_j|}})) \\ & \quad = e(r_j P, x_1 s Q_i) \cdot e(P_{pub}, x_2 r_j Q_{Y_2} + \ldots \\ & \quad + x_{|G_j|} r_j Q_{Y_{|G_j|}}) \\ & \quad = e(X_1, x_1 S_i) \cdot e(P_{pub}, x_2 X_2 + \ldots + x_{|G_j|} X_{|G_j|}). \end{aligned}$$

Then, U_i can recover the session key

$$K_j = Y_j \oplus H_2(e(X_1, x_1 S_i) \cdot e(P_{pub}, \sum_{i=2}^{|G_j|} x_i X_i)).$$

(iv) Self-healing: Without loss of generality, suppose U_i misses the broadcast message for a session $t < j$. As far as it belongs to the session group G_t, it picks up the polynomial z_t from broadcast message B_j and forms the $|G_t| \times |G_t|$ matrix A_t as operations in the procedure of Key recovery. Then, U_i solves the following system of equations.

$$(x_1, x_2, \ldots, x_{|G_t|}) \times A_t = (1, 1, \ldots, 1).$$

With $(x_1, x_2, \ldots, x_{|G_t|})$, U_i gets

$$(x_1, x_2, \ldots, x_{|G_t|}) \times \begin{pmatrix} Q_i \\ Q_{Y_2} \\ \vdots \\ Q_{Y_{|G_t|}} \end{pmatrix} = Q_{Y_1}.$$

After that, with knowledge its private key S_i, U_i computes $e(P_{pub}, r_t Q_{Y_1})$ as follows:

$$\begin{aligned} & e(P_{pub}, r_t Q_{Y_1}) \\ &\quad = e(P_{pub}, r_t(x_1 Q_i + x_2 Q_{Y_2} + \ldots + x_{|G_t|} Q_{Y_{|G_t|}})) \\ &\quad = e(P_{pub}, r_t x_1 Q_i) \cdot e(P_{pub}, r_t(x_2 Q_{Y_2} + \ldots \\ &\quad + x_{|G_t|} Q_{Y_{|G_t|}})) \\ &\quad = e(r_t P, x_1 s Q_i) \cdot e(P_{pub}, x_2 r_t Q_{Y_2} + \ldots \\ &\quad + x_{|G_t|} r_t Q_{Y_{|G_t|}}) \\ &\quad = e(X_1, x_1 S_i) \cdot e(P_{pub}, x_2 X_2 + \ldots + x_{|G_t|} X_{|G_t|}). \end{aligned}$$

Finally, U_i recovers the lost session key

$$K_t = Y_t \oplus H_2(e(X_1, x_1 S_i) \cdot e(P_{pub}, \sum_{i=2}^{|G_t|} x_i X_i))$$

If more than one broadcast message get lost, the operation of key recovery is the same as aforementioned.

(v) Adding or revoking user: If a new user U_{new} apply for joining the session j, the GM checks its validity firstly. If it is an authorized user, in the procedure of *broadcast*, the GM constructs a new $(|G_j| - 1) \times |G_j|$ matrix and computes new $Q_{Y_i}(1 \le i \le |G_j|)$ which should includes Y_{new}.
If a user U_{rov} is revoked in session j, what the group manager should do is constructing a new $(|G_j| - 1) \times |G_j|$ matrix and computing new $Q_{Y_i}(1 \le i \le |G_j|)$ which should excludes Q_{rov}.
The adding and revoking operations are very efficient in our scheme. For the condition that more than one user join or revoke, the operations preform as aforementioned.

This scheme achieves some good properties. In terms of storage overhead, each user stores only the GM's public key and its public/private key pair. The GM's public key $P_{\text{pub}} = sP$ which is a point on G_1 and the order of G_1 is q. The length of the GM's public key is $2\log q$. Because H_1 is a mapping from $\{0,1\}^*$ to G_1 , so the length of the personal public key $Q_i = H_1(ID_i)$ is $2\log q$. The private key is the multiplication of the master key $s \in \mathbb{Z}_q^*$ and the public key, so the size of private keys is $2\log q$. Therefore, the storage overhead for each group user is $6\log q$. Therefore, the storage overhead for each user is a constant.

The communication overhead comes from the broadcast message $B_j = \{z_1, \ldots, z_j\}$. z_j is composed of $X_i(1 \le i \le |G_j|)$ and Y_j. The size of $X_i(1 \le i \le |G_j|)$ is $2\log q$ and the size of Y_j is $\log q$ bits. Therefore, the length of z_j equals to $(2|G_j| + 1)\log q$, which increases in direct proportion to $|G_j|$. Consequently, the size of broadcast message B_j is $\sum_{l=1}^{j}(2|G_l| + 1)\log q$ bits, which is related to the total number of users $\sum_{i=1}^{j}|G_i|$ in all the communication groups for different sessions and increases in direct proportion to session number j. Therefore, this scheme is only suitable for small-scale networks.

All the computation in the procedure of *Key recovery* is as follows: (1) Solving a set of linear equations with $|G_j|$ variables; (2) $|G_j| + 1$ scalar multiplications in the group G_1; (3) $|G_j|$ additions in the group G_1; (4) Two pairings computation; (5) One hash computation; (6) One XOR operation. Generally speaking, bilinear pairing computation is more time-consuming than scalar multiplication, let alone addition, hash and XOR operation. Therefore, the main computation overhead comes form (4). The computation overhead of the users at Key recovery stage is two pairing operations.

This scheme is collusion-free for any coalition of unauthorized users, including the revoked users and the newly joined users. In our scheme, if a user wants to obtain the session key K_i, it should compute $e(X_1, x_1 S_i)$ in the procedure of Key recovery with personal key S_i. Therefore, only the authorized users can recover the session key. In addition, due to the difficulty of solving discrete logarithm problem (DLP), any coalition of non-authorized users cannot derive the private keys of authorized users from their public keys. The personal private key has nothing to do with the number of revoked users and can be reused as long as it is not disclosed. Most important of all, this paper presented technique to perform mutual-healing between neighboring nodes. As far as we know, it is the first self-healing key distribution scheme using bilinear pairings. it is also the first time to explore technical details of mutual-healing in self-healing key distribution schemes.

6. THREE CONSIDERATIONS RELATED TO SELF-HEALING KEY DISTRIBUTION SCHEMES

In this section, we discuss three considerations which can be used to strengthen the robustness of the basic self-healing key distribution schemes. The first one is sponsorization, the second one is mutual healing and the last one is authentication on broadcast messages.

6.1. Sponsorization

The motivation for sponsorization is to give dynamism to the general scheme, allowing an authorized subset of users in the group to invite a new user for one session without the interference of the GM. This feature has been considered in other distributed protocols such as group key distribution schemes in [41] and [42]. Sáez [17] considered the process that a coalition of users sponsors a user outside the group for one session. For the sake of security, it is required that only a coalition of authorized users in an access structure can perform this action. A secure unicast channel between each sponsor and the sponsored user is necessary in order to fulfill the sponsorization. It is a rigorous requirement which is infeasible for wireless sensor networks. In addition, the operations introduces too much communication and computation overheads. The motivation is good while the realization is troublesome.

6.2. Mutual healing

More et al. [11] pointed out that the protocol discussed in [10] suffers from inconsistent robustness. That is, for certain sessions, if broadcast messages get lost, the user cannot recover the session keys, no matter how many other broadcast messages are received. Subsequently, they used a sliding window mechanism to make error recovery consistently robust: after the initial *Setup* procedure, any lost key can be recovered as long as two sufficiently close broadcast messages—one before it and one after it—are received. Similar technique is discussed in [33]. The minimum size of the window can be dynamically adjusted according to the condition of networks. Both [33] and [11] guarantee that authorized users can recover window size number of session keys as long as they receive broadcast messages. However, how to recover the session key if the last broadcast message gets lost or more than sliding window number of broadcast messages get lost are never addressed clearly. Therefore, it is virtually impossible to make users completely self-healing according to the existing self-healing key distribution mechanism. In view of some concrete applications, such as live and pay-per-view TV, have strict requirement of freshness. They would better lose only a limited number of broadcast messages. Therefore, it is meaningful to explore counterpart measures to deal with the aforementioned issues.

