A Secure Coordination-based Data Dissemination Protocol for Mobile Sinks in Sensor Networks

by

Le Xuan Hung

Presented to
the Faculty of the Graduate School of Computer Engineering
in Partial Fulfillment of the Requirements
for the Degree of
Ph.D.

Department of Computer Engineering
Kyung Hee University
Korea

February, 2009

A Secure Coordination-based Data Dissemination Protocol for Mobile Sinks in Sensor Networks

Le Xuan Hung

Department of Computer engineering, Kyung Hee University

Thesis Committee:

Chae, Oksam. Professor

Lee, Heejo. Professor

Hong, Seon Choong. Professor

Huh , Eui-Nam. Professor

J. d'Auriol, Brian. Professor

Lee, Sungyoung. Professor

Acknowledgement

First and foremost, I would like to express my gratitude to my supervisor, Prof. Sungyoung Lee, who gave me the opportunity to do this Ph.D research. He gave me the freedom to try out new ideas and gave me continuous support during the research.

I particularly appreciate Prof. Young-Koo Lee and Prof. Heejo Lee for their guidance and continue support throughout my PhD study. I also would like to thank Prof. Brian J. d'Auriol for his great help on my dissertation.

I am also grateful to all members of Ubiquitous Security Team, as well as the Ubiquitous Computing Group, for their collaboration and friendship during my study.

I would like to thank my parents and my elder bother, who always support my decision to continue my study. I also appreciate my parents-in-law. They are helping me a lot and take care of our baby. Without their help, I would not be able to continue my PhD.

Last but not least, I would like to express my deepest gratitude to my wife, Le Nhat Tu, for her love, patient and fully support.

Le Xuan Hung

February 2009,

Seoul, Korea

Abstract

Advances in wireless communications and *Micro Electro-Mechanical Systems* (MEMS) have motivated the development of extremely small, low-cost sensors that possess sensing, signal processing and wireless communication capacities. Sensors have very scarce resources in terms of memory, energy, communication, and computation capacities.

There are a number of motivations behinds this study. First, routing protocols play an important role in wireless sensor networks to deliver sensed data from the network to users. Due to the short communication range, an efficient multi-hop routing scheme is very essential. Second, for many sensor network applications such as military, home land security, users (sinks) need to access the network while they are moving. Thus, the routing protocol must support sink mobility. Third, a secure end-to-end transmission is very crucial. Sensed data is very critical and needed to protect integrity, and confidentiality.

However, there are many non-trivial challenges ahead. Routing protocols require energy-efficiency due to its limited power supplies, short delay, high packet delivery ratio, and long network lifetime. On the other hand, maintenance of secure source-to-sink communication demands a high cost due to two main reasons: (1) in order to disseminate data to a mobile sink, the sensor network must be aware of the sink's location; therefore the mobile sink has to frequently broadcast its current location to the whole sensor network; doing this causes lots of communication overhead; (2) intermediate nodes on a new routing path need to exchange secret information to establish a secure communication; this requires not only communication and computation overhead, but also an efficient key management scheme to provide a secrete key infrastructure for sensor networks, (3) several studies in sensor networks have shown that routing protocols face with many well-know security vulnerabilities including spoofed, altered, or replayed routing information, selective forwarding, sinkhole attacks, *Sybil* attacks, wormhole attacks, and HELLO

flood (unidirectional link) attacks. Tackling those problems would be a non-trivial task.

Previous studies on sink mobility have mainly focused on efficiency and effectiveness of data dissemination without security consideration. Also, studies and experiences have shown that considering security during design time is the best way to provide security for sensor network routing. This dissertation presents an energy-efficient secure routing for mobile sinks in sensor networks, called *Secure COordination-based Data dissEmination for mobile sinks* (SCODE). In SCODE, the network is partitioned into a virtual hexagonal plane. Nodes in the same cell negotiate so that only one node stays awake, while the other may fall into sleeping mode. The proposed routing algorithm is a cell-based approach, which is more flexible than node-based and location-based approaches. In order to increase security and efficiency, the key management scheme and routing protocol are considered together during the design time. Moreover, an inspecting system, a type of autonomous diagnosis system, is proposed to defend against node compromise attacks and recover the routing path under attacks. Furthermore, this presentation also presents an simplified version of SCODE, which replaces the hexagon by a square topology. Compared with hexagonal topology, the square brings less computational complexity, more applicable to various applications.

Analysis and simulation-based evaluations are conducted to evaluate the proposed protocol and compare with existing approaches. The security analysis demonstrates that the proposed scheme can defend against aforementioned attacks in sensor network routings. The analysis and simulation-based evaluations show that the SCODE significantly reduces communication overhead, energy consumption, average delay, while it always delivers more than 97% of packets successfully. The SCODE also works well with large scale networks in which hundreds to thousands of users may simultaneously access to sensor nodes.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recent advances in electronic and communication technology have resulted in large scale sensor networks with hundreds or thousands of unattended sensors. These distributed wireless networks have enabled various important wireless applications, including real-time multimedia communication, medical applications, surveillance, and home networking applications.

Current researchers are focusing on important issues in the design of wireless sensor networks, that is two key resources – communication bandwidth and energy efficiency. Each sensor device which is in a small size has a limited radio communication and power. Moreover, recharging batteries of thousands of sensors in a hostile or remote environment is almost impossible. Such limitations require emergence of communication techniques to increase the amount of bandwidth per user and innovative design techniques and protocols to use available energy efficiently. Communication protocols must be designed to adapt to current conditions instead of being designed for worst-case conditions.

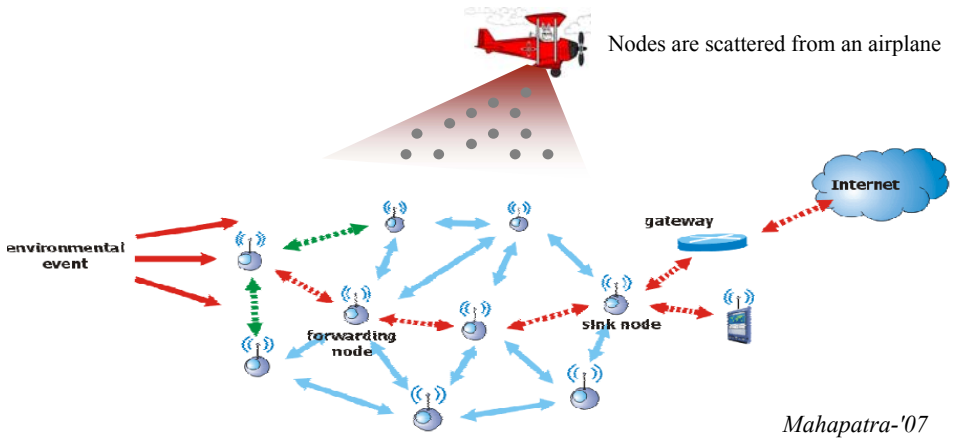


Fig. 1. A Sensor network example.

A typical example of sensor networks is shown in Fig. 1. Sensor nodes may be scattered from an airplane. After deployment, they negotiate with each other to collect event or environment information and report to users. In sensor networks, a data source, which is a sensor node, is usually located where environmental activities of interest take place. Once a stimulus appears, sensor nodes surrounding the target are in charge of sensing and aggregating. Then only one node collects useful information and disseminates through the sensor network to the sink. A sink such as a PDA is equipped by a user, used to gather data from the sensor network. Users may receive sensed data directly from the sensor network, or via Internet.

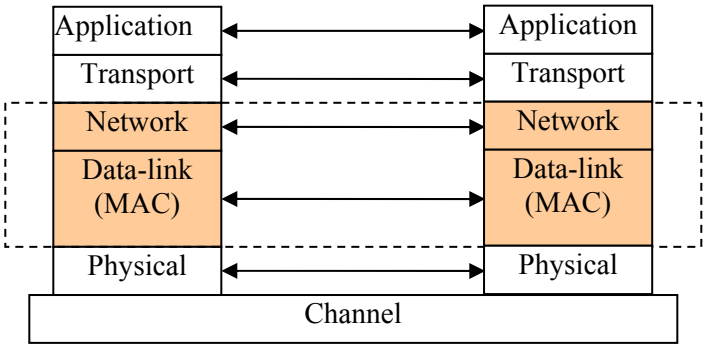


Fig. 2. Wireless sensor network layers

Though a wireless sensor network composes of five layers as illustrated in Fig. 2, the main focus in this study lies on the network layer and medium access control (MAC) layer (data-link layer). Network protocols deal with how to disseminate data from a source to a sink efficiently. The MAC layer provides efficient medium access control for the sensor networks. In addition, MAC provides protocol in order to turn off the radio to conserver energy and reduce collision such as MAC 802.11 protocol.

1.1 Wireless sensor networks

1.1.1 Applications of WSNs

WSNs have many applications which basically are classified into main the following categories:

- **Military applications**: For military use, sensor networks are mainly used in area monitoring. For example, soldiers use the sensor network to detect tanks or enemies in the battle field. Sensor networks can also be used for monitoring the status of friendly troops and the availability of equipment and the ammunition in a battle field.
- **Healthcare applications**: Sensor networks can provide interfaces for disabled, integrated patient monitoring. It can monitor and detect elderly people's behavior, e.g., when a patient has fallen. These small sensor nodes allow patients a greater freedom of movement and allow doctors to identify pre-defined symptoms earlier. The small installed sensor can also enable tracking and monitoring of doctors and patients inside a hospital.
- **Traffic monitoring**: A traffic sensor network collects data on travel speed, lane occupancy, and vehicles counts by installing sensor nodes along highways. This collected data makes it possible to calculate the average speeds or travel times and enable service such as telemetries. Traffic information can inform drivers not only how to get point from another but also how long it will take to get there, or even direct them to another route in case of traffic jam.
- **Other commercial applications**: The advance of wireless sensor networks leads to many agricultural, industrial and commercial applications. Some examples are herd monitoring, building virtual keyboards, managing inventory, monitoring product quality, interactive toys and transportation management.

1.1.2 Sensor node architecture

The design of actual sensor nodes is critical to the success of the design of a wireless sensor network. Each node must provide necessary operation while keeping its size and product cost extremely low. Typically, a sensor node consists of components as described in Fig. 3.

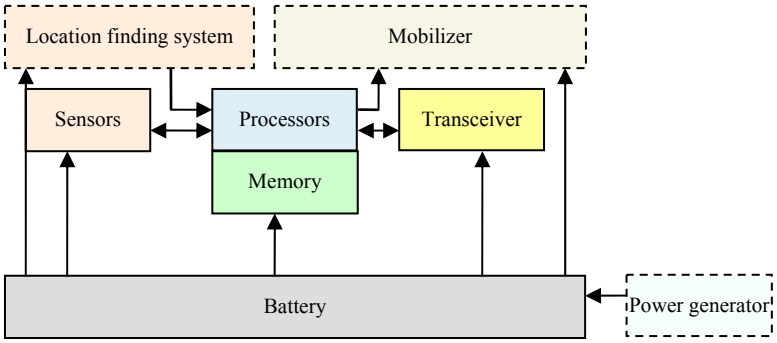


Fig. 3. Sensor Node Architecture

- **Embedded processors**: quite simple embedded processors, such as the Atmel or the Texas Instruments MSP 430. A decisive characteristic here is, apart from the obviously important power consumption, an answer to the important question whether and how these microcontrollers can be put into various operational and sleep modes, how many of these sleep modes exist, how long it takes and how much energy it costs to switch between these modes. Also, the required chip size, computational power and on-chip memory are important.

- **Sensors**: These sensing units are usually composed of two subunits: sensors and analog-to-digital converters (ADCs). The analog signals produces by the sensors based on the observed phenomenon are converted to digital signals by the ADC, and the fed into the processing unit.

- **Transceiver**: the common transceiver used nowadays is radio transceivers, including the RFM TR1001 or Infineon or Chipcon devices. Typically, ASK or FSK is used, the Berkeley PicoNodes use OOL modulation. Advanced radio concepts like ultra-wide band are under discussion, but their impact is not yet clear. A crucial step forward would be the introduction of a reasonably working wake-up radio concept which could either wake up all nodes in the vicinity of a sender or even only some directly addressed nodes. A wake-up radio allows a node to sleep and to be wakened up by suitable transmissions from other nodes, using

only a low-power detection circuits. Transmission media other than radio communication are also considered, for example optical communication.

- **Batteries**: Battery of node provides require energy for sensor node but can not provide much. Due to energy limitation, power consumption in sensor networks is very crucial issues. Currently, many efforts focus on how to conserve power in sensing, computation and communication.

- **Memories**: The operating system or, rather, run-time environment for such system is also a hot debated issue in the literature. On the other hand, minimal memory footprint and executions overhead are required. Flexible means to combine protocol building blocks are necessary, since a simple, layered architecture is unlikely to be optional and since it can be expected that meta information has to be used in many places in a protocols stack (e.g., information about location, received signal strength, etc. has an influence on many different protocol functions). Consequently, I believe that structures like black-boards, publish/subscribe or tuple spaces are an interesting starting point for the run-time environments for such nodes.

- There are also other subunits that are application-dependent. Most of the sensor network routing techniques and sensing tasks require knowledge of location with high accuracy. Thus, it is common that a sensor node has a location finding system. A mobilizer may sometimes be needed to move sensor nodes then it is required to carry out the assignment tasks.

All of these units may need to fit into a march-box size module. The required size may be smaller than even a cubic centimeter, which is light enough to remain suspended in the air. Apart from the size, there are some other stringent constrains for sensor nodes. These nodes must consume extremely low power, operate in high volumetric densities, and have low production cost, be dispensable and autonomous, operate unattended, and be adaptive to the environment.

1.1.3 Communication architecture

The protocol stack used by sensor nodes is given in Fig. 4. This protocol stack combines power and routing awareness, integrates data with networking protocols, communicates power efficiently through the wireless medium, and promotes cooperative efforts of sensor nodes. This protocol stack consists of physical layer, data link layer, network layer, transport layer, application layer, and management planes including power management plane, mobility management plane, and task management plane.

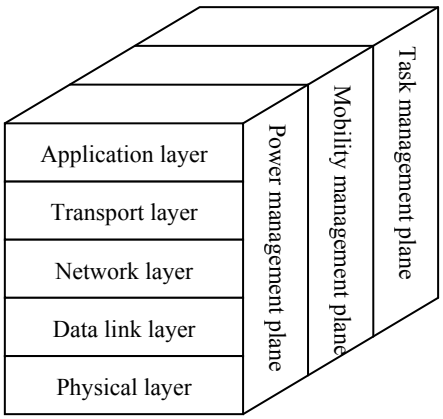


Fig. 4. The sensor networks protocol stack

The physical layer deals with data transmission and reception. The medium access control (MAC) protocol must be power aware and able to minimize collision with neighbors' broadcast. The network layer is in charge of routing the data supplied by the transport layer. The transport layer helps to maintain the flow of data if the sensor networks application requires it. Depending on the sensing tasks, different types of application software can be built and used on the application layer. In addition, the power, mobility, and task management planes manage power, movement and task distribution among sensor nodes. These planes help sensor nodes coordinate the sensing task and lower the overall power consumption.

1.1.4 Routing protocols in WSNs

Due to limited power supply and not possible to recharge, the energy consumption is the most concerned in sensor networks. Designing routing protocols is a very challenging research issue. Data delivery plays the most important role in sensor networks. Data, dynamically acquired from the environment, travel through the network towards the sink, offering low-latency real-time information that was previously hard or infeasible to get.

There are basically three types of schemes concerning data delivery: continuous, event driven and observer-initiated. According to the first one, sensor nodes send their measurement to the sink at a specified rate, while in the event-driven model nodes send the measurement data to the base station whenever they detect some type of activity that is worth reporting. In the observe-initiated scheme, the sink itself issues queries to any node in the network or to all nodes within a specific area, resulting in sensors collecting data and sending them back to the base station.

From the perspective of routing communication, routing protocols for wireless sensor networks can broadly be classified into two categories: multi-hop routing protocols and clustering approaches.

(a) *Multi-hop Routing*

The basic idea of multi-hop routing is that data is relayed through some hops before reaching the sink using short communication. Two first routing protocols for wireless networks, Destination-Sequenced Distance Vector (DSDV) and Ad hoc On-demand Distance Vector (AODV), are originated from routing protocols for wire networks, distance vector routing and link-state routing respectively. However, there are some difficulties with these protocols. The periodic message needed to maintain valid routes may not only congest the network, they may also drain the limited battery supply of a portable node. Dynamic source routing (DSR), solves this problem by only creating routes on an on-demand basic. This minimizes the amount of overhead needed in creating

routes, at the expense of latency in finding a route when it is needed. These are ad-hoc, self-configuring protocols that are robust to node failures.

Recently, there has been much work on "power-aware" routing protocols for wireless networks [61][62]. In these protocols, optimal routes are chosen based on the energy at each node along the route. Routes that are longer but use nodes with more energy than the nodes along the shorter route are preferred. This helps avoid "hot-spot" in the network, where a node is often used to route other nodes' data, and it helps to evenly distributed energy dissipation.

(b) *Clustering*

Another method of wireless communication is to use clustering approach, similar to a cellular telephone network. The basic idea is that sensor nodes are grouped into clusters. Each cluster has a clusterhead which is the most powerful node of the cluster. The clusterhead is in charge of collecting data from its members, processing and forwarding to the sink. Clustering enable bandwidth reuse and can thus increase system capacity. Using a clustering approach enables better resource allocation and helps improve power control [28]. In addition, the hierarchical structure obtained using clustering can help overcome some of the problems with node mobility.

While conventional cellular networks rely on a fixed infrastructure, new research is focusing on ways to deploy clustering architectures in ad-hoc fashion, without the assistance of a fixed infrastructure. Early work by Baker *et al.* developed linked cluster architecture. Using the distributed linked cluster algorithm (LCA), nodes are assigned to be either ordinary nodes, clusterhead nodes, or gateways act as the backbone network, transporting data between clusters. This enables robust networking with point-to-point connectivity.

Heinzelman *et al* have proposed LEACH protocol [32]. This is a fully distributed cluster formation and communication algorithm where there are no fixed clusterhead node and cluster as well. LEACH is an emerging approach that minimizes energy dissipation. By randomly selecting cluster-heads, it can

aggregate data at the cluster heads to reduce amount of data for transmission. LEACH mainly concerns the function of the application, the need of easy development, and the severe energy constrains of the nodes. By computing locally to reduce the amount of transmitted data, network configuration and operation is done using local control, and Media Access Control (MAC) and routing protocols enable low-energy networking, LEACH provides the high performance needed under the tight constraints of the wireless channel.

In a static clustering protocol for microsensor networks, nodes are organized into clusters initially, and these clusters and the clusterheads remain fixed throughout the lifetime of the network. Nodes transmit their data to the clusterhead node during each frame of data transfer, and the clusterhead forwards the data to the sink. Since data from nodes located closed to each other are highly correlated, the clusterhead node aggregates the signals to reduce the actual amount of data that must be transmitted to the sink. Since the clusterhead must transmit the data to the end-user via shared wireless channel, if the clusterhead could not aggregate the data, there would be no advantage to using this approach over an approach where each node sent its data directly to the sink..

1.1.5 Sink mobility

In many sensor networks applications, sink mobility is very essential. For example, in a battle field a soldier is using a PDA or laptop computer to collect an enemy tank location while he is moving (see Fig. 5); another example is a user on a moving vehicle is checking the traffic condition though the sensor network to avoid congested routes (Fig. 6).

There are a number of efforts having been focused on several sink mobility issues in the sensor networks. One of early approaches to solve the issues of data dissemination to mobile sinks is two-tier data dissemination (TTDD) [12] in which data sources use a geographic mesh to broadcast their data and sinks subscribe to the data at their nearest mess point. Another effort [63] attempts to solve the network lifetime issue in sensor networks with sink mobility.

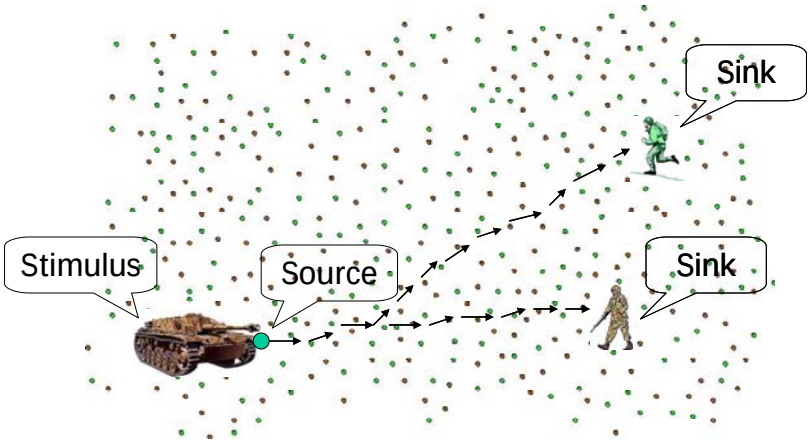


Fig. 5. A mobile soldier is observing the enemy tank location

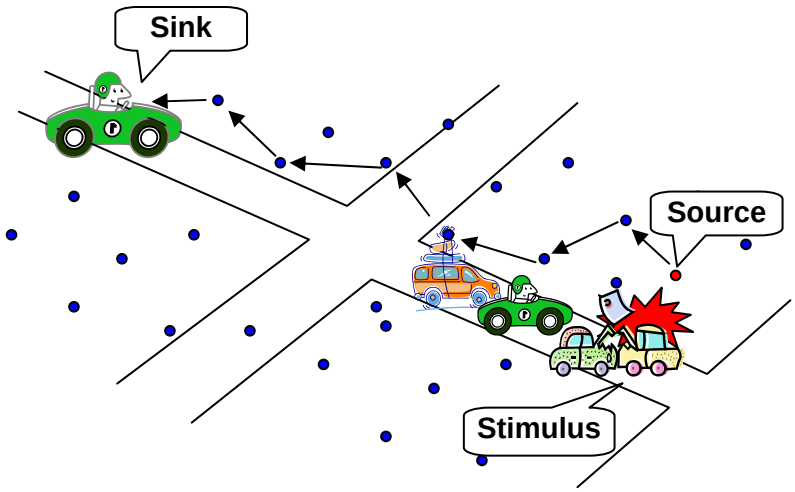


Fig. 6. A user on the moving vehicular checks traffic congestion status

1.2 Motivation

In large scale sensor networks, a secure multi-hop routing protocol from sources to mobile sinks is very essential. This is motivated by the following reasons.

(1) **Multi-hop**: Routing protocols play an important role in wireless sensor networks to deliver sensed data from the network to users. Due to the short communication range, sensors cannot report their sensed data to the base stations / sinks within one hop transmission. Thus, multi-hop communication is needed. However, sensors are very resource-

constraint, and applying existing routing mechanisms for ad hoc networks is not applicable. Since transmission and receipt cost of one bit over 100m spend the same energy as that of execution of 3000 instructions [65], an efficient multi-hop routing scheme for sensor networks is major factor to achieve energy efficiency and prolongs sensor network lifetime.

(2) **Sink mobility - supported**: For many sensor network applications such as military applications, home land security, users (sinks) need to access the network while they are moving. Examples are mentioned in Section 1.1.5: a soldier uses PDA or laptop computer to trace an enemy tank location while he is moving in the batter field; or a user on a moving vehicular checks the traffic congestion status ahead before proceeding. Without a support for sink mobility, routing protocols would not be able to apply in such applications.

(3) **Secure**: Security is very critical in many sensor network applications. Many routing protocols would not be applicable without security support. After deployment, sensors are usually left unattended, thus they are prone to be compromised by adversaries. On the other hand, the sensor communication is wireless, so it is very vulnerable from many attacks (see Chapter 2 for details).

1.3 Focus of the dissertation

Routing techniques in sensor networks can be classified based on network structure into flat-based, hierarchical-based, and location-based [64], see Fig. 7. A routing protocol is considered adaptive if certain system parameters can be controlled in order to adapt to the current network conditions and available energy levels. These protocols can be further classified into multipath-based, query-based, negotiation-based, QoS-based, or coherent-based routing techniques depending on the protocol operation. Each category is further

classified into two sub-categories: one supports only for static sinks, and the other supports for mobile sinks.

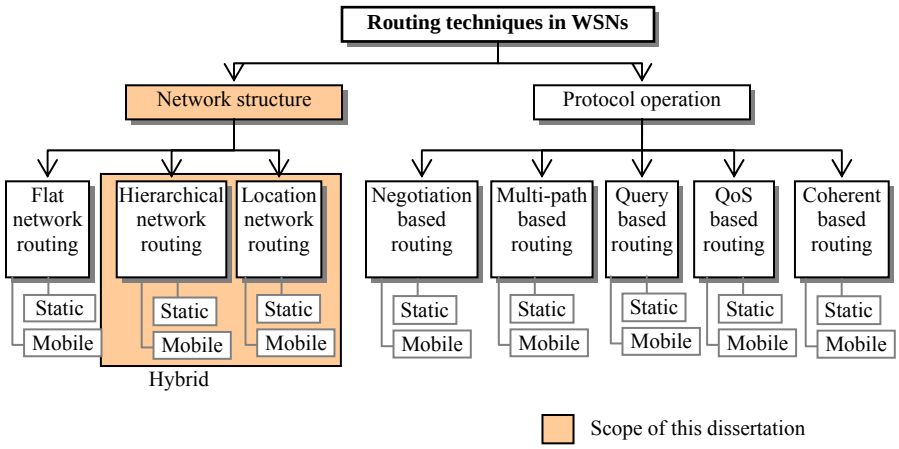


Fig. 7. Routing protocols in sensor networks: A taxonomy

This study is focused on hierarchical network routing and location network routing and supports for sink mobility. Thought these approaches require more cost for location awareness such as GPS or predistribution management, they are much more efficient than the others. The study is therefore focused more on the elimination of location awareness issues and tried to reduce its cost.

My major concern is energy efficiency. Therefore, in my proposed protocol I apply network structure-based techniques because it can be the best way to achieve efficiency. In this case, protocol operation-based such as multi-path based routing would not be a good choice because of extra communication which may significantly increase communication cost.

