SurvSec Security Architecture for Reliable Surveillance WSN Recovery from Base Station Failure

by

Mohamed Helmy Mostafa Megahed

Thesis submitted to the

Faculty of Graduate and Postdoctoral Studies

In partial fulfillment of the requirements

For the Ph.D. degree in Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering

Faculty of Engineering

University of Ottawa

©Mohamed Helmy Mostafa Megahed, Ottawa, Canada, 2014

ABSTRACT

Surveillance wireless sensor networks (WSNs) are highly vulnerable to the failure of the base station (BS) because attackers can easily render the network useless for relatively long periods of time by only destroying the BS. The time and effort needed to destroy the BS is much less than that needed to destroy the numerous sensing nodes.

Previous works have tackled BS failure by deploying a mobile BS or by using multiple BSs, which requires extra cost. Moreover, despite using the best electronic countermeasures, intrusion tolerance systems and anti-traffic analysis strategies to protect the BSs, an adversary can still destroy them. The new BS cannot trust the deployed sensor nodes. Also, previous works lack both the procedures to ensure network reliability and security during BS failure such as storing then sending reports concerning security threats against nodes to the new BS and the procedures to verify the trustworthiness of the deployed sensing nodes. Otherwise, a new WSN must be re-deployed which involves a high cost and requires time for the deployment and setup of the new WSN. In this thesis, we address the problem of reliable recovery from a BS failure by proposing a new security architecture called Surveillance Security (SurvSec).

SurvSec continuously monitors the network for security threats and stores data related to node security, detects and authenticates the new BS, and recovers the stored data at the new BS. SurvSec includes encryption for security-related information using an efficient dynamic secret sharing algorithm, where previous work has high computations for dynamic secret sharing. SurvSec includes compromised nodes detection protocol against collaborative work of attackers working at the same time where previous works have been inefficient against collaborative work of attackers working at the same time.

SurvSec includes a key management scheme for homogenous WSN, where previous works assume heterogeneous WSN using High-end Sensor Nodes (HSN) which are the best target for the attackers. SurvSec includes efficient encryption architecture against quantum computers with a low time delay for encryption and decryption, where previous works have had high time delay to encrypt and decrypt large data size, where AES-256 has 14 rounds and high delay. SurvSec consists of five components, which are:

1. A Hierarchical Data Storage and Data Recovery System.
2. Security for the Stored Data using a new dynamic secret sharing algorithm.
3. A Compromised-Nodes Detection Algorithm at the first stage.
4. A Hybrid and Dynamic Key Management scheme for homogenous network.
5. Powerful Encryption Architecture for post-quantum computers with low time delay.

In this thesis, we introduce six new contributions which are the followings:

1. The development of the new security architecture called Surveillance Security (SurvSec) based on distributed Security Managers (SMs) to enable distributed network security and distributed secure storage.
2. The design of a new dynamic secret sharing algorithm to secure the stored data by using distributed users tables.
3. A new algorithm to detect compromised nodes at the first stage, when a group of attackers capture many legitimate nodes after the base station destruction. This algorithm is designed to be resistant against a group of attackers working at the same time to compromise many legitimate nodes during the base station failure.
4. A hybrid and dynamic key management scheme for homogenous network which is called certificates shared verification key management.
5. A new encryption architecture which is called the spread spectrum encryption architecture SSEA to resist quantum-computers attacks.
6. Hardware implementation of reliable network recovery from BS failure.

The description of the new security architecture SurvSec components is done followed by a simulation and analytical study of the proposed solutions to show its performance.

ACKNOWLEDGMENTS

I would like to express my gratitude and heartiest thanks to my supervisors, Prof. Dimitrios Makrakis and Prof. Hussein Mouftah. Their breadth of knowledge, vision for the future, and enthusiasm for research has been an inspiration to me. Prof. Dimitrios Makrakis and Prof. Hussein Mouftah are motivators, facilitators, challengers, and above all good friends. Their supervision has been invaluable and my life has been enriched personally, intellectually and professionally by working with them.

Again, I would like to thank Prof. Dimitrios Makrakis and Prof. Hussein Mouftah for their insights into sensor networks. Their advice in setting up my research direction was a great help, while their encouragement, criticism and feedback have greatly enhanced and strengthened my research. My thanks also go to the staff and my fellow students at the lab.

It has been a privilege interacting with these wonderful, bright and talented people whose advice, feedback and friendship have made my PhD experience educational, especially Prof. Carlisle Adams, Prof. Ashraf Matrawy, Dr. Benod, Dr. Bidi Ying and Dr. Jose.

I would like to thank my parents, Helmy and Nagwa, and my brother, Ahmed, and my sister, Amal who encouraged me to finish my PhD.

Also, I would like to thank Col. Sherif El Shemy and Major General Mohamed El Keshky for everything they have done for me.

Finally, I would like to thank my colleagues and friends in Egypt, who have worked with me on my dissertation, as they must also be noted. These include but are not limited to Brigadier General Essam Abdel Waness, and Col. Hisham Dahshan.

Table of Contents

List of Tables

List of Figures

List of Symbols, Abbreviations and Nomenclature

Symbol	Definition
ANMP	Ad-hoc Network Management Protocol
BS	Base Station
CA	Certificate Authority
ECC	Elliptic Curve Cryptography
h	Hash
ID_{CA}	Identity of Certificate Authority
K	Number of Compromised Sensor Nodes
K_L	Individual Key Length
LEAP	Localized Encryption and Authentication Protocol
LFSR	Linear Feedback Shift Register
LIDS	Local Intrusion Detection System
MAC	Message Authentication Code
N	Number of Sensor Nodes
N_{CH}	Number of Cluster Heads
N_S	The Number of Nodes Under Security Manager
N_{SEC}	Number of Security Managers
PKI	Public Key Infrastructure
PRNG	Pseudo Random Number Generator
QC	Quantum Computer
Q_{CA}	CA Public Key
q_{CA}	CA Private Key
QoSS	Quality of Security Service
R	Number of Rounds
SKKE	Symmetric Key Key Exchange
SM	Security Manager
SNMP	Simple Network Management Protocol
sNMP	Sensor Network Management Protocol
SPN	Substitution Permutation Network
SS	Secret Sharing
SSEA	Spread Spectrum Encryption Architecture
SurvSec	Surveillance Security
WSN	Wireless Sensor Network
WSNMP	Wireless Sensor Network Management Protocol
A	The Percentage of Sleep Nodes

List of Publications

[1] Mohamed Megahed, Dimitrios Makrakis, and Bidi Yang "SurvSec: A New Security Architecture for Reliable Network Recovery from Base Station Failure of Surveillance WSN", the 2nd International Conference on Ambient Systems Networks and Technologies, ANT 2011, September 17-19, 2011, Niagara Falls, Canada.

[2] Mohamed Megahed, and Dimitrios Makrakis, "Secure Network Recovery from Base Station Failure of Surveillance WSN in Hostile Environment", the 14th International Conference on Aerospace Sciences& Aviation Technology, ASAT – 14, May 24 – 26, 2011, Cairo, Egypt.

[3] Mohamed Megahed, Dimitrios Makrakis, and Bidi Yang "SurvSec: A New Security Architecture for Reliable Network Recovery from Base Station Failure of Surveillance WSN", ELSEVIER Procedia Computer Science Journal, Volume 5, September 2011, Pages 141-148.

[4] Mohamed Megahed, and Dimitrios Makrakis, "Location based, Hybrid and Dynamic Key Management Scheme for SurvSec Security Architecture", 6th International Scientific Conference of the Military Technical College, ICEENG, 29-31 May 2012.

[5] Mohamed Megahed, and Dimitrios Makrakis, "Overlapped Groups-based Compromised Nodes Detection for SurvSec Security Architecture", 6th International Scientific Conference of the Military Technical college, ICEENG, 29-31 May 2012.

[6] Mohamed Megahed, Dimitrios Makrakis and Hisham Dahshan, "Distributed Compromised Nodes Detection Scheme at First Stage for SurvSec Security Architecture", SENSORCOMM 2012, the Sixth International Conference on Sensor Technologies and Applications.

[7] Mohamed Megahed, Dimitrios Makrakis and Hisham Dahshan, “Certificates Shared Verification Key Management for SurvSec Security Architecture”, SENSORCOMM 2012, the Sixth International Conference on Sensor Technologies and Applications.

[8] Mohamed Megahed, and Khaled Hussein, “Hardware Implementation of Reliable Network Recovery from Base Station Failure of Surveillance WSN”, International Journal of Engineering Research and Technology (IJERT) Journal, Volume 2, Issue 12, December 2013, Pages 3293-3308.

CHAPTER 1

INTRODUCTION

In this chapter, we offer an overview of the research area of interest. In particular, we provide a statement of the problem that we have tackled throughout this thesis. We also provide a brief sketch of our solution for the stated problem. We mention the contributions that we have made to this field of research. And finally, we provide the outline of the thesis.

1.1 Overview

In the past few years, wireless sensor networks (WSNs) have seen considerable and still growing interest from the scientific and engineering communities due to their potential use in many applications such as monitoring and disaster management.

Sensors are empowered with limited data processing engines and storage units and in the majority of cases their batteries cannot be changed, making them power constrained. In many cases, collected information has to go through several to many hops before reaching the sink (usually called a "base station"). Also, due to the many limitations of WSNs, data have to be sent simultaneously through multiple multi-hop paths before reaching the base station, in order to reach a certain level of reliability (i.e. data delivery rate). Sensor nodes are used to probe their surroundings and report any abnormal events over wireless communication links, often over multi-hop paths, to the base station (BS).