The idea of mutual-healing was proposed in [23] by Bohio *et al.* The so-called mutual healing is, if a user has missed more than a fixed number of broadcast messages or the last broadcast message, it can apply for assistance from its neighboring nodes. The neighboring users in the same session group cooperate with each other forwarding broadcast messages which the neighboring users miss. By this way, the authorized users can get the missed broadcast messages in a timely and efficient manner. Thus the robustness of self-healing key distribution schemes is strengthened. It was claimed in [23] that there are two requirements for mutual healing: the authentication of requesting users and the users' authorization for the requested session keys. On the one hand, in order to avoid attacks on their limited resource, effective authentication of requesting user must be developed to identify misbehaving users. On the other hand, as messages are broadcasted in the form of plain text, anyone can read them. We argue that the second authentication is unnecessary. Instead, the neighboring users need to forward only the broadcasted message which corresponds to the requested session key. If the requesting user is authorized for the session, it would be able to recover the session key. However, even unauthorized users can receive broadcast messages, they cannot recover the session key.

The paper [23] proposed only a general idea without exploring technical details. We targeted the problem with bilinear pairings technique in [24]. The definition and security model for mutual healing key distribution was formalized in [40]. By binding the identity and location of nodes, the authors explored the ways to realize mutual healing. A pair-wise key is the byproduct of authentication. The pair-wise key can be used for private communication between two nodes. The authentication process is helpful to identify cooperating or misbehaving nodes. However, the prerequisite of [40] is the awareness of the identity and location of each node. This prerequisite implies that the scheme [40] is suitable for static wireless sensor networks. Although the mutual-healing technique is more useful in mobile network, tt is not trivial to realize mutual healing in mobile environment.

6.3. Authentication on broadcast messages

In self-healing key distribution schemes, session keys are masked by some private elements. The broadcast messages for key distribution are transmitted in plain text form. The existing self-healing key distribution schemes focus on the exact construction and improvement of performance. It is supposed that the broadcast channel can keep the integrity of broadcast message. However, in real network environment, the adversaries can launch various attacks, such as substitute a false message for a legitimate one and modify broadcast message, on broadcast message easily. This assumption is not compatible with real application. In order to keep the correctness and availability of a self-healing key distribution scheme, it should be possible for the users in the communication group to verify that the broadcast messages have not been substituted or modified in transit. Attempts to add an integrity checking property to existing protocols often focus on cryptographic authentication mechanisms. Aside from the limited resource that make digital signature schemes impractical, authentication in sensor networks poses serious complications. It is unclear how to establish trust especially in large-scale ad hoc deployments. Adding security often fails even in systems without these additional constraints. Therefore, efficient and effective authentication on the identity of the GM and the integrity of broadcasts are of great importance. Otherwise, the correctness of the recovered session keys cannot be guaranteed. Designing the authentication mechanism for wireless networks is an important research topic at present.

Intuitionally, a symmetric key, TEK, can be used to encrypt broadcast messages by the GM and decrypt broadcast messages by the group users. In dynamic group communication, TEK must be refreshed on each change of membership. Jiang *et al.* [21] used TEK to prevent non-legitimate nodes from having access to the secret broadcast contents. The TEK is renewed periodically instead of an every node topology change. Han *et al.* [34] considerate encrypting broadcast messages by TEK, thus the integrity of broadcast messages can be kept.

Two constructions of TEK were proposed in [34]. The first construction brought too much computation load for the GM. The second construction is only feasible to the self-healing key distribution scheme proposed in [34]. It is meaningful to explore practical ways to construct and renew TEK.

7. COMPOSITIVE ANALYSIS AND FUTURE WORK

According to the state-of-the-art research, self-healing key distribution is the most suitable way to establish session key for large and dynamic group communication over unreliable wireless networks. It is one of the branch of key management and has received much attention. In this section, first, we make a compositive analysis on existing schemes and then we outline the problems to be solved.

7.1. Compositive analysis on exiting schemes

Although unconditionally secure self-healing key distribution schemes can keep strict security requirement, they should meet some lower bounds in terms of storage and communication overheads. Computationally secure self-healing key distribution schemes relax the security slightly and achieve some nice features, such as constant storage overhead. As far as the cryptographic primitives are concerned, self-healing key distribution schemes are limited to polynomial secret sharing, vector space secret sharing, hash chain and SDR mechanism. The authors of this paper tried to design bilinear pairing-based self-healing key distribution scheme and have made a little progress.

Figure 7 provides taxonomy of papers on self-healing key distribution schemes. The figure is a directed acyclic graphs where nodes represent papers. Directed edges show predecessor/successor relations among the papers. There is an edge from a paper to another one if the latter provides improvement for the solution proposed by the former. Papers are ordered over a horizontal time axis according to their publication dates. Vertical axis groups papers under five categories: (1) polynomial secret sharing, (2) SDR, (3) vector space secret sharing, (4) hash chain and (5) bilinear pairing-based self-healing key distribution schemes. The style of edge in between two nodes represents the problem in which an improvement is provided. A paper may based on more than one cryptographic primitive; therefore, corresponding may be reachable from more than one origin paper, and there may be more than one edge with different styles in between two papers.

Every category has its features, requirements and goals. In summary, polynomial secret sharing is the most common technique used to realize self-healing key distribution. It performs easily. However, the maximum number of revoked

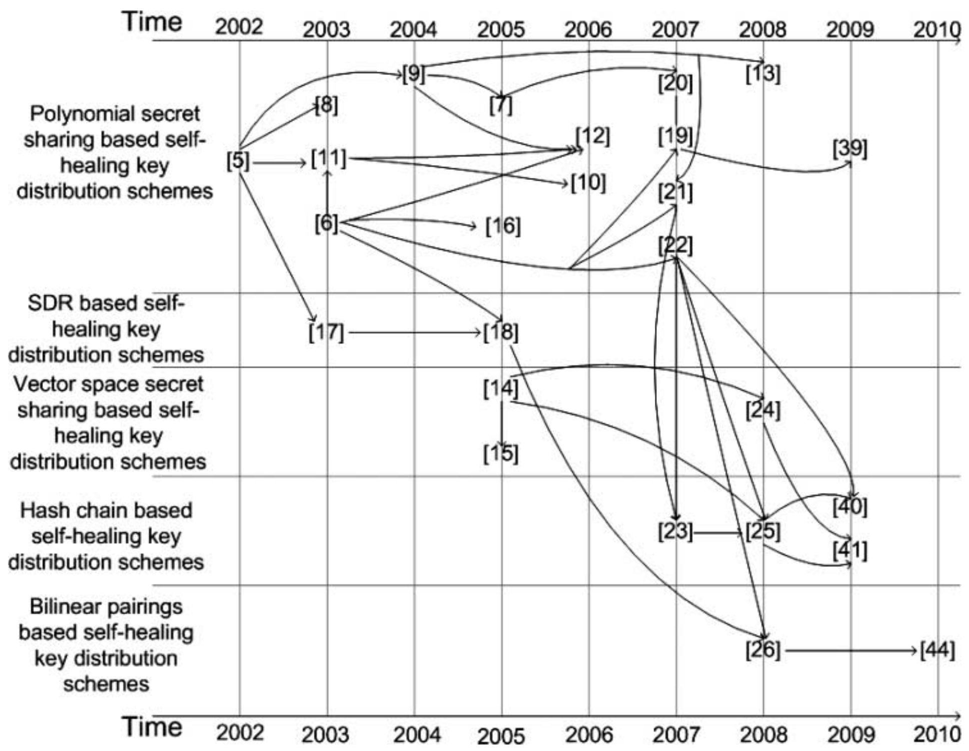

Time
2002
2003
2004
2005
2006
2007
2008
2009
2010
Polynomial secret sharing based self-healing key distribution schemes
SDR based self-healing key distribution schemes
Vector space secret sharing based self-healing key distribution schemes
Hash chain based self-healing key distribution schemes
Bilinear pairings based self-healing key distribution schemes
[5]
[8]
[9]
[11]
[6]
[7]
[12]
[10]
[16]
[20]
[13]
[19]
[21]
[22]
[39]
[17]
[18]
[14]
[15]
[24]
[40]
[23]
[25]
[41]
[26]
[44]

FIGURE 7. Taxonomy of the papers on self-healing key distribution schemes. The graph is a directed acyclic graph where nodes represent papers, and edges represent predecessor/successor relations among solutions provided by the papers. Style of an edge represents the problem on which destination paper provides improvements.

users is constraint to the degree of the polynomial. In addition, in the procedure of *Key recovery*, Lagrange's interpolation formula should be used in order to recover the secret polynomial, thus leading to much computation overhead.