1.4 Problem statement

This study is focused on solving the problem of design a secure multi-hop routing protocol for mobile sinks where (1) sensors are very limited in power, and (2) the network is very vulnerable from many common attacks [33]. The problems are further described as follows:

(1) **Power constraint**: Routing protocols require energy-efficiency due to limited power of sensors, usually with two AA batteries. Applying existing approaches for ad hoc wireless networks is not appropriate due to their high cost of communication, computation, and memory. On the other hand, maintenance of secure source-to-sink communication demands a high cost due to two main reasons. First, in order to disseminate data to a mobile sink, the sensor network should be aware of the sink's location. Thus, the mobile sink has to frequently broadcast their current location to the whole sensor network. Doing this causes lots of communication overhead. Second, intermediate nodes on a new routing path need to exchange secret information to establish a secure communication; this requires not only communication and computation overhead, but also an efficient key management scheme to provide a robust and efficient key infrastructure for sensor networks,

(2) **Very vulnerable from many common attacks** [33]: After deployment, sensors are usually left unattended and easy to be physically compromised. An adversary can capture one or more nodes, injects some malicious code into them to cause threats or receives information from the network. On the other hand, due to wireless communication, an adversary can easily eavesdrop the transmission or launch serious attacks. Therefore, it is not surprising that sensor network routings are very vulnerable from many security threats. Several studies in sensor networks have shown that routing protocols face with many well-know security attacks including spoofed, altered, or replayed routing information, selective forwarding, sinkhole attacks, *Sybil* attacks, wormhole attacks, and *HELLO* flood (unidirectional link) attacks. Those attacks will be further explained in Chapter 2.

1.5 Contributions

In this dissertation, a secure routing protocol for mobile sinks is proposed. To my best knowledge, this is the first and novel routing protocol considering both

security and efficiency for mobile sinks in sensor networks. The major contribution lies in three aspects: energy-efficiency, security, and autonomous diagnosis system.

(1) **Efficiency**: The routing protocol is energy-efficient while the average delay and success ratio are comparable with existing approaches.

The protocol spends approximately 0.005 J of energy for each 64-bytes data packet transmission. If each sensor is attached with two Lithium Ion AA batteries (11,050 J), then the network can deliver about 2.2 million data packets. The average response time to users is around $0.02 \div 0.06$ second, which indicates that the protocol can work in a real-time manner. More importantly, over 97% of packets are always transmitted successfully.

An intensive analysis and simulation show that the protocol achieve better energy-efficient compared with other approaches. Compared with TTDD [12] and DD [11], the proposed protocol spends less energy, about 60.2% and 43.5% amount of TTDD and DD, respectively.

(2) **Security robustness**: It is proved that the proposed protocol is secure against common attacks in sensor networks routing mentioned earlier including spoofed, altered, replayed routing information, selective forwarding attacks, sinkhole, worm hole, Sybil attack, HELLO flood attack. Compared with existing routing protocols such as Directed Diffusion, GEAR, TTDD, LEACH, TEEN, PEGASIS, SeRINS, SEEM, this dissertation provides a completed security solution (Section 4.9).

On the other hand, a robust key management scheme is proposed to provide a key infrastructure for the routing. The proposed key management solves the memory and security robustness issues. Compared to CPKS [57], the proposed key management scheme

requires less memory and has better security robustness against node compromise attacks (Section 4.8).

(3) **Autonomous diagnosis system**: An inspecting system, a type of autonomous diagnosis system, is presented. It provides self-defense and self-healing mechanisms for sensor networks. In sensor networks, nodes are usually left unattended after deployment. Therefore, they are easy to be physically compromised by any adversaries. They also may be malfunctioning due to surrounding harsh environment. The inspecting system is proposed so as each node can keep watching on another. If some node is compromised or becomes malfunctioning, then the network can detect it and eliminates that node from the routing operations. The recovery phase in the inspecting system cover the routing path after eliminating the compromised or malfunctioning node.

Another benefit of the inspecting system is that it can be extended to predict any potential attacks in the networks.

1.6 Outline of the dissertation

The chapters of this dissertation are organized as follows

Chapter 1: Introduction

Chapter 2: In this chapter, attacks in sensor network routing are discussed. A number of related works are also discussed.

Chapter 3: An overview of SCODE is presented.

Chapter 4: This chapter presents the SCODE including key management scheme and secure routing protocol. Analysis and simulation-based evaluation are described in comparison with existing approaches. SCODE's key management scheme is compared with CPKS [57], and the routing protocol is compared with SeRINS [39], DD [11], TTDD [12], etc.

Chapter 5: A simplified version of SCODE is discussed. It simplifies the hexagonal topology to the square topology. Compared with the hexagon, the square brings better flexibility and more widely use in various applications.

Chapter 6: Conclusion and future works.

Chapter 2

Threat model and related work

2.1 Threat model

Many sensor network routing protocols are quite simple, and for this reason are sometimes susceptible to attacks from the literature on routing in ad-hoc networks. Attacks on sensor network routing have been discussed in several papers [33]-[36]. Most of the attacks fall into one of the following categories: spoofed, altered, or replayed routing information [33][34]; selective forwarding [33]; sinkhole [33]; *Sybil* [35]; wormhole [36]; and *HELLO* flood (unidirectional link) attacks [33]. I briefly describe those attacks on sensor networks as follows.

2.1.1 Spoofing, altering, or replaying routing information

The most direct attack against a routing protocol is to target the routing information exchanged between nodes. By spoofing, altering, or replaying routing information, the adversaries can create routing loops, attract or repel network traffic, extend or shorten source routes, generate false error messages, partition the network, increase end-to-end latency, etc.

An example of a spoofing attack is shown in Fig. 8 occurred in TinyOS beaconing protocol. Since routing updates are not authenticated, it is possible for any node to claim to be a base station and become the destination of all traffic in the network.

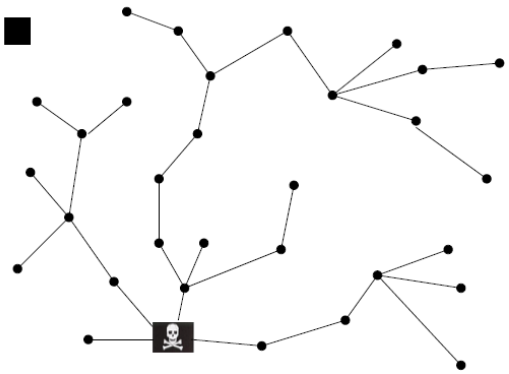


Fig. 8. An adversary spoofing a routing update from a base station in TinyOS beaconing

2.1.2 Selective forwarding attacks

Malicious nodes may refuse to forward certain messages and simply drop them, ensuring that they are not propagated any further. Another form of this attack is an adversary selectively forwards packets, i.e. she is interested in suppressing or modifying packets originating from a few selected nodes, reliably forwards the remaining traffic and limits suspicion of her wrongdoing.

2.1.3 Sinkhole and wormhole attacks

In sink-hole attacks, the adversary attracts nearly all the traffic from a particular area through a compromised node, creating a metaphorical sinkhole with the adversary at the center. With wormhole attack, adversary tunnels messages received in one part of the network over a low latency link and replays them in a different part. Wormhole attacks more commonly involve two distant malicious nodes colluding to understate their distance from each other by relaying packet along and out-of-bound channels available only to the attacker.

An example is shown in Fig. 9. The adversary first creates a wormhole between two colluding laptop-class nodes, one near the base station and one near the targeted area. The first node forwards (authenticated) routing updates

to the second one through the wormhole, which participates normally in the protocol and rebroadcasts the routing update in the targeted area. Since the "wormhole" routing update will likely reach the targeted area considerably faster than it normally would have through multi-hop routing, the second node will create a large routing sub-tree in the targeted area with itself as the root. As seen in Fig. 9, all traffic in the targeted area will be channeled through the wormhole, enabling a potent selective forwarding attack

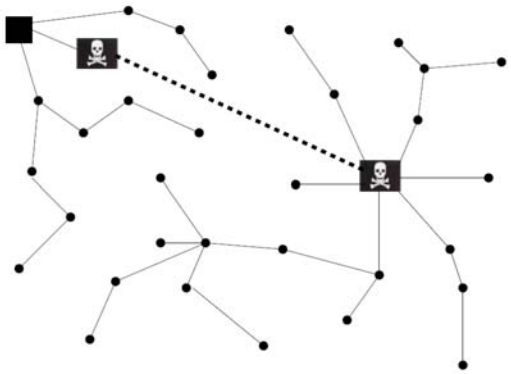


Fig. 9. A laptop-class adversary using a wormhole to create a sinkhole in TinyOS beaconing

2.1.4 *Sybil* attacks

A single node presents multiple identities to other nodes in the network. In particular, a *Sybil* attack causes a significant threat to geographic routing protocols. Using a *Sybil* attack, an adversary can cheat as many nodes at different locations.

For example, without too much additional effort, an adversary can dramatically increase her chances of success by mounting a *Sybil* attack. As depicted in Fig. 10, an adversary can advertise multiple bogus nodes surrounding each target in a circle (or sphere), each claiming to have maximum energy. By intercepting transmissions sent to each of the bogus nodes, the adversary maximizes her chances for placing herself on the path of any nearby

data flow. Once on that path, the adversary can mount a selective forwarding attack.

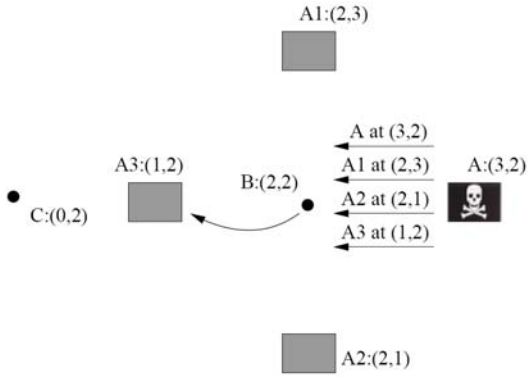


Fig. 10. The *Sybil* attack against geographic routing.

Adversary A at actual location (3,2) forges location advertisements for non-existent nodes A1, A2, and A3 as well as advertising her own location. After hearing these advertisements, if B wants to send a message to destination (0,2), it will attempt to do so through A3. This transmission can be overheard and handled by the adversary A.

2.1.5 HELLO flood (unidirectional link) attacks

A laptop-class attacker may broadcast routing or other information with large enough transmission power and convinces every node in the network that the adversary is its neighbor. As a consequence, these nodes only relay packages to the attacker's laptop.

If a laptop-class adversary has a powerful transmitter, it can use a HELLO flood attack to broadcast a routing update loud enough to reach the entire network, causing every node to mark the adversary as its parent. Most nodes will be likely out of normal radio range of both a true base station and the adversary. As shown in Fig. 11, the network is crippled: the majority of nodes are stranded, sending packets into oblivion. Due to the simplicity of this

protocol, it is unlikely there exists a simple extension to recover from this attack. A node that realizes its parent is not actually in range (say by using link layer acknowledgements) has few options short of flooding every packet. Each of its neighbors will likely have the adversary marked as its parent as well.

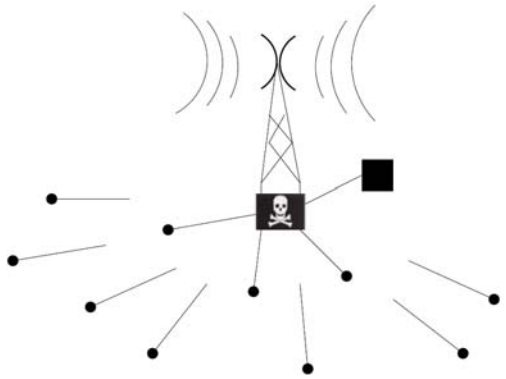


Fig. 11. HELLO flood attack against TinyOS beaconing.

A laptop-class adversary that can retransmit a routing update with enough power to be received by the entire network leaves many nodes stranded. They are out of normal radio range from the adversary but have chosen her as their parent.

2.2 Related work

Research on the sensor network routing has been carried out for nearly a decade. Though a number of routing protocols have been proposed for sensor networks, they mostly focus on efficiency and effectiveness of data dissemination, regardless either security or sink mobility issues. In the following sections, I will briefly discuss on those approaches. For a straightforward description, I first mention about such approaches which support sink mobility, and then approaches which consider security for routing.

2.2.1 Routing protocols support sink mobility

There are a number of routing protocols aiming to support mobile sinks in WSNs such as LEACH [32], Directed Diffusion [11], TTDD [12] which are summarized as follows.

Heinzelman [32] introduces a clustering algorithm for WSNs, called LEACH. Though LEACH is designed for static sinks, it can be used for mobile sinks as well thank to one-hop communication between sources and sinks. In LEACH, sensors are organized into clusters. Each cluster has one cluster head (CH) which collects and aggregates information from its members (non-CH sensors in the same cluster) and passes on information to the base station (BS), see Fig. 12. However, LEACH has a number of shortcomings. LEACH assumes every node can directly reach the base station by transmitting with sufficiently high power. However, one-hop transmission directly to the base station is not feasible in large-scale WSNs due to the resource-limitations of sensors. On the other hand, LEACH is vulnerable from several attacks including HELLO flood, selective forwarding, and *Sybil* attacks.

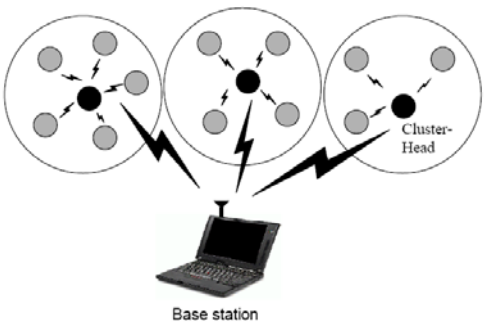


Fig. 12. LEACH routing protocol

Directed Diffusion (DD) [11] is a popular data aggregation paradigm for wireless sensor networks. It is a data-centric and application-aware paradigm, in the sense that all data generated by sensor nodes is named by attribute-value pairs. In DD, the base station requests data by broadcasting interests, which describes a required task to be implemented by the network. The interest is

defined using a list of attribute-value pairs such as name of objects, interval, duration and geographical area. Each node receiving the interest can cache it for later use. As the interest is broadcasted through the network hop-by-hop, gradients are setup to draw data satisfying the query towards the requesting node. A gradient is a reply link to the neighbor from which the interest was received. It contains the information derived from the received interest's fields, such as the data rate, duration and expiration time. Each sensor that receives the interest, sets up a gradient toward the sensor nodes from which it received the interest. This process continues until gradients are setup from the sources all the way back to the base station. In this way, several paths can be established, so that one of them is selected by reinforcement. The sink resends the original interest message through the selected path with a smaller interval, hence reinforcing the source node on that path to send data more frequently. Fig. 13 shows examples of DD ((a) sending interests, (b) building gradients and (c) data dissemination). DD suggests that each mobile sink needs to continuously propagate its location information throughout the sensor field, so that all sensor nodes get updated with the direction of sending future data reports. However, frequent location update from multiple sinks leads to both increased collisions in wireless transmissions and rapid power consumption of the sensor's limited battery supply.

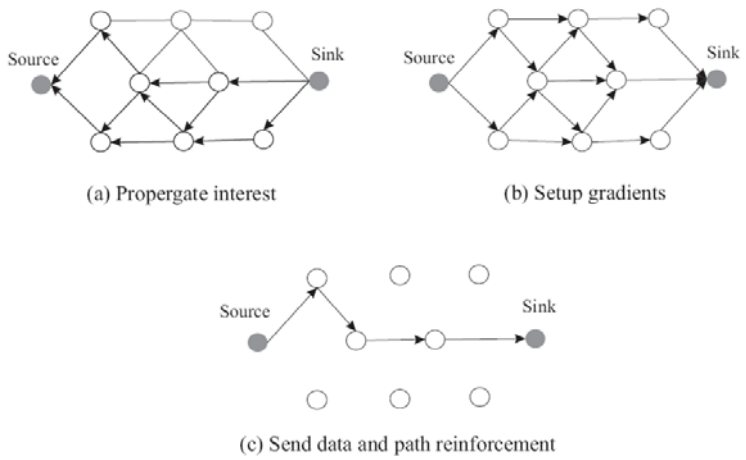


Fig. 13. Three phases of Directed Diffusion protocol

Two-Tier Data Dissemination (TTDD) [12] provides data delivery to multiple mobile base-stations based on a decentralized architecture. It assumes there are homogeneous sensors, each aware of its own location and generally stationary. There are multiple sources and mobile sinks, which query the network to collect sensing data. Instead of broadcasting their location information to all sensor nodes, TTDD uses a two-tier data dissemination model to deal with the sink mobility problem and reduce energy consumption. In TTDD, each data source uses a grid structure to divide the topology into cells as shown in Fig. 14. Only sensors located at a cell boundary need to forward the data. The data sink proactively builds the two-tier grid structure throughout the network and sets up forwarding points in the sensors closest to the grid boundaries, which are called dissemination nodes. The lower tier is the cell at the sink's current location and the higher tier contains the dissemination nodes at cell boundaries. The data sink only floods the query within its own cell. When the nearest dissemination node in the cell receives the query, it forwards it to its adjacent dissemination node in another cell. This process continues until the query reaches the producer or one of the dissemination nodes that have the corresponding data. During the query propagation, the network establishes the reverse path towards the sink, so that the data could be forwarded on the same path as that of the query propagation. TTDD exploits local flood within a local cell of a virtual grid which sources build proactively. However, it does not optimize the path from the source to the sinks. When a source communicates with a sink, the restriction of grid structure may multiply the length of a straight-line path by $\sqrt{2}$. Also, TTDD frequently renews the entire path to the sinks. It therefore increases energy consumption and connection loss ratio.

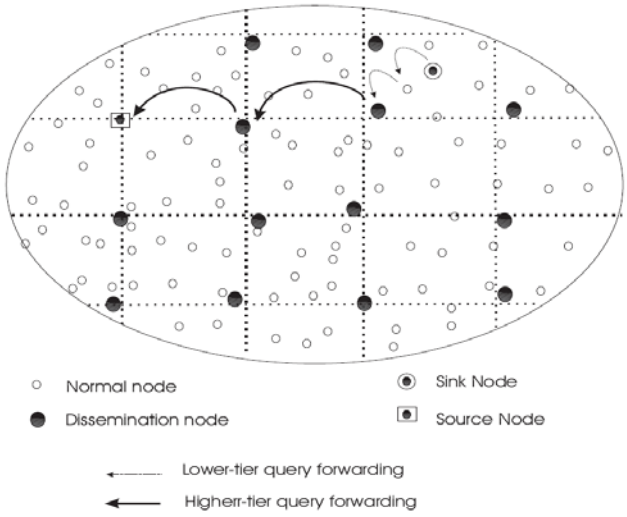


Fig. 14. Two-tier grid structure in TTDD

Since those multi-hop routing protocols did not consider security, it is not surprising that they face many security problems from spoofed, altered, or re-played routing information, selective forwarding, sinkhole, *Sybil*, wormhole, and HELLO flood (unidirectional link) attacks.

2.2.2 Routing protocols support security

Recent works have taken into account of both performance and security such as SecRout [37], SEEM [38], SeRINS [39], and TTSR [40].

The SecRout protocol [37] uses the symmetric cryptography to secure messages, and uses a small cache in sensor nodes to record the partial routing path (previous and next nodes) to the destination. It guarantees that the destination will be able to identify and discard the tampered messages and ensure that the messages received are not tampered. In SecRout, the routing packet and data packet are very small because they only include the partial path information. It does not use the source routing because in source routing the identities of the traversed intermediate nodes are accumulated in the route request. Also, it creates the route first, and then forwards the data to the sink node along the route. It does not flood the data packet to the sink node because

the data packet has much bigger size than the routing packet. It uses the two-level architecture, where the cluster head aggregates the data, then sends it to the sink node along the route. Only a high efficient symmetric cryptography is used in SecRout.

SEEM [38] is based on the knowledge that base stations are typically many orders of magnitude more powerful than common sensor nodes. They might have workstation or laptop-class processors, memory, and storage, AC power, and high-bandwidth links for communication amongst themselves. SEEM takes full advantage of this feature and adopts a scheme similar to the well-known Client/Server software architecture, in which server process everything and clients are only responsible for submitting requests to and displaying the responses from servers, and sometimes doing some simple computations. The authors make the clients being thin clients, i.e., clients only sends requests to and receives responses from the server and do not do any energy-consuming and computation-intensive jobs. In SEEM, the base station takes the role of the server and all sensor nodes work as clients. Base station does everything, from querying specific sensing data, broadcasting control packets, routing paths selection and maintenance to work as the interface to the outside networks, such as the Internet. Sensor nodes is only responsible for basic functions, such as sensing data, forwarding packets on behalf of other nodes and sending sensing data to the base station, leaving the energy consuming and vulnerable functions, routing paths selection and maintenance, to the base station.

SeRINS [39] is focused on detecting and isolating compromised nodes. Each node keeps multiple parent nodes in the routing tree and forwards packets through alternate paths to the one of its parent node so that packets from descendent nodes of a compromised node have an opportunity to bypass the node which arbitrarily drops them. Otherwise, every packet from the descendent nodes of a compromised node would be always forwarded via the compromised node whose position is the root of its downstream nodes as shown in Fig. 15. SeRINS copes with the problem of the compromised node, which advertises

inconsistent routing information, by the scheme, neighbor report system. Using neighbor report system, a base station gets to know which node is compromised, and then it broadcasts the information about the compromised node (e.g., the node ID, the key ring of the compromised node, etc.) to the whole network. By network-wide revocation of the cryptographic keys of the compromised node, SeRINS is able to exclude the compromised node from the network. However, the major shortcoming of those approaches is that they assume the BS is stationary and all sensor nodes know the BS's location. This assumption makes them fail to work in case the BS (sink) is mobile. Moreover, an adversary can compromise a number of nodes surrounding the base station and altered the routing information packet hop 1 to larger one (e.g. 10), thus no packet can reach the base station. Therefore, SeRINS is vulnerable from altering routing information attacks.

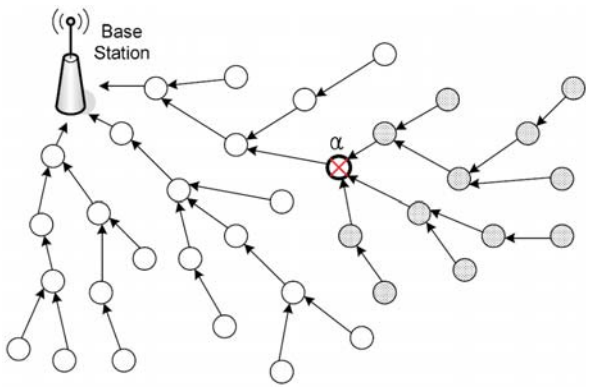


Fig. 15. Secure routing through multiple paths in SeRINS

TTSR [40] is a secure and efficient routing protocol for heterogeneous sensor networks (HSNs) which takes advantage of powerful high-end sensors (H-sensors) in an HSN. In an HSN, the BS, H-sensors and L-sensors form hierarchical network architecture. The basic idea of routing in HSNs is to let each L-sensor sends data to its cluster head (an H-sensor). An H-sensor may aggregate data from multiple L-sensors and remove redundant data, and then send compressed data to the BS via the H-sensor backbone. Transmissions in the backbone have longer range and may use a different frequency than

transmissions among L-sensors. Based on the above two layer communication architecture, TTSR is designed a secure and efficient routing protocol for HSNs, and it is referred to as Two-Tier Secure Routing (TTSR) protocol. TTSR consists of two parts: secure routing within a cluster (among L-sensors), and secure routing across clusters (among H-sensors). TTSR is only suitable for HSNs with sufficient powerful sensor nodes, not large-scale homogeneous WSNs. Besides, relying only on some particular nodes makes them prone to deplete there energy sooner or later. Using fixed coordinators makes attackers easy to choose 'right' nodes to compromise.

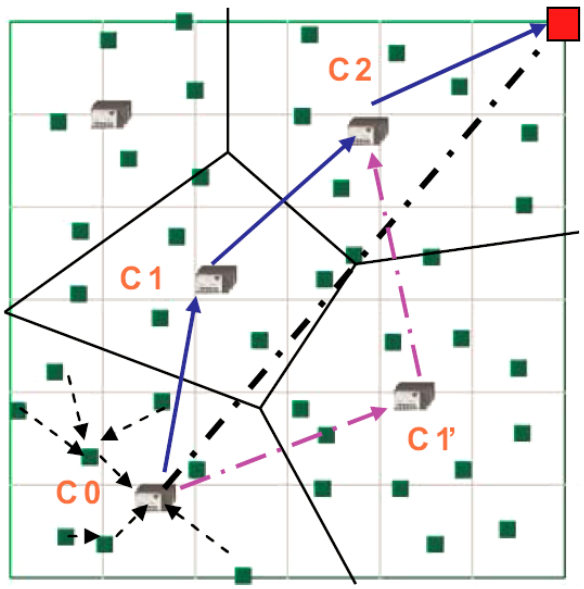


Fig. 16. Routing through powerful H-sensors in TTSR

My protocol overcomes those shortcomings by considering security and routing protocol at the design time. The scheme is based on GAF, along with a hexagonal deployment model to achieve better security and efficiency. It securely disseminates 97% data packets to the mobile sinks successfully and even more efficiently than non-secure approaches like TTDD and DD.

Chapter 3

SCODE Overview

This chapter provides an overview of my proposed protocol *Secure COordination-based Data dissEmincation for mobile sinks in sensor networks* (SCODE). Components and operations of the SCODE are first presented. Then, possible attacks and countermeasure are described.

3.1 Components and operations

SCODE comprises of three major components: secure routing, autonomous diagnosis system, and network topology management, as illustrated in Fig. 17.