Many WSNs deployments are mission' critical, such as surveillance [1] and one of the key challenges which WSN needs to address is security. The general objective of such an application is to alert the control unit in advance to the occurrence of events of interest in hostile regions. The event of interest will vary according to its mission which might be the presence of moving vehicles or target detection or other events where there are several types of sensors such as Vibration, Motion, Tracking, Video, and Infrared sensors which can be used for surveillance applications [2]. Obviously, for successful detection and tracking for surveillance WSN requires that the application obtain the current position of the target in the region of interest with acceptable precision and confidence then, this information has to be reported to the BS within an acceptable latency [3]. However, surveillance WSN application requirements are longevity, adjustable sensitivity, stealthiness, security, effectiveness, fault tolerance and reliable recovery from failure [4], [5]. With their deployment, various novel security attacks have appeared. The aims of these attacks are usually to compromise legitimate nodes, eavesdropping, traffic analysis, physical attacks or to disrupt data flow.

We believe that, the worst attack scenario will be done by a group of attackers. First, they will launch physical attacks against the BSs including jamming and destruction then they will compromise many legitimate nodes during the BSs failure to control the deployed network security and to cover their unauthorized intrusions.

1.2 Research Motivations and Objectives

The Base Station is a critical part of a WSN and an entire WSN can be rendered useless by taking down its BS. Since the BS is a single point of failure, once its location is revealed, an adversary can jam or destroy it, thereby rendering ineffective the entire WSN. Physical attacks against BS are the most efficient and dangerous attacks, since the energy, time, and effort needed to destroy a small number of BSs are much less than that needed to destroy large number of sensor nodes.

1.2.1 Motivations

Our motivations are the followings:

1- The high probability of BS failure as a single point of failure to render the whole WSN ineffective and hence reduce the WSN lifetime where physical attacks towards BS specifically target the reduction of the WSN life time because the new deployed BS does not trust the deployed sensor nodes of the WSN. Therefore, we designed new security architecture for reliable network recovery from base station failure.

2- The high computational power of dynamic secret sharing algorithms to secure the stored data on WSN.

3- The high probability for group of attackers to compromise many legitimate nodes from the surveillance WSN. Therefore, we designed compromised nodes detection algorithm at the first stage resistant to group of attackers.

4- The usage of public key based key management scheme has drawbacks of high computational complexity and the usage of symmetric key based key management scheme has large class of attacks such as man-in-the middle attack. Therefore, we designed hybrid key management scheme and it is also dynamic key management scheme using our new certificates shared verification for fast key management.

5- The rapid speed towards a building quantum computer increases the probabilities towards breaking symmetric key cipher systems and asymmetric key cipher systems. Therefore, we designed the spread spectrum encryption architecture which is resistant to quantum computer attacks with high speed.

1.2.2 Objectives

Our goal is to design a new security architecture called surveillance security (SurvSec) to solve the problem of BS failure of surveillance WSN in hostile environment where SurvSec includes the followings:

1- Reliable WSN recovery from BS failure of surveillance WSN in hostile environment by enabling the newly deployed BS to trust the deployed sensor nodes. SurvSec deploys the concept of network distributed security for the Surveillance WSN by proposing hierarchical Security Managers (SM) within the hierarchical layered WSN

architecture to resemble the distributed security of the cellular mobile networks. Each SM will securely store the security related information for its downstream nodes of the hierarchical architecture,

2- Continuously storing security related data of the network sensor nodes using a designed dynamic secret sharing algorithm that uses distributed users tables,
3- Designing of a new compromised nodes detection algorithm to detect compromised nodes at the first stage against collaborative work of attackers working at the same time to capture and compromise nodes.
4- Developing hybrid and dynamic key management scheme based on hybrid key management scheme to resist compromised node attack, and collusion attack.
5- Developing of new encryption architecture called spread spectrum encryption architecture to resist quantum computer attacks with high speed.

1.3 SurvSec Five Phases

SurvSec is not comprehensive security architecture. SurvSec has only five components which are the followings: reliable network recovery from BS failure, secure security reports storage using new dynamic secret sharing algorithm, new compromised nodes detection algorithm, new hybrid and dynamic key management system and the spread spectrum encryption architecture for post quantum computing. SurvSec has five phases:

1- First phase, continuous secure storage of security related data of sensor nodes,
2- Second phase, BS failure where the last layer nodes near the BS of the hierarchical WSN architecture cannot listen to the BS periodic beacons.
3- Third phase, authentication of the newly deployed BS,
4- Fourth phase, reliable network recovery from BS failure to enable the newly deployed BS to trust the deployed sensor nodes. If the new BS does not trust the network sensor nodes, the network administrator has to follow the order of two expensive solutions:
    a- First; he must test the whole network sensor nodes using software-based attestation to verify the memory contents of the nodes to detect malicious nodes

and hence revoke them. This solution is expensive in terms of the time and energy required.

b- Second; he needs to deploy a new WSN if a large number of sensor nodes are found to be malicious. This solution is expensive in terms of the time and WSN money cost required.

5- Fifth phase, security threats recovery to delete its stored security related data.

Indeed, it is crucial to protect a BS against both software-based and physical attacks. Several intrusion tolerant techniques have been developed to protect a BS against software-based remote attacks such as denial of service (DoS) attacks that flood the BS with packets, and remote spoofing of the BS to misdirect legitimate sensor data [6].

Software-based techniques cannot protect BS against physical attacks. Therefore, some works have been done to address the problem of protecting a BS against physical attacks through concealing its geographic location in the network [7]. Monitoring and analyzing the volume and the direction of packets traffic towards the BS can reveal the direction towards BS and hence the location of the BS [8]. The higher layer nodes near the BS forward a greater volume of packets than the lower layer nodes away from the BS where BS is located at the boundary of the field.

A relevant question in the above approach of reliable recovery from BS failure of surveillance WSN is that whether we can use a secure multi-path routing to multiple destination static or mobile BSs to defend against these attacks and to provide fault tolerance against BS failure. The attackers can destroy all base stations.

Also, one of the most important performance metrics for BS security of WSN is the average ratio of connected sensor nodes after the failure of BSs which represents the fault tolerance of the network [9]. This metric will be our future work towards failure of multiple BSs network.

1.4 The Main Problem

Our work considers the worst attack scenario in which a group of attackers' first launch physical attacks against the BS then compromise many legitimate nodes to control

the network security and to cover their unauthorized intrusions thus the new BS cannot trust the deployed sensor nodes. Despite using the best electronic countermeasures, intrusion tolerance systems and anti-traffic analysis strategies to protect the BSs, an adversary still can destroy them.

To the best of our knowledge, no work has been done on securing the surveillance WSN during the time between the BS failure and the new BS deployment. This is the perfect time for a group of attackers to compromise many legitimate nodes and hence destroy the security of the whole WSN. Also, there is not any work that describes how the new BS will verify the trustworthiness of the deployed WSN. Otherwise, a new WSN must be deployed which means high cost and long time for new WSN deployment. Moreover, if the attackers isolate legitimate nodes by means of physical jamming in the absence of the BS, these nodes must be reported as untrusted sensor nodes to the BS and we need to verify their trustworthiness. Therefore, for mission critical applications such as surveillance WSN, if the BS fails, we propose to address this problem for the reliable WSN recovery from single BS failure as shown in Figure 1.1 through employing our new designed security architecture which is called Surveillance Security (SurvSec) to continuously and securely store the security related data of the sensor nodes in multiple replicas, and to send the stored data to the new BS after it is authenticated. Therefore, the **first problem** is BS failure. Stored security reports need to be encrypted for transmission and this is the **second problem** where we use dynamic secret sharing to secure stored data but it has high computational power to securely store the security reports. We need new dynamic secret sharing algorithm with low computational power.

The stored data must be encrypted which is unlike traditional surveillance networks that only transmit using mechanisms that guarantee integrity and authenticity while confidentiality is not required [10]. However, encryption of stored data will prevent eavesdropping and traffic analysis during data recovery after the new BS deployment. BS failure can be alleviated such as work discussed in [11] by the use of multiple base stations as shown in Figure 1.2 where multiple base stations are deployed along the periphery of the field, and allowing each base station to act as a data sink however as mentioned in [9], multiple BSs failure is an important performance metric which must be

considered [9]. Therefore, if the BS failed and the network nodes are not trusted by the new BS, the network must be redeployed. Previous works for compromised nodes detection algorithms cannot work against group of attackers working at the same time to compromise many legitimate nodes. This is the **third problem** where we need compromised nodes detection algorithm at first stage against group of attackers working at the same time to compromise legitimate nodes.