Vector space secret-sharing-based self-healing key distribution schemes consider a monotone-decreasing family of rejected subset of users instead of a monotone decreasing threshold structure. This general case makes the self-healing scheme more flexible and suitable for practical application. In addition, the constructions are general in the sense that they depend on the particular public mappings and for different choices of public mappings different self-healing key distribution schemes can be attained. The length of broadcast messages also depends on the particular function.

Hash chain has many elegant features and can be used to design self-healing key distribution schemes. The constructions do not need to send the recorders of revoked subsets of users in order to perform self-healing, yielding reduction in the communication cost. Both forward and backward secrecy and partial collusion resistance property can be assured. However, the collusion of the revoked nodes whose life cycles do not expire and the newly joined nodes can recover the session keys that they are not authorized to.

One of the remarkable properties of SDR-based self-healing key distributions is that the personal keys of users are independent of the number of sessions. Because of this property, the SDR-based schemes do not require a redistribution of any personal keys after the number of sessions exceeds the estimation. Another remarkable property of SDR schemes is that they can revoke any number of users and remain secure against their collusion. As in the original SDR algorithm, the number of subsets increases when users join and leave. The communication complexity depends upon how many additional subsets Na introduces. The value of Na depends on the group size, the number of the newly joined users and the revoked users in each rekeying period and the value of session number m.

Tian *et al.* presented a self-healing key distribution scheme by using bilinear pairings. The scheme achieved several nice features as aforementioned. However, the communication overhead increases with the number of authorized users in the communication group. How to reduce the communication overhead is an interesting problem to be solved.

Table 5 summarizes the advantage and disadvantage of each class of self-healing key distribution schemes.

TABLE 5. Summary of different class of self-healing key distribution schemes.

Class	Features
Polynomial secret sharing	Advantage: It performs easily. Disadvantage: High computation over head; The maximum number of revoked users is constraint to the degree of the polynomial.
Vector space secret sharing	Advantage: A monotone decreasing family of rejected subset of users; Disadvantage: Some public mapping may introduce large communication overhead.
Hash chain	Advantage: Efficient in computation; Disadvantage: can not resist collusion attack.
SDR	Advantage: Do not require re-distribution of any personal keys; can revoke any number of users; Disadvantage: The communication depends on the number of subsets, which increases with the number of joined and revoked users.
Bilinear pairings	Advantage: Efficient in computation; Problem to be solved: How to reduce the communication cost?

7.2. Future work

Further research effort should be made to meet the lower bounds on communication overhead in unconditionally secure self-healing key distribution scheme without increasing too much storage and computation overheads. Finding efficient constructions which exhibit a good tradeoff between user memory storage and broadcast size is also an interesting open problem.

The hash condition featured with limited onboard computing resource and unsupervised environment for wireless nodes makes them volunerable to attacks. The limited ability of individual nodes to thwart failure or attack makes ensuring network availability more difficult [43]. Sponsorization and mutual healing can be seen as countermeasures against fault and intrusion. The communication and computation overheads of sponsorization are too high to afford for nodes in wireless sensor networks. Mutual healing, that is, if a user has missed more than a fixed number of broadcast messages or the last broadcast message, it can get assistance from its neighboring nodes. It seems that mutual healing is a practical way. However, many exisiting schemes are not practical technique to realize the idea since it was proposed in [23] by Bohio *et al.* By combining the identity and location of nodes, we explored the means to realize it. It should be noted that locating node is a resource-consuming work. It is interesting to explore practical way of mutual healing.

It is supposed that there is only one GM in general self-healing key distribution schemes. These schemes can easily be extended to a multi-GM self-healing key distribution schemes. Each GM can dynamically broadcast messages into an arbitrary group of receiver determined by it. This extension raises the issue of data source or broadcaster authentication. We may acquire some ideas from authenticated broadcast encryption scheme such as [44].

In the pioneering work [10], the possibility of long-lived personal key scheme was discussed. The reusage of personal key was also discussed in [11] and [13]. However, it seems that none of them is feasible. In order to keep security, new personal keys are periodically distributed to groups users through either by unicast or by off-line means. Both are inefficient for a large communication group. It would be interesting to consider other personal key distribution methods. Specifically, we would like to be able to periodically distribute new personal keys via broadcast, so that after a user is added to the group, no further secure unicast channel between it and the GM is necessary.

Finally, new techniques to implement self-healing key distribution, apart from those we have just described, represent another interesting target for researchers, due to the suitability of this approach to key distribution. In addition, when the resource of nodes in wireless networks is not limited, the asymmetric keys-based self-healing key distribution will be a promising research direction.

8. CONCLUSION

With the wide application of wireless networks, as one of the basic security service, self-healing key distribution scheme will receive more attention. In this paper, we reviewed most existing self-healing key distribution schemes. We clarified the security requirements of self-healing key distribution schemes according to their special application environment. Then we classified the schemes according to different cryptographic primitives and give an insight to their features and goals. We made a thorough comparison on performance and delineated their similarities and differences through tables and figures. We have also discussed sponsorization, mutual healing and authentication techniques which can be used to strengthen the robustness of self-healing key distribution schemes. We pointed out several open research problems in self-healing key distribution schemes and suggested the future research topics in this field. Our analysis made it clear that there is no unique solution that satisfies all requirements. The optimal combination of bandwidth efficiency and robustness against link loss under a given power consumption should be sought in future. Also, secure and efficient key revocation remains an challenge for bilinear pairing based self-healing key distribution schemes.

FUNDING

This project is partially supported by the Australian Research Council Grants: LP110100602, LP100200538; LP100100404 and DP0985838.

REFERENCES

[1] Setia, S., Koussih, S., Jajodia, S. and Harder, E. (2000) Kronos: A Scalable Group Re-keying Approach for Secure Multicast. *Proc. IEEE Symp. Security Privacy*, Berkeley, CA, USA, 14–17 May, 2000, pp. 215–228.

[2] Rafaeli, S. and Hutchison, D. (2003) A survey of key management for secure group communication. *ACM Comput. Surv.*, **35**, 309–329.

[3] Eltoweissy, M., Wadaa, A., Olariu, S. and Wilson, L. (2005) Group key management scheme for large-scale wireless sensor network. *Data Commun. Topol. Control Ad Hoc Netw.*, **3**, 668–688.

[4] Younis, M., Ghumman, K. and Eltoweissy, M. (2006) Location-aware combinatorial key management scheme for clustered sensor networks. *IEEE Trans. Parallel Distrib. Syst.*, **17**, 865–882.

[5] Staddon, J., Miner, S., Franklin, M., Balfanz, D., Malkin, M. and Dean, D. (2002) Self-healing Key Distribution with Revocation. *Proc. IEEE Symp. Sec. Privacy*, Oakland, CA, USA, 12–15 May, 2002, pp. 241–257.

[6] Liu, D., Ning, P. and Sun, K. (2003) Efficient Self-healing Key Distribution with Revocation Capability. *Proc. 10th ACM Conf. Comput. Commun. Sec.* New York, NY, USA, pp. 231–240, ACM.

[7] Hong, D. and Kang, J. S. (2005) An efficient key distribution scheme with self-healing property. *IEEE Commun. Lett.*, **9**, 759–761.

[8] Blundo, C., D'Arco, P. and Listo, M. (2003) A New Self-healing Key Distribution Scheme. *Proc. Eighth IEEE Int. Symp. Comput. Commun.*, vol. 2, pp. 803–808.

[9] Blundo, C., D'Arco, P., Santis, A. D and Listo, M. (2004) Design of self-healing key distribution schemes. *Design Codes Cryptogr.*, **32**, 15–44.

[10] Zou, X. and Dai Y. S. (2006) A Robust and Stateless Self-healing Group Key Management Scheme. *Int. Conf. Commun. Technol. (ICCT*'06), November 2006, pp. 1–4.