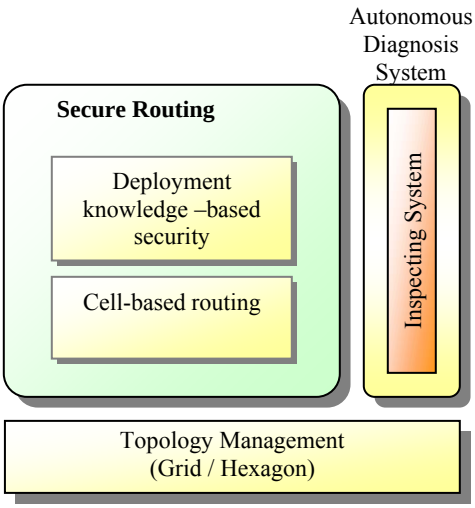


Fig. 17. Components of SCODE

The underlying component is network topology management (Section 4.4). In SCODE, the sensor network plane is partitioned in a virtual cell structure. Cells can be hexagons or squares. Each node is assigned a cell ID. With the hexagonal network topology, nodes are assigned cell ID and key material (symmetric polynomials) before deployment phase. In this case, deployment points which are center points of virtual hexagons are first determined. Each

group of sensors will be scattered to a deployment point, either by an airplane or other means. Since nodes are assigned a cell ID and key material before hand, they can participate in key exchange and cell-based routing protocol without knowledge of their physical locations (Section 4.5). With the grid network topology, nodes are aware of its location information and should detect their cell ID after deployment phase. In fact, the grid topology is a simplified version of the hexagonal topology but the security remains the same (Chapter 5).

The secure routing component includes a cell-based routing algorithm and deployment knowledge-based security. Differentiated with existing approaches, the SCODE is a cell-based routing protocol in which the routing algorithm is based on the cell ID of nodes, instead of node ID or physical location information (Section 4.6). The security of the routing protocol is achieved mainly based on my new key management scheme and message authentication code (MAC). The key management scheme is based on deployment knowledge to reduce the memory requirement and enhance security robustness. MAC is employed to partly provide resilience against many kinds of attacks in sensor network routing. It requires only 4 bytes attached with each packet to provide authentication and integrity of the packet.

In order to defend against node compromise attacks, I introduce an inspecting system which is a type of autonomous diagnosis system (Section 4.7). It provides self-defense, self-healing for sensor networks. In this system, nodes inspect each other and can detect if a node is compromised or malfunctioning. Once a compromised node is detected, it will notify to other nodes so as to eliminate that node from joining into the routing or other processes.

Fig. 18 describes the internal processes of SCODE consisting of the following steps.

(1) Key materials (polynomials) are loaded into different groups of sensor nodes.

(2) A virtual grid or hexagonal network plane is determined. Groups of sensors are scattered to deployment points so that the expected location of each node will reside within a cell. There is only one node in each in the awake state, called coordinator, to handle routing operation. Other nodes may be either in awake or sleep state to conserve their energy.

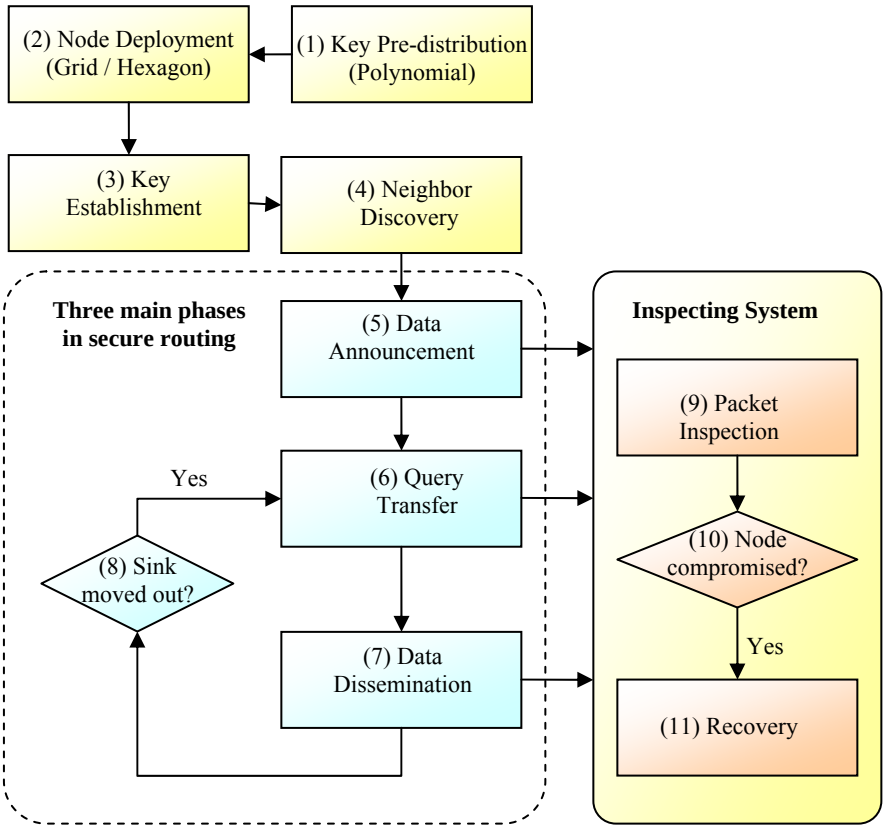


Fig. 18. SCODE workflow

(3) After deployment, nodes broadcast their ID and cell ID to setup pairwise keys. Two neighboring nodes can compute their pairwise key on their own if they share at least a common polynomial. After that, cluster keys as well as a global key are established.

(4) Then, nodes enter the neighboring discovery phase. They broadcast HELLO messages using their pairwise key. Use of the pairwise key eliminates the HELLO flood attacks in sensor networks routing.

(5) When an event (stimulus) occurs, sensor nodes surrounding it negotiate with each other, and one of them becomes a source to generate and send reports. The source broadcast data announcement messages to all coordinators. To preserver integrity of data announcement messages, they are hashed using the source's key.

(6) A user (sink) wants to query information about the event. It contacts with the closest coordinator, called agent, and sends a query message. The query message is then forwarded through coordinators based on the cell-based routing algorithm. Each intermediate node stores a routing table containing uplink nodes towards to the sink for afterward data dissemination phase. The query message is encrypted with a 4-byte MAC value to preserver its integrity. It is also for node authentication.

(7) Receiving the query, the source starts generating report and sends to the sink through the query routing path. Similar to query messages, data messages are also encrypted with the source key, and attached with a 4-byte MAC value generated by pairwise key.

(8) Periodically, the sink checks its location. If it moves out to another cell, it sends a new query again to establish a new routing path.

(9) Packets sent out from a node will be inspected by its surrounding coordinators, called inspectors.

(10) A compromised node will be detected if it is doing something wrong, such as dropping the packet, or selectively forwarding the packet.

(11) If the inspectors detect a node compromised, they will send alert message to other nodes. The alert message is encrypted by the pairwise key, so that no malicious nodes can send fake alert

information. Upon receiving the alert message, other nodes trigger the recovery phase which eliminates the compromised node from participating into routing process and cover the routing path.

3.2 Possible attacks and countermeasure

3.2.1 Outsider attacks

As mentioned in Chapter 2, attacks on sensor network routing fall into five categories: (1) spoofed, altered, replayed routing information attacks, (2) selective forwarding attacks, (3) sinkhole or wormhole attacks, (4) Sybil attacks, and (5) HELLO flood attacks. This section briefly explains how the SCODE defends against these threats from outsider attackers (Section 4.9).

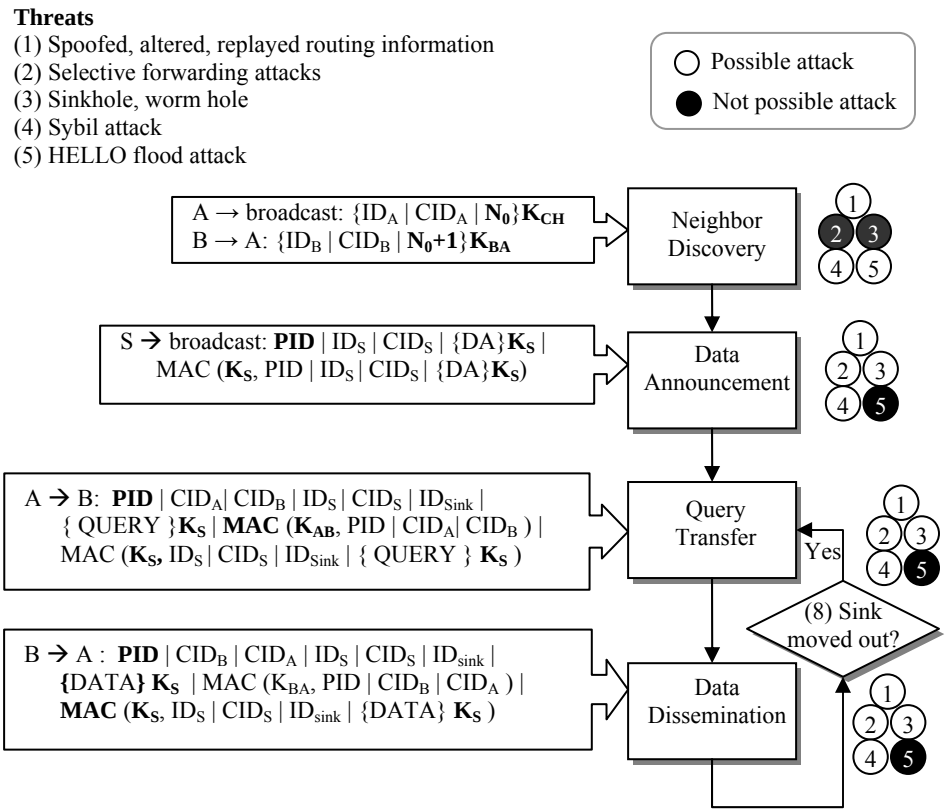


Fig. 19. Possible attacks and countermeasures

Fig. 19 illustrates the possible threats and countermeasures in the SCODE. In the neighbor discovery phase, an adversary may launch (1), (4), and (5) attacks but not (2) and (3). For other phases including data announcement, query transfer, and data dissemination, an adversary may launch (1), (2), (3), and (4) attacks but not (5). In the neighbor discovery attacks, HELLO flood attacks are defended by using a cluster key and pairwise key. In all phases, attacks using spoofed, altered, replayed routing information, sinkhole, wormhole, and Sybil attack are defended by the source key and 4-byte MAC using pairwise key. Selective forwarding attacks are eliminated by using the packet ID (PID) attached with each packet.

3.2.2 Insider attacks (node compromise attacks)

The inspecting system provides self-defense, self-healing mechanism against node compromise attacks. An adversary can compromise a number of nodes, turns them to malicious nodes, and launch any kind of aforementioned attacks. In this case, some defending mechanisms are no longer useful because malicious nodes have secret keys.

I propose an inspecting system which takes advantage of neighboring coordinators to detect if a node performs any wrongdoing. For each coordinator, there are six inspection nodes, called inspectors, which are neighboring coordinator of that node. For example, in Fig. 20 there are six inspectors A, C, D, E, F, and G around node B. Since inspectors receive the same packets with the receiver, so they can inspect the message and detect if the sender doing something wrong. Once the compromised node is detected, the inspectors send alert message to other nodes. The alert message is encrypted by the pairwise key, so that no malicious nodes can send fake alert information. Upon receiving the alert message, other nodes trigger the recovery phase which eliminates the compromised node from participating into routing process and cover the routing path.

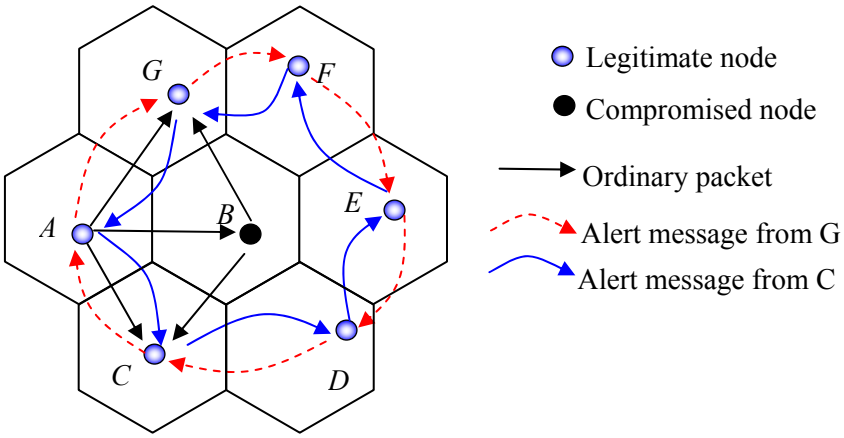


Fig. 20. Inspecting system

Chapter 4
Secure Coordination-based Data Dissemination for Mobile Sinks

4.1 Introduction

This chapter presents the proposed protocol, *Secure COordination-based Data dissuEmination* protocol for mobile sinks, called SCODE [42]. SCODE is a secure and energy-efficient routing protocol supporting sink mobility in sensor networks.

In the SCODE, the network topology is predetermined. Key materials (symmetric polynomials) are preloaded into each group of sensors before deployment. After deployment, they exchange their IDs with each other to establish a key infrastructure. They also negotiate with each other so that only one node within a cell stays awake, called coordinator, to handle routing while the others fall into sleeping mode for the sake of energy saving. The SCODE is based on *Geographical Adaptive Fidelity* (GAF) [6] to establish such a coordination network. The routing protocol takes advantage of off-the-shelf security primitives (e.g. HMAC, TEA) and the proposed inspecting system to provide security robustness.

In the following sections, GAF is first briefly described. Then the SCODE protocol will be presented including the node deployment mechanism, key distribution scheme, secure routing protocol, and the mutual inspecting system.

4.2 Geographical Adaptive Fidelity (GAF)

In SCODE protocol, sensor nodes within a cell periodically negotiate among each other to elect the coordinator in every round. For each round, only one

node stays active to be a coordinator, while the others fall into sleeping mode. Doing this significantly reduces the energy consumption because nodes in the idle state spend much more energy as compared with the sleeping state. Analysis in [66] has shown that energy consumption ratio for *sleep:idle:receive:transmit* is 0.13:0.83:1:1.4. It also reduces the network congestion because the number of nodes participating in transmission/reception is decreased. On the other hand, frequent change of coordinator role helps the particular nodes not running out of its energy quickly. Therefore, it can prolong nodes as well as the network lifetime. Geographical Adaptive Fidelity (GAF) protocol [6] is employed in order to control nodes in different states and transition.

The GAF conserves energy by identifying nodes that are equivalent from a routing perspective and then turning off unnecessary nodes, keeping a constant level of routing fidelity.

4.2.1 Determining node equivalence

Even with location information, it is not trivial to find equivalent nodes in a network. Nodes that are "equivalent" between some nodes may not be equivalent for communication between others.

To address this issue, the GAF uses location information and virtual grids to determine node equivalent. Two nodes are equivalent if they locate in the same virtual cells. The size of each virtual cell is determined based on the nominal radio range R. Assume that the virtual grid is a square with r units on a side as show in Figure 1. In order to meet the definition of virtual cell, the distance between two possible farthest nodes in any two adjacent cells, must not be larger than R. For example, $node_1$ and $node_5$ in Fig. 21 are at the end of the long diagonal connecting two adjacent cells. Therefore, we get $r^2 + (2r)^2 \leq R^2$ or $r \leq R/\sqrt{5}$.

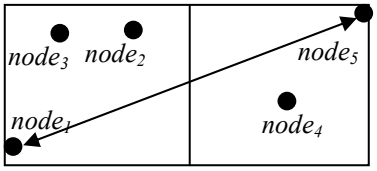


Fig. 21. Example of virtual grid in GAF

4.2.2 GAF state transitions

In the GAF, nodes are in one of three states: sleeping, discovery, active. A state transition diagram is shown in Fig. 22.

Initially a node starts out in the discovery state, in which it turns on its radio and exchanges discovery message to find other nodes within the same cell. The discovery message is a tuple of node ID, cell ID, estimated node active time (*enat*), and node state. As described above, a node uses its location and cell size to determine the cell ID.

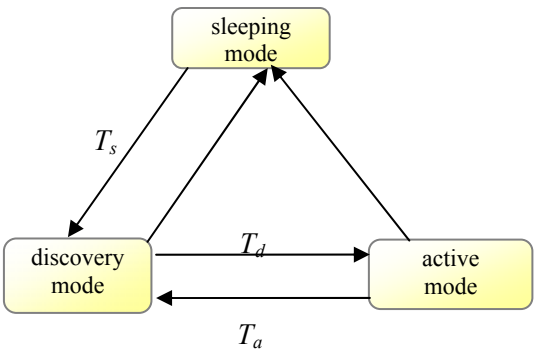


Fig. 22. State transitions in GAF

When a node enters discovery state, it sets a timer for T_d seconds. When the timer fires, the node broadcasts its discovery message and enters state active. The timer can also be suppressed by other discovery messages. This timer reduces the probability of discovery message collision. When a node enters active, it sets a timeout value T_a to define how long this node can stay in active state. After T_a, the node will return to the discovery state. While active, the node periodically re-broadcast its discovery messages at an interval of T_d.

A node in discovery or active states can change state to sleeping when it can determine some other equivalent nodes will handle routing. When transitioning to sleeping, a node cancels all pending timers and powers down its radio. A node in the sleeping state wakes up after an application-dependent sleep time T_S and transitions back to discovery.

The GAF leaves choices of many parameters including *enat*, T_d, T_a, node rank, T_s to application. Applications may wish to optimize these choices, for example, perhaps trading increased packet loss for greater energy savings. enat can be set to the expected node lifetime, conservatively set by assuming the node will constantly consume energy at a maximum rate until it dies. GAF chose T_d as a uniform random value between 0 and some constant. This approach avoids contention from synchronized discovery message. The GAF uses T_a to accomplish load balancing. Node ranking in GAF is chosen to maximize network lifetime by selecting which nodes handle routing. Rank is determined by several rules. For example, nodes with longer expected lifetime higher rank. GAF employs a load balancing strategy so that all nodes remain up and running together for as long as possible.

4.3 Assumptions and notations

SCODE is based on the following assumptions:

(1) Due to resource constraints, sensor nodes are not equipped with tamper-resistant hardware. If an adversary successfully compromises a sensor, then she can obtain all key material, data, and code stored on that node.

(2) Sinks are powerful nodes, moving within the sensor network field. Sinks are also aware of its location. Since sinks are mobile, sensors cannot know sink locations.

(3) Sinks are trusted.

Table 1 provides notation descriptions which will be used through out the paper.

Table 1 Notations

Notation	Description
ID_A	Identification of node A
$CID_A(X,Y)$	Cell identification of node A, indicating X and Y axes of the cell
PID	Packet Sequence Number (packet ID)
R	Radio range of a sensor node
r	Cell size
K_A	A secret key held by node A
K_{AB}	A shared key between A and B.
MAC (*K, M*)	Message authentication code of message M using a symmetric key *K*
{*M*}K	Message M is encrypted with a key *K*
N_0 , N_1	*Nonces*, one-time random number generated by nodes
A → broadcast: *M*	Node *A* broadcasts a message *M*
A → *B*: *M*	Node *A* sends a message *M* to node *B*

4.4 Hexagonal network deployment

4.4.1 Deployment model

In my proposal, the target area is divided in a hexagonal grid. This model is practical in realistic scenarios, when sensor nodes in each group are delivered together, such as using aircraft to drop groups in sequence, so expected adjacent groups have better chance of being close to each other on the ground. Based on different deployment methods, the deployment distributions follow some specific probability distribution functions (*pdf*). The *pdf* may be a uniform distribution [48] or two-dimensional *Gaussian* [49] distribution. In this paper, for the sake of simplify in analysis, I use *Gaussian* distribution, which is also widely studied and used in practice. Other distributions could be applied as well.

Supposed a sensor networks contains *N* nodes, which is split into *G* groups, each group is distributed following *Gaussian* distribution. When the

deployment point of group G_i is at (x_i^o , y_i^o), the *pdf* for a node n_i belongs to group G_i is as follows:

$$f(n_i(x_i,y_i)\,|\,n_i \in G_i) = \frac{1}{2\pi\sigma^2} e^{-[(x_i-x_i^o)^2+(y_i-y_i^o)^2]/2\sigma^2} = f(x_i - x_i^o, y_i - y_j^o) \quad (5)(1)$$

where (x_i,y_i) is the coordinate of node n_i in the group G_i and σ is the standard deviation of the distribution. The value of σ depends mainly on the height of aircraft when dropping sensor groups. I define a cluster is a set of three adjacent groups and there are three types of cluster. At any group $G_{i,j}$, there are 1-cluster containing $G_{i,j-1}$ and $G_{i-1,j}$, 2-cluster containing $G_{i,j+1}$ and $G_{i-1,j+1}$, and 3-cluster containing $G_{i+1,j}$ and $G_{i+1,j+1}$. In local area, each cell follows two-dimensional normal distribution, but the distribution on whole target field is nearly uniform. The hexagonal group-based deployment model could be seen in Fig. 23. For example, from group (2, 2) in Fig. 23, group (3, 2) and (3, 3) belong to *1-cluster* (2, 2). Group (2, 1) and group (1, 2) belong to *2-cluster* (2, 2). And *3-cluster* (2, 2) consists of group (2, 3) and group (1, 3). Therefore, each group consists of three cells, and every cell belongs to three groups.

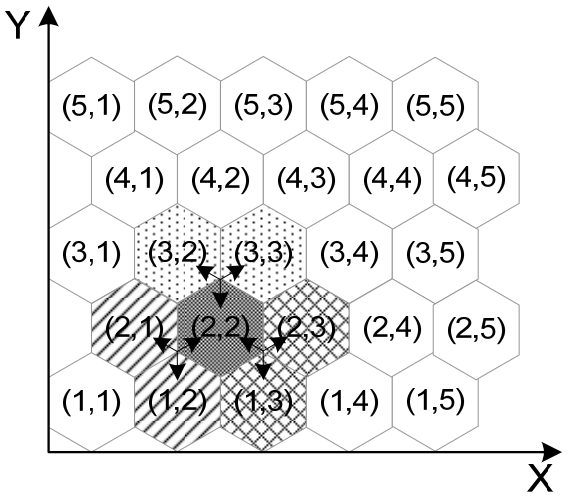


Fig. 23. Hexagonal group-based deployment model

4.4.2 Cell size determination

The cell size strongly effects to routing hops. If the cell size is small, then the number of transmission hops between the source and sink increase. However, it the cell size is too large, then two nodes within the adjacent cells cannot communicate, and thus make the protocol failed. Therefore, the cell size must be selected so that it guarantees both connectivity and efficiency. In SCODE, for two nodes within two adjacent hexagons can communicate with each other, the cell size must be determined so that $d_1 \leq R$ and $d_2 \leq R$ (Fig. 24).

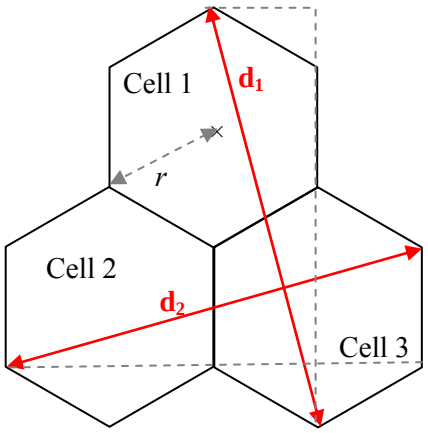


Fig. 24. Nodes within adjacent cells must be connected

I define r as the radius of the hexagon. So:

$$d_1 = r.5/\sqrt{2};\ d_2 = r.\sqrt{13}$$

Since $d_2 \leq d_1$, I just need $d_1 \leq R$. Therefore:

$$d_1 = r.5/\sqrt{2} \leq R \Leftrightarrow r \leq R\sqrt{2}/5$$

So the radius r of the hexagon must be satisfied $r \leq R\sqrt{2}/5$ to guaranty connectivity of every two neighbors. For efficiency, r should be equal to $R\sqrt{2}/5$. In conclusion, the cell size is decided as follows

$$r = R\sqrt{2}/5$$

4.5 Key distribution scheme

4.5.1 Introduction

Key management is an essential cryptographic primitive upon which other security primitives are built. Most security requirements, such as privacy, authenticity, and integrity, can be addressed by building upon a solid key management framework. In fact, a secure key management scheme is the prerequisite for the security of these primitives, and thus essential to achieving secure infrastructure in WSNs.

In this section, I will explain the proposed key management scheme that exploits hexagonal deployment knowledge to achieve more secure and efficient compared with existing ones. The key management scheme is an improvement of Blundo‘s scheme [50] because Blundo’s scheme is not able to apply directly to WSNs due to its memory overhead for storing keys. The proposed scheme is targeted to SCODE. However, it can be suitable for other uses.

4.5.2 Related work

The first scheme is proposed by Eschenauer and Gligor [51]. In this system, a large key pool is generated off-line and each sensor picks a random subset of keys from the key pool. Any two nodes in the communication range can talk to each other only if they share a common key. Depending on the size of the key pool and the number of sensor nodes in the network, this design may achieve different connectivity and resilience.

Perrig and Song [52] later proposed an approach using the similar idea, but increased the intersection sharing keys between key-rings from one key to some $q > 1$ keys. It is shown that, by increasing the value of q, network resilience against node capture is improved. Du, Deng, Han and Varshney suggested a key predistribution model by applying deployment knowledge [53]. In their design, entire network was divided into groups. Each group implements the basic

random key predistribution as in [51]. The key pool of a group shared α keys with horizontal groups' key-pools and β keys with diagonal groups' key-pools.

The key-matrix solutions are based on the idea of Blom [54]. He recommended a key predistribution scheme making certain that any pair of members in a group is able to calculate the common sharing key. Denote *N* is the number of sensor nodes in the network, let G be a generator matrix of size (t+1)×N over finite field F_q and let D be a secret random matrix (t+1)×(t+1) with elements in F_q. From the matrix G and D, construct a N×N symmetric K whose entries will be the pairwise keys between nodes. The matrix K is equal to $K = (D \cdot G)^T \cdot G$. Each node *i* stores a corresponding row *i* of private matrix $A = (D \cdot G)^T$. If node i want to communicate with node j, then it computes the inner product of row vector it stores with the j-th column of G to obtain the common key $K_{i,j}$. Multiple-space key predistribution of Du, Deng, Han and Varshney [7] combined the Blom's method with the basic random key predistribution of Eschenauer and Gligor [51] for applying to sensor networks. In this approach, they denoted the set of keys that each tuple <*D,G*> can generate a key space. Each node in the network stored randomly τ spaces from ω pre-generated spaces. Based on probabilistic, any two nodes could share a common space, which may compute a common secret key. Later, Du, Deng, Han and Varshney also applied pre-deployment knowledge to propose DDHV-D scheme in [55]. It is the combination of multiple-space key predistribution [56] with the random predistribution scheme applied deployment knowledge [55] All the key-matrix solutions have threshold t-secure property. It means that no more than t nodes are compromised by attackers then the communications between non-compromised nodes are still secured.