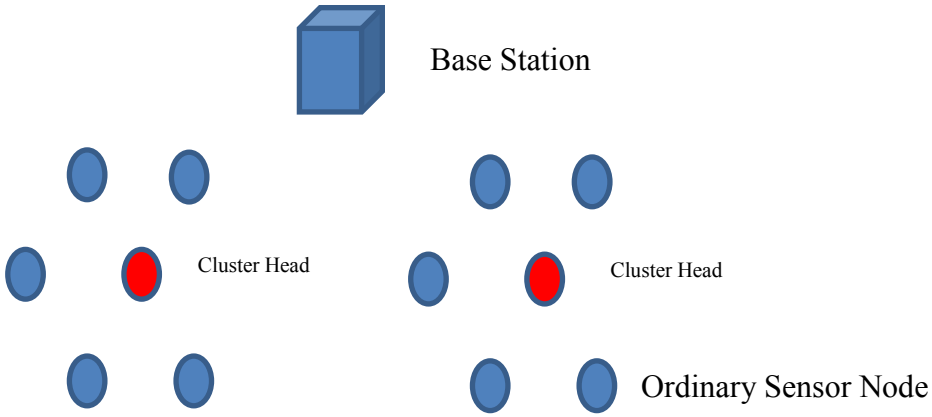


Figure 1.1, Wireless Sensor Network with a Single Base Station

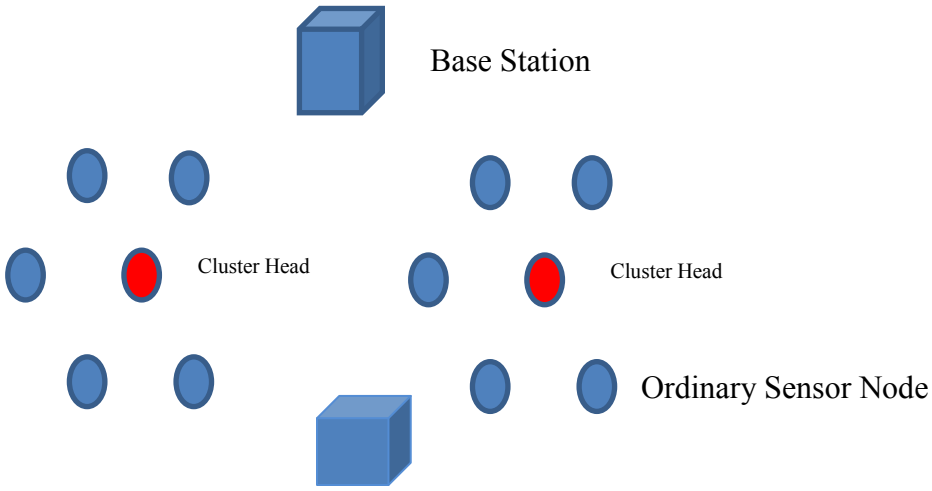


Figure 1.2, Wireless Sensor Network with Multiple Base Stations

We need a new key management system for SurvSec security architecture, where previous works assume High end Sensor Nodes (HSNs) for hybrid key management and these sensor nodes are the best target for the attackers. This is the **fourth problem**.

Therefore, we designed hybrid and dynamic key management for homogenous WSN. We need an efficient high speed encryption architecture for post quantum computing where previous works assume that we use AES-256 which has 14 rounds and it has high delay to encrypt and decrypt large data size and this is the **fifth problem**. Finally, we designed the spread spectrum encryption architecture against the quantum computer with high speed where previous works have high time delay.

1.5 Threat Model

We assume that the sensor nodes are organized in a hierarchical architecture which is a tree-like network routing structure around the base station. The base station is the root node of the tree. Each node has a number of child nodes that are its downstream nodes, and a parent node that is its upstream node. Every sensor node processes the sensed data from all of its child nodes and itself, and sends the result to its parent node. Each node has an activity range. If the distance between two sensor nodes is no more than the activity range, the pair of nodes can send and receive data to and from each others. We assume group of attackers are attacking the WSN.

For the capabilities of the adversaries, we assume that:

1- It is very difficult for the adversaries to obtain sufficient global information to destroy the entire sensor nodes network. Instead, the adversaries are assumed to have knowledge of the wireless sensor network BS location. As a result, the adversaries' threats to the BS are to jam the communication medium, destroy the BS, spoof the BS, or flood the BS. All of previous attacks will result in BS failure.

2- After the BS failure, the adversaries can capture sensor nodes and are capable of compromising sensor nodes to obtain all of their information, e.g. symmetric keys. In addition, the adversaries can reprogram a sensor node to convert it into a malicious node. But we assume that an adversary needs some time to compromise a node.

3- The adversaries have a jamming range. Within the jamming range, the adversaries can generate radio signals to interfere with signals generated by sensor nodes or BS.

4- The adversaries can receive any data from any sensor node or BS, if the distance is less than the activity range. Although it is easy to send a stronger data signal to a larger range than a normal sensor node's range, it is difficult to receive data from a sensor node that is further than the activity range, since it needs very sensitive and expensive equipments.
5- The adversaries are mobile and can physically move from place to place which means high capabilities to compromise many legitimate nodes.
6- The adversaries do not have global information about the whole WSN, and cannot jam the entire network.

1.6 Network Model

We consider a surveillance sensor network that is composed of a large number of sensor nodes with a unique ID. The following assumptions have been made in the formulation of the SurvSec security architecture:

1- All sensor nodes are static.
2- The secure hierarchical data storage and recovery system needs a pre-configuration to allow the base station to choose from the network topology some sensor nodes inside the hierarchical architecture to be security managers (SM), to allow dynamic security for the stored data through dynamic secret sharing between sensor nodes of the hierarchical architecture, and to allow reliable data recovery from the stored data.
3- The compromised nodes detection algorithm needs a pre-configuration to allow overlapped groups formation to protect the network from compromise nodes attack.

1.7 Research Methodology and Results

SurvSec can work with other network rather than WSN in case of base station failure this network can be cloud computing.

1.7.1 Research Methodology

The methodology of this research started with the understanding of surveillance WSN security, current weaknesses and new demands. Then, the existing fault management protocols and security protocols was investigated. Our methodology to conduct the research is similar to the DSR methodology for information systems research [207].

The methodology to conduct our research is divided into four steps which are the followings:

1- Modeling and design of new security architecture for reliable WSN recovery from BS failure of surveillance WSN in hostile environment.
2- Simulation of the designed model.
3- Performance analysis of the designed model.
4- Hardware implementation of reliable network recovery from base station failure.

The new security architecture called SurvSec is designed with its ingredients. MATLAB is used as a simulation tool to see the performance of the new designed security architecture.

MATLAB is used as the whole architecture is designed at the application layer where the whole architecture considers only messages between nodes and operations at nodes. The first step in the development of the SurvSec security architecture is to design hierarchical security managers for reliable network recovery from base station failure. Then we design a secure data storage system using dynamic secret sharing. The second step is done through the simulation of SurvSec secure data storage system. The third step is done through the simulation of SurvSec new compromised nodes detection algorithm at first stage. The fourth step is to simulate SurvSec hybrid and dynamic key management system. The fifth step is to measure the performance of the spread spectrum encryption architecture. Finally, the sixth step is the hardware implementation of reliable network recovery from base station failure.

1.7.2 Results

The result from this research will be the simulation results for new security architecture and hardware implementation of reliable network recovery from base station failure. These results will lead to the development of new security architecture for surveillance wireless sensor network called SurvSec which can provide the followings:

1- New security architecture for reliable network recovery from BS failure of surveillance WSN in hostile environment.
2- New data storage system for security related data using dynamic secret sharing.

3- New Compromised nodes detection scheme against group of attackers at first stage.

4- Hybrid and dynamic key management scheme for homogenous WSN.

5- Spread spectrum encryption architecture for post quantum cryptography.

1.8 Thesis Contributions

In this work, we present a novel recovery approach from BS failure that includes detecting BS failure, continuously monitoring the network security issues to store the sensitive data, and send the stored data to the new BS after deployment to enable efficient recovery from BS failure while maintaining the operation of the network.

In this thesis, we introduce six contributions as follows:

1. The first contribution is the design and simulation of a new security architecture called Surveillance Security (SurvSec) for reliable network recovery from BS failure of surveillance WSN. SurvSec relies on distributed sensor nodes named as security managers (SMs) to securely store the security related data of the sensor nodes in distributed manner within the hierarchical architecture of the surveillance WSN. SurvSec employs Security Managers (SM) to deploy the concept of distributed security for WSN to resemble the cellular mobile networks. SurvSec has hierarchical data storage and data recovery system for the security related data of the sensor nodes.
2. The second contribution is the design of dynamic security system for the stored data through dynamic secret sharing algorithm using distributed users tables.
3. The third contribution is the design and simulation of SurvSec compromised node detection algorithm to detect compromised nodes at the first stage when the attackers capture the nodes. This algorithm is designed to be resistant against collaborative work of attackers to compromise many legitimate nodes at the same time. This new algorithm is designed to be based on the formation of overlapped groups to detect compromised nodes.

4. The fourth contribution is the design of hybrid and dynamic key management scheme for homogenous network resistant to compromised node attack, and collusion attack.
5. The fifth contribution is the design of new encryption architecture called spread spectrum encryption architecture which is a family of encryption architectures designed to resist quantum computer attacks with high speed.
6. The sixth contribution is the hardware implementation of reliable network recovery from base station failure.

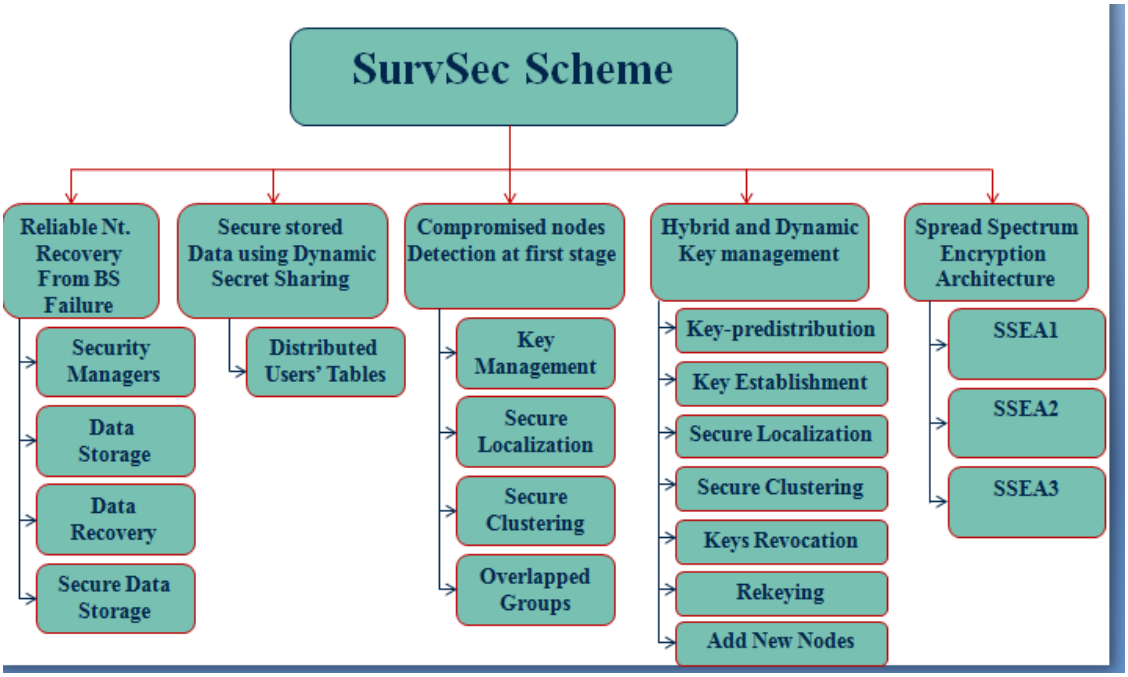


Figure 1.3, SurvSec Components

Figure 1.3 shows the five components of SurvSec security architecture and all of its subcomponents.