[11] More, S. M., Malkin, M., Staddon, J. and Balfanz, D. (2003) Sliding Window Self-healing Key Distribution with Revocation. *Proc. 2003 ACM Workshop on Survivable and Self-regenerative Systems: in Association with 10th ACM Conference on Computer and Communications Security*, Fairfax, VA, USA, 31 October, 2003, pp. 82–90. ACM, New York, USA.

[12] Blundo, C., D'Arco, P. and Santis, A. D. (2006) On self-healing key distribution schemes. *IEEE Trans. Inform. Theory*, **52**, 5455–5467.

[13] Tian, B. and He, M. (2008) Self-healing key distribution scheme with novel properties. *Int. J. Netw. Sec.*, **7**, 147–152.

[14] Sáez, G. (2005) On Threshold Self-healing Key Distribution Schemes. *Cryptography and Coding*, vol. 3796 of Lecture Notes in Computer Science, pp. 340–354, Springer, Berlin, Heidelberg.

[15] Sáez, G. (2005) Self-healing Key Distribution Schemes with Sponsorization. *Int. Federation Inform. Processing IFIP'05*, vol. 3677 of Lecture Notes in Computer Science, pp. 22–31, Springer, Berlin, Heidelberg.

[16] Sun, H., Lin, D. and Xue, R. (2005) An Improved Efficient Self-healing Group Key Distribution. *Int. Symp. Commun. Inform. Technol. (ISCIT'05)*, vol. 1, Beijing, China, October 2005. pp. 192–196.

[17] Zhu, S., Setia, S. and Jajodia, S. (2003) Adding Reliable and Self-healing Key Distribution to the Subset Difference Group Rekeying Method for Secure Multicast. *In Group Communications and Charges: Technology and Business Models. Proceedings of the 5th COST 264 International Workshop on Networked Group Communications, NGC 2003*, pp. 107–118, Springer, Berlin, Heidelberg.

[18] Bohio, M. J. and Ali, M. (2005) Self-healing group key distribution. *Int. J. Netw. Sec.*, **1**, 110–117.

[19] Dutta, R. and Mukhopadhyay, S. (2007) Improved self-healing key distribution with revocation in wireless sensor network, *IEEE Wireless Communications and Networking Conference (WCNC'07)*, Kowloon, Hong Kong, 11–15 March, 2007, pp. 2963–2968.

[20] Dutta, R. and Mukhopadhyay, S. (2007) Designing Scalable Self-healing Key Distribution Schemes with Revocation Capability, *Parallel and Distributed Processing and Applications*, vol. 4742 of Lecture Notes in Computer Science, pp. 419–430. Springer, Berlin, Heidelberg.

[21] Jiang, Y., Lin, C., Shi, M. and Shen, X. (2007) Self-healing group key distribution with time-limited node revocation for wireless sensor networks. *Ad Hoc Netw.*, **5**, 14–23. Security Issues in Sensor and Ad Hoc Networks.

[22] Dutta, R., Chang, E. C. and Mukhopadhyay, S. (2007) Efficient Self-healing Key Distribution with Revocation for Wireless Sensor Networks Using One Way Key Chains. *Proc. 5th Int. Conf. Appl. Cryptogr. Netw. Sec.*, vol. 4521 of ACNS'07, Zhuhai, China, pp. 385–400, Springer, 2007.

[23] Kausar, F., Hussain, S., Park, J. H. and Masood, A. (2007) Secure Group Communication with Self-healing and Rekeying in Wireless Sensor Networks. In *Mobile Ad-Hoc and Sensor Networks*, vol. 4864, pp. 737–748. Springer, Berlin, Heidelberg.

[24] Dutta, R., Mukhopadhyay, S., Das, A. and Emmanuel, S. (2008) Generalized Self-healing Key Distribution Using Vector Space Access Structure. *NETWORKING 2008 Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet*, vol. 4982, pp. 612–623, Springer, Berlin, Heidelberg.

[25] Tian, B., Han, S., Dillon, T. S. and Das, S. (2008) A Self-healing Key Distribution Scheme Based on Vector Space Secret Sharing and One Way Hash Chains. *Proc. 2008 Int. Symp. World of Wireless, Mobile and Multimedia Networks*, Newport Beach, CA, USA, 23–26 June, 2008, pp. 1–6, IEEE Computer Society, 2008.

[26] Tian, B., Han, S. and Dillon, T. S. A Self-healing and Mutual-healing Key Distribution Scheme Based on Bilinear Pairings. *Proc. 2008 IEEE/IFIP Int. Conf. Embedded and Ubiquitous Computing*, vol. 2, Shanghai, China, 17–20 December, 2008, pp. 208–215. IEEE Computer Society.

[27] Naor, D., Naor, M. and Lotspiec, J. (2001) Revocation and Tracing Schemes for Stateless Receivers. *Proc. 21st Annu. Int. Cryptol. Conf. Adv. Cryptol., CRYPTO'01*, pp. 41–62. Springer, Berlin, Heidelberg.

[28] Shamir, A. (1984) Identity-based Cryptosystems and Signature Scheme. *Proc. CRYPTO 84 Adv. Cryptol.*, pp. 47–53. Springer, Berlin, Heidelberg.

[29] Boneh, D. and Franklin, M. (2001) Identity Based Encryption from the Weil Pairing. *Proc. 21st Ann. Int. Cryptol. Conf. Adv. Cryptol., CRYPTO'01*, pp. 213–229. Springer, Berlin, Heidelberg.

[30] Du, X., Wang, Y., Ge, J. and Wang, Y. (2005) An ID-based broadcast encryption scheme for key distribution. *IEEE Trans. Broadcasting*, **51**, 264–266.

[31] Cover, T. M. and Thomas, J. A. (1991) *Elements of Information Theory* (2nd edn). Wiley-Interscience, New York, USA.

[32] Kim, Y., Perrig, A. and Tsudik, G. (2000) Simple and Fault-tolerant Key Agreement for Dynamic Collaborative Groups. *Proc. 7th ACM Conf. Comput. Commun. Sec.*, Athens, Greece, pp. 235–244. ACM.

[33] Wood, A. D. and Stankovic, J. A. (2002) Denial of service in sensor networks. *Computer*, **35**, 54–62.

[34] Kurnio, H., Safavi-Naini, R. and Wang, H. (2003) A Group Key Distribution Scheme with Decentralised User Join, *Security in Communication Networks*, Lecture Notes in Computer Science, pp. 146–163. Springer, Berlin, Heidelberg.

[35] Daza, V., Herranz, J. and Sáez, G. (2003) Constructing general dynamic group key distribution schemes with decentralized user join. *Information Security and Privacy*, vol. 2727, pp. 464–475. Springer, Berlin, Heidelberg.

[36] Mu, Y., Susilo, W., Lin, Y. and Ruan, C. (2004) Identity-based Authenticated Broadcast Encryption and Distributed Authenticated Encryption. *Advances in Computer Science-ASIAN* '04, vol. 3321, pp. 169–181. Springer, Berlin, Heidelberg.

[37] Kogan, N., Shavitt, Y. and Wool, A. (2006) A practical revocation scheme for broadcast encryption using smartcards. *ACM Trans. Inform. Syst. Sec.*, **9**, 325–351.

[38] Shannon, C. E. and Weaver, W. (1948) A mathematical theory of communication. *Bell Syst. Tech. J.*, **5**, 379–423, 623–656.

[39] Han, S., Tian, B., He, M. and Chang, E. (2009) Efficient threshold self-healing key distribution with sponsorization for infrastructureless wireless networks. *IEEE Trans. Wirel. Commun.*, **8**, 1876–1887.

[40] Dutta, R., Mukhopadhyay, S. and Dowling, T. (2009) Trade-off between Collusion Resistance and User Life Cycle in Self-healing Key Distributions with t-Revocation. *Second Int. Conf. Appl. Digital Inform. Web Technol.*, August 2009, pp. 603–608.