The basic idea of polynomial key generation was proposed by Blundo et al [50]. It uses symmetric polynomial evaluations to obtain a pairwise key. The detail of this method will be described in the next section. This proposal is t-collusion resistant against node captured with property: compromise of less than

t+1 node doesn't reveal any information about keys of other nodes. Derived from above method and basic random key predistribution [51], Liu and Ning introduced random subset assignment key predistribution model [57]. Instead of generating large key-pools and creating key-rings, this scheme creates a large polynomials pool and assigned each node a subset of polynomials from the pool. Then two nodes can only communicate to each other when they shared at least one common polynomial. It is shown that this solution increased the resilience comparing with Eschenauer and Gligor's model [51]. Further solution using predeployment knowledge is Closet Polynomials Predistribution Scheme (CPKS) of Liu and Ning [58][59]. Most of these solutions indicated the trade-off between security and performance.

4.5.3 Blundo's key predistribution scheme

Blundo's scheme in [50] uses *n* variables polynomials with *t-degree* to establish key distribution for *t-secure n-conference*. Applied to pairwise key between two entities, key predistribution server randomly generates a bivariate *t-degree* polynomial $f(x,y) = \sum_{i,j=0}^{t} a_{ij} x^i y^j$ over a finite field F_q, where q is a large enough prime number that could accommodate a cryptographic key. The function $f(x,y)$ is symmetric meaning that $f(x,y) = f(y,x)$. Each node having unique integer ID i loads the information of $f(i,y)$ from the polynomial $f(x,y)$. Then any two nodes i and j can compute the key $k_{i,j} = f(i,j)$ at node i and $k_{j,i} = f(j,i)$ at node j. Because of symmetric property, I have $k_{i,j} = k_{j,i}$ so that two nodes have a common pairwise key.

Each node must store $t+1$ coefficients, each coefficient costs $\log_2 q$ bits. So the memory storage requirement for each node in this model is $(t+1)\log_2 q$ bits. The analysis in [50] shows that, this scheme is unconditionally secure and t-collusion resistant. It means that as long as no more than t nodes are

compromised, the attacker knows nothing about the pairwise key between any two non-compromised nodes.

This basic proposal is not able to apply directly to sensor networks due to its memory overhead for storing keys. The size of memory depends exponentially on the size of the network, so it is not useful for such resource-constraint devices like sensor nodes using only this model. I will focus on this problem by using predeployment knowledge and showing that it will take more advantages than other polynomial-based schemes applied expected location knowledge.

4.5.4 Proposed key management scheme using hexagonal deployment knowledge

I define a key-space as derived from a bivariate polynomial in Blundo's scheme. A node N_A picks a key-space $f_{u,v}(x, y)$ if it carries the coefficients of $f_{u,v}(N_A \oplus nonce_A, y)$, where $nonce_A$ is a random value of node A and will be described in the *key predistribution* phase. When two nodes are in the same key-space, they could calculate a pair-wise shared key to setup a secure channel.

The proposed scheme allows sensor nodes to find a common key space with each of their neighbors after deployment. It has totally three phases: *key predistribution*, *direct key establishment*, and *indirect key establishment*. The key predistribution phase is carried out to preload the credential information to each sensor node before deployment. After setting up, two sensor nodes can establish a direct key between them if they share at least a common key-space, otherwise, they could agree on an indirect key according to the indirect key establishment phase.

(a) *Key Predistribution Phase*

The purpose of this phase is to assign key materials to each node. Based on these key materials, neighboring nodes could setup pair-wise keys after deployment.

This task is done by an offline server. At first, the server will generate a polynomial pool $\mathcal{F}$ containing enough *t-degree* symmetric bivariate polynomials for every cluster. Then it distributes each polynomial to all sensor nodes in each cluster. Because each cell belongs to three clusters, every node has to store knowledge of three *t-degree* bivariate polynomials. In other words, each node needs to pick three key-spaces. The detail algorithm for polynomials predistribution is shown in Fig. 25.

After finishing the phase, every sensor node stores node IDs, three space IDs, random values, and three vectors of coefficients equivalent to three key-spaces. These key materials will be used to setup pair-wise keys in the next phase.

Algorithm 1: Polynomials predistribution

Data: set of nodes $\mathcal{N}$ in the network, set of clusters Ψ, polynomial pool $\mathcal{F}$

Result: Preload key materials to every sensor nodes in the networks

1 **foreach** *sensor node* N_A *in* $\mathcal{N}$ **do**
2 Generate and insert to preloaded data of N_A a random value $nonce_A$;
3 **end**
4 **foreach** *cluster* C_i *in* Ψ **do**
5 Get a bivariate polynomial $f_i(x, y)$ from $\mathcal{F}$;
6 **foreach** *node* N_A *in* C_i **do**
7 Compute $f_i(N_A \oplus nonce_A, y) = \sum_{j=0}^{t} b_j y^j$;
8 Insert to preloaded data of N_A: $\{b_j \| j = 0, .., t\}$; Insert id of this polynomial, called space-id f_i to N_A,
9 **end**
10 Remove $f_i(x, y)$ from $\mathcal{F}$;
11 **end**

Fig. 25. Polynomial predistribution

(b) *Direct Key Establish Phase*

After deployment, every sensor node discovers the sharing key-space with its neighbors. Assume that node N_A with three space IDs f_i, f_j, f_k needs to discover shared key-space with its neighbors. It broadcasts a 1-hop discovery message *Key-Space Discovery Message* (KSDM) containing the following information:

$$N_A, nounce_A, H(f_i \oplus nounce_A), H(f_j \oplus nounce_A), H(f_k \oplus nounce_A)$$

where H is the hashing function and $\oplus$ is the XOR operation.

When a neighbor of A, let's call B, receives this message, it finds out that it could share three, one or no common key-space with A. Similarly, node A also receives B's KSDM message and finds out common key-spaces. If the sharing is at least one common key-space, the pair-wise key between B and A is calculated at B as follows:

$$K_{BA} = f(N_B \oplus nounce_B, N_A \oplus nounce_A) \quad (6)$$

After getting K_{BA}, node B deletes value $nonce_A$ from its memory. The process of computing pair-wise key at A is similar. Because of the symmetric property of bivariate polynomials, $K_{AB} = K_{BA}$. After this phase, every node stores a list of pair-wise keys with its neighbors, beside the key-space information and a random value in previous phase.

(c) *Indirect Key Establishment Phase*

In case there is no common key-space between two neighboring nodes, it is needed to establish a path key through one or more intermediate nodes. My solution for this problem is as follows.

After the *direct key establishment* phase, every node A knows a set of secure neighboring nodes, denoted as S_A. Node A wants to establish a pair-wise shared key with its neighbor B, but B and A do not share any key-space. In this situation, A generates a session key, called K_S, and find in S_A a node C that have the same group ID with node B or neighboring group ID of group containing node B. Node A then sends a message containing K_S encrypted by key K_{AC} to node C. In turn, node C sends to B a session key through a secure channel protected by the key K_{CB}. The key K_S then is used as pair-wise shared key between two nodes A and B.

After above three phases, every node stores a table containing neighbor IDs and pair-wise shared keys equivalently. The existence of key materials allows sensor networks to be able to add new nodes for replacement later.

(d) *Establishing/Revoking Keys of New/Existing Sensors*

To add a new sensor, the key setup server only needs to predistribute the related polynomial shares to the new node, similar to predistribution phase. Since the size of key-space is limited, the more sensors are added, the lower the security in that cell becomes.

The revocation method is also straightforward. Each sensor node only needs to store a black list IDs of compromised sensors that share at least one bivariate polynomial with itself. If there are more than t compromised nodes sharing the same polynomial, the non-compromised nodes that have this polynomial will remove this polynomial and all related compromised nodes.

4.6 Secure routing protocol

After deployment and key predistribution phase, nodes maintain three types of keys: unique individual key K_A that each node A shares with a sink; a cluster key K_{CH} shared among all nodes in the same cell; and a pair-wise shared key K_{AB} shared between a node/sink A and its neighbor B (Node B is defined as a neighbor of node A if and only if B coexists in the same or adjacent cell of A).

The proposed secure routing protocol includes secure neighboring discovery phase, data announcement phase, query transfer and route establishment phase, and data dissemination phase. In the discovery phase, nodes broadcast HELLO message to discover their neighborhoods using the cluster key and pairwise key. Those keys are used to defend against HELLO flood (unidirectional) attacks. The other phases are main phase of routing algorithms. The security is mainly based on pairwise key and the cluster key along with HMAC and lightweight cipher algorithm.

4.6.1 Secure neighboring discovery

After deployment, each node, say *A*, broadcasts a HELLO message in order to discover its neighborhoods. This message is encrypted by A's cluster key K_{CH}:

$$A \rightarrow broadcast: \{ID_A \mid CID_A \mid N_0\}K_{CH}$$

Each receiving node *B* decrypts the message and checks if it is a neighbor of *A*. If yes, it replies to *A* node ID and cell ID along with a nonce value N_0 encrypted by the pair-wise shared key K_{AB}.

$$B \rightarrow A: \{ID_B \mid CID_B \mid N_0+1\}K_{CH}$$

Receiving node *A* decrypts the message using the pair-wise shared key. It then checks if the nonce N_0 is the one it has broadcasted. If it is, A accepts B as a neighbor and updates its neighborhood table.

The above two-way handshake protocol can avoid (or defend against) the unidirectional link problems (or attacks). For example, if an attacker uses node *A* which is a more powerful node such as a laptop with longer transmission range than *B*, then *A* can send a message to *B* directly, but *B* cannot send a message to *A* in one-hop. However, node *B* still thinks that *A* is a one-hop neighboring node, and various problems may arise, for example *B* will not relay messages to its neighbors but *A*, consequently these messages will be dropped.

4.6.2 Three main phases

(a) *Phase 1: Secure data announcement*

When a stimulus is detected, a source *S* propagates a *Data-Announcement* (DA) message to all coordinators using a flooding mechanism. The message contains source ID, cell ID and a MAC:

$S \rightarrow broadcast$:

$$PID \mid ID_S \mid CID_S \mid \{\text{DA}\}K_S \mid \text{MAC}\,(K_S, PID \mid ID_S \mid CID_S \mid \{\text{DA}\}K_S)$$

Every coordinator stores a few piece of information for the route discovery, including the information of the stimulus and its cell ID. Since the coordinator role might be changed every time, the cell ID is the best solution for nodes to know the target it should relay the query to. To avoid indefinite storing data-announcement message in each coordinator, the source attaches a timeout parameter in each message. Within the timeout interval, if the coordinator has not received any further data-announcement message, it removes the information of the stimulus and the source location to free the cache.

(b) Phase 2: Secure Query Transfer and Route Discovery

Receiving a data-announcement message, a sink looks up in its $\{(ID_{node}, K_{node})\}$ table, finds the key K_S shared with the source, and uses it to decrypt the message. It then constructs a query message using K_S and sends back to the source via coordinators as follows:

- The sink first sends to its agent (which is the closest coordinator):

Sink → Agent:

$PID \mid CID_{This} \mid CID_{Next} \mid ID_S \mid CID_S \mid ID_{Sink} \mid \{ \text{QUERY} \} K_S \mid \text{MAC} (K_{SinkAgent}, PID \mid CID_{This} \mid CID_{Next}) \mid \text{MAC} (K_S, ID_S \mid CID_S \mid ID_{Sink} \mid \{\text{QUERY}\} K_S)$

where CID_{This} is cell ID of the agent, and CID_{Next} is cell ID of the next cell (in this case $CID_{This} = CID_{Sink}$, and $CID_{Next} = CID_{Agent}$). The source key K_S is used to encrypt the query content. It is also used to build a *MAC* of ID_S, CID_S, and ID_{Sink} in order to provide data authentication and data integrity of the information sent from the sink. The pair-wise shared key $K_{SinkAgent}$ is used to build a *MAC* of *PID,* CID_{This}, and CID_{Next} in order to provide data authentication and data integrity of the packet sent from the current node.

- The agent computes next cell ID (which is the closest cell to the destination) towards the source, and then forwards the packet to it.

Agent → *Nextcell*:

$PID \mid CID_{This} \mid CID_{Next} \mid ID_S \mid CID_S \mid ID_{Sink} \mid \{ \text{QUERY} \} K_S \mid \text{MAC}$

($K_{AgentNextl}$, PID | CID_{This}| CID_{Next}) | MAC (K_S, ID_S| CID_S | ID_{Sink} | { QUERY } K_S)

Receiving coordinator checks whether its cell ID is the same as CID_{Next} in the message. If yes, it computes next cell ID, and relays the message to it. Each node maintains a routing table which stores the departure cell ID, the destination cell ID, the uplink cell ID (which it receives the message) and the downlink cell ID (which is next cell ID CID_{Next}).

In SCODE, each node can compute the next cell to which it relays the packet. The algorithm is solely cell-based algorithm, which relies on the cell ID instead of node ID or node's location information.

During the *secure neighboring discovery* phase (Section 4.6.1), each coordinator recognizes its neighbors as well as their cell IDs. Also, each coordinator determines which cell is void (no node in that cell). When a node receives a query message, it selects among non-void adjacent cells the closest one to the destination. The selection can be easily carried out based on distance between the centers of cells, regardless the location of the coordinator belong to that cell.

An example is illustrated in Fig. 26, $sink_1$ selects the coordinator in cell [4, 2] as its agent. The coordinator [4, 2] maintains a list of neighboring coordinators {[4, 3], [3, 2], [4, 1]}. When it receives the query from the $sink_1$ targeting to the source in cell [1, 3], it computes and compares the distances among d_1([4, 3], [1, 3]), d_2([3, 2], [1, 3]), and d_3([4, 1], [1, 3]). Since d_2 is the smallest, it selects [3,2] as a next cell to relay the packet. Likewise, the query is finally forwarded to the source along the path {$sink_1$, [3, 2], [3, 3], [2, 3], [1, 3], S}.

SCODE is a cell-based routing protocol to flexibly handle routing operation based on a dynamic coordination network. A coordinator role can be changed to any node within the cell. In such a dynamic scenario, cell ID would be the best choice to find a routing path, rather than node ID or node location. For that reason, the protocol trades the optimum to energy conservation and flexibility.

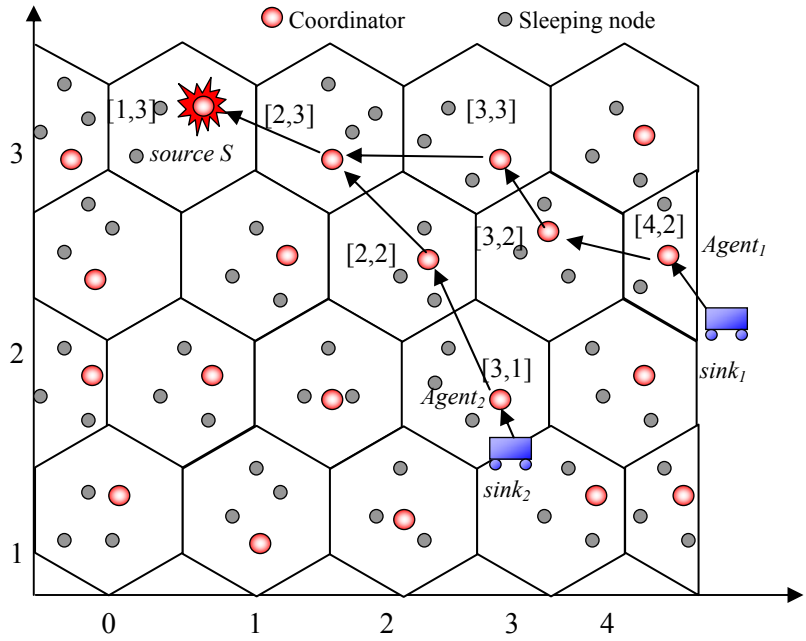


Fig. 26. Routing in the SCODE

(c) *Phase 3 - Secure Data Dissemination*

When the source receives a query from a sink, it starts generating and transmits data. The data message is encrypted by the source key K_S. The source first sends to its uplink node A:

$S \rightarrow A$:

$$PID \mid CID_{this} \mid CID_{Next} \mid ID_S \mid CID_S \mid ID_{sink} \mid \{\text{DATA}\}\ K_S \mid MAC\ (K_{SA}, PID \mid CID_{this} \mid CID_{next}) \mid MAC\ (K_S, ID_S \mid CID_S \mid ID_{sink} \mid \{\text{DATA}\}\ K_S)$$

where $\text{CID}_{\text{this}} = \text{CID}_{\text{S}}$, and $\text{CID}_{\text{next}} = \text{CID}_{\text{A}}$.

Receiving a data packet, node B checks whether the data packet is forwarded to it or not by comparing its cell ID with CID_{Next} in the packet. If not, it keeps the packet for a short time for inspecting purpose before dropping it (see Section 4.7). Otherwise, it computes next cell ID, changes PID, CID_{this}, CID_{next} and compute a new MAC to replace MAC (K_{SA}, $PID \mid CID_{this} \mid CID_{next}$). After that, it relays the packet to the next cell.

When the message reaches the sink, the sink verifies MAC value in the message to ensure its data authentication and integrity. It then decrypts the DATA by using the source key K_S.

4.6.3 Handling sink mobility

Fig. 27 illustrates how SCODE handles sink mobility during routing. Periodically, a sink checks the distance between itself and its agent. If it recognizes that it has move out of transmission range of the agent, it has to compute its new cell ID and selects the coordinator in that cell as its new agent. The cell ID is computed by the formula (1). Then, the sink re-sends a query to the source to establish a new data dissemination route. A new data dissemination route is established by using the same mechanism described in Section 4.6.2.

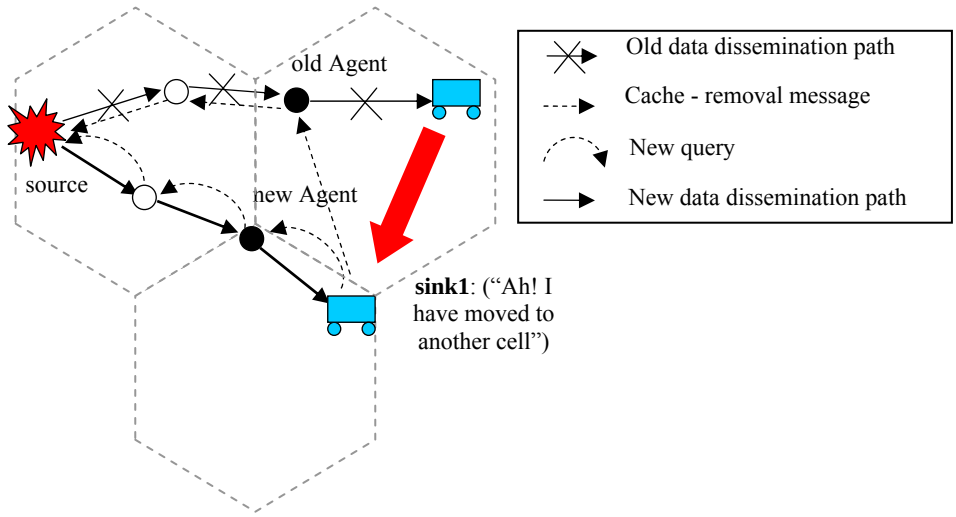


Fig. 27. Handling sink mobility in SCODE

By re-sending a new query only when the sink moves out of the agent’s transmission range, SCODE reduces significantly the number of queries compared with other approaches. Hence, collision and energy consumption is reduced. Also, the number of loss data packet is decreased. In case the sink moves into a void grid, it selects the closest coordinator to act as its agent

4.6.4 Coordinator election

As aforementioned, the proposed scheme is based on GAF to establish a coordination network. During discovery phase, a node with the highest ranking will stay awake and play a role of a coordinator while the others fall into sleeping mode. The node ranking is determined by an application-dependent ranking procedure (it can be an arbitrary ordering of nodes to decide which nodes would be active, or it can be selected to optimize overall system lifetime). In GAF, a node with a longer expected lifetime is assigned a higher rank. This rule put nodes with longer expected lifetime into use first. However, this is very vulnerable. An adversary can use a compromised node to advertise the highest rank so that this node can be active all the time.

Practically, it is very hard (perhaps not possible) to absolutely defend against node compromise attacks for GAF. Therefore, I modify the discovery process and the ranking rule of GAF in order to reduce the risk without lessening its advantages. A coordinator is orderly selected according to node ID at each discovery round. For example, supposed there are m nodes $\{ID_1, ID_2, ..., ID_m\}$ ($ID_1 < ID_2 < ... < ID_m$) in a certain cell, if node ID_i is a coordinator at the current round, then ID_{i+1} would be a coordinator at the next round; if $i = m$ then "$i + 1$" would be 1. By doing this way, the adversary cannot make her node active all the time at will. The compromised node is active only if it is a coordinator at the given round. On the other hand, my enhancement reduces communication overhead compared with GAF since nodes do not need to broadcast discovery messages.

4.7 Inspecting system

One of the most concerns in sensor network security is node compromise attacks. Due to resource limitation, sensor nodes are not equipped with any tamper-resistant hardware. Once a node is compromised, an adversary can extract all information stored in that node including all key materials. After a legitimate node is compromised, it becomes a malicious node. She then can use

this node to perform various types of attacks to the networks. Defending against node compromise in routing is a non-trivial task.

Therefore, an *inspecting system* is proposed to detect if a node performs any wrongdoing. For each node, there are six inspectors which are its neighboring coordinators. For example, in Fig. 28 six inspectors of B are *A, C, D, E, F*, and *G*.

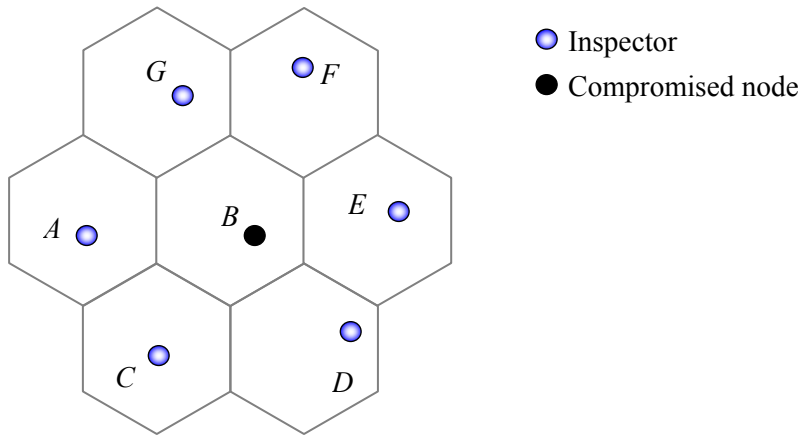


Fig. 28. Six neighboring coordinators of B play a role of inspectors on B

4.7.1 Inspection

a) *A malicious node altered message information*

During query transfer and data dissemination phase, an adversary may use a compromised node to modify the message, then forward to a wrong next-cell.

The query transfer message has the following format:

$$\begin{aligned} A \rightarrow B:\ & PID \mid CID_A \mid CID_B \mid ID_S \mid CID_S \mid ID_{Sink} \mid CID_{Sink} \mid \{QUERY\}K_S \\ & \mid MAC\ (K_{AB},\ PID \mid CID_A \mid CID_B\ ) \\ & \mid MAC\ (K_S,\ ID_S \mid CID_S \mid ID_{Sink} \mid \{\ QUERY\ \}\ K_S\ ) \end{aligned}$$

And the data dissemination message has the following format:

$$\begin{aligned} A \rightarrow B:\ & PID \mid CID_A \mid CID_B \mid ID_S \mid CID_S \mid ID_{sink} \mid CID_{Sink} \mid \{DATA\}K_S \\ & \mid MAC\ (K_{AB},\ PID \mid CID_A \mid CID_B) \\ & \mid MAC\ (K_S,\ ID_S \mid CID_S \mid ID_{sink} \mid \{DATA\}\ K_S\ ) \end{aligned}$$

Since the structure of query transfer message is similar to data dissemination message, I only discuss the countermeasure on the data dissemination message. An example is shown in Fig. 29, upon receiving message from node A, malicious node B modifies routing information such as ID_S, CID_S, ID_{Sink}, CID_{Sink} in the message and sends to E.

$$\begin{aligned} B \rightarrow E:\ & PID \mid CID_B \mid CID_E \mid ID_S^* \mid CID_S^* \mid ID_{sink}^* \mid CID_{Sink}^* \mid \{DATA\}K_S \\ & \mid MAC\ (K_{BE},\ PID \mid CID_B \mid CID_B) \\ & \mid MAC\ (K_S,\ ID_S \mid CID_S \mid ID_{sink} \mid \{DATA\}\ K_S\ ) \end{aligned}$$

When B sends the message, G and C also receive the same message as E does. They check the previous message that they received when A sends B. If either ($ID_S \neq ID_S^*$), ($CID_S \neq CID_S^*$), ($ID_{sink} \neq ID_{sink}$), or ($CID_{Sink} \neq CID_{Sink}^*$) is true, then G and C regard B as a compromise node.

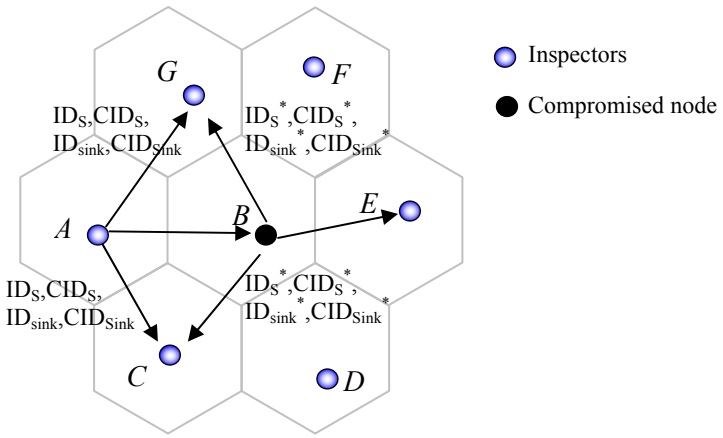


Fig. 29. Malicious node modifies the source/sink information in the message

b) *A malicious node changes the next cell ID so that the message forwarded to a wrong route*

The compromised node *B* attempts to modify the next cell ID in the message. For example, instead of placing CID_E in the message and forwarding it E, B inserts CID_F as follows:

$$B \rightarrow F: \ PID \mid CID_B \mid CID_F^* \mid ID_S \mid CID_S \mid ID_{sink} \mid CID_{Sink} \mid \{DATA\}K_S$$
$$\mid MAC\ (K_{BF}, PID \mid CID_B \mid CID_B)$$
$$\mid MAC\ (K_S, ID_S \mid CID_S \mid ID_{sink} \mid \{DATA\}\ K_S\ )$$

Since G and C also received the same message from A as B did, so they can compute the correct next cell ID the B must forward to, which would be E. If B attempts to change CID_E to CID_F, it can be detected by C and G.