The description of the new security architecture SurvSec components is given followed by an analytical analysis of the ingredients of SurvSec to evaluate the performance of our newly designed security architecture.

1.9 Organization of the Thesis

The rest of thesis is organized as shown in Figure 1.4 and the work is done as follows: Chapter 2 presents the background of the surveillance WSN security. Chapter 3 describes the overall design of the new security architecture SurvSec for reliable network recovery from BS failure. Chapter 4 describes SurvSec secure data storage and recovery system to continuously store the security related data of the network sensor nodes. Chapter 5 describes SurvSec compromised nodes detection algorithm. Chapter 6 describes SurvSec hybrid and dynamic key management scheme. Chapter 7 describes SurvSec new encryption architecture which is named spread spectrum encryption architecture. Chapter 8 is the hardware implementation of reliable network recovery from base station failure. Finally, Chapter 9 is the conclusion and future work.

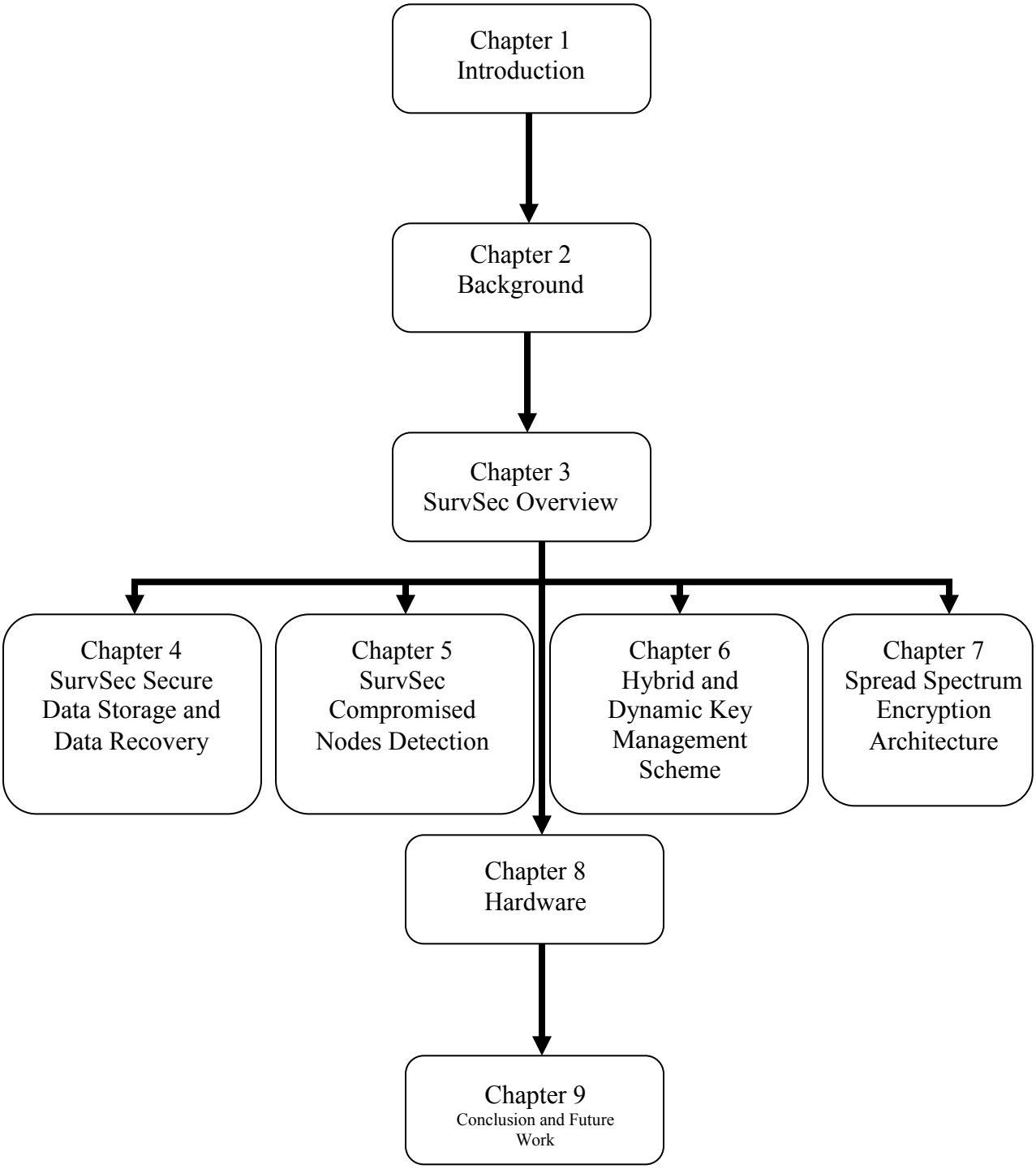


Figure 1.4, Thesis Organization

CHAPTER 2

SURVEILLANCE WSNS SECURITY – BACKGROUND

In this chapter we provide an overview of the related work in this area of research and point out their strengths and weaknesses. We have divided the literature into eight parts corresponding to the surveillance WSN, evaluation of the surveillance WSN, enhancing the base station security, features needed for an efficient surveillance WSN, security issues for the WSN, attacks on WSN, security protocols and fault management protocols.

2.1 Surveillance WSN Systems

In this chapter, background information about surveillance WSN security is introduced. Section 2.2 presents an evaluation of surveillance WSN security. Section 2.3 presents the enhancement techniques for the base station security. Section 2.4 presents the features needed for an efficient surveillance WSN. Section 2.5 presents the security issues for sensor networks. Section 2.6 presents the attacks on sensor networks. Section 2.7 presents the security protocols. Section 2.8 presents the fault management protocols. Finally, section 2.9 provides the summary.

A wireless sensor network (WSN) is comprised of small and low-cost sensors with limited computational and communication power. The objective is to sense the environment and communicating the information to the data collection center. Many areas of employment are investigated for WSNs ranging from the monitoring of endangered animals populations to military surveillance. Surveillance wireless sensor networks are deployed at perimeter or border locations or battlefields to detect unauthorized intrusions. For *deterministic deployment* of sensors, the security of the deployed sensor network can be determined sufficiently well by analysis in advance of

the region of deployment. However, when *probabilistic deployment* is required, determining the deployed network security becomes challenging [12].

In this thesis, we concentrate on a surveillance WSN whose duty is intrusion detection in applications such as border surveillance against penetration by hostile elements or perimeter protection. Sensors are deployed to a region; they wake up, organize themselves as a network, and start sensing the area for intrusion. Our work focuses on a hierarchical WSN architecture which is formed of clusters with the ability of re-cluster to solve communication links failures. When a sensor detects an intrusion, the event is communicated to the base station so that appropriate action is taken.

Because of the energy constraints of sensor devices of surveillance WSN; such systems necessitate *energy-efficient surveillance* to ensure the longevity of surveillance missions. The surveillance system must allow a group of co-operating sensor devices to detect and track the positions of moving vehicles in an energy efficient and stealthy manner. Surveillance WSN systems can trade off energy-awareness and surveillance performance by adaptively adjusting the sensitivity of the system [5].

It is of a great practical importance to provide *differentiated surveillance* service for various target areas with different degrees of security requirements [13]. Differentiated surveillance refers to providing different degrees of sensing coverage for a sensor network according to different requirements such as energy conservation and low communications overhead between nodes.

The wide varieties of sensors have been incorporated into the spectrum of surveillance WSN platforms to provide flexible and different sensing capabilities using motion detection, tracking, and monitoring sensors. We motivate the security problems that surveillance sensor nodes networks face by first evaluating the surveillance WSN security then second developing new security architecture to solve some of the problems to elevate the security level.

2.2 Evaluation of Surveillance WSN Security

Securing the surveillance WSN is challenging and requires that a surveillance WSN be robust against an increasing number of threats and to support all security properties such as: confidentiality, integrity, authentication and availability. We must identify the threats and vulnerabilities to surveillance WSN starting from the radio layer to the application layer. The threats to a sensor network are different from the threats to Mobile Ad-Hoc network. Such existing network security mechanisms developed for Mobile Ad-Hoc networks are a poor fit for WSN because of unattended nature of WSN.

Attackers may deploy a few malicious nodes with similar hardware capabilities to act as the legitimate nodes during the network setup or after the network setup. The malicious node might collude to attack the system cooperatively. The attackers may come upon these malicious nodes by capturing some legitimate nodes to physically overwrite their memories. Sensor nodes may not be tamper resistant and if an adversary compromises a node, he can extract all the key material, data, and code stored on the node. While tamper resistance might be a viable defense for node compromise attack, we cannot use it as a general purpose solution because of its high cost [14].