[41] Dutta, R., Mukhopadhyay, S. and Dowling, T. Generalized Self-healing Key Distribution in Wireless Ad Hoc Networks with Trade-offs in User's Pre-arranged Life Cycle and Collusion Resistance. *Proc. 5th ACM Symp. QoS Sec. Wireless Mobile Networks*, Tenerife, the Canary Islands, Spain, 26–30 October, 2009, pp. 80–87. ACM, New York, USA.

[42] Zhang, J. and Varadharajan, V. (2010) Wireless sensor network key management survey and taxonomy. *J. Netw. Comput. Appl.*, **33**, 63–75.

[43] Tseng, Y. (2007) A secure authenticated group key agreement protocol for resource-limited mobile devices. *Comput. J.*, **50**, 41–52.

[44] Tian, B., Han, S., Hu, J. and Dillon, T. S. (2011) A mutual-healing key distribution scheme in wireless sensor networks. *J. Netw. Comput. Appl.*, **34**, 80–88.

A Key Management Protocol for Multiphase Hierarchical Wireless Sensor Networks

Biming Tian, Song Han, Sazia Parvin, Tharam S. Dillon
DEBI Institute, Curtin University
Enterprise Unit 4, De Laeter Way, Technology Park
Bentley, Perth, Australia 6102
Email: {biming.tian, sazia.parvin }@postgrad.curtin.edu.au
{song.han, tharam.dillon}@cbs.curtin.edu.au

***Abstract*—The security of Wireless Sensor Networks (WSNs) has a direct reliance on secure and efficient key management. This leaves key management as a fundamental research topic in the field of WSNs security. Among the proposed key management schemes for WSNs security, LEAP (Localized Encryption and Authentication Protocol) has been regarded as an efficient protocol over the last years. LEAP supports the establishment of four types of keys. The security of these keys is under the assumption that the initial deployment phase is secure and the initial key is erased from sensor nodes after the initialization phase. However, the initial key is used again for node addition after the initialization phase whereas the new node can be compromised before erasing the key. A time-based key management scheme rethought the security of LEAP. We show the deficiency of the time-based key management scheme and proposed a key management scheme for multi-phase WSNs in this paper. The proposed scheme disperses the damage resulting from the disclosure of the initial key. We show it has better resilience and higher key connectivity probability through the analysis.**

***Keywords*-Key management, key predistribution, Wireless Sensor Networks(WSNs);**

I. Introduction

Wireless Sensor Networks (WSNs) have gained wide applications ranging from civilian to military use. A typical WSN is composed of a great number of sensor nodes. These sensor nodes have limited battery power, weak data processing capability and short radio range. Most importantly, sensor nodes are often randomly spread out over specific regions and work in unattended environment. They are prone to all kinds of attacks thus security becomes the first concern. In order to keep communication secure, sensitive data should be encrypted and authenticated. Therefore, key management, which is a prerequisite of encryption and authentication, should be addressed carefully.

Among all the key management mechanisms for WSNs, key pre-distribution mechanism provides a nice tradeoff between storage overhead and processing power, and is considered as the most suitable mechanism for WSNs. However, most of key predistribution schemes consider a homogeneous topology and only support the establishment of pairwise keys. Even though homogeneous networks are simple and efficient for small network scale, such networks lack scalability due to ”one-affect-n” effect in node addition and revocation. In a hierarchical WSN, the effect of node addition and revocation can be localized into a cluster thus scalability is achieved.

In a hierarchical WSN (HWSN), various types of communication may happen. The base station broadcasts control commands to the whole network. Control node multicasts messages within the cluster. A node communicates with its neighboring nodes by unicasting. Therefore, network-wide key, cluster key, and pairwise key are required to satisfy different types of secure communication. Zhu et al. [1] devised a scheme called localized encryption and authentication protocol (LEAP) for hierarchical WSNs. LEAP supports establishment of individual keys, pairwise keys, cluster keys, and a global key. Different keys are used to handle the different types of packets. The security of all types of keys relys on that of the initial key. As many existing key management protocols, LEAP assumes that the initial key is secure during the initialization phase and is erased from the memory of sensor nodes when the initialization phase finishes. The authors regarded the scheme is secure under such an assumption. However, the same key should be used again for node addition and replacement. According to the assumption, some new nodes may be captured at any time after the initialization phase. That is, the new deployed nodes could be captured before removing the initialization key. The security of the scheme is threatened by the attacks launched after the initialization phase.

Jang et al. [2] improved LEAP by introducing a time-based key management protocol. The scheme strengthens the security with a new notion of probabilistic time intervals. However, the scheme does not guarantee the perfect key connectivity. In addition, the pairwise key does not exclusively belong to the two end nodes. Those nodes which have the same initial key or master key can calculate the other nodes' pairwise keys as the ID of each node is public. To address these security and performance issues, we present an elegant key management scheme in this paper.

This paper is organized as follows: We describe the LEAP protocol [1] and the time-based key management scheme in [2] and then talk about the security problems of them in Section II. We propose a key management protocol for multiphase hierarchical WSNs in Section III. We analyze its performance and security in Section IV and conclude this paper in Section V.

II. Related Works

A. Key Predistribution Schemes

Eschenauer et al. proposed the pioneering work [3]. It is a random key predistribution scheme. Initially a large key pool of P symmetric keys and their identities are generated. Each sensor randomly draws $k(k \ll P)$ keys from the key pool without replacement. These k keys and their identities form the key-chain for a sensor node. In the shared-key discovery phase, two neighbor nodes exchange and compare the list of identities of keys in their key-chains. If two sensors have at least one key in common, they can setup a secure link directly. Otherwise, the path-key establishment procedure is triggered to setup a link between two neighbors. The nodes still can establish a secure channel under the help of one or more intermediate nodes. The advantage of the random key predistribution scheme is that there is no computational overhead to generate pairwise keys between sensor nodes. The main shortcoming of the scheme in [3] is that a large number of keys could be disclosed by compromising a few nodes. The basic scheme in [3] was further improved by Chan et al. in [4] from two different aspects. Two variations are proposed: the $q(q \geq 2)$-composite scheme and the multipath key reinforcement scheme. The variations make it more difficult to compromise a node.

The schemes [3] and [4] share a common shortcoming. That is, they only explore the way to establish the pairwise keys between nodes but not other types of keys. Obviously, it is not enough for different communication manners in HWSNs. In addition, both of [3] and [4] allow dynamic addition, however, their key pools do not evolve with time. As a result, if the network encounters a long-term attack, the newly deployed nodes may be preloaded with some already compromised keys. It is possible to discover all the keys in the key pool if an attacker continues his/her attack. One naive countermeasure is to refresh key pool when some nodes are added to the network. The attacker cannot deduce the key ring of the newly deployed nodes with the knowledge of the key materials of the compromised nodes. However, this renewal introduces unexpected consequences. That is, sensor nodes deployed at different time slots cannot establish pairwise keys because they do not have the common initial key. How to achieve connectivity between nodes deployed at different time slots in a dynamically renewed key pool is an open problem to be answered.

To the best of our knowledge, RoK [8] is the first key predistribution scheme adapted to multiphase WSNs. In this scheme, sensor nodes which run out of power will be removed from the network and new sensor nodes need to be periodically deployed to assure network connectivity. Correspondingly, the predistributed keys have limited lifetimes and the key pool should be refreshed periodically. This scheme overcomes the drawback of the general key predistribution schemes. [3] [4]. The security of the network does not degrade with time. Zo-RoK [9] takes advantage of prior deployment knowledge in order to reduce the size of key ring. In this way, the resiliency of the network against node capture attacks increases with a smaller key ring of each node. However, deployment knowledge is not always available in WSNs. Lately, a random generation material (RGM) key predistribution scheme was proposed in [5]. In this scheme, the lifetime of the whole network is divided into generations. Each generation has its own random keying material and pairwise keys are established by two nodes is only known by the nodes in two generations which the two nodes belong to. Nodes deployed in other generations other than the two generations which the nodes belong to have no access to the pairwise key. We lend this idea to improve the performance of the LEAP scheme in this paper.

B. LEAP

Zhu et al. [1] devised a key management scheme which is abbreviated as LEAP for hierarchical WSNs. LEAP offers establishment methods of individual keys, pairwise keys, cluster keys, and a global key. Different keys are used to handle the different types of packets. The establishment of cluster keys and the global key mainly depends on the the established pairwise keys, so we omit the establishment of them here and focus on the description of the establishment of pairwise keys.