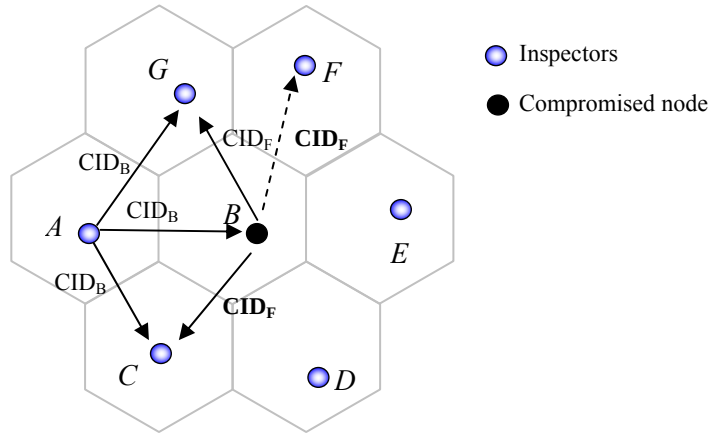


Fig. 30. Malicious node modifies the next-cell ID information in the message

4.7.2 Recovery

a) *Node compromise notification*

Once the compromised node is detected, the inspectors alert to other coordinators and eliminate that node from participating in the routing process. Coordinators consider that cell a void cell and establish another route by finding a round path. For example in Fig. 31, supposed the coordinator *A* is legitimate node, and *B* is a compromised node. Node *C* and *G* are common inspectors of *A* and *B*, so they can receive the message sent out from *A* and *B* and can detect if *B* is doing something wrong. When *C* and *G* detect *B* as a compromised node, they send an alert message to the neighboring coordinator of *B* including *A, D, E,* and *F*. The alert message sent from *C* will go through *C→D→E→F→G→A→C*. The alert message sent out from *G* will go through *G→F→E→D→C→A→G*. This duplicated alert message makes sure that the

compromised node is detected by two inspectors. The alert message is encrypted by the pairwise key between the sender and the receiver. For example, *G* sends an alert message to *F*:

$$G \rightarrow F: ID_G, CID_G, ID_B, CID_B, \text{MAC}(K_{GF}, ID_G, CID_G, ID_B, CID_B) \quad (2)$$

where ID_G, CID_G is node ID and cell ID of the inspector which detects compromised node, and ID_B, CID_B is the node ID and cell ID of the compromised node.

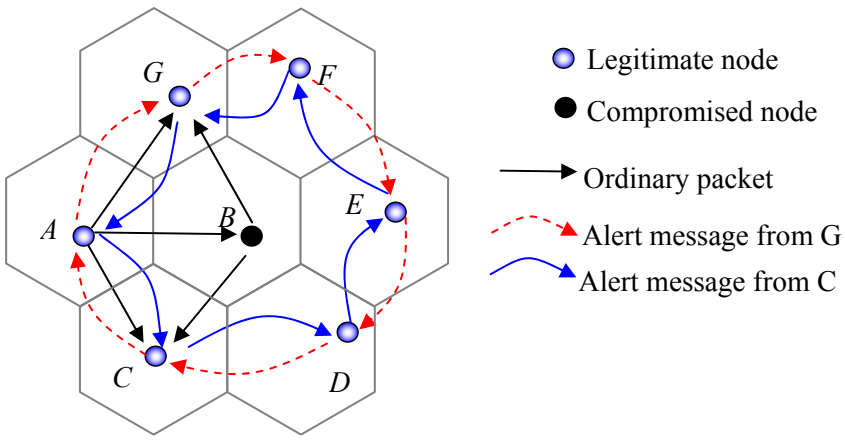


Fig. 31. Notification if a compromised node is detected

b) *Routing path recovery*

Once all inspectors get notification about the compromised node B, they send message to all nodes in the cell B. Those nodes eliminate node B and elect a new coordinator. In order to do this, they just increase the coordinator ID in their table by 1, and consider that node as a new coordinator (Fig. 32).

In most cases, an adversary does not compromise only one node, but a group of nodes. If all nodes in cell B are compromised, then the new coordinator election may become useless. To solve this problem, I go a step further and employ the by-pass mechanism. The mechanism is illustrated in Fig. 33.

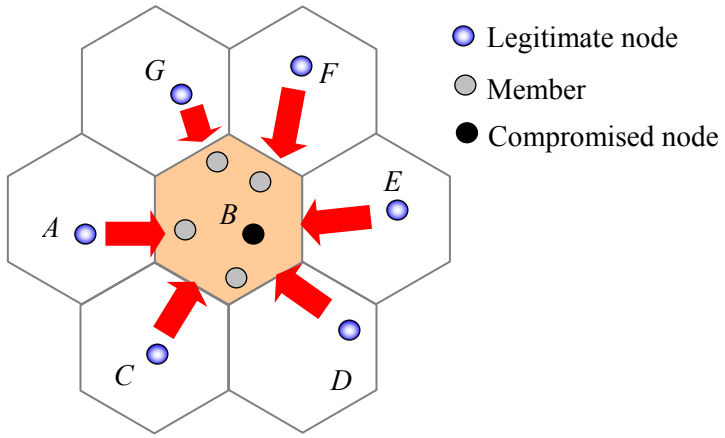


Fig. 32. A new coordinator is elected to eliminate the compromised node B

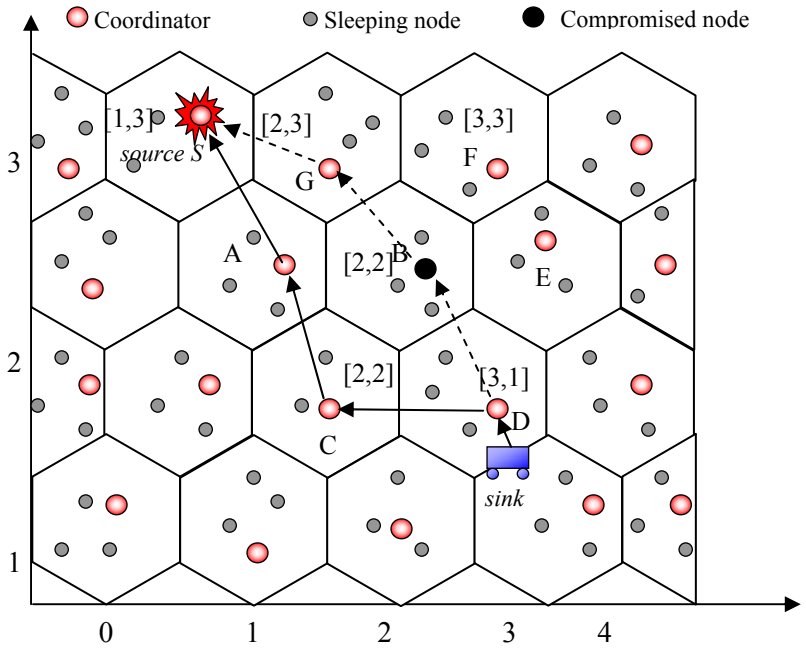


Fig. 33. Routing path recovery after compromised node is detected

As the routing path {sink, D, B, G, S} goes through the compromised node B, D and G send a removal message to all nodes along the routing path. Then, node D sends a new query on behalf of the sink to C to establish a new one as {sink, D, C, A, S}.

4.8 Security analysis of key management scheme

In this section, I am focusing on security and efficiency of the key management scheme. I discuss communication and memory overhead, network connectivity including local connectivity and global connectivity, and the enhanced resilience against node compromise of this version. Local connectivity is the probability a node could connect with neighbor nodes within its transmission range. Global connectivity is the ratio of the number of sensor nodes forming the largest isolated connected component in the final key graph *G* to the size of the whole network. In node compromise attacks, adversaries usually launch node compromise attacks to eavesdrop secure channels in the network, or using key materials revealed from compromised nodes to perform node replication attacks. In this regard, I discuss whether nodes compromised attacks could be used for eavesdropping or not.

4.8.1 Network key connectivity

Supposed $A(n_i, n_j)$ is the event node n_i is a neighbor of node n_j, $B(n_i, n_j)$ is the event that share at least one common key-space. The local connectivity could be calculated as:

$$P_{local} = P(B(n_i, n_j) \mid A(n_i, n_j)) = \frac{P(B(n_i, n_j) \cap A(n_i, n_j))}{P(A(n_i, n_j))}$$

Probability that a node $n_i \in G_i$ is a neighbor of node $n_j(x_j, y_j)$ is the integral of *pdf* $f(n_i)$ over the circle around node n_j with radius *R*.

$$P(n_j(x_j, y_j)) = \iint_{G_i, \|(x,y),(x_j,y_j)\| \le R} f(n_i(x, y))dxdy$$

Because n_j distribute in group G_j following (5) the probability that $n_i \in G_i$ is a neighbor of $n_j \in G_j$:

$$P(A(n_i, n_j) \,||\, G_i, G_j) = \iint_{G_j} P(n_j(x_j, y_j)) f(n_j(x, y))dxdy$$

Hence,

$$P(A(n_i,n_j)) = \sum_{G_i \in \psi} \sum_{G_j \in \psi} P(n_i \in G_i) P(n_j \in G_j) P(A(n_i,n_j) \,||\, G_i, G_j)$$

Denoted $S(G_i)$ is set of neighboring groups of G_i I have

$$P(B(n_i,n_j) \cap A(n_i,n_j)) = \sum_{G_i \in \psi} \sum_{G_j \in S(G_i)} P(n_i \in G_i) P(n_j \in G_j) P(A(n_i,n_j) \,||\, G_i, G_j)$$

Because a sensor node is chosen in a given group with an equal probability, I have the local connectivity can be calculated as

$$P_{local} = \frac{\sum_{G_i \in \psi} \sum_{G_j \in S(G_i)} P(A(n_i,n_j) \,||\, G_i, G_j)}{\sum_{G_i \in \psi} \sum_{G_j \in \psi} P(A(n_i,n_j) \,||\, G_i, G_j)}$$

Denoted $d = a \times \sigma$ is the distance between two deployment points of two neighboring cells. This value has impacts on the local connectivity and global connectivity in the network. If the deployment distribution follows *Gaussian* distribution, there are 99.87% nodes of a group reside within range 3σ from its deployment point. Therefore, if the value d is much larger than 6σ, almost every nodes in a group reside in its cell area, and the neighboring nodes are from its own group. In this case, the local connectivity is very high, but the network is totally partitioned into isolated components, meaning global connectivity is very low. In case of the value d is smaller, the local connectivity may be low, but the global connectivity is high. So, choosing suitable value of d affects the network connectivity.

In the simulation, I change different values of d according to a. Along with this, the ratios of local connectivity and global connectivity also have various values as shown on Table 2 and Fig. 34.

Table 2 Network connectivity

a	**Local connectivity**	**Global connectivity**
0.4	0.0787	0.6546

0.6	0.1577	0.9290
0.8	0.2524	0.9704
1.0	0.3643	0.9921
1.5	0.6036	0.9990
2.0	0.7720	0.9994
2.5	0.8617	0.9998
3.0	0.9226	0.9999
3.5	0.9555	0.9999
4.0	0.9657	1

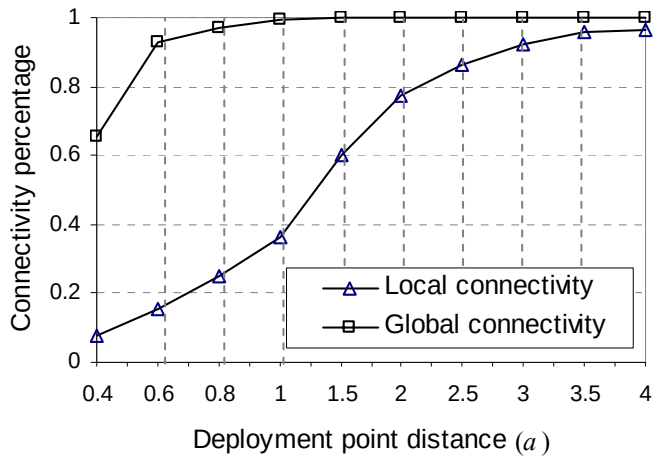


Fig. 34. Network connectivity vs. Deployment point distance (a)

When the distance between two deployment points of two neighboring cells is too low (a = 0.4; 0.6; 0.8 or 1.0), at any node A, there are many nodes of non-neighbor cells distributed around it. These nodes do not share any key-space with node A. So the local connectivity and global connectivity are reduced.

From Fig. 34, it is easy to see that my model gains high local and global connectivity when choosing suitable value of deployment point distances. With value a=1.5, the global connectivity is 0.9990, meaning that only 0.01% number of nodes in the network are waste.

4.8.2 Communication and memory overhead

The network lifetime is a critical goal in designing protocols for wireless sensor networks. In my proposal, I minimized the broadcast data requirement in establishing direct key between neighboring nodes. My 1-hop broadcast

message length is *sizeof(ID) + sizeof(nonce) + 3 × sizeof(hash)*. Comparing with other models in [51][52], the broadcast messages in key discovery phase contain hundreds of key, to achieve high connectivity..

The memory size for storing key materials derived from polynomials is $M = 3\times (t + 1) \log_2 q + sizeof(ID) + sizeof(nonce)$ (bits). This value, along with the number of nodes sharing a polynomial, affects to the resilience against node compromise attacks. This issue will be discussed in more detail in the following section. For CPKS [59], each node shares polynomial with its four adjacent cells, thus the memory size is $4\times (t + 1) \log_2 q + sizeof(ID)$. Fig. 35 shows the memory comparison between CPKS and SCODE's key management, which are almost similar (with $t = 200$, $q = 128$ (bits), $ID = 64$ (bits), $nonce = 32$ (bits)).

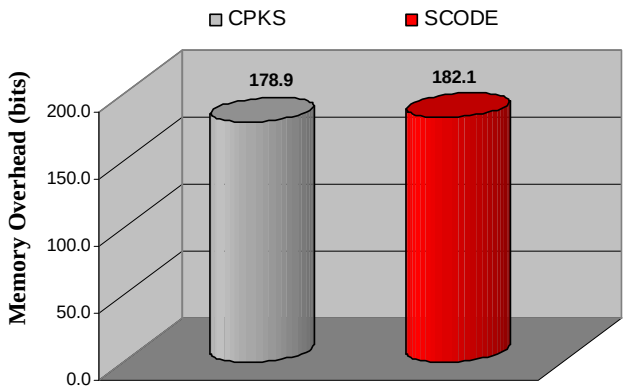


Fig. 35. Comparison of memory overhead between CPKS and SCODE's key management scheme

4.8.3 Resilience against node compromise attacks

Because the working environments of sensor networks usually are hostile, it's easy for sensor nodes to be captured and revealed information. Adversaries could get all the pair-wise keys in compromised nodes, therefore they could break a number of secure links, including all links from these nodes and maybe other links between non-compromised nodes. I evaluate the proposed model in term of the resilience against node capture. That is when x nodes are

compromised, how much is probability to reveal a polynomial, meaning to disclose direct key between non-compromised sensor nodes.

The analysis in [50] shows that the polynomial-based scheme has *t-secure* property: unless more than t polynomial shares of a bivariate polynomial are disclosed, adversaries would not know about the non-compromised node's pairwise keys which are established using this polynomial. Thus, the security of my model depends on the average number of sensor nodes sharing the same polynomial, which is the number of sensor nodes expected to be located in three neighboring hexagon cells.

I have described the deployment model in previous section. Denoted the average number of sensor nodes that are expected to be located in a cell is N_c, the average number of sensor nodes sharing a polynomial can be computed by:

$$N_G = 3N_c = 3\varpi S_c = \frac{3\sqrt{3}a^2\sigma^2\varpi}{2}$$

In this formula, ϖ is the network density.

As described in previous section, the memory requirement for storing key materials is $M = 3\times (t + 1) \log_2 q$ *(bits)*, so the degree of bivariate polynomials is:

$$t = \left\lceil \frac{M}{3} \right\rceil - 1$$

As long as $N_G \le t$, my scheme is perfect resistance against node captures. In other words, compromising of sensors does not lead to the compromise of direct keys shared between non-compromised sensors.

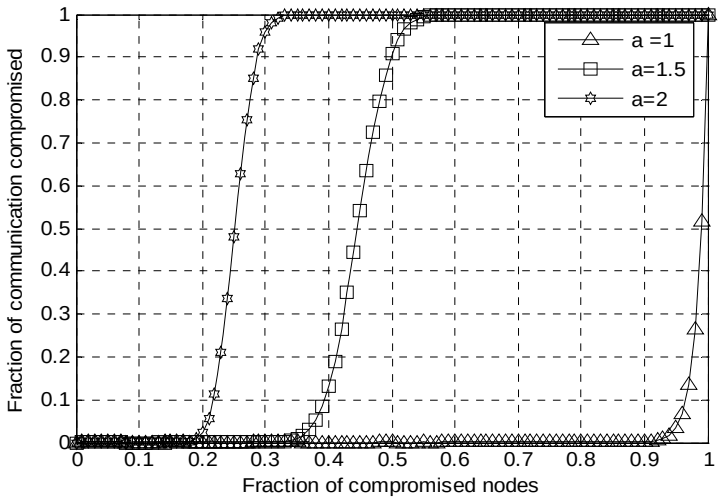


Fig. 36. Network resilience against node compromised attack with different deployment point distances.

According to the analysis in [48], I consider a random attack here. I assume a fraction p_c of sensor nodes in the network have been compromised by an attacker. Among N_G sensor nodes that have polynomial shares, the probability that exactly i sensor nodes have been compromised can be evaluated by:

$$P_c(i) = \binom{N_G}{i} p_c^i (1 - p_c)^{N_s - i}$$

So, the probability that a bivariate polynomial is compromised can be calculated by:

$$P_c = 1 - \sum_{i=0}^{t} P_c(i)$$

In Fig. 37, it can be seen that the longer the distance is (i.e. the larger cell size is), the more vulnerable the resilience against node compromised attacks is. Because when the cell size is larger, there are more sensor nodes in a cell sharing a key-space, leading to lower the security.

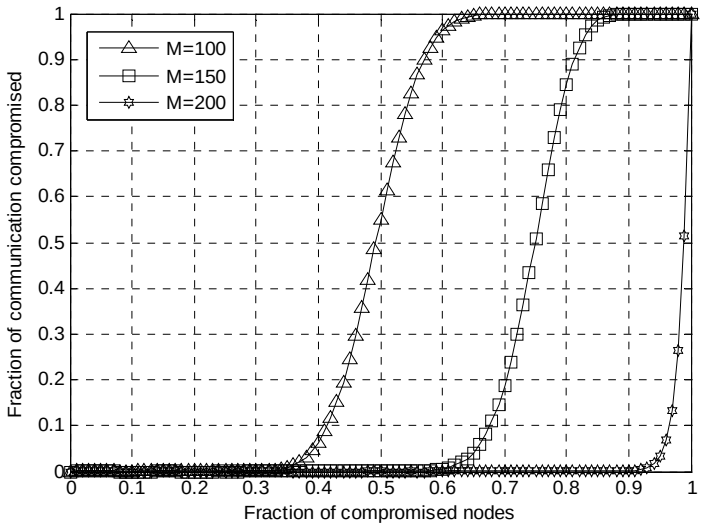


Fig. 37. Network resilience against node compromised attacks with different memory sizes.

In Fig. 37, with more memory, the resilience of network is strengthened, because the degree of polynomials is higher.

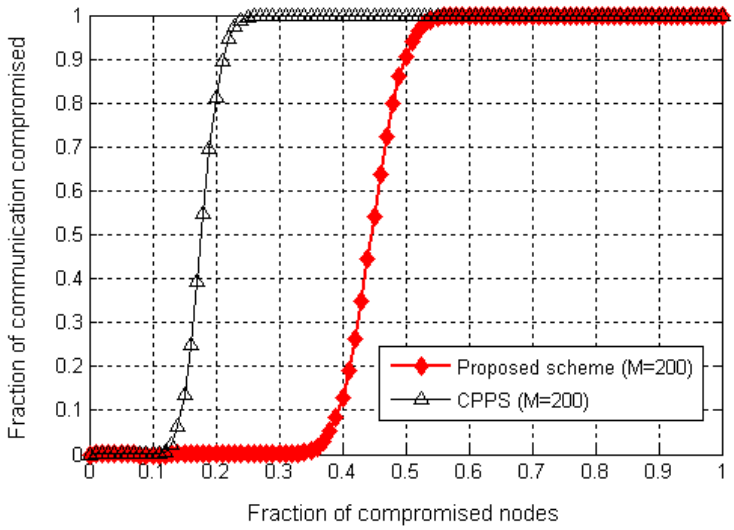


Fig. 38. Comparison the fraction of communication compromised.

Comparing with other models in [51][52][57], the proposed key management scheme has better security in term of resilience against node compromised. In Fig. 38, the proposed scheme gains better security than CPKS [58][59] with the same cryptographic keys storage (M = 200).

4.9 Security analysis of routing protocol

In this section, I analyze the security of SCODE in terms of sensor network routing attacks mentioned in Chapter 2: spoofed, altered, or replayed routing information, selective forwarding, sinkhole, Sybil, wormhole, *HELLO* flood (unidirectional link) attacks. I briefly discuss how SCODE can defend against these attacks.

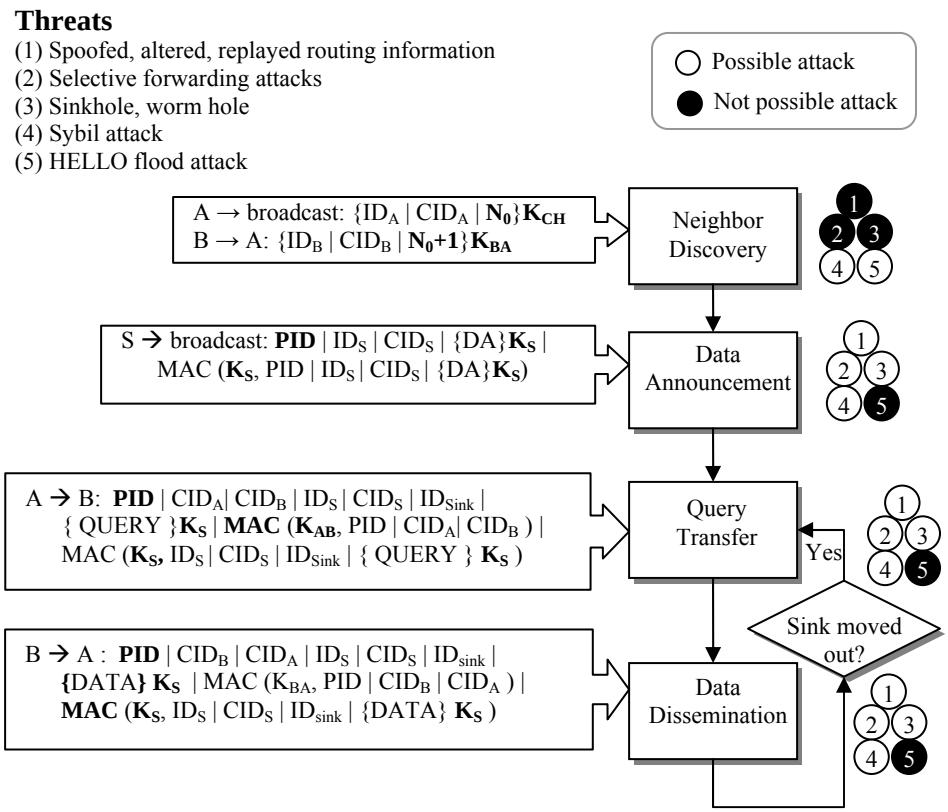


Fig. 39. Attacks and countermeasures in SCODE

Fig. 39 illustrates possible threats and countermeasure of SCODE. During four phases of routing protocol, it is possible for an adversary to launch a

number of attacks. For example, during neighbor discovery phase, she can launch spoofed, altered, replayed routing information attacks, Sybil attack, or HELLO flood attacks. She however would not be able to launch selective forwarding attacks, and sinkhole, worm hole on this phase. In order to defend against those attacks, SCODE employs a number of security primitives including message authentication code, cipher algorithms, with pairwise key, cluster key or global secret key. Next sections will further describe the countermeasure.

4.9.1 Defending against spoofing, altering, or replaying routing information attacks

Let $A \rightarrow B$: M denotes a node A sends a message M to node B, where A is a malicious nodes, B is legitimate node, and M can be data announcement message, a query, or data packets. It is not possible for A to spoof, alter or replay M and sends to B without detection. That means she cannot create routing loops, attract or repel network traffic, extend or shorten source routes, generate false error messages, partition the network, increase end-to-end latency, etc.

There are three cases as follows:

- M is data announcement message. The malicious node A may modify the source information (ID, CID) to cheat the sink (Fig. 40).

  A → B: $PID|ID_S^*|CID_S^*|\ \{DA\}\mathbf{K_S}\ |\ MAC\ (\mathbf{K_S}, PID\ |\ ID_S\ |\ CID_S|\{DA\}\mathbf{K_S})$

  Upon receiving the message, the sink uses K_S to build the MAC value:

  $$MAC\ (\mathbf{K_S}, PID\ |\ ID_S^*\ |\ CID_S^*|\{DA\}\mathbf{K_S})$$

  Since

  $$MAC\ (\mathbf{K_S}, PID\ |\ ID_S^*\ |\ CID_S^*|\{DA\}\mathbf{K_S})$$

  $$\neq\ MAC\ (\mathbf{K_S}, PID\ |\ ID_S\ |\ CID_S|\{DA\}\mathbf{K_S})$$

The sink detects that the source information has altered. This it rejects the message.