A WSN is like any other data exchange network with generic vulnerabilities including: eavesdropping, spoofing, message integrity, denial-of-service and physical compromise [15].

The Surveillance WSN lifetime is directly related to both the energy resources of the sensor nodes which can be extended by energy-aware protocols and the security status of network sensor nodes. Therefore, the Surveillance WSN must be able to adapt to changes of the network security threats and the different environmental conditions.

Also, network failure, partial or wholly, may not only be due to the power exhaustion of the sensor nodes where physical destruction attacks can take place on a number of sensor nodes. When a group of sensors are intentionally destroyed by attackers, this leads to uncovered areas in the surveillance WSN which must be recovered or replaced along with the failure distribution of power-deprived sensor nodes.

Moreover, area failure of sensor nodes may occur through the presence of strong physical jamming. Since sensor failures are common, the fault tolerance protocols of the network should report these events to the base station because loss of individual sensors or a group of sensors should not stop the task accomplishment of the WSN.

The surveillance WSN only transmits using mechanisms that guarantee integrity and authenticity while confidentiality is not required. Also, surveillance WSN in hostile environments poses unique security challenges as they are left unattended.

There are a number of security solutions to the security threats issues inherent in a surveillance wireless sensor network but there are still some problems which need to be solved. One of the security problems is that surveillance WSNs are highly vulnerable to the failure of BS. This is because the attackers can easily render the whole network useless by only destroying the BS. The efforts needed to destroy the BS is much less than that needed to destroy the whole network and this attack scenario will give the attackers the best chance to compromise many legitimate nodes and hence destroy the network security.

Previous works have tackled BS failure by deploying a mobile BS or by using multiple BSs which requires extra cost. Also, previous works lack both the procedures to ensure network reliability and security during the BS failure such as storing then sending reports concerning security threats against nodes to the new BS. Also, there is no information about the procedures used to verify the trustworthiness of the deployed network by the new BS; otherwise a new WSN must be re-deployed which carries a high cost and it requires time for the re-deployment of the new WSN. Moreover, the probability of a single BS failure is high as a single point of failure.

2.3 Enhancing the Base Station Security

There are number of security attacks that target the BS such as denial-of-service by flooding, denial-of-service by jamming, and physical destruction of the BS. Therefore, there are a number of proposed strategies designed to secure the sensor network BS

against these threats that can lead to the failure of the BS. These protocols are summarized as:

1- Location concealing of BS through privacy algorithms [16],

   The location privacy of the base station requires ultimate protection due to its crucial position in wireless sensor networks. In [16], an efficient scheme is proposed, consisting of anonymous topology discovery and intelligent fake packet injection (IFPI), to protect the location privacy of the base station.

2- Using multiple mobile BSs [17],

3- Intrusion tolerant software [18],

   Intrusion-tolerant routing protocol for wireless **se**nsor **n**etwork**s** (INSENS) constructs forwarding tables at each node to facilitate communication between sensor nodes and a base station. This minimizes computation, communication, storage, and bandwidth requirements at the sensor nodes at the expense of increased computation, communication, storage, and bandwidth requirements at the base station. INSENS does not rely on detecting intrusions, but rather tolerates intrusions by bypassing the malicious nodes. An important property of INSENS is that while a malicious node may be able to compromise a small number of nodes in its vicinity, it cannot cause widespread damage in the network.

4- Relocation of the BS in the network topology [19],

5- Using multipath routing to multiple BSs [20],

6- Anti-traffic analysis strategies such as random fake paths to confuse the adversary [6],

7- Random areas of high communication activities [17],

8- Confusion of address and identification fields in packet headers via hashing functions [8].

Despite using the best electronic countermeasures, intrusion tolerance and anti-traffic analysis strategies to protect the BS, an adversary still can destroy it.

2.4 Features Needed for an Efficient Surveillance WSN

Our work is towards base station failure of the hierarchical architecture of surveillance WSN in hostile environment. We need effective security architecture for reliable network recovery from BS failure of surveillance WSN because BS failure has high probability as single point of failure. The BS needs to have the ability to monitor all of network security threats, security changes of the network and all of the security related data of the sensor nodes to trust the sensor nodes decisions and measures then to trust the network. This ability can be added through enabling the sensor nodes to continuously store the security related data of the downstream nodes' security threats against the network and the nodes in security reports. These security reports are stored encrypted and therefore we need key management system to distribute the keys for these nodes. The key management system must be hybrid and dynamic key management system to prevent large class of attacks including compromised nodes attack, and collusion attack.

Therefore, to effectively defeat physical destruction attacks to BS, we propose to continuously and securely store security reports of the network where the BS should have the ability to monitor the network security status to be able to trust the deployed nodes.

In this thesis, we propose a new security architecture called surveillance security (SurvSec) which can provide reliable network recovery from BS failure of surveillance WSN through the accurate in-time security reports of the current security status of the network. Also, SurvSec is resilient to collaborative work of attacker to compromised legitimate nodes through SurvSec compromised nodes detection algorithm. Moreover, SurvSec has hybrid and dynamic key management system. Finally, SurvSec has strong encryption architecture.

SurvSec shows the importance of achieving distributed security for the network by adding distributed security managers to resemble cellular networks.

In the next section, we will describe the design goals of WSN security, the four security services for the sensor networks and the key management systems for the WSN

2.5 Security Issues for Sensor Networks

The security of WSN can be breached in many ways. A remote end user accessing the base station can be prevented from doing so in a variety of ways. Communication between the base station and sensor nodes can be blocked. Another way to breach the security is to destroy the base station itself then spoof the base station and deceive nodes into routing all packets to the spoofed base station instead of the real base station. A third threat is eavesdropping.

A WSN is influenced by the fact that the computing resources of the nodes are highly limited, transmission rate of the radio, energy lifetime of the nodes and by the framework and deployment environment of the sensor network. Therefore, the three tough issues which have to be considered when designing the security of WSN which are its nature of being wireless, sever resource constraints and deployment environment. Moreover, security techniques should seek to implement the following general goals [21]:

1- Communication efficiency by low communications overhead.
2- Computational efficiency by low computations overhead.
3- Energy efficiency by reducing the energy.
4- Bandwidth efficiency by reducing the bandwidth.
5- Storage efficiency by low storage size.
6- Intrusion tolerance due to compromised nodes.
7- Fault tolerance.
8- Scalability.

There are a lot of security issues concerning wireless sensor networks according to the increasing number of threats which will be described in Section 2.6. In this section, we describe the required design goals of sensor network security which depend on knowing what needs to be protected. Also, we describe the security goals suited to the unique constraints of the sensor networks such as confidentiality, authentication, data integrity and availability. The last security issue we will describe in this section is the key management system.

2.5.1 Design Goals of Sensor Networks Security [22]

The design goals of sensor networks security are the followings:

1- Robust design: security design should build trustworthy system out of unattended sensor nodes and should have the ability to detect and react when needed.

2- Component-based security: security must be provided to all of the components of WSN as well as to the network. Security must secure the whole chain. Two classes of security can be provided which are host-based security such as local intrusion detection and network-based security such as secure routing and secure aggregation.

3- Adaptive security: WSNs have numerous combinations of sensing communication and computing technologies, sensors are deployed from very sparse to highly dense. Depending on the traffic characteristic, environment of deployment and the security threats faced, the sensor networks have to adopt themselves. e.g., in a good environment where the probability of security attack is low they should use a low level of security.

4- Quality of security service (QoSS): an important issue is how to trade off between the QoS parameters such as communication and computations overheads while providing security.

2.5.2 Security Services for Sensor Networks [23]

The security goals are classified as primary and secondary. The primary services are data confidentiality, data authentication, data integrity, and data availability. The secondary services are data freshness, self-organization, time synchronization and secure localization.

2.5.2.1 Data Confidentiality

Confidentiality is the ability to conceal messages from a passive attacker so that any message communicated via the sensor network remains confidential. This is the most important issue in network security. Confidentiality in WSN considers the followings:

1- A sensor node should not reveal its data to its neighbours. E.g., in a sensitive military application where an adversary has injected some malicious nodes into the network,

confidentiality will preclude them from gaining access to information regarding other nodes.

2- Establishing and maintaining confidentiality is important when node identities and keys are being distributed to establish a secure communication channel among sensor nodes.

2.5.2.2 Data Authentication

Authentication ensures the reliability of the message by identifying its origin. Attacks in sensor networks do not just involve the alteration of packets. Therefore, the receiving node needs to be able to confirm that a packet received does in fact come from the node claiming to have sent it. In other words, data authentication verifies the identity of senders. Data authentication is achieved through symmetric or asymmetric mechanisms where sending and receiving nodes share secret keys to compute the message authentication code (MAC).

2.5.2.3 Data Integrity

Data integrity in sensor networks is needed to ensure the reliability of the data and refers to the ability to confirm that a message has not been tampered with, altered or changed while on the network. Even if the network has confidentiality measures in place, there is still a possibility that the data's integrity has been compromised by alterations.

2.5.2.4 Data Availability

Availability determines whether a node has the ability to use the resources and whether the network is available for the messages to communicate. Since complex security measures entail a higher consumption of energy and computation power, keeping resource constrained sensor networks available is challenging. However, failure of the base station or cluster heads' availability will eventually threaten the entire sensor network. Thus availability is of primary importance for maintaining an operational network.

2.5.3 Key Management Systems for Sensor Networks

Efficient key distribution and management mechanisms are needed besides lightweight ciphers. Many key establishment techniques have been designed to address the trade off between limited memory and security, but which scheme is the most effective is still debatable.

It is important to examine the different requirements, constraints and evaluation metrics of sensor networks as well as single network-wide key scheme, which is the simplest of key management techniques, before discussing the various key management techniques.