- Individual key: This is an unique key that is shared between the base station and each sensor node [1]. The key is preloaded into each node's memory before being deployed. The individual key is calculated as $K_u^m = f_{K^m}(u)$ where f is a pseudo-random function and K^m is the system key known only to the base station and u represents the ID of the node u.
- Pairwise key: Each node shares a pairwise key with each of its immediate neighbors. Similar to the scheme in [3], there are four stages of pairwise key establishment: key predistribution, neighbor discovery, pairwise key establishment, and key erasure. During the initial stage of key predistribution, node u is loaded with a initial key IK by the controller and drives the master key $K_u = f_{IK}(u)$. For neighbor discovery, node u first initialize a timer to activate during a time slot of T_{min}, then it broadcasts a HELLO message containing its ID to discover its neighbors. The neighboring node v responds to node u with an acknowledgement (ACK) message containing its ID if it receives node u's HELLO message. The ACK message of v is authenticated using its master key K_v which is derived from IK. Node u verifies the Message Authentication Code (MAC) of v by generating the master key K_v with IK. The neighbor discovery stage can be denoted as:

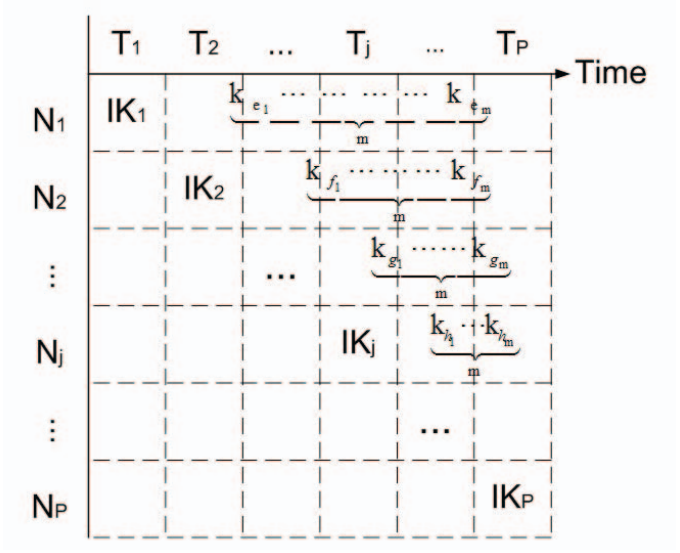


Figure 1. Key materials preloaded to nodes at different time slots in scheme [2].

$$u \rightarrow * : u;$$

$$v \rightarrow u : v, MAC(K_v, u \mid v).$$

In the stage of pairwise key establishment, node u calculates the pairwise key K_{uv} shared with node v, as $K_{uv} = f_{K_v}(u)$. Node v can also derive K_{uv} in the same way. K_{uv} serves as their pairwise key. In the final stage, when its timer expires after T_{min}, node u erases IK and all the masters keys of its neighbors, which it computed in the neighbor discovery stage. Even though an adversary captures a node, the communication between the captured node and another node cannot be decrypted without the key IK.

C. The Time-based Key Management Scheme

With the motivation of minimizing the portion of compromised network when the initial key IK is disclosed, Jang et al. split the lifetime of a sensor network into P time slots and each time slot is assigned with an initial key. As depicted in Figure 1, T_j and N_j represent a time slot and a group of node deployed during that time slot T_j, respectively. If a node will be deployed at time slot T_j, the sensor node is preloaded with the initial key IK_j and m master keys of randomly-chosen time slots. Then the newly deployed node can establish pairwise keys with nodes which are deployed at same or different time slots. Three situations exist for the establishment of pairwise keys.

1) All nodes in the same group $N_j (1 \leq j \leq P)$ are able to establish pairwise keys with each other using the initial key IK_j during the time slot T_j.
2) Then, they are able to establish pairwise keys with other nodes which are deployed at different time slots, but have the master key derived from the current initial key. Suppose u is a node deployed at time slot T_j and v is a node deployed before T_j. If the node v has the master key K_{vj} which is derived from the initial key IK_j for time slot T_j, the node v can compute a pairwise key $K_{uv} = f_{K_{vj}}(u)$. The node u is also able to generate a master key of v, $K_{vj} = f_{IK_j}(v)$.
3) Finally, a pair of sensor nodes that do not share any keying material but are in wireless communication range can establish pairwise keys via proxy nodes.

D. The Security Problems of LEAP and The Time-based Key Management Scheme

As many existing key management protocols, LEAP assumes that sensor nodes are secure during the initialization phase and can be compromised after the phase. However, such an assumption could be incorrect. Security of LEAP mainly depends upon the initial key which is erased from sensor nodes after the initialization phase. However, the same initial key IK should be used again for node addition after that phase while the new node can be captured before removing the initial key. Therefore, the initial key IK should never be used for node addition in LEAP after the initial time T_{min}. Different initial keys are used for different time slots in the time-based key management scheme [2]. The threat caused by the disclosure of the initial key is eliminated. However, the key connectivity is constrained by the number of preloaded master keys m and the order of the current time slot. If m is far less than the lifetime P of the network, the key connectivity $\frac{m}{P-i}$ is far less than 1 at the time slots $j(j \leq P/2)$. On the contrary, if m is close to P, higher key connectivity can be achieved with heavy burden on storage. We consider the security problem of the established pairwise key between two nodes. The pairwise key does not exclusively belong to the two end nodes. As shown in Figure 2 in [2], Nodes of group N_1, N_2, and N_6 are preloaded with master key K_{u7}, the pairwise keys between any two groups of them are known by the other group. In addition, m master keys of randomly-chosen time slots are preloaded to the nodes when they are deployed to the network without taking the lifetime of nodes into consideration. Suppose a node who can survive at most G_w time slots is deployed at the j-th time slot with m master keys of randomly-chosen time slots. Those master keys of the time slots from $(j+G_w)$-th to P-th would never be used. They waste scarce memory of sensor nodes.

III. Our Construction

Motivated by the random key predistribution scheme in [5], we propose a novel key management for multi-phase hierarchical WSN. In this scheme, the time domain of the network is split into many time slots. In this paper, we call the time slot generation as well. It is assumed that there are totally P generations. Sensor nodes are usually powered by battery. It is assumed that a node may live at most for G_w generations. Each generation has its initial key which is constructed by a key distribution center. There is no relation between the initial keys for different time slots. This property prevent the attackers from concluding the previous and future initial keys. Wireless sensor network are set up for longer lifetime as compared to that of sensor nodes. Therefore, new nodes need to be replenished in some generations in the case of node capture attack or depletion of battery to provide continuity of network. It is supposed that an attacker can get all the key materials stored in the captured node. In order to achieve connectivity between nodes belonging to different generations and resiliency, the keys that are used to establish pairwise keys should evolve in a different way, independent of evolution of the initial keys. Table I presents the symbols used in our proposed key management scheme.

Table I
SYMBOLS USED IN THE PROPOSED KEY MANAGEMENT SCHEME

P	The initial key pool size and the whole life time of the network
KR^j	The key ring of nodes deployed at generation j
K^j_{uv}	Pairwise key between nodes u and v which deployed at generation j
K^{gh}_{uv}	Pairwise key between nodes u which deployed at generation g and v which deployed at generation v where $1 \leq g < h \leq g + G_w - 1$
$H(\cdot)$	Secure hash function

A. Predistribution of Key Materials

In the proposed scheme, as depicted in Figure 2, the group G_j deployed at generation T_j are assigned with an initial key IK_j and $G_w - 1$ master keys in order to establish links with nodes deployed at the same or different generations. The initial key is reserved for the establishment of secure communication with nodes deployed in the same generation. The other $G_w - 1$ master keys are used to establish secure links with the groups deployed at subsequent generations. Different from the time-based key management scheme in [2], the master keys in our scheme are constructed in an ingenious way. These master keys are transformed from the initial keys for the generations within the the node's generation window G_w. For a group deployed at generation j, their sub-keyring $S - KR^j$ containing the master keys for the subsequent generations $j + 1 \leq i < j + G_w - 1$, is given as follows:

$$S - KR^j = \{K_{j,i} | K_{j,i} = H(IK_i \parallel j)\}, \tag{1}$$

where IK_i is the initial key for the i-th generation.