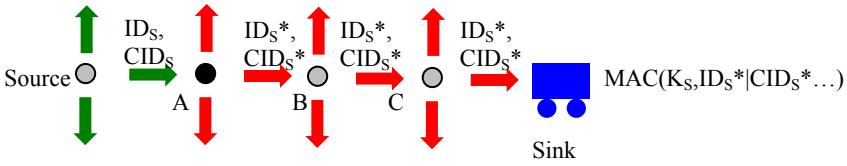


Fig. 40. A malicious node A alters the source information

- If M is query or data message, the attacks can be presented by inspecting system (see Section 4.7).

4.9.2 Defending against selective forwarding attacks

Let $A \rightarrow B$: M denotes the node A sends a message M to node B, where A is a malicious nodes, B is legitimate node, and M can be a data announcement message, a query, or data packets. It is not possible for an adversary using A to launch selective forwarding attacks without detection of the legitimate node B. That means A may refuse to forward, drop, or selectively forward the message M to B.

During three main phases (data announcement, query transfer, data dissemination), the message M is attached with the packet ID (PID), as follows:

- **PID** | ID_S | CID_S | {DA}K_S
  | MAC (K_S, **PID** | ID_S | CID_S | {DA}K_S)
- **PID** | CID_A| CID_B | ID_S | CID_S | ID_{Sink} | { QUERY }K_S
  | MAC (K_{AB}, **PID** | CID_A| CID_B) | MAC (K_S, ID_S | CID_S | ID_{Sink}
  | { QUERY } K_S)
- **PID** | CID_B | CID_A | ID_S | CID_S | ID_{sink} | {DATA} K_S
  | MAC (K_{BA}, **PID** | CID_B | CID_A)
  | MAC (K_S, ID_S | CID_S | ID_{sink} | {DATA} K_S)

If A drops or selectively forwards M, B can easily detect that by checking the packet ID (Fig. 41). Any change of PID can also be detected by B, because B

computes the MAC value of message including the PID and compares with the received MAC. If they are different, then B ensures that M has been modified. Therefore, it is not possible for the adversary to launch selective forwarding attacks without detection of legitimate nodes.

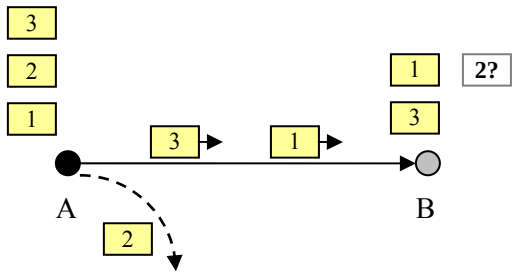


Fig. 41. Malicious node *A* attempts to launch selective forward attacks.

4.9.3 Defending against sinkhole and wormhole attacks

Let $A \rightarrow B \rightarrow C$ denotes a communication between a legitimate node *A*, malicious node *B*, and a malicious node *C*. *B* and *C* may be powerful laptop-class nodes. Then it is not possible for *B* and *C* to launch wormhole or sinkhole attacks. That means an adversary cannot cheat on *A* that $B \rightarrow C$ is the shortest path to the destination so that she can attract nearly all traffic of *A* through B.

According to Section 4.6.2 (query transfer and path discovery phase), a query-receiving node *A* selects the closest cell to the destination to relay the message. The closest cell ID can be computed by itself based on its cell ID and the destination's cell ID. A does not concern about physical distance between B and the destination. Therefore, if B is not in the closest cell, then A will not consider it as next hop though B may say to A that $B \rightarrow C$ is the shortest path to the destination (Fig. 42).

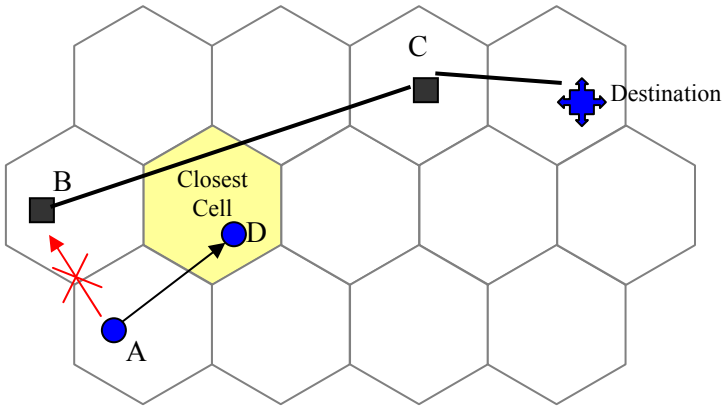


Fig. 42. Malicious nodes B and C cannot launch sinkhole/wormhole attacks

4.9.4 Defending against *Sybil* attacks

Let *A* is a legitimate node, and ID_A denotes ID of a malicious node *A*. It is not possible for an adversary to use A to presents multiple identities to B, for example A says that its ID is ID_C.

This attack can be easily defended in SCODE. In query transfer and data dissemination process, a MAC is presented using the pairwise shared key of the sender and the receiver. If A says its ID is ID_C, then the pairwise key would be K_{CB}, but in fact A cannot computes such a key.

$A \rightarrow B$: PID | CID_A| CID_B | ID_S | CID_S | ID_{Sink} | {QUERY}K_S
| **MAC** (K_{AB}, PID | CID_A| CID_B)
| MAC (K_S, ID_S | CID_S | ID_{Sink} | { QUERY } K_S)

$A \rightarrow B$: PID | CID_B | CID_A | ID_S | CID_S | ID_{sink} | {DATA} K_S
| **MAC** (K_{BA}, PID | CID_B | CID_A)
| MAC (K_S, ID_S | CID_S | ID_{sink} | {DATA} K_S)

In broadcasting process, a node uses the cluster key K_{CH} to encrypt the message with a nonce value N_0 and requires the receiver reply N_0+1 encrypted by pairwise shared key K_{BA}.

$$A \rightarrow \text{broadcast: } \{ID_A \mid CID_A \mid \mathbf{N_0}\}\mathbf{K_{CH}}$$

$$B \rightarrow A: \{ID_B \mid CID_B \mid \mathbf{N_0+1}\}\mathbf{K_{BA}}$$

Therefore, no node can pretend to be other nodes. As consequent, the Sybil attack does not work.

4.9.5 Defending against HELLO flood (unidirectional link) attacks

Let A is a legitimate node, which initiates a HELLO broadcast message, and *B* is a powerful laptop-class node and B's location is out of A's communication range. Then it is not possible for B to cheat A as it is neighboring node.

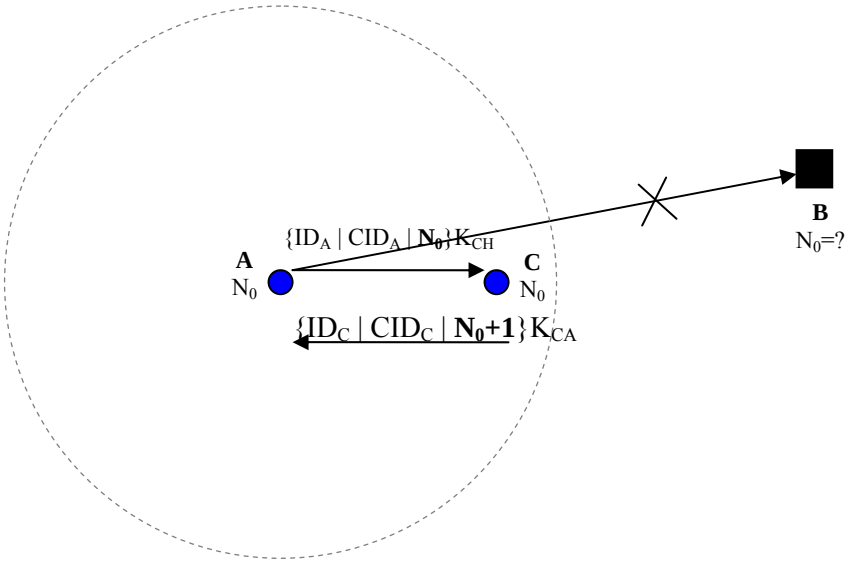


Fig. 43. Malicious node B would not be able to cheat A that it is a neighbor

In neighbor discover phase, A broadcasts a *HELLO* message encrypted by the cluster key K_{CH} with a nonce value N_0.

$$A \rightarrow \text{broadcast}: \{ID_A \mid CID_A \mid \mathbf{N_0}\}\mathbf{K_{CH}}$$

It requires a receiver, say C, replies message encrypted by the pairwise shared key with received nonce value plus 1.

$$C \rightarrow A: \{ID_C \mid CID_C \mid \mathbf{N_0+1}\}\mathbf{K_{CA}}$$

In this case, since B is out of A's communication range, B would not receive the message, thus has no idea what N_0 is. It therefore would not be able to reply N_0+1 to A as:

$$B \rightarrow A: \{ID_B \mid CID_B \mid \mathbf{N_0+1}\}K_{BA}$$

In conclusion, Table 3 provides a summary comparison between SCODE and existing routing protocol. It shows that SCODE is resilient from all sophisticated attacks in sensor network routing.

Table 3 Comparison on communication security of routing protocols[1]

Resilient? / Protocol	Spoofing, altering, or replaying routing information	Selective forwarding attacks	Sinkhole and wormhole	Sybil attacks	HELLO flooding attacks
Directed Diffusion	✗	✗	✗	✗	✗
GEAR, TTDD	✗	✗	N/A	✗	N/A
LEACH, TEEN, PEGASIS	N/A	✗	N/A	N/A	✗
SeRINS	✗	✓	✓	✗	✓
SEEM	✗	✓	✓	✗	✗
SCODE	✓	✓	✓	✓	✓

4.10 Communication overhead analysis

This section presents the communication overhead analysis of the SCODE. I also compare SCODE with TTDD [12] and other sink-oriented data dissemination approaches (henceforth called SODD) such as Directed Diffusion [11], GRAB [14]. Since query aggregation and data aggregation

[1] 'N/A' means *not possible attack* on the given routing protocol. 'X' icon means vulnerability while 'V' icon means resilience against the given attacks.

techniques are adopted in SCODE, TTDD and SODD as well, I do not consider these aggregations when I compare communication overhead. I assume a similar model and notations to those defined in [12].

Let's consider N nodes deployed uniformly in a sensor field of A square meters. Each cell has a size $r = R/\sqrt{5}$ as computed in [6], where R is nominal radio range of sensor nodes. This cell size guarantees connectivity of every two adjacent cells. There are k sinks moving with max speed v, while receiving d data packets from a source in a time period of T. I assume that each insecure data packet is l_{data} (bytes) size long, and the other insecure messages are l_{query} (bytes) size long (here, the other messages means broadcast messages to find neighbors, query messages, *cache-removal* messages, and *data-announcement* messages). As discussed in TinySec [43], the choice of 4-bytes MAC is not detrimental in the context of WSNs. So I use CBC-MAC to generate 4-bytes MACs for every message. As described in Section 4, each data message is added 2 MACs (i.e. 8 bytes). Thus, each secure data message will have $l_{data}+8$. Likewise, each secure query message will have a size of $l_{query}+8$ bytes, and each secure *cache-removal* message and secure *data-announcement* message will be $l_{query}+4$ byte long. The secure broadcasting message to find neighbor has a same size as the insecure broadcasting message because my encryption computation does not increase the size of encrypted information. To model sink mobility, I assume each sink traverses m cells ($m \le 1 + vT/(R/\sqrt{5})$. Consequently, each sink has to send a *cache-removal* message and a *query* m times, and receives d / m data packets between two consecutive location updates.

Suppose there are $c \times c$ cells in the whole sensor field (where $c = \left\lceil \sqrt{A}/(R\sqrt{5}) \right\rceil$; $\lceil x \rceil$ is the smallest number larger than x). I analyze the communication overhead in the worst-case, i.e. the source and the sink are furthest away from each other.

For a query from a source to reach a sink, it traverses c cells throughout the sensor field, in other words, it traverses throughout c hops. Therefore, the

overhead is $c(l_{query}+8) + c(l_{query}+4) = 2c.l_{query}+12c$, including sending *cache-removal* messages.

Similarly, the overhead to deliver d / m data packets from a source to a sink is $c.(d/m)$. Since I have k mobile sinks, the total overhead to receive d data packets is:

$$k.m.(2c.l_{query}+12+c.d.l_{data}/m) = k.m.c.(2l_{query}+12)+k.c.d.l_{data}$$

Plus the overhead $c^2.(l_{query}+4)$ for *data-announcement* message to reach all coordinators using flooding mechanism and the overhead in broadcasting hello message to find neighbors Nl_{query}, I have:

$$CO_{SCODE}=k.m.c.(2l_{quer}+2)+k.c.d.l_{data}+c^2.(l_{query}+4)+N.l_{query}$$

In SCODE, there is no overhead for constructing the grid, since each node computes its grid ID based on its geographical location. Also, there is no communication overhead for coordination election process from GAF because each node elects itself based on its ID. Therefore, the total overhead of SCODE is:

$$CO_{SCODE} = k.m.c.(2l_{query}+12)+k.c.d.l_{data}+c^2.(l_{query}+4)+N.l_{query} \quad (2)$$

For TTDD, as analyzed in [12], the total overhead is:

$$CO_{TTDD} = N.l + \frac{4N}{\sqrt{n_1}}l + k.m_1.n_1.l + k.c_1(m_1.l + d(l_{data}+8))\sqrt{2N} \quad (3)$$

And for the *SODD,* the communication overhead without considering aggregation is:

$$CO_{SODD} = k.m_1.N.l + k.c_1.d(l_{data}+8)\sqrt{N} \quad (4)$$

where m_1 is the number of cells that a mobile sink traverses ($m_1 \le 1+vT/\alpha$, where α is the TTDD's cell size), n_1 is the number of nodes in each cell ($n_1 = N\alpha^2 / A$), and $c_1\sqrt{N}$ is the average number of sensor nodes along the straight-line path from the source to the sink. ($0 < c \le \sqrt{2}$).

For example, the sensor field A = 2000m×2000m. The number of mobile sinks k is 4, moving with speed v = 10 (m/s). I suppose m and m_1 reach their maximum value, i.e. $m = 1 + vT / R / \sqrt{5})$ and $m_1 = 1 + v.T / \alpha$, where the nominal radio range $R=250$ (m), $T=200$ ($seconds$), and TTDD's cell size α = 200 (m). Suppose $c_1 = 1$, l_{query} = 36 (bytes), l_{data} = 64 (bytes) and d = 100 data packets. I vary the number of nodes N from 0 to 10,000 in order to show the predominance of SCODE in node density.

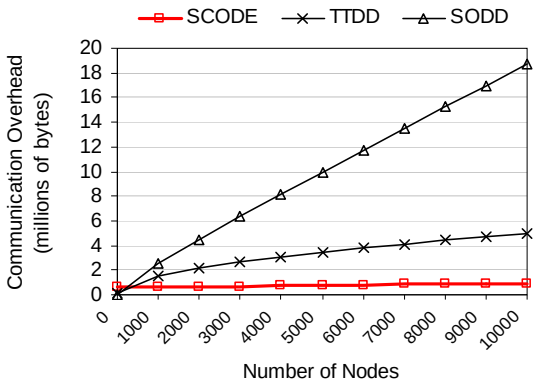


Fig. 44. Communication overhead vs. number of nodes

Fig. 44 plots the communication overhead of SCODE compared with TTDD and SODD. In this figure, the communication overhead of SCODE is much less than TTDD and SODD as the node density increases. The reason is that, in SCODE, only the coordinators participate into sending and receiving packets. Therefore, the communication overhead only mostly depends on the number of cell c^2 , instead of the number of nodes N. Whereas, in TTDD and SODD, most of the nodes participate in communication process, thus the total overhead increases as the number of nodes increases.

4.11 Simulation-based evaluation

In this section, I evaluate routing performance of SCODE and compare it with other protocols. The evaluation is carried out to evaluate its real-time performance including energy consumption, packet delivery ratio, and latency.

4.11.1 Simulation model

I simulated SCODE on SENSE (Sensor Network Simulator and Emulator) [44] and compared with other approaches TTDD and DD. The network comprises of 400 nodes randomly deployed in a 2000m × 2000m field. I use the same energy model used in *ns-2.1b8a* [45] that requires 0.66W, 0.359W and 0.035W for transmitting, receiving and idling respectively. I set the power consumption rates of RC5 according to [46] for encryption, MAC computation, and random number generation are 0.65W, 0.48W, and 0.36W, respectively. As analyzed in [43][46], I set the time consumption for encryption of 64 bits with RC5 0.26 ms, for generating 64 pseudorandom bits takes 0.26 ms, and computing a MAC on 32 bits requires 0.13 ms. The simulation uses MAC 802.11 *Distributed Coordination Function* (DCF) and nominal transmission range of each node is 250m. *Two-ray ground* [47] is used as the radio propagation model. Each data packet has 64 bytes, query packets and the others are 36 bytes long. Additional bytes for MACs and nonce values are also put into each message. The default number of sinks is 8 moving with speed 10 m/s according to *random way-point model*. Two sources generate different packets at an average interval of 1 second.

4.11.2 Simulation results

In SCODE, I study the impact of the number of sinks, sink's speed, and the density of the network since these evaluations show how the protocol works well in mobile-sink sensor networks. I measure the energy consumption, average delay (average response time to users), and success ratio (total number of packets has been delivered successfully). The metric (energy consumption, average delay, success ratio) is foremost important. Energy consumption evaluates how the protocol costs, which is a key factor to make the protocol feasible or not. The average delay and success ratio must be guaranteed to be appropriate. If the energy consumption is low, but high average delay (e.g. 5

second…), and success ratio is low (e.g. 50%), then the protocol would not be used.

(a) *Impact of Sink Number*

For the sensor network area of 2000m×2000m, I set the number of sink varying from 1 to 8. Each sink moves with the maximum speed 10m/s and a 5-second pause time. The number of total nodes and the number of sources are not changed.

Fig. 45 shows total energy consumption of SCODE as the number of sinks varies from 1 to 8. It demonstrates that SCODE is more energy efficient than SCODE, TTDD, and DD. This is because of three reasons. First, SCODE is based on a coordination network, so that nodes in each cell negotiate among themselves to turn off its radio to significantly reduce energy consumption. Meanwhile, TTDD and DD must turn on all nodes to participate in routing. Second, SCODE is optimized a number of transmission hops between sources and sinks that is based on the cell size to maximize the communication distance between two adjacent cells.

Fig. 46 plots the average end-to-end delay of SCODE. The figure shows that the average delay of SCODE is less than that of TTDD. In Fig. 47 it shows that the success rate of SCODE is always above 97%. It means that SCODE delivers most of packets successfully.

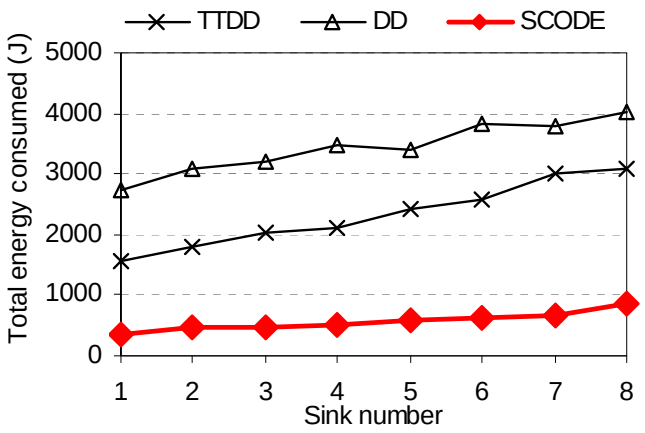


Fig. 45. Energy consumption vs. sink number

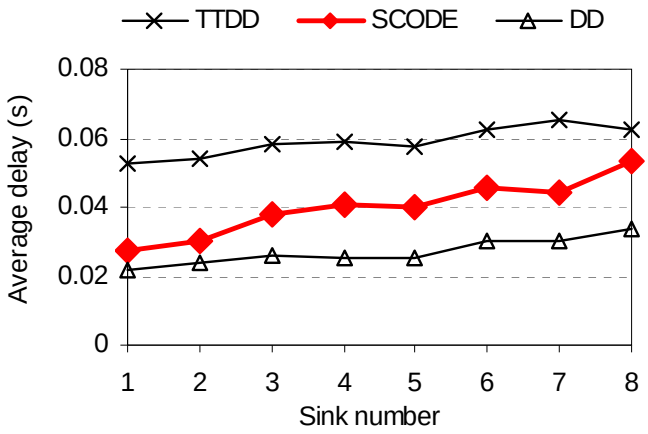


Fig. 46. Average delay vs. sink number

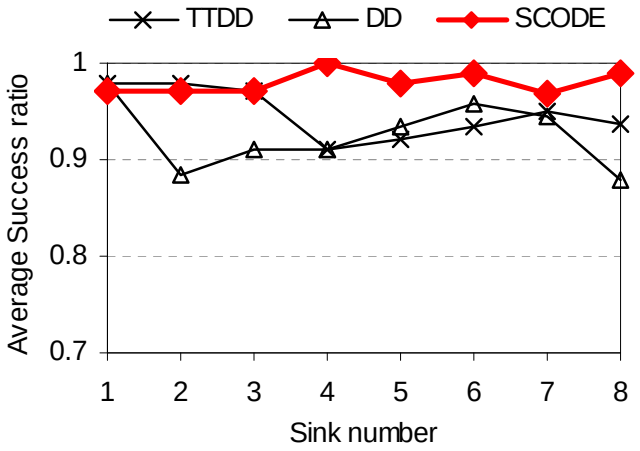


Fig. 47. Success ratio vs. sink number

(b) *Impact of Sink Mobility*

In order to see the impact of sink mobility, I ran the simulations for different sink speeds (0 to 30m/s). In this experiment, the network consists of 8 mobile sinks and 400 sensor nodes. The number of sources does not change.

Fig. 49 shows the energy consumption as the sink speed changes. In both low and high speeds of the sinks, it shows that the total energy consumed is much less than TTDD. The reason is because, aside from above reasons, SCODE reduces the number of re-transmissions of query and up dating sink's locations while the sinks are moving. The query only needs to resend as the sink moves to

another cell. In contrast, TTDD sends more messages to propagate new location of the sinks throughout the sensor field to all sensor nodes.

Fig. 48 shows the delay of SCODE. In Fig. 49, as the sinks speed up, the average success ratio is always above 97%. This results show that SCODE handles mobile sinks efficiently

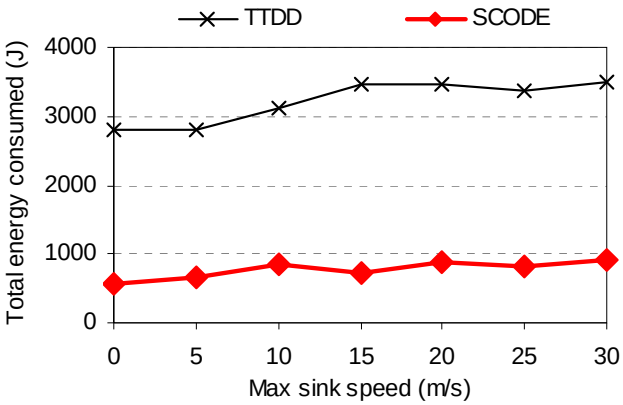


Fig. 48. Energy consumption vs. sink speed

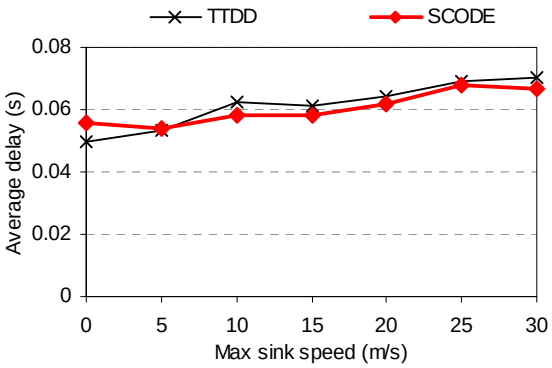


Fig. 49. Average delay vs. sink speed

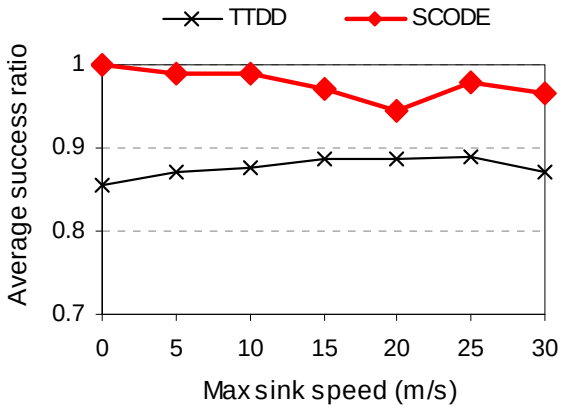


Fig. 50. Success ratio vs. sink speed

(c) *Impact of Node Density*

To evaluate impact of node density on SCODE, I vary the number of nodes from 200 to 600. The number of sinks is 8. Each sink keeps moving with speed 10m/s as the default setting. The number of sources is 2. The sensor field size is not changed.

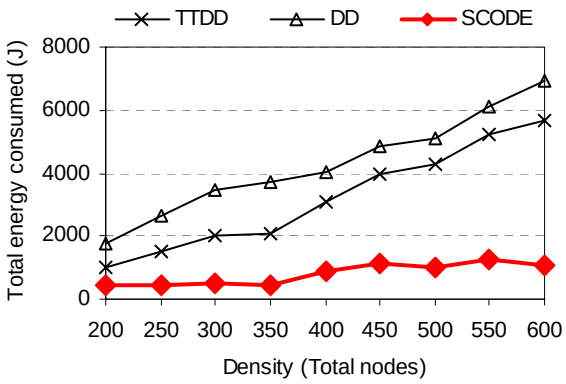


Fig. 51. Energy consumption vs. density

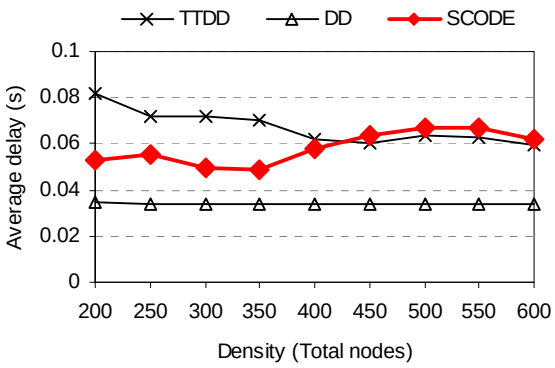


Fig. 52. Average delay vs. network density

Fig. 51 shows the energy consumption with different node densities. The figure demonstrates that SCODE consumes less energy than TTDD and DD. As the number of nodes increases, the total energy consumed slightly increases while that of TTDD and DD significantly increases. This is mainly because SCODE turns off radio most of the time. Therefore, energy is consumed mostly by the coordinators. Whereas, in TTDD and DD, nodes do not participate in communication still consume much energy in idling mode. Fig. 52 shows the average delay of SCODE which is comparable with TTDD.