Sensor networks' key establishment technique employed in a given sensor network should meet several requirements to be efficient. These requirements include supporting in-network processing and facilitating the self-organization of nodes. However, the key establishment technique for a secure application must minimally incorporate authenticity, confidentiality, integrity, scalability, and flexibility [24].

A key establishment technique is not judged only based upon its ability to provide secrecy of transferred messages, but must also meet certain other criteria for efficiency to face vulnerability of adversaries, including the three resistance to replication nodes, revocation of compromised nodes, and resilience to ensure that if a node is captured, it will not reveal secret information about other nodes [24].

Key management schemes [25–40] in WSN can be classified as follows:

1- Single network-wide key,
2- Pair wise key establishment,
3- Trusted base station,
4- Public key schemes using elliptic curve cryptography [33–36],
5- Key pre-distribution schemes (random key pre-distribution scheme [28], Random pair wise key scheme [26], Key management schemes using deployment knowledge [31], Location dependent key management scheme [39], Location aware combinatorial key management [39]),
6- Dynamic key management [37],
7- Hierarchical key managements (LEAP [30], Heterogeneous sensor networks [32, 33]).

From the above classifications, in many key management schemes static administrative keys or keys that are never updated are adequate to manage administrative events such as membership management or re-clustering. However, for long lived and hostile environment networks, the survivability of these keys cannot be assumed. Therefore, in hostile environment, and long-lived WSN operates unattended where its nodes are highly prone to capture, dynamic key management system is needed. This key management system must support nodes additions and revocations, re-clustering and administrative key updates to maintain the WSN security and survivability. Also, location-based key management is needed to restrict the attacks within a small location area and can be used to have less storage and communications overhead when compared with non-location based design [40].

It is clear that the security problem in WSN becomes more challenging when dealing with the group security as this grouping impose additional overhead in terms of network management. Several works have addressed the problem of group key management [41-45]; however, each of them relies on a specific and different grouping concept.

Specifically, location-based key management is resilient to compromised node attack as it is useless for a group of adversaries to capture number of nodes from the whole sensor network and even if they captured a large number of nodes in one division, the effect will be limited to that division but the adversaries will not have the capacity to destroy the whole network [46].

SurvSec will have hybrid and dynamic key management system to defeat large class of attacks in the unattended hostile environment.

Table 2.1, Key Management Functions in Static and Dynamic Keying [46]

Administrative keys	Static keying	Dynamic keying
Key assignment	Once at pre-deployment	Multiple times
Key generation	Once at pre-deployment	Multiple times
Key distribution	All keys are pre-distributed to nodes prior to deployment	Subset of keys are re-distributed to some nodes as needed
Re-keying	Not applicable	Multiple times, require a small number of messages
Handling node capture	Revealed keys are lost and may be used to attack other nodes	Revealed keys are altered to prevent further attacks

2.6 Attacks on Sensor Networks

Wireless Sensor Networks are vulnerable to security attacks due to their broadcast nature of the wireless transmission medium. Furthermore, wireless sensor networks have an additional vulnerability because nodes are often placed in a hostile environment where they are left unattended and they are not physically protected. Attacks are classified into several classifications which are according to the capability of the attacker, attacks on information on transit, host-based versus network-based attacks, based on the protocol stack [14] and based on the attacker mobility.

2.6.1 Based On the Capability of the Attacker [14]

2.6.1.1 Outsider versus insider attacks

Outsider attacks are defined as the attacks from external nodes which do not belong to the WSN and insider attacks are defined as attacks from the legitimate nodes of a WSN or a node misbehaving or a node operating in a malicious way. To overcome these attacks, we require robustness against outsider attacks, resilience to insider attacks, and resilience to node compromise attacks.

2.6.1.2 Passive versus active attacks

Passive attacks include eavesdropping or traffic analysis within a WSN and active attacks involve some modifications of the data stream or the creation of a false data stream.

2.6.1.3 Mote-class versus laptop-class attacks

In mote-class attacks, an adversary can attack a WSN by using a few nodes with similar capabilities to the network nodes; in laptop-class attacks, an adversary can use more powerful devices such as a laptop to attack a WSN. Laptop-class attacks have greater transmission range, processing power, and energy reserves than the network nodes.

2.6.2 Attacks on Information in Transit [14]

In WSN, sensor nodes monitor the changes of specific values and report to the base station according to a pre-defined threshold. While sending the report, the information in transit may be attacked to provide the wrong information to the base stations. These attacks are the followings:

2.6.2.1 Interruption

Communication links in sensor networks can become unavailable. This type of attack threatens service availability. The main purpose of interruption is to launch Denial-of-Service (DoS) attacks. DoS attack can aim all WSN protocol stack layers.

2.6.2.2 Interception

Sensor network can be compromised by an adversary by gaining unauthorized access to sensor nodes. This threatens message confidentiality. The main purpose is to eavesdrop on the information carried in the exchanged messages.

2.6.2.3 Modification

An adversary not only can access the data but also can tamper with it. This threatens message integrity. The main purpose is to confuse or mislead the parties involved in the communication protocol. This type of attack usually threatens the network layer and the application layer, because of the richer semantics of these layers.

2.6.2.4 Fabrication

An adversary can inject false data and therefore, compromises the trustworthiness of information. This threatens message authenticity. The main purpose is to confuse or mislead the parties involved in the communication protocol.

2.6.2.5 Replaying existing messages

This operation threatens message freshness. The main purpose of this operation is to confuse or mislead the parties involved in the communication protocol that is not time-aware.

2.6.3 Host Based versus Network Based [14]

2.6.3.1 Host-based attacks

This type of attack has three classes. User compromise: This involves compromising the users of a WSN, e.g. by cheating the users into revealing information such as passwords or keys about the sensor nodes. Hardware compromise: This involves tampering with the hardware to extract the program code, data and keys stored within a sensor node. Also, the attacker might attempt to load his program in the compromised node. Software compromise: This involves breaking the software running on the sensor nodes to change the applications running on a sensor node.

2.6.3.2 Network-based attacks

This type of attack has two perspectives: layer-specific compromises where the attack is targeting which layer, and protocol-specific compromises where the attack is targeting a protocol on the layer. Also, this includes all the attacks on information in transit.

2.6.4 Based On Protocol Stack [14]

This section discusses the WSN attacks targeting protocol stack layers. This can be summarized in Table 2.2 according to the five layers: physical layer, data link layer, network layer, transport layer and application layer.

Table 2.2, Sensor Networks Layers Attacks'

Layer	Attack
Physical Layer	Jamming, Radio Interference, and Tampering or Destruction
Data Link Layer	Exhaustion, Collision, Unfairness, Denial-of-Service Attack (DoS) and Sybil Attack
Network Layer	Sinkhole, Hello Flood, Node Capture, Selective Forwarding, Sybil Attack, Wormhole Attack, Spoofed/Altered/Replayed Messages, Acknowledgement Spoofing, DoS, and Misdirection
Transport Layer	Flooding, DoS, and De-synchronization
Application Layer	Overwhelm, and Path-based DoS Attack

2.6.5 Based On the Mobility of the Attacker [14]

The attackers can be classified as static attackers and mobile attackers. The mobile attacker has high capability to compromise many legitimate nodes.

2.7 Security Protocols

There are different security protocols proposed and implemented for use with wireless sensor networks. In [29], Perrig et al. proposed Security Protocols for Sensor Networks, SPINS, a suite of security protocols optimized for sensor networks. It consists of two secure building blocks, SNEP and µTESLA, which run on top of TinyOS, a small, event driven operating system for sensor nodes. Secure Network Encryption Protocol, SNEP, is used to provide confidentiality through encryption and authentication, in addition to integrity, using a message authentication code (MAC) and µTESLA protocol based on delayed key disclosure is used for authentication and suffers from the denial-of-service attacks (DoS). In [47], Karlof et al. designed the replacement for the unfinished SNEP, known as TinySec. Essentially, it provides similar services, including authentication, message integrity, confidentiality and replay protection. There are two

packet formats defined by TinySec. These are TinySec-Auth, for authenticated messages, and TinySec-AE, for authenticated and encrypted messages.

Localized Encryption and Authentication Protocol (LEAP) was proposed by Zhu et al as a key management protocol for sensor networks, motivated by the observation that different types of messages propagated in wireless sensor networks have different security requirements. Lightweight, energy efficient operation and robustness and survivability in the face of node compromise, are the main design goals of this protocol [30].

Heo and Hong proposed a new method of authenticated key agreement [48]. It is based on a Public Key Infrastructure (PKI) and Elliptic Curve Cryptography (ECC). The Security Manager (SM) gives static domain parameters such as the base point and elliptic curve coefficients to prospective network nodes. Devices use these initial parameters to establish permanent public keys and ephemeral public keys, which are in turn used to secure the network data. After calculating a public key, a node sends this to the SM, which could have a public key list for all nodes in the network.

ZigBee is an industrial consortium, which was designed to build a standard data link communication layer for use in ultra low power wireless communications. ZigBee specification outlines the design of the ZigBee network layer (NWK) that operates just above the PHY and MAC layers specified by the IEEE802.15.4 standard. Additionally, it contains descriptions, protocols and algorithms relating to the application support layer (APS), ZigBee device objects (ZDO) and profile (ZDP), the application framework and ZigBee security services [49].

The concept of a “Trust Center” is introduced in the specification. Generally, the ZigBee coordinator performs this duty. The coordinator allows other devices to join the network and distributes the appropriate keying information. There are three roles played by the “Trust Center”; 1: trust manager, whereby authentication of devices requesting to join the network is carried out, 2: network manager, maintaining and distributing network keys, and 3: configuration manager, enabling end-to-end security between devices [49]. There are two modes of operation; Residential Mode and Commercial Mode. Running the

former, low security residential applications are accounted for. The latter is designed for high-security commercial applications.