"This process " can be detailed as follows. In order to form the sub-keyring $S - KR^j$, the key distributor first picks

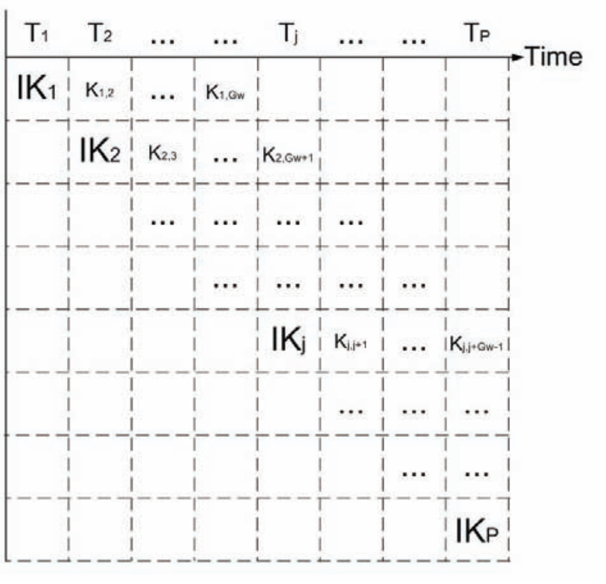


Figure 2. The key rings preloaded to nodes at different generations.

up $G_w - 1$ initial keys for the subsequent $G_w - 1$ generations. Each of these keys is appended with the generation number of the group, which is j, and hashed using a secure hash function like SHA-1 [6] or SHA-256 [7], depending on the key size. These hashed values are stored in the sub-keyring. In this way, we customize the keys belonging to a subsequent generation to be used in another generation without storing the actual initial keys, owing to the one-way property of the secure hash functions.

To sum up, the keyring of a sensor node contains (1) the initial key assigned to the current generation; (2) the transformed master keys for up to $G_w - 1$ subsequent generations. More formally, for a node deployed at generation j, its key ring KR^j is shown as follows:

$$KR^j = \{IK_j, K_{j,j+1}, \ldots, K_{j,j+G_w-1}\}. \tag{2}$$

That is, each sensor node stores one initial key and $G_w - 1$ master keys for each upcoming generation. Because a sensor node may communicate with nodes at most $G_w - 1$ next generation, the maximum number of keys in the key ring of a particular node is G_w.

B. Key Establishment

After the keyring is created, the nodes are deployed over the sensor field. Two situations exist for the establishment of pairwise keys.

1) Since all sensor nodes deployed at generation j contain the initial key IK_j, they can establish pairwise keys using IK_j. Suppose two nodes u and v belong to the same generation j, they compute their pairwise keys as follows:

- After a node u computes a master key $K_u^j = f_{IK_j}(u)$, node u broadcasts a HELLO message with its ID and generation j and then waits for a response from the neighboring node v which is deployed in the same generation. Node v sends node u a response message including its ID and MAC.

$$u \to * : u, j, nonce;$$

$$v \to u : v, MAC(K_v^j, u \mid v).$$

- Both u and v can compute a pairwise key $K_{uv}^j = f_{K_v^j}(u) = f_{K_u^j}(v)$.

2) If the two nodes belong to different generations, the pairwise key creation process is different. Let us suppose that the node u deployed at generation g and another node v deployed at generation $h (1 \geq g < h \leq g + G_w - 1)$. They computer their pairwise keys as follows:

$$K_{uv}^{gh} = f_{K_v^{gh}}(u) = f_{K_u^{gh}}(v)$$

where $K_v^{gh} = f_{K_{gh}}(v)$ and $K_u^{gh} = f_{K_{gh}}(u)$. Node u is already preloaded with the key K_{gh} before deployment. By using this key, node u can calculate K_u^{gh}. However, from the point of view of node v, the master key K_{gh} is the master key for the previous generation g. Therefore, it is not in node v's keyring. Fortunately, node v can calculate K_{gh}. As discussed in the previous subsection, $K_{gh} = H(IK_h \parallel g)$, where H is a secure one-way hash function. Node v is deployed at generation h, and therefore it stores the initial key IK_h. By using IK_h, it calculates the key K_v^{gh} and then calculates K_{uv}^{gh}.

IV. Security and Performance Analysis

A. Security Analysis

The goal of this section is to evaluate the security of our proposal and compare it with that of the time-based key management scheme proposed in [2].

We suppose that when a node is compromised, the key material stored in the node will be extracted by the adversary. The key material will be utilized to attack the rest of network. In [2], the resilience of schemes is described as that the additional portion of network that an adversary can compromise using the key material obtained from x compromised nodes. We still use this definition in this section. The security of the LEAP scheme depends on the security of the initial key IK. The whole network can be compromised once the initial key K_I is disclosed. The damage resulting from a disclosure of an initial key IK is localized by the time-based key management scheme [2]. In order to provide connectivity between nodes deployed at different generations, the preloaded master key for different generation is the same. An compromised initial key IK_a at generation T_a only affect the nodes deployed at generation T_a rather than the whole network. However, as we mentioned in Section II, the pairwise key does not exclusively belong to the two end nodes. If three nodes are preloaded with the same master key, the pairwise keys between any two groups of them are known by the other group. Once a node is captured, the pairwise keys shared by other two nodes will be compromised as well.

In our scheme, the pairwise key K_{uv}^{gh} exclusively belongs to the two end nodes. For example, the pairwise key used between node u deployed at generation g and v deployed at generation h is confined to the two end nodes deployed at these two generations. Nodes deployed at generations other than g and h have no access to this key. This is because the master key K_{gh} can be computed by a node if and only if this node has been deployed at generation h and has an initial key IK_h in its key ring. As a result, a sensor node w, which is deployed at any other generation l cannot compute a master key K_{gh}. Three conditions exist according to the value of l

- $l < h$. The node w needs IK_h to compute K_{gh}. Even though the node w has the master key $K_{lh} = H(IK_h \parallel l)$, it cannot derive IK_h from K_{lh} due to the one-way property of the secure hash function H.
- $h < l \leq g + G_w - 1$. The node u is preloaded with the master key $K_{gl} = H(IK_l \parallel g)$ and the node v is preloaded with the master key $K_{hl} = H(IK_l \parallel l)$. Even the node w can calculate K_{gl} and K_{hl}, it cannot derive IK_h or K_{gh} due to the one-way property of the secure hash function H.
- $l > g + G_w - 1$. The node u is power off at the generation l.

It is clear that a master key K_{gh} is known only by the nodes of the generation g and h. No node deployed at any other generation can compute the key that is unique to the generation g and h. Hence an attacker has to spend extra effort if s/he wants to acquire the pairwise key between the nodes u and v that are deployed at the generations g and h, respectively. This has an advantage over the scheme in [2] in restricting the information that an attack acquires if s/he captures a node.

B. Performance Analysis

Key Connectivity. The key connectivity of a group N_t which is deployed at generation t of the time-based key management scheme [2] is assessed from two aspects. One is N_t's probability of sharing keying materials with prospective sensor nodes, $p_{pros}(t)$ and the other is the probability of sharing keying materials with predeployed sensor nodes and nodes deployed at the current generation G_t, $p_{pre}(t)$. According to the scheme, when a sensor node is deployed at generation t, only the predeployed nodes which have the master key, which is derived from the initial key IK_t of the generation t, can establish pairwise key with it. Because each node is preloaded with the initial key of the

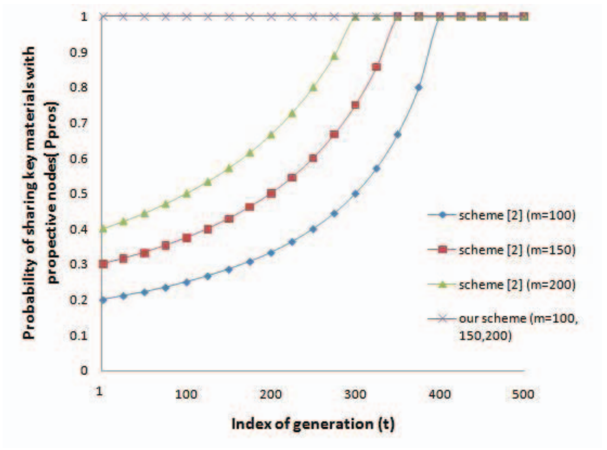


Figure 3. The comparison of the probabilities that N_t establishes pairwise key with prospective groups in our scheme and the scheme in [2].