4.11.3 Scalability

In the previous sections, only 8 sinks were deployed so that SCODE could be comparable with existing approaches. However, this number does not reflect the scalability of SCODE. In a large scale sensor network, hundreds to thousands of users may be simultaneously access the networks.

Therefore, I extend the simulation with 10,000 nodes deployed in a area of 10,000m ×10,000m. The number of sinks varies from 100 to 1000.). The energy consumption linearly increases as the number of sinks increases. The average delay is always around 0.12 second, and success ratio is around 100%, which indicates that SCODE works well with a large number of users

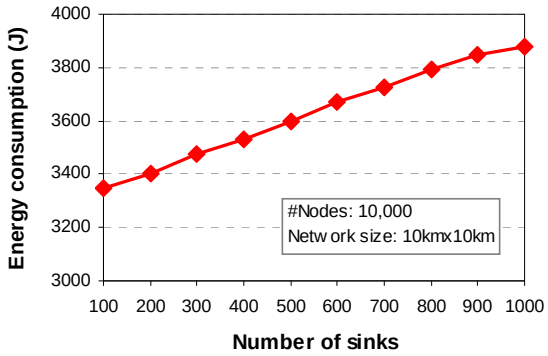


Fig. 53. Energy consumption vs. number of sinks

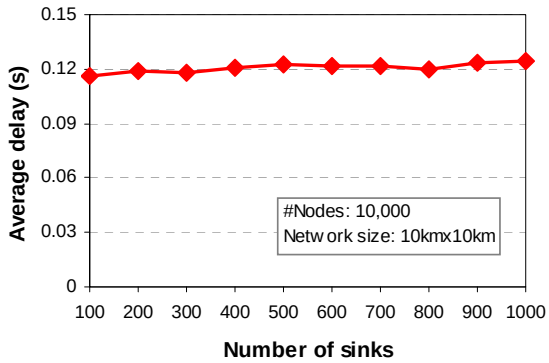


Fig. 54. Average delay vs. number of sinks

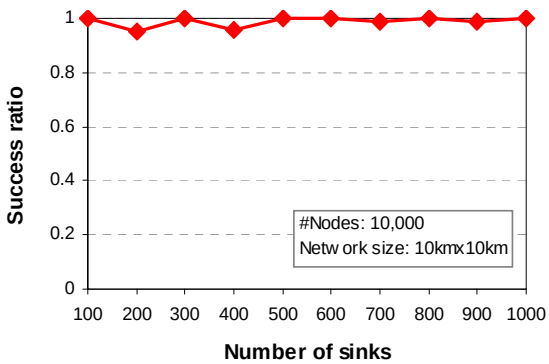


Fig. 55. Success ratio vs. number of sinks

Chapter 5
Simplified SCODE

5.1 Introduction

This chapter presents a simplified version of the SCODE. Though the SCODE has showed a better performance than existing approaches, it is too much depended on human intention and the topology is fixed.

Firstly, the deployment phase requires administrators determine the deployment points in advance. Each group of sensors should be deployed exactly to those points with much intention so that sensor nodes will not reside out of the predetermined cells.

Secondly, the node deployment model must be uniform or Gaussian distribution. This limits SCODE to be applied in wide applications.

Third, the topology cannot be changed once it is deployed. With advances in MEMS technology nowadays, sensors can be dynamically adjusted their power to carry out some application purposes. Change of power leads to the change of node communication range. In that case, the fixed network topology would not work correctly. Therefore, there is a need to recalculate the cell size to adapt new communication range. For a hexagonal topology, each node needs to know three cells to compute the new cell size as shown in Fig. 24.

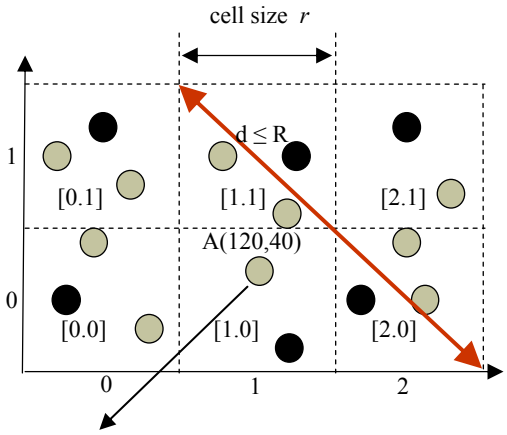
Compared with the hexagon, the square topology is a simplifier so as to apply in different scenarios. The square requires a simpler computation of cell size, thus it is more flexible to the cell-size change. The square does not require any specific deployment distribution. It can be uniform or non-uniform distribution. However, the square is less efficient than the hexagon because it produces a larger number of cells.

5.2 Grid (square) topology formation

Supposed there are N sensor nodes deployed in the network area of a virtual grid. Each node itself computes cell ID [X, Y] based on its coordinate (x, y) as follows:

$$X = \lfloor x/r \rfloor, Y = \lfloor y/r \rfloor \quad (1)$$

An example is shown in Fig. 56. Coordinate of node A is (120, 40). Assume that the cell size r = 100m. It computes X= $\lfloor x/r \rfloor = \lfloor 120/100 \rfloor = 1$, Y = $\lfloor y/r \rfloor = \lfloor 40/10 \rfloor = 0$. So the cell ID of A is (1,0).



Example: node (x,y) = (120,40), r = 100(m)

X= $\lfloor x/r \rfloor = \lfloor 120/100 \rfloor = 1$

Y = $\lfloor y/r \rfloor = \lfloor 40/10 \rfloor = 0$

Fig. 56. Nodes compute their cell ID based on their coordinate

In the simplified SCODE, nine surrounding cells are considered as adjacent cells. For every two nodes in adjacent cells communicate with each other within one-hop transmission, the diagonal distance d must not be larger than R. Therefore $d^2 = (2r)^2 + (2r)^2 \le R^2$ or $r \le R/\sqrt{8}$.

5.3 Routing algorithm in grid topology

In grid topology, the routing algorithm is also revised to adapt to a new grid structure. This algorithm replaces the algorithm in phase 2: query transfer and routing path discovery (Section 4.6.2). Fig. 57 illustrates the algorithm.

```
1. function next_cell_calculation()
2. input: CID_S, CID_this
3. output: CID_next
4. ΔX = CID_S.X – CID_this.X ;
5. ΔY = CID_S.Y – CID_this.Y;
6. φX = (ΔX == 0)?0:ΔX/|ΔX|;
7. φY = (ΔY == 0)?0:ΔY/|ΔY|;
8. CID_next.X = CID_this.X + φX;
9. CID_next.Y = CID_this.Y + φY;
10.if (lookup_neighbor_table(CID_next) == FALSE)
11.      round_way_calculation();
12.}
```

Fig. 57. The algorithm to find next cell

It is based on cell ID, instead of node ID or node's location. The algorithm is described as follows:

- [**Lines 4-7**] It first computes the disparities φX, φY between source's cell and current node's cell. For example, cell ID of the current node is CID_{this}(X,Y) = [3,1]; and cell ID of the source is CID_S(X, Y) = [0,3]. Then:

$$\Delta X = CID_S.X - CID_{this}.X = 0 - 3 = -3$$

$$\Delta Y = CID_S.Y - CID_{this}.Y = 3 - 1 = 2$$

So, the disparities are:

$$\varphi X = (\Delta X == 0)?0:\Delta X/|\Delta X| = -3/|3| = -1$$

$$\varphi Y = (\Delta Y == 0)?0:\Delta Y/|\Delta Y| = 2/|2| = 1 .$$

- [**Lines 8-9**] Now, cell ID of the next hop is calculated as following:

$$CID_{next}.X = CID_{this}.X + \varphi X = 3 + (-1) = 2$$

$$CID_{next}.Y = CID_{this}.Y + \varphi Y = 1 + 1 = 2 \quad .$$

i.e. $CID_{next} = [2,2]$.

- [**Line 10-11**] The node then looks up in its neighboring table, if there exists a coordinator in the next cell, then it forwards the message to that cell; otherwise it considers the next cell a void cell. In this case, it finds a round way to avoid that cell. The round way calculation is simple: I can set the disparity either φX or φY equal to *0* and re-calculate again the next cell ID in step 2.

For example in Fig. 58, the $sink_1$ sends a query to the source along the route {$sink_1$, [4,1], [3,2], [2,3], [1,3], *S*}. However, with the $sink_2$, the cell [3.0]'s coordinator can not find any neighboring node in cell [2.1] (due to void cell). Therefore, it finds the round way as {$sink_2$, [3,0], [3,1], [2,2], [1,3], *S*}. A query from the sink is re-transmitted when the sink moves to another cell.

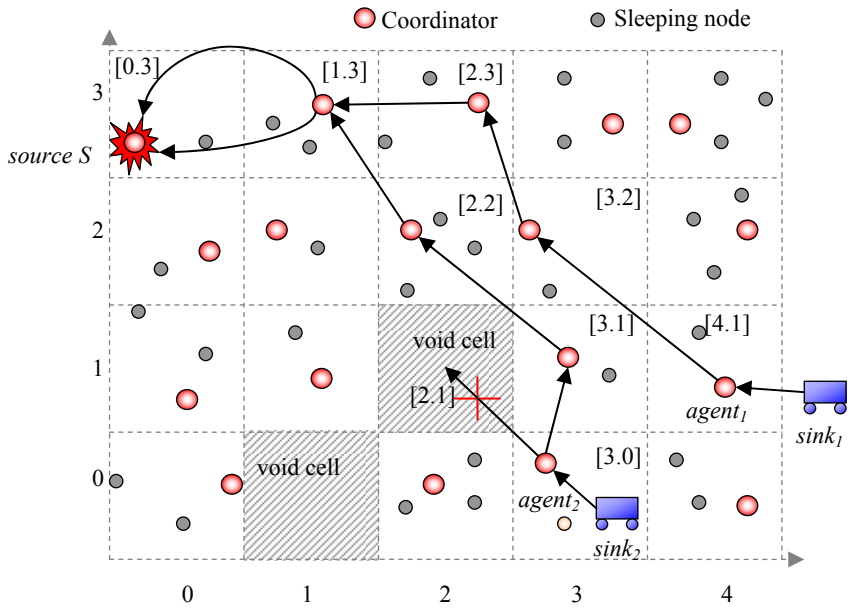


Fig. 58. SCODE routing scheme.

Theorem 5.1. The *next_cell_calculation* algorithm guarantees that a message will reach its destination eventually.

Proof: Suppose the current node is not the destination ($ID_{this} \neq ID_S$ and $CID_{this} \neq CID_S$) and the node needs to compute the next cell ID. I proof that computed next cell is always towards to the destination.

Since $CID_{this} \neq CID_S$, I have at least $CID_{this}.X \neq CID_S.X$ or $CID_{this}.Y \neq CID_S.Y$. This means that at least $\varphi X \neq 0$ or $\varphi Y \neq 0$. Therefore, the computed next cell must locate between the current cell and the destination cell, i.e. the computed next cell is towards to the destination.

5.4 Security in grid topology - base routing

Compared with hexagonal topology, all the security of SCODE maintains the same. However, the inspecting system considers fours neighboring coordinators as inspectors, not six as hexagonal topology. For example, in Fig. 59 nodes A, C, D, and E are inspectors of node B. Therefore, nodes C and E can detect B if it is compromised based on communication between A, B and D.

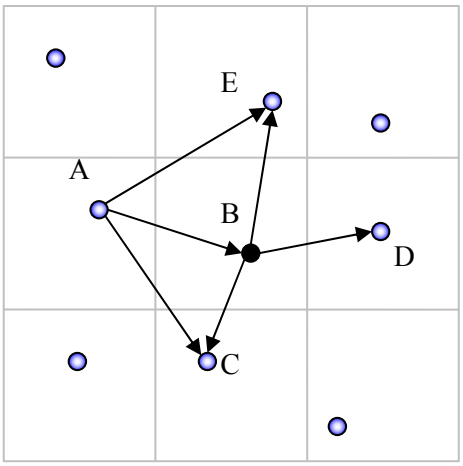


Fig. 59. Inspecting system in grid topology

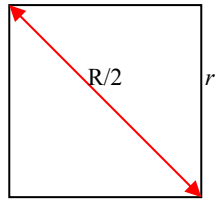
5.5 Grid vs. hexagonal topology

Compared with the hexagon, the square topology requires a simpler computation of cell size, thus more flexible to the cell-size change. Square does not require any specific deployment distribution. It can be uniform or non-

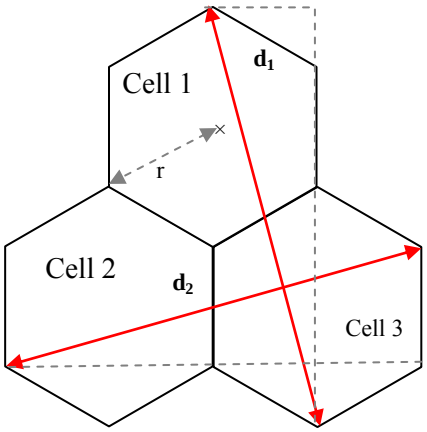
uniform distribution. However, square is less efficient than hexagon because it produces a larger number of cells. The following paragraphs briefly present the differences between square and hexagon in terms of flexibility, distribution, and efficiency.

5.5.1 Computational complexity

In case of square, each node requires knowledge of its own cell to compute the cell ID based on the communication range. With hexagon, it requires knowledge of three adjacent cells to decide the cell size, as shown bellows:



$r^2 + r^2 = (R/2)^2 => r = R/\sqrt{8}$



$d_1 = r.5/\sqrt{2};\ d_2 = r.\sqrt{13}$

$d_1 = r.5/\sqrt{2} = R \Leftrightarrow r = R\sqrt{2}/5$

Fig. 60. Cell size computation in grid and hexagonal topology

This becomes very important in such a case that nodes need to automatically adjust their transmission power for some application-specific purpose. The transmission range of the operating communication system can be predicted by using the Friis free-space [67] formula as:

$$R_{\max} = f(P_t) = \frac{\lambda}{4\pi}\sqrt{\frac{P_t G_t G_r (1 - |\Gamma_r|^2)}{P_r}}$$

where Γ_r = the operating wavelength,

P_t = the power transmitted by the sensor,

G_t = the gain of the transmitting antenna,

G_r = the gain of the receiving antenna,

P_r = the receiver sensitivity, and

$|G_r|^2$ = the reflected power coefficient of the receiving antenna.

5.5.2 Deployment distribution

Section 4.4 presents hexagonal deployment model. The deployment points, which are centers of hexagons, are first determined. Each group of sensors are then scattered from an airplane to a deployment point as shown in Fig. 61. The probability distribution functions (pdf) may be uniform distribution or Gaussian distribution.

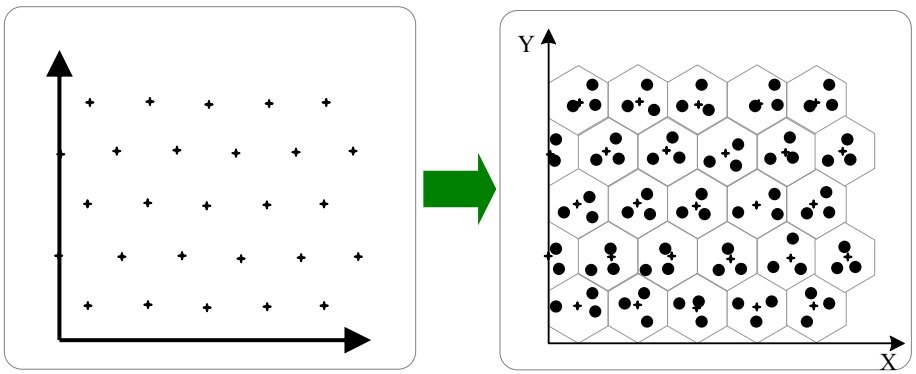


Fig. 61. Hexagonal deployment model

In the square topology, the deployment distribution is not restricted to any specific distribution. It can be uniform or non-uniform (Gaussian, poison, etc) as illustrated in Fig. 62.

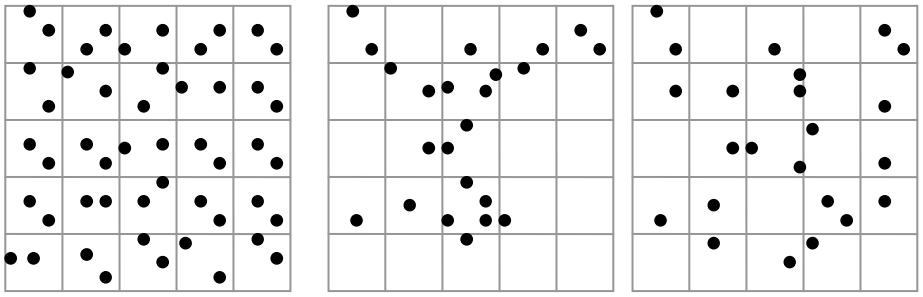


Fig. 62. Possible distributions in the square topology

5.5.3 Efficiency

Hexagonal topology is more optimal than grid and gains better efficiency. This is because a hexagon is approximate to the circle. With a less number of cells, it can cover the entire network area. That means it reduces the number of hops in every source-sink communication.

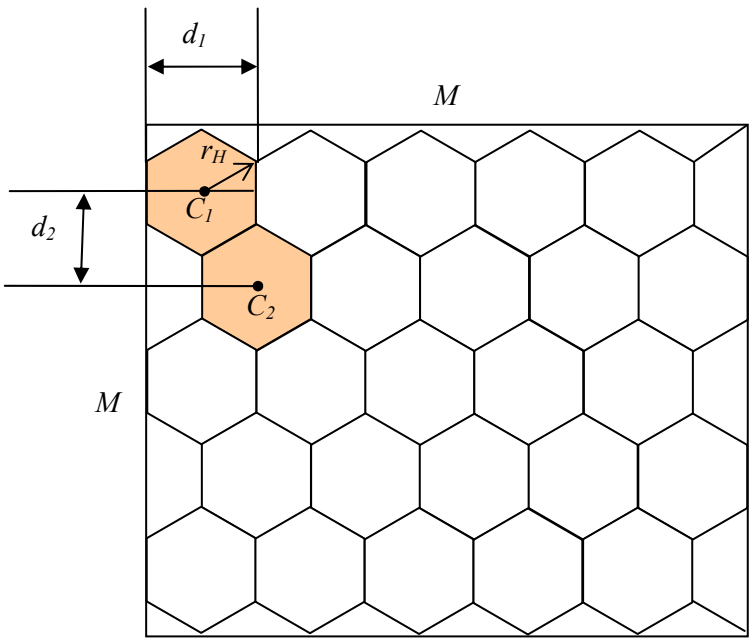


Fig. 63. Number of cells in hexagonal topology

Assume that the network is an area of $M \times M$, and the sensor's communication range is R. As presented in Section 4.4.2, the cell size of hexagonal topology $r_H = R\sqrt{2}/5$. The distance d_1 in Fig. 63 is

$$M.5/(R.\sqrt{6})\ d_1 = 2.r_H\sqrt{3}/2 = r_H\sqrt{3} = R\sqrt{6}/5$$

Therefore, the number of cells N_{HX} in X axis is:

$$N_{HX} = \frac{M}{d_1} = \frac{M}{R\sqrt{6}/5} = \frac{5M}{R\sqrt{6}}$$

The distance d_2 between cell C_1 and cell C_2 in Fig. 63 is

$$d_2 = r_H + r_H/2 = r_H.3/2 = R.3/\sqrt{10}$$

The number of cells N_{HY} in Y axis is

$$N_{HY} = \frac{M}{d_2} = \frac{M\sqrt{10}}{3R}$$

So the total number of cells in hexagonal topology is

$$N_H = N_{HX}.N_{HY} = \frac{5\sqrt{10}.M^2}{3\sqrt{6}.R^2}$$

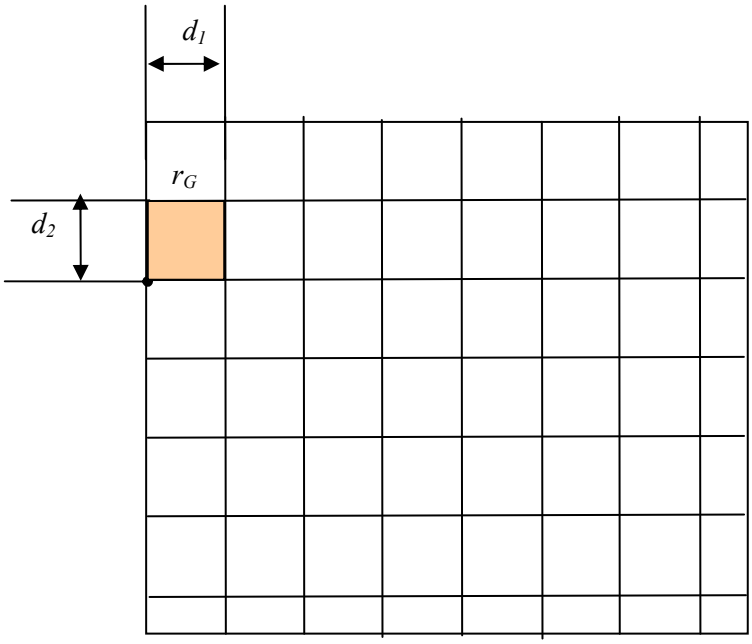


Fig. 64. Number of cells in grid topology

In case of grid topology, as presented in Section 5.2, the cell size $r_G = R/\sqrt{8}$. The distances d1 and d2 in Fig. 64 are equal and equal to the cell size r_G. Therefore, the number of cells N_{GX} in X axis is:

$$N_{GX} = M / d_1 = M / r_G = M\sqrt{8} / R$$

Likewise, the number of cells N_{GY} in X axis is:

$$N_{GY} = M / d_2 = M / r_G = M\sqrt{8} / R$$

So the total number of cells in grid topology is:

$$N_G = N_{GX}.N_{GY} = \frac{8M^2}{R^2}$$

Assume M=2000, R = 250, then a comparison of the number of cells between grid and hexagonal topology is given in Fig. 65. It shows that the number of cells in grid is around 3.7 folds larger than that of hexagonal topology. The number of cells closely relates to the number of transmission hopes. It means

that routing path is square is longer than hexagonal routing path. For example, in Fig. 66 the number of cells in grid is 12 and in hexagonal topology is 6; as a consequence, the number of hops in grid is 12, while it is only 6 in hexagonal topology. Reducing number of transmission hops means reducing energy consumptions and average delays.

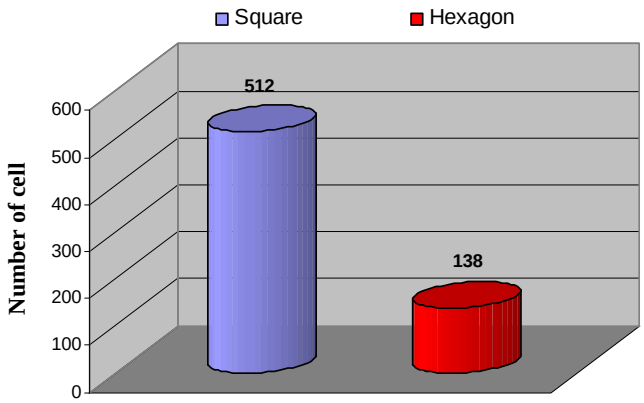


Fig. 65. Comparison between number of cells in grid and hexagonal topology

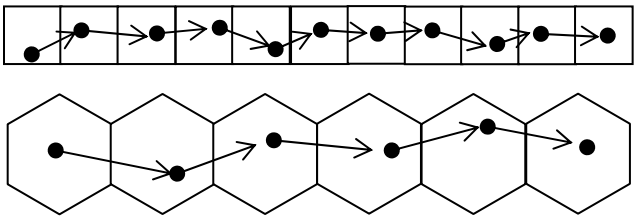


Fig. 66. With the same distance, the number of hops in grid (11) is larger than that of hexagonal topology (6)

In order to verify above stated optimization between grid and hexagonal topology, I have performed the same simulation with Section 4.11. Energy consumption and average delay are measured in Fig. 67 and Fig. 68. We can see that grid topology (employed in simplified SCODE) has less energy efficiency and longer delay time than hexagonal topology (employed in SCODE).

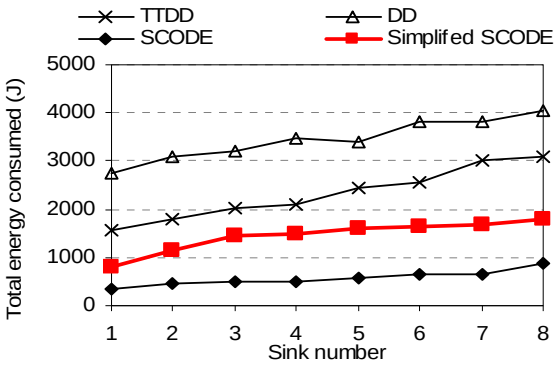


Fig. 67. Energy consumption vs. sink number

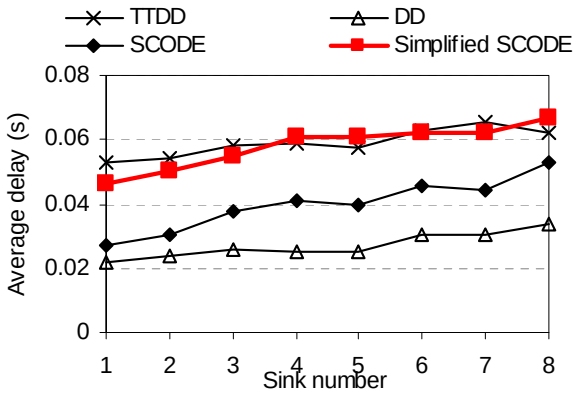


Fig. 68. Average delay vs. sink number

Finally, the Table 4 concludes comparison between square and hexagon. It shows that grid is simpler in computation, but less optimal (less efficient) compared with hexagonal topology. Therefore, grid is suitable for flexible communication range (i.e. cell size is frequently changed) while the hexagonal topology is the best for static topology.

Table 4 Comparison between grid and hexagonal topology

	Grid	**Hexagon**
Flexibility	High	Low
Distribution	Any	Gaussian/Uniform
Efficiency	Low	High

Chapter 6
Conclusion and future work

In this study, I presented a first, novel energy-efficient secure routing and key management scheme for mobile sinks in sensor networks, namely SCODE*plus*. It is a significant enhancement of the previous study, *Secure COodination-based Data dissEmination protocol for mobile sinks* (SCODE). Besides several improvements, this study was recognized by a careful consideration of security (key management scheme) during the design time. Those not only enhance security but also increase efficiency of the proposed scheme.

In SCODE, the network is partitioned into a virtual hexagonal plane. Nodes in the same cell negotiate so that only one node stays awake, while the other may fall into sleeping mode. The proposed routing algorithm is a cell-based approach, which is more flexible than node-based and location-based approaches. In order to increase security and efficiency, the key management scheme and routing protocol are considered together during the design time. Moreover, an inspecting system, a type of autonomous diagnosis system, is proposed to defend against node compromise attacks and recover the routing path under attacks. SCODE is also simplified with grid topology. Compared with hexagonal topology, grid brings less computational complexity, but less optimal. Therefore, the simplified SCODE is recommended in cases the sensor node's communication range may be frequently changed to adapt application-specific purpose.

Analysis and simulation-based evaluations are conducted to evaluate proposed protocol and compare with existing approaches. The security analysis demonstrates that the proposed scheme can defend against common attacks in sensor networks including node compromise attacks, replay attacks, selective forwarding attacks, sinkhole and wormhole, *Sybil* attacks, *HELLO* flood attacks.

The performance evaluation both in mathematical analysis and simulation shows that the SCODE*plus* significantly reduces communication overhead, energy consumption, packet delivery latency while it always delivers more than 97 percentages of packets successfully.

For the future work, I will focus more on inspecting system. Most of attacks are not randomly occurs, but follow some logical behaviors. For example, if one sensor node is compromised and causes a threat, say selective forwarding attacks, then it is likely that some surrounding nodes may be compromised soon after. If such compromise and attacks are predicted, then the sensor network would have a more robust countermeasure. For example, once a compromised node is detected, the routing algorithm should find the pass way to avoid that 'inspected' area. Another interesting issue is user authentication. Since sensed data is critical and confidential, only authorized person should be allowed to access. This will be further studied as well.

Bibliography

[1] M. Weiser. The computer for the twenty-first century. Scientific American, 265(3):94{10, 1991.

[2] HomeRF Project. http://www.homerf.org 1999

[3] I.F. Akyidiz et al, "Wireless Sensor Network: A survey", *computer networks*, Vol3.38, pp.393-422, March 2002

[4] K. Akkaya, M. Younis "A Survey on Routing Protocols for Wireless Sensor Networks" Elsevier Ad Hoc Network Journal, 2005, vol 3, pp. 325-349

[5] Yan Yu, Ramesh Govindan, Deborah Estrin: Geographical and Energy Aware Routing: a recursive data dissemination protocol for wireless sensor networks (2001). UCLA Computer Science Department Technical Report UCLA/CSD-TR-01-0023, May 2001

[6] Y. Xu, J. Heidemannn, and D. Estrin. "Geography-informed energy conservation for ad hoc routing". *In Proc. of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2001)*, Rome, Italy, July 2001.

[7] S. Singh and C.S. Raghavendra. "Pamas: Power aware multi-access protocol with signalling for ad hoc networks". *ACM CCR,* July 1998.

[8] S. Tilak et al., "A Taxonomy of Wireless Microsensor Network Models" *in ACM Mobile Computing and Communications Review (MC2R)*, June 2002.

[9] L.Robinson. Japan's new mobile broadcast company: Multimedia for cars, trains, and hand-helds. In *Advance Imaging*, page 18-22, July 1998.

[10] Bluetooth Project. http://www.bluetooth.com, 1999.

[11] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, F. Silva. "Directed diffusion for wireless sensor networking" *Networking, IEEE/ACM Transactions* on Volume: 11 Issue: 1 , Feb. 2003 Page(s): 2 -16.

[12] Fan Ye, Haiyun Luo, Jerry Cheng, Songwu Lu, Lixia Zhang. "Sensor Networks: A two-tier data dissemination model for large-scale wireless sensor networks" *Proceedings of the Eighth Annual ACM/IEEE International Conference on Mobile Computing and Networks (MobiCOM 2002)*, Sept 2002, Atlanta, GA

[13] Joanna Kulik, Wendi Heinzelman, Hari Balakrishnan. "Negotiation-based protocols for disseminating information in wireless sensor networks" *ACM Transaction* on Volume 8, Issue 2/3 March-May 2002.

[14] F. Ye, S. Lu, L Zhang. "GRAdient Broadcast: A Robust, Long-lived, Large Sensor Network" http://irl.cs.ucla.edu/papers/grab-tech-report.ps, 2001

[15] Wensheng Zhang; Guohong Cao; La Porta, T. "Data dissemination with ring-based index for wireless sensor networks" Network Protocols, 2003. *Proceedings. 11th IEEE International Conference* on , 4-7 Nov. 2003 Pages:305 – 314

[16] G. J. Pottie and W. J. Kaiser. "Embedding the internet: wireless integrated network sensors". *Communications of the ACM*, 43(5):51–58, May 2000.

[17] M. Stemm and R.H Katz. "Measuring and reducing energy consumption of network interfaces in hand-held devices". *IEICE Transaction and communication*, E80-B(8): 1125-1131, Aug. 1997

[18] Y.Xu, J.Hendemann, and D.Estrin. "Adaptive energy-conserving routing for multihop ad hoc networks". Technical Report TR-2000-527, USC/Information Sciences Institute, Oct. 2000. Available at ftp://ftp.isi.edu.isis-pubs/tr-527.pdf

[19] Y. Xu, J. Heidemannn, and D. Estrin. "Geography-informed energy conservation for ad hoc routing". *In Proc. of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2001)*, Rome, Italy, July 2001.

[20] S.T. Hedetniemi, S.M. Heterniemi, and A. Liestman, "A survey of gossiping and broadcasting in communication networks" networks, vol. 18, pp.129-134, 1998.

[21] I.Krontiris, F. Nikakis "Secure and Efficient Data Delivery in Sensor Networks" MS thesis

[22] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva " A Performance Comparison of Multi-hop Wireless Ad hoc network routing protocols". In Proc 4th ACM International Conference on Mobile Computing and Networking (Mobilecom'98) October 1998

[23] X. Lin and I. Stojmenovic. "Power-Aware routing in ad hoc wireless networks" In SITE, University of Ottawa, TR-98-11, December 1998.

[24] S. Park and M. Srivastava. “Power Aware routing in sensor networks using Dynamic Source Routing” In ACM MONET Special Issue on Energy Conserving Protocol in Wireless Networks, 1999

[25] S. Singh, M. Woo, and C.S. Raghavendra “Power-aware routing in mobile ad hoc networks”. In Proceeding of the Fourth Annuanl ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom ’98), October 1998.

[26] T.Kwon and M.Gerla “Clustering with Power Control” IN Proceeding MILCOM ’99, volume 2, November 1999

[27] V. Garg and J. Wilkes “Wireless and Personal Communications Systems” Prentice Hall PTR, New Jersey, 1996

[28] K. Pahlavan and A. Levesque “Wireless Information Networks” John Willey & Son, Inc, New York, 1995

[29] D. Baker, A. Ephremides, and J. Flynn “The design and simulation of mobile radio network with distributed control” IEEE Journal on Selected Areas in Communications, 2(1):226-237, January 1984.

[30] C. Lin and M. Gerla “Adaptive Clustering for mobile wireless networks” IEEE Journal on Selected Areas in communications 15(7):1265-1275, September 1997.

[31] A. McDonald and T. Znati. “A mobility-based framework for adaptive clustering in wireless ad hoc networks” IEEE journal on selected area in communications 17(8): 1466-1486, August 1999.

[32] W.B. Heinzelman “Application-specific protocol architecture for wireless networks” Ph.D. dissertation, Dept of Electrical Engineering and Computer Science, MIT.

[33] Karlof, C.; Wagner, D. Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures. In Proceeding of First IEEE International Workshop on Sensor Network Protocols and Applications (SPNA), AK, USA, May 2003; pp. 113-127.

[34] Wood,A.D.; Stankovic,J.A. Denial of service in sensor networks. Computer, 2002, 35(10), 54–62.

[35] Douceur, J. R. The Sybil attack, In Proceeding of 1st International Workshop on Peer-to-Peer Systems, MA, USA, Mar, 2002; pp. 251– 260.

[36] Hu, Y.-C.; Perrig, A.; Johnson, D. B. Wormhole detection in wireless ad hoc networks. Technical Report TR01384, Department of Computer Science, Rice University, June 2002.

[37] Yin, J.; Madria, S.K. SecRout: A Secure Routing Protocol for Sensor Networks. In Proceeding of the 20th International Conference on Advanced Information Networking and Applications (AINA), Vienna, Austria, April 2006; pp. 393-398.

[38] Nasser, N.; Chen, Y. SEEM: Secure and energy-efficient multipath routing protocol for wireless sensor networks. Comput. Commun., 2007, 30, 2401-2412.

[39] Lee, S.B; Choi, Y.H. A secure alternate path routing in sensor networks. Comput Commun. 2006, 30,153-165.

[40] Xiaojiang D., Mohsen G., Yang X., Hsiao-Hwa C. Two Tier Secure Routing Protocol for Heterogeneous Sensor Networks. IEEE Trans. Wirel. Commun., 2007, 6(9), 3395-3401.

[41] Kim, H.S.; Abdelzaher, T.F.; Kwon, W.H. Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks. In Proceedings of the First ACM International Conference on Embedded Networked Sensor Systems, CA, USA, November 2003; pp.193-204.

[42] Hung L.X.; Lee S.Y.; Lee Y.-K; Lee H.J. SCODE: A Secure Coordination-based Data Dissemination to Mobile Sinks in Wireless Sensor Networks. IEICE Trans. Commun., 2009, E92-B(01), -.

[43] Karlof, C. ; Sastry, N.; Dagner, D. TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. In Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys04), Baltimore, Maryland, November 2004; pp. 162-175.

[44] Chen, G. and Branch, J. and Pflug, M.J. and Zhu, L. and Szymanski, B., SENSE: A Sensor Network Simulator, Advances in Pervasive Computing and Networking, Springer, New York, NY, 2004, pp. 249-69.

[45] NS-2, http://www.isi.edu/nsnam/ns/.

[46] Q. Xue and A. Ganz, “Runtime security composition for sensor networks” in Proc. IEEE Veh. Technol. Conf., Oct. 2003, pp. 105–111.

[47] T. S. Rappaport. Wireless communications, principles and practice. Prentice Hall, 1996.

[48] Liu, D.; Ning, P. Improving Key Predistribution with Deployment Knowledge in Static Sensor Networks. ACM Trans. Sens. Netw., 2005, 1(2), 204–239.

[49] Du, W.; Deng, J.; Han, Y.; Varshney, P. A Key Predistribution Scheme for Sensor Networks Using Deployment Knowledge. IEEE Trans. Depend. Secure, 2006, 3(1), 62–77.

[50] Blundo, C.; Santis, A.D.; Herzberg, A.; Kutten, S.; Vaccaro, U.; Yung, M. Perfect-Secure Key Distribution of Dynamic Conferences. In Proceeding of Advances in Cryptography, BalatonfÜred, Hungary; pp. 471–486.

[51] L. Eschenauer, V. D. Gligor. "A key-management scheme for distributed sensor networks", In Proceedings of the 9th ACM conference on Computer and communications security, November 2002.

[52] H. Chan, A. Perrig, D. Song. "Random key predistribution schemes for sensor networks" In Proceedings of the 2003 IEEE Symposium on Security and Privacy, May 2003.

[53] W. Du; J. Deng; Y. S. Han; S. Chen; P.K Varshney. ”A key management scheme for wireless sensor networks using deployment knowledge”, In 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom’04), Hong Kong, China, March 21-25, 2004.

[54] R. Blom, “An optimal class of symmetric key generation systems”, In Proc of EUROCRYPT ’84, pp. 334-338, 1985.

[55] W. Du, J. Deng, Y. S. Han, P. Varshney. "A Key Predistribution Scheme for Sensor Networks Using Deployment Knowledge". In IEEE Transactions on Dependable and Secure Computing, Volume 3 , Issue 1 (January 2006).

[56] W. Du, J. Deng, Y. S. Han, P. K. Varshney. "A pairwise key predistribution scheme for wireless sensor networks", In Proceedings of the 10th ACM conference on Computer and communications security, 2003.

[57] D. Liu, P.Ning. "Establishing pairwise keys in distributed sensor networks", In ACM Transactions on Information and System Security, February 2003.

[58] D. Liu; P. Ning. “Location-based pairwise key establishments for static sensor networks”, In Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks (SASN '03), October 2003.

[59] D. Liu, P.Ning. "Improving Key Predistribution with Deployment Knowledge in Static Sensor Networks", In ACM Transactions on Sensor Networks, Vol. 1, No. 2, November 2005, pp. 204-239.

[60] Ngo Trong Canh, Tran Van Phuong, Young-Koo Lee, Sungyoung Lee, and Heejo Lee. "A Location-aware Key Predistribution Scheme for Distributed Wireless Sensor Networks," In The 15th IEEE International Conference on Networks (ICON ’07), pp.188-193, 19-21 Nov. 2007.

[61] X. Lin and I. Stojmenovic. “Power-Aware routing in ad hoc wireless networks” In SITE, University of Ottawa, TR-98-11, December 1998.

[62] S. Park and M. Srivastava. “Power aware routing in sensor networks using Dynamic Source Routing” In ACM MONET Special Issue on Energy Conserving Protocol in Wireless Networks, 1999.

[63] Gatzianas, M.; Georgiadis, L. A Distributed Algorithm for Maximum Lifetime Routing in Sensor Networks with Mobile Sink. IEEE Transaction on Wireless Communications, 2008, Vol 7(3), pp. 984-994.

[64] Al-Karaki, J. N. and A. E. Kamal, "Routing Techniques in Wireless Sensor Networks: A Survey", to appear in IEEE Wireless Communications.

[65] G. Pottie and W. Kaiser, “Wireless integrated network sensors,” Communications of the ACM, vol. 43, pp. 51–58, May 2000.

[66] Chen, B.; Jamieson, K.; Balakrishnan, H.; Morris, R. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. Wireless Networks, 2002, 8(5), 481-494.

[67] P.V. Nikitin and K.V.S. Rao, “Theory and Measurement of Backscattering From RFID Tags,” IEEE Antennas and Propagation, vol. 48, No. 6, December 2006, pp. 212-216.

List of publications

Patents

[1] Sungyoung Lee, Young-Koo Lee, Heejo Lee, Le Xuan Hung. **Controlling Hospital Employee's Access to Hospital Information**. Korea (Domestic) Patent, Application No. 10-2008-0062962.

Journals

[1] Le Xuan Hung, Sungyoung Lee, Young-Koo Lee, and Heejo Lee. **SCODE: A Secure Coordination-based Data Dissemination for Mobile sinks in Sensor Networks**. IEICE Transaction on Communication, 2009, Vol E92-B(01) (SCI)

[2] Le Xuan Hung, Ngo Trong Canh, Sungyoung Lee, Young-Koo Lee, and Heejo Lee. **An Energy-Efficient Secure Routing and Key Management Scheme for Mobile Sinks in Wireless Sensor Networks using Deploying Knowledge**. Journal of Sensor, Special Issue "Wireless Sensor Technologies and Applications", 2008, Vol.8(11) (SCIE, Impact Factor 1.573) (to appear).

[3] Le Xuan Hung, Sungyoung Lee, Young-Koo Lee, and Heejo Lee. **Activity-Oriented Access Control Scheme to Hospital Information and Services**. IEEE Transaction on IEEE Transaction on Information Technology in Biomedicine (SCI, IF 1.5) (conditional acceptance).

[4] Le Xuan Hung, Sungyoung Lee, Young-Koo Lee, and Heejo Lee. **An Energy-Efficient Distributed User Authentication Scheme for Heterogeneous Sensor Networks**. IEICE Transaction on Communication (SCI) (under review).

Conferences (22)

[1] Le Xuan Hung, Hassan J., Riaz A.S , S.M.K. Raazi, Y. Weiwei, NT Canh, P.T.H. Truc, Sungyoung Lee, Heejo Lee, Yuseung Son, Miguel Fernandes, Miso (Hyoung-IL) Kim, Yonil Zhung. Activity-based Security Scheme for Ubiquitous Computing Environments. 27th IEEE International Performance Computing and Communications Conference (IPCCC), Texas, USA, 7-9 December 2008 (to appear).

[2] Le Xuan Hung, Riaz A.S, Hassan J., S.M.K. Raazi, Y. Weiwei, NT Canh, P.T.H.Truc, Sungyoung Lee, Heejo Lee, Yuseung Son, Miguel Fernandes. Activity-Oriented Access Control for Ubiquitous Environments. 6th Annual IEEE Consumer Communications & Networking Conference (CCNC 2009) – Track on Ubiquitous Application & Security Services. Las Vegas Jan 10-13, 2009 (to appear).

[3] Ngo Trong Canh, Le Xuan Hung, P.T.H. Truc, Tran Hoang Hai, Young-Koo Lee and Sungyoung Lee. Enhanced Group-based Key Management Scheme for Wireless Sensor Networks using Deployment Knowledge. will appear in proc. of the 6th Annual IEEE Consumer Communications & Networking Conference (IEEE CCNC 2009), Las Vegas, Nevada, USA, Jan 2009 (to appear)

[4] Hassan J., Le Xuan Hung, Riaz A.S, S.M.K. Raazi, Y. Weiwei, NT Canh, Sungyoung Lee, Yuseung Son, Miguel Fernandes. Image-Feature based Human Identification Protocols on Limited Display Devices. Workshop on Information Security Applications. Jeju Island, Korea September 23-25, 2008.

[5] Le Xuan Hung, Sungyoung Lee, Young-Koo Lee, and Heejo Lee. Activity-centered Access Control to Hospital Information. Proceeding of 13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications August (RTCSA 2007) September 21-24, 2007. Daegu, Korea pp. 488-496

[6] Pho Duc Giang, Le Xuan Hung, Riaz Ahmed Shaikh, Sungyoung Lee, Young-Koo Lee and Heejo Lee. A Trust-Based Approach to Control Privacy Exposure in Ubiquitous Computing Environments. IEEE International Conference on Pervasive Services (ICPS 2007), July 15-20, Istanbul, Turkey pp. 149-152.

[7] Nguyen Ngoc Diep, Le Xuan Hung, Yonil Zhung, Syngyoung Lee Young-Koo Lee and Heejo Lee. Enforcing Access Control Using Risk Assessment. 4th European Conference on Universal Multiservice Networks (ECUMN) 14-16 February, 2007 - France pp. 419-424

[8] Pho Duc Giang, Le Xuan Hung, Sungyoung Lee, Young-Koo Lee, Heejo Lee. A Flexible Trust-Based Access Control Mechanism for Security and Privacy Enhancement in Ubiquitous Systems. International Conference on Multimedia and Ubiquitous Engineering 2007

[9] Phuong Van Tran, Le Xuan Hung, Young-Koo Lee, Heejo Lee, Sungyoung Lee, "TTM: An Efficient Mechanism to Detect Wormhole Attacks in Wireless

Ad-hoc Networks". IEEE Consumer Communications and Networking Conference, Wireless Sensor Network Track (CCNC), Las Vegas, USA, Jan 11-13, 2007

[10] Weiwei Yuan, Donghai Guan, Le Xuan Hung, Sungyoung Lee, and Young-Koo Lee. A Trust Model with Dynamic Decision Making For Ubiquitous Environments. The 14th IEEE International Conference on Networks (ICON2006), Singapore, Sep 13-15, 2006

[11] Le Xuan Hung, Tran Van Phuong, Pho Duc Giang, Yonil Zhung, Sungyoung Lee and Young-Koo Lee. Security for Ubiquitous Computing: Problems and Proposed Solution. The 12th IEEE International Conference on Embedded and Real-Time Computing Systems and Application (RTCSA 2006), Sydney. 16-18 August, 2006

[12] Le Xuan Hung, Pho Duc Giang, Yonil Zhung, Tran Van Phuong, Sungyong Lee and Young-Koo Lee. A Trust-based Security Architecture for Ubiquitous Computing Systems. IEEE Intelligent and Security Informatics (ISI-2006), May 23-24, 2006 San Diego, USA. ISBN: 3-540-34478-0. LNCS Vol. 3975/2006pp 753-754

[13] Le Xuan Hung, Nguyen Ngoc Diep, Yonil Zhung, Sungyoung Lee and Young-Koo Lee. A Flexible and Scalable Access Control for Ubiquitous Computing Environments. IEEE Intelligent and Security Informatics (ISI-2006), May 23-24, 2006 San Diego, ISBN: 3-540-34478-0. LNCS Vol. 3975/2006 pp 688-689

[14] Tran Van Phuong, Le Xuan Hung, Seong Jin Cho, Sungyoung Lee, and Young-Koo Lee. An Anomaly Detection Algorithm for Detecting Attacks in Wireless Sensor Networks. IEEE Intelligent and Security Informatics (ISI-2006), May 23-24, 2006 San Diego, USA. ISBN: 3-540-34478-0. LNCS Vol. 3975/2006. pp 735-736

[15] Pho Duc Giang, Le Xuan Hung, Yonil Zhung, Sungyoung Lee, and Young-Koo Lee. A Home Firewall Solution for Securing Smart Spaces. IEEE Intelligent and Security Informatics (ISI-2006), May 23-24, 2006 San Diego, USA. ISBN: 3-540-34478-0. LNCS Vol. 3975/2006. pp 760-761

[16] Le Xuan Hung, Hassan Jammeel, Seong Jin Cho, Yuan Weiwei, Sungyoung Lee and Young-Koo Lee, A Trust Model for Uncertainty in Ubiquitous Environments.. IEEE Intelligent and Security Informatics (ISI-2006), May 23-

24, 2006 San Diego, USA. ISBN: 3-540-34478-0. LNCS Vol. 3975/2006pp 755-757

[17] Le Xuan Hung, Hassan Jameel, Sungyoung Lee, and Young-Koo Lee. Modeling Trust in Ubiquitous Environment. FINA 2006, in conjunction with the IEEE 20th International Conference on Advanced Information Networking and Applications (AINA 2006) 18-20 April 2006 in Vienna, Austria.

[18] Hung Le Xuan and Sungyoung Lee. Two Energy-Efficient Routing Algorithms for Wireless Sensor Networks. The 4th International Conference on Networking (ICN'05) Reunion Island April 17-21, 2005. ISBN 3-540-25339-4, LNCS Vol 3420/2005 pp. 698-705

[19] Hassan Jameel, Le Xuan Hung, Umar Kalim, Ali Sajjad, Sungyoung Lee and Young-Koo Lee. A Trust Model for Ubiquitous Systems based on Vectors of Trust Values. IEEE International Symposium on Multimedia (ISM2005). San Francisco, California USA, December 13, 2005. ISBN:0-7695-2489-3, pp 674 ? 679

[20] Le Xuan Hung, Sungyoung Lee and Young-Koo Lee. A Key-Exchanging Scheme for Distributed Sensor Networks. The 2005 IFIP International Conference on Intelligence in Communication Systems (INTELLCOMM 05), Canada. October 17-19, 2005. ¨ÏIFIP, Springer-Verlag, ISBN 0-387-29121-0, pp 271-279

[21] Le Xuan Hung, Dae Hong Seo, Sungyoung Lee, and Young-Koo Lee, "Minimum-Energy Data Dissemination in Coordination-based Sensor Networks", The 11 IEEE International Conference on Embedded and Real-Time Computing Systems and Application (RTCSA), Hong Kong. 17-19 August, 2005

[22] Le Xuan Hung and Sungyoung Lee. A Coordination-based Data Dissemination for Wireless Sensor Networks. The International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP 05) Melbourne, Australia December 14~17, 2004.

List of abbreviations

AODV	Ad hoc On-demand Distance Vector
ADC	Analog-to-Digital Converter
BS	Base Station
CID	Cell Identification
DCF	Distributed Coordination Function
DD	Directed Diffusion
DSDV	Destination-Sequenced Distance Vector
DSR	Dynamic Source Routing
GAF	Geographic Adaptive Fidelity
GEAR	Geographic and Energy Aware Routing
GeRaf	Geographic Random Forwarding
GPS	Global Position System
GRAB	GRAdient Broadcast
GSR	Global State Routing
HSN	Heterogeneous Sensor Network
ID	Identification
KSDM	Key-Space Discovery Message
LEACH	Low Energy Adaptive Clustering Hierarchy
LEAP	Localized Encryption and Authentication Protocol
MAC	Message Authentication Code

	Medium Access Control
NS-2	Network Simulator 2
PDA	Personal Digital Assistant
QoS	Quality-of-Service
RC5	Rivest Cipher 5 algorithm
SCODE	Secure Coordination-based Data dissemination
SecRout	Secure Routing
SEEM	Secure and Energy-Efficient Multipath routing
SENSE	Sensor Network Simulator and Emulator
SODD	Sink-Oriented Data Dissemination
SPIN	Sensor Protocols for Information via Negotiation
TinyOS	Tiny Operating System
TTDD	Two-Tier Data Dissemination
TTSR	Two-Tier Secure Routing
XOR	Exclusive OR operation
WSN	Wireless Sensor Network