In Residential Mode, the Trust Center will allow devices to join the network, but does not establish keys with the network devices. It therefore cannot periodically update keys and allows for the memory cost to be minimal, as it cannot scale with the size of the network.

In Commercial Mode, it establishes and maintains keys and freshness counters with every device in the network, allowing centralized control and updating of keys. This results in a memory cost that could scale with the size of the network [49]. This could be managed through means of clustering, for example.

There are three types of keys specified for use in ZigBee security services; the Master Key, the Link Key and the Network Key. Master keys are installed first, either in the factory or out of band. They are sent from the Trust Center and are the basis for long-term security between two devices. The Link Key is a basis of security between two devices and the Network Keys are the basis of security across the entire network. Link and Network Keys, which are installed either in the factory or out of band, employ symmetrical key-key exchange (SKKE) handshake between devices. The key is transported from the Trust Center for both types of keys. This operation occurs only in Commercial Mode, as Residential Mode does not allow for authentication.

TinyECC security architecture is another variation of elliptic curve cryptography for TinyOS [50]. It supports a number of motes including the MICAz, and supports all elliptic curve operations over the finite field.

There is no security protocol which guarantees the security of the WSN during the time between the BS failure and the deployment of a new base station. Also, there is no security protocol which describes how the new base station verifies the trustworthiness of the deployed sensor nodes.

2.8 Fault Management Protocols

Since BS and nodes of WSNs are prone to failure due to energy depletion, hardware failure, communication link errors, software attacks and physical attacks, therefore, fault tolerance is one of the critical issues in WSNs. Fault tolerance is defined as the ability of the system to deliver a desired level of functionality in the presence of faults [51]. Since the sensor nodes and the BS in hostile environment face high probability of destruction or failure or capture by attackers, fault tolerance should be seriously considered in missions' critical applications such as surveillance WSN in hostile environment.

In [52], fault recovery mechanism in single-hop sensor networks was studied. The proposed fault recovery scheme is designed such that it can deal with failure of sensor nodes, including the sink node. The basic idea of the scheme is to partition the sensor memory into two parts, namely, data memory and redundant memory. The data memory is used to store sensed data and data recovered from failures of other sensor nodes. The redundant memory is used to store redundant data for future recovery. The recovered data is distributed in the memories of the non faulty sensors to be sent to the sink when it becomes available.

Fault management frameworks address faults as part of a larger network management structure. Such solutions approach the fault management at a higher level by designing the management infrastructure and information model. These frameworks can be complemented by the specific fault detection and recovery techniques discussed previously. A number of such frameworks have been introduced for either ad hoc networks or wireless sensor networks.

While Simple Network Management Protocol (SNMP) has been one of the management protocol used in wired networks [53, 54], there exist studies on the design of management protocols for ad hoc networks. For instance, Ad Hoc Network Management Protocol (ANMP) [55] uses hierarchical clustering to reduce the number of messages exchanged between the manager and the agents. Moreover, there are a number of management systems that have been designed and developed specifically for WSNs.

These systems include Digest [56], Sympathy [57], NOSY [58], SNMS [59], AgletBus [60, 61], MANNA [62, 63], WinMS [64], WSNMP [65], and sNMP [66].

We will present a brief overview of the management protocols in WSNs:

1- Digest [56] is architecture used to monitor WSNs with different levels of details, and it focuses on the design of continuous computing summaries of network properties.

2- Sympathy [57] is a tool for debugging and detecting failures in sensor networks.

3- NOSY [58] is a centralized network monitoring system that keeps track of the progress of code dissemination, adjusts sensor reporting frequency, pulls information from an individual sensor, and reboots a node if no messages are received for an extended period of time.

4- SNMS [59] is a middleware layer that provides a set of management services such as remote power management, enumerating sensor nodes, monitoring physical parameters of sensors.

5- AgletBus [60, 61] is a management middleware that provides consistent and transparent framework for both inter- and intra-nodal coordination and management. Similar to SNMS, it includes services such as leader election, event forwarding and power management.

6- MANNA [62] is a policy-based network management system for WSNs. Depending on the network topology and characteristics (homogeneous vs. heterogeneous), MANNA assigns different roles (network managers or agents) to various sensor nodes. These nodes exchange request or response messages with each other for management purposes. MANNA forms a basis for fault management [63], one of several network management services supported by this architecture. Fault management in MANNA mainly relies on the coverage area maintenance service and the failure detection service.

7- WSN Management System (WinMS) [64] is an adaptive policy-based management system for WSNs. End users predefine management parameter thresholds on sensors that are used as event triggers, and specifies management tasks to be executed when the events occur. WinMS adapts to the changes of network conditions by reconfiguring the network according to current events as well as predicting future

events. WinMS Advantages are its lightweight TDMA protocol that provides energy-efficient management, data transport and local repair.

8- WSN management protocol (WSNMP) [65] is proposed as a management architecture protocol that monitors WSN with minimum overhead, collects the management data and finally manages the network efficiently by periodically reconfiguring the network. Also, it detects the fault by identifying the non-response nodes and reconfigures the routing path.

9- Sensor Network Management Protocol (sNMP) [66] has two functions. It defines sensor models to represent the current state of the network and various network management functions. It also provides algorithms and tools to collect network state through the network management functions. Models for sensors include network topology, energy map, and usage patterns. The correlation between the energy map and network topology can be used to identify weak areas in the network.

Different approaches for fault management suffer from the following problems [67]**:**

1- It is very challenging to apply existing fault management architecture from one application to another due to application specific nature of WSNs.

2- Most existing approaches mainly focus on failure detection. However, there is still no comprehensive solution for fault management in WSNs from the management architecture perspective.

3- Different mechanisms proposed for fault recovery are not directly relevant to fault recovery in respect of the network system level management (e.g. network connectivity and network coverage area etc).

4- Fault recovery mechanisms are mainly application specific (e.g. gateway recovery, and common node recovery) and focus on small region or individual nodes thereby are not fully scalable.

5- Some decentralized approaches require the network to be pre-configured, which is very costly for resource constrained WSNs.

6- Some management frameworks require the external human manager to monitor the network management functionalities e.g. sNMP.

7- Some schemes only consider permanent faults and avoid other faults such as Transient.
8- Most existing approaches in WSNs isolate failed or misbehaving nodes directly from the network communication, but there is no adequate fault recovery procedure available.

Also, all existing fault management approaches lack the procedures to allow new BS to trust the deployed sensors network after BS failure in mission critical applications such as surveillance WSN in hostile environment.

Table 2.3 presents the fault management approaches categorization.

In [73], sympathy can detect sink failure if no node is able to hear the sink but hearing other nodes. Remedial action will involve changing the sink placement or examining sink metrics for bugs or other connectivity issues. Sympathy did not include any procedures between BS failure and new BS deployment which is a must in surveillance WSN in hostile environment.

All of the above fault management protocols lack the procedures for secure and reliable network recovery from BS failure which are important issues for mission critical applications such as surveillance WSN in hostile environment.

Table 2.3, Fault Management Approaches Categorization [67]

Schemes	Management System Organization	Types of Faults & Failure addressed	Action taken
Sympathy	Centralized Hierarchical, Pro-active monitoring	Node self, Network faults, Sink fault, Crash & time-out omission failures	Fault Detection & Diagnosis
MANNA	Centralized + Distributed Passive monitoring	Node faults	Detection, Diagnosis & Recovery
WinMS	Centralized + Distributed (Hierarchical) Pro-active monitoring	Node faults (week or faulty)	Detection & Recovery
WSNMP	Centralized + Distributed (Hierarchical Clustering based)	Node faults, Network faults	Detection & Recovery
Cluster-based approach [68, 69]	Centralized + Distributed	Node faults (energy failures), Network faults (network connectivity), Permanent faults	Detection & Recovery
Passive diagnosis of WSNs [70]	Centralized + Hierarchical, Probabilistic approach, Passive monitoring	Node faults, Network faults, Transient faults	Detection, Diagnosis & Recovery
Efficient Tracing of failed Nodes [71, 72]	Centralized, Active monitoring	Node faults, Route Faults	Detection, Diagnosis & Recovery

Table 2.4 presents the evaluation of different fault management approaches.

Table 2.4, Evaluation of Fault Management Approaches [65]

Protocol	Energy efficiency	Robustness	Adaptability	Memory efficiency	Scalability
MANNA	No	N/A	N/A	N/A	No
SNMS	Yes	Yes	No	Yes	No
sNMP	Yes	No	No	Yes	Yes
WinMS	Yes	Yes	Yes	Yes	No
SNMP	N/A	No	No	No	No

2.9 Summary

In this chapter, we show the background of surveillance WSN. We describe the evaluation of surveillance WSN security. We describe the techniques used to enhance the base station security. We provide a brief discussion about the features needed for an efficient surveillance WSN. In this section, we describe the problem of base station failure and the solution by the reliable network recovery from BS failure of surveillance WSN in hostile environment to increase the lifetime of the network and to verify the trustworthiness of the deployed sensor nodes through continuously storing the security-related data of the network. Also, SurvSec shows the importance of achieving distributed security for the network by adding distributed security managers (SMs) to resemble cellular networks, and the importance of designing a hybrid and dynamic key management system for SurvSec to prevent wide range of attacks. We describe the security issues for sensor networks. We describe the attacks on sensor networks. We describe the security protocols such as TinySec, SPINS, LEAP and ZigBee. Finally, we describe the fault management protocols.

CHAPTER 3

SURVSEC: A NEW SECURITY ARCHITECTURE

In this chapter, we describe the five components of SurvSec security architecture. These components are security managers for reliable network recovery from base station failure, secure data storage and recovery, compromised nodes detection at the first stage against collaborative work of attackers working at the same time, hybrid and dynamic key management scheme and spread spectrum encryption architecture for post-quantum computer. Also, we state the design goals and the evaluation metrics.

3.1 Introduction

In this chapter, an introduction to security of surveillance WSN is introduced. Section 3.2 presents the requirements for SurvSec design. Section 3.3 presents SurvSec design goals and evaluation metrics. Section 3.4 presents the threat model. Section 3.5 presents the assumptions and the network setup for SurvSec security architecture. Section 3.6 presents an overview of SurvSec security architecture. Finally, section 3.7 presents the summary.

To the best of our knowledge, there is not any scheme in the open literature addresses the base station failure. The current security schemes proposed for wireless sensor networks lack the ability of providing reliable network recovery in the case of base station failure. This challenge is quite serious in case of mission critical deployments such as deployments of surveillance wireless sensor networks in hostile environment.

In hostile environments, the probability of base station failure is high since, as a single point of failure, it is a natural target for the adversary. Also, the time and efforts required to destroy a base station is considerably less compared to what is needed to neutralize the actual WSN. Indeed, even excluding "physical attacks", cyber attacks to the base station can be quite effective. Despite using the best electronic counter measures, intrusion tolerance systems, and anti-traffic analysis strategies to protect the BS, an adversary still can destroy it. It should be noted that by having BS destroyed, the attacker's effort to compromise legitimate nodes becomes easier. Even if a new base station is deployed or the existing one is recovered, the new base station cannot verify the trustworthiness and security of the deployed sensor nodes, thus compromised nodes might remain operational for quite long time.

Software-based attestation [74-78] is not valid as a solution for the whole sensor nodes of the network because it will take long time to be completed and it will consume considerable amount of energy, which might lead the individual sensor nodes to energy depletion. In addition, during the attestation time, the area covered by the surveillance system is not protected and the acquired information cannot be trusted. Deployment of a new sensor networks is not an effective or smart solution, since this results in high cost and long-time leaving the protected area uncovered by a reliable WSN for the duration of deploying and establishing the new WSN.

To the best of our knowledge, there is no contribution in the open literature addressing the situation a user has to deal with from the time the BS fails (due to hostile attack or accidently) to the time the WSN is operational again. Also, we haven't found any research explaining how the new BS can verify the trustworthiness of the existing sensor nodes. By lacking the ability to verify the trustworthiness of the existing sensor nodes, a user has no choice but to "scrape" the existing deployment and proceed with a new one, despite the deficiencies associated with this choice (e.g. high cost and long duration of having unreliable coverage of the deployed WSN).

Our work addresses this important issue and strives to provide practical answers to this challenging problem. Based on our work, we propose a new security architecture we named Surveillance Security (SurvSec). SurvSec is capable of maintaining security

information even during the BS failure periods. This is accomplished through two steps. The first step is storing the security-related data until the recovery of the BS or the deployment of new BS. The second step is sending the stored data to the recovered BS or the new BS after it is authenticated

Furthermore, BS failure shows the importance of the continuous storage of the security reports of monitored security threats towards the WSN through securely storing the security-related data of sensor nodes. The stored security-related data will be sent to the new BS during the recovery process. These procedures will result in reliable recovery from such attack and also, they will maintain the WSN lifetime where physical attacks towards BS specifically target the reduction of the WSN lifetime.

3.2 Requirements for SurvSec Design

In this section, we summarize the most important requirements for SurvSec security architecture design which are the followings:

1- *Processing and Data Storage*: the processing power and data storage capabilities of WSN nodes are considerably limited and require the use of computational efficient algorithms (for the energy saving purposes) and small software footprints (for the memory saving codes purposes).

2- *Reliability*: it is very important to have the network work without any human involvement/intervention. This is because the whole or part of the network might be located at inaccessible sites where sensor nodes are deployed unattended as well as humans might not have the capacity to identify and respond to very time-sensitive critical messages in due time such as considering the case of a nuclear plant generating electric energy, with the sensors indicating “core meltdown”, or the case of WSN deployed around a very sensitive military related facility. In both cases there is not much margin for failure.

3- *Power*: the energy reserves available to a WSN node are generally very limited and are derived from 2-3 AAA batteries. Nodes are expected to run for extended

periods of time vary from (1 - 2 years) on this internal energy reserve. Thus the design should be energy efficient.

4- *Cost*: the cost of WSN deployments must not be adversely impacted by the inclusion of security services as the cost is often a major factor for WSN technology.
5- *Scalability*: the design of any security architecture composed of security components must ensure network scalability is preserved in order to allow all the future expansions of the WSN.

3.3 SurvSec Design Goals and Evaluation Metrics

Our design goal for SurvSec security architecture is to provide the following security services for surveillance WSN in hostile environment:

1- *Security Managers*: SurvSec proposed to choose some sensor nodes to serve as security managers where these nodes will be responsible for adding distributed security concept to the WSN and therefore, the security managers will store the security-related information of its downstream sensor nodes. The security managers are chosen by the BS every two layers of sensor nodes.
2- *Secure Data Storage and Recovery System*: SurvSec permitted to continuously store security information of the sensor nodes in order to allow reliable network from base station failure. The stored security related data should be handled in a manner that reduces storage overhead. The data must be securely stored to prevent eavesdropping on the network security status during the storage process. The encryption scheme must be capable of preventing the attacker from acquiring and revealing the security-related data on sensor nodes after a node compromise attack. This will be achieved through the deployment of a new security scheme which is based on the use of dynamic secret sharing [79-82]. Also, a secure data storage system must allow for reliable data recovery of the stored data.
3- *Compromised Nodes Detection*: SurvSec implemented a new algorithm for the early detection of compromised nodes. Also, the detecting technology should be

resistant to collaborative work of attackers which target the simultaneous compromise of many legitimate nodes.

4- *Hybrid and Dynamic Key Management Scheme*: SurvSec implemented an efficient key management scheme resistant to compromised nodes attack, and collusion attack.
5- *Spread Spectrum Encryption Architecture*: SurvSec implemented a new encryption architecture resistant to quantum computer.

Our evaluation metrics for the SurvSec security architecture are satisfied as follows:

1- Secure data storage system resilient to node compromised and traffic analysis attacks with minimal data storage. Also, the data storage is resilient to multiple nodes failure.
2- Reliable stored data recovery to the new base station.
3- Resiliency to compromise node attack.
4- Low communications overheads.
5- Low computations overheads.
6- Low storage overheads.
7- Deployment of adaptive security concept [83-86].
8- Low setup time.
9- Scalability.

3.4 Threat Model

Surveillance WSN imposes a wide range of attacks and the worst attack scenario is the following planned attack: when group of attackers' first launch physical attacks against the BS to destroy the BS. Then, they compromise many legitimate nodes to control the network security and to cover their unauthorized intrusions where the new BS cannot trust the deployed sensor nodes.

In this work, we focus on the effect of collaborative work of group of mobile attackers to destroy the security of surveillance WSN. They first destroy the base station

then compromise many nodes at the same time to eavesdrop, send false data, change correct alarms, physically corrupt, modify, and capture the stored security-related data of the network. The attackers have the following capabilities:

(1) The attackers can eavesdrop on all of the traffic of the WSN.
(2) The attacker can determine the nodes that are communicating with each other's to secure the stored data using the secret sharing.
(3) The attackers are mobile to compromise many nodes after the base station failure.
(4) The attackers have the capabilities to jam the base station or part of the network.

3.5 Assumptions and Network Setup for SurvSec

We consider a surveillance sensor network that is composed of a large number of sensor nodes with a unique ID. The following assumptions have been made in the formulation of the SurvSec security architecture:

1- All sensor nodes are static.
2- The secure hierarchical data storage and recovery system needs a pre-configuration to allow the base station to choose from the network topology some sensor nodes inside the hierarchical architecture to be security managers (SM), to allow dynamic security for the stored data through dynamic secret sharing between sensor nodes of the hierarchical architecture, and to allow reliable data recovery from stored data.
3- The compromised nodes detection algorithm needs a pre-configuration to allow overlapped groups formation to protect the network from compromise nodes attack.

3.6 Overview of SurvSec Security Architecture

In this section, we present SurvSec, a suite of security services for hierarchical surveillance WSN in hostile environment. Our goal is to address the problems discussed in section 3.1 which are the base station failure, collaborative work of mobile adversaries against sensor nodes and increasing the quality of security services by implementing

dynamic secret sharing for the stored data. First, we will give an overview of our designed security architecture then the contents of SurvSec security reports then the security architecture operational phases and finally the security architecture components. Figure 3.1 describes SurvSec's five phases.

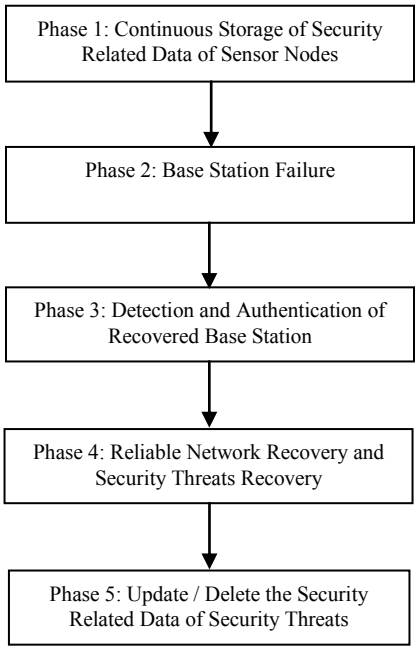


Figure 3.1, SurvSec Security Architecture Phases of Operations