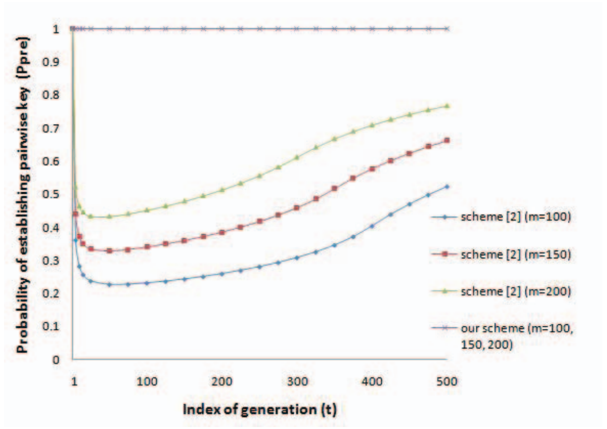


Figure 4. The comparison of the probabilities that N_i establishes pairwise key with pre-deployed sensors and the sensors being deployed in the same generation in our scheme and the scheme in [2].

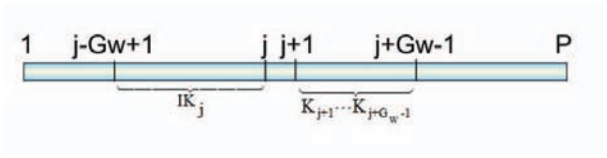


Figure 5. Nodes deployed at generation j can establish pairwise nodes with at most $2G_w - 1$ generations with 100% probability.

current generation and m master keys of randomly chosen generations after deployment, the probability of sharing key material with other prospective nodes is $\frac{m}{P-t}$ where P is the number of total generations (Please refer to [2] for the process of calculation). The variation $1 \leq t < P$. When $P - t \leq m$, the master keys of all remaining generations will be preloaded to the nodes deployed at generation t so that the key connectivity probability $p_{pros}(t)$ is 1 and keeps 1 for the subsequent generations. We make a comparison of the probabilities that N_t establishes pairwise key with prospective groups in our scheme and the scheme in [2] in Figure 3. In order to facilitate the comparison, we keep the size of key pool (the same as the number of generations of a network) 500. The three curves in Figure 3 demonstrate the conditions of [2] when m equals 100, 150, and 200, respectively. As shown in the figure, the probability p_{pros} increases as the index the generation t increases. In our scheme, the group N_t can always share keying material with groups in its generation window $[t, t + G_w - 1]$ with 100% probability. The variation $1 \leq m \leq P-1$, it is easy to reach the conclusion that the more master keys a sensor node has in [2], the higher probability of key connectivity becomes. In our scheme, more than $G_w - 1$ preloaded master keys do not increase the key connectivity probability as $G_w - 1$ master keys are enough for reaching 100% probability.

In terms of $p_{pre}(t)$, it can be calculated by using p_{pros}. If we suppose that sensor nodes are uniformly distributed at each generation, p_{pre} can be calculated as:

$$p_{pre}(t) = (\sum_{i=1}^{t-1} p_{pros}(i) + 1)/t,$$

where 1 means that a sensor node can establish pairwise keys with nodes deployed at the same generation with 100% probability. In fact, it is the average value of the probabilities of key connectivity with each preloaded and the current group of sensor nodes. Figure 4 describes the change tendency of probability p_{pre} for various m when the number of generation P is fixed to 500.

Different from the scheme in [2], when a sensor node is deployed at generation t, it is preloaded with the initial key IK_t and $G_w - 1$ master keys for the subsequent $G_w - 1$ generations. The node can establish pairwise keys with nodes in at most $2G_w - 1$ generations with 100% probability as long as both of them are still viable. As described in Figure 5, a node deployed at generation j can establish pairwise keys with nodes deployed from generation $j - G_w + 1$ to generation j by using its initial key IK_j and establish pairwise keys with nodes deployed from generation $j + 1$ to generation $j + G_w - 1$ by using the proloaded master keys for each generation. In time-based scheme [2], a node is preloaded with m master keys of randomly- chosen generations. Some of these master keys might not be used before the node is power off. Suppose a node is deployed at generation j and is viable at at most G_w generations, those master keys of the generations from $j+G_w$ to P are useless.

Storage Overhead. In terms of storage overhead, it is determined by the memory of a node and generations that a node can survive. In a practical application of network deployment model, the beginning of new generation means the occurrence of node additions Therefore, the number of generations is approximately equal to the number of node addition. Therefore, the higher frequency of node additions, the larger number of generations a network has. According to our scheme, a sensor node surviving 100 generations has to store 100 keys including one initial key and 99 master keys. These node can establish pairwise keys with nodes in 199 generations with 100% probability. The modern sensor nodes such as MICA-Z have 128KB program memory, 4KB runtime memory, and 512KB external memory [10]. Suppose the size of a key is 128bits, our scheme requires only 1.6KB memory. Our scheme has reasonable storage requirement for modern sensor nodes.

V. Conclusion

The LEAP key management mechanism of LEAP is welcomed due to its multiple keying mechanism. However, the security of all types of keys is mainly depends on that of an initial key. It is assumed that the initial deployment phase is secure and the key is erased from sensor nodes after the initialization phase. However, the same key should be used again for node addition after that phase while the new node can be captured before removing the initial key. A time-based key management scheme was proposed to eliminate the effect of disclosure of the initial keys. The time-based scheme split the time domain of network into many time slots. Each time slot has its own initial key. This scheme does disperse the damage resulting from the disclosure of the initial key. However, the scheme reduces the probability of key connectivity. In contrast, the proposed scheme in this paper keeps 100% key connectivity within the node's lifetime time without degrade security. The established pairwise key is exclusively known only by the nodes of the generations which the two end nodes belong to. No node deployed at other generations can compute the pairwise key.

References

[1] S. Zhu, S. Sanjeev, and J. Sushil. *LEAP: efficient security mechanisms for large-scale distributed sensor networks*, Proceedings of the 10th ACM conference on Computer and communications security. Washington D.C., USA. ACM. 2003.

[2] J. Jang, T. Kwon, and J. Song. *A Time-Based Key Management Protocol for Wireless Sensor Networks*, Information Security Practice and Experience, pp.314-328, 2007.

[3] L. Eschenauer and V. Gligor, *A Key Management Scheme for Distributed Sensor Networks*, Proceedings of the 9th ACM Conference on Computer and Communications Security, pp.4147, 2002.

[4] H. Chan, A. Perring, and D. Song, *Random Key Predistribution Scheme for Sensor Networks*, Proceedings of the IEEE Symposium on Security and Privacy, 2003.

[5] M. Ergun, A. Levi, and E. Savas, *Increasing Resiliency in Multi-phase Wireless Sensor Networks: Generationwise Key Predistribution Approach*, The Computer Journal Advance Access, Published by Oxford Univerisity Press on behalf of The British Computer Society, May 11, 2010.

[6] FIPSPUB180-1. (1995) http://www.itl.nist.gov/fipspubs/fip180-1.htm (accessed July 12, 2010).

[7] FIPS PUB 180-2. (2002) http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf (accessed July 12, 2010).

[8] C. Castelluccia, and A. Spognardi, *RoK: A Robust Key Predistribution Protocol for Multi-phase Wireless Sensor Networks*, Proceedings of SecureComm 20073rd International Conference on Security and Privacy in Communications Networks, pp.351-360, 2007.

[9] K. Kalkan, S. Yilmaz, O. Z. Yilmaz, A. Levi, *A Highly Resilient and Zone-based Key Predistribution Protocol for Multiphase Wireless Sensor Networks*, Proceedings Q2SWinet095th ACM International Symposium on QoS and Security for Wireless and Mobile Networks, Tenerife, Spain, pp.2936. 2009.

[10] Crossbow Technology (http://www.xbow.com)(Accessed July 20,2010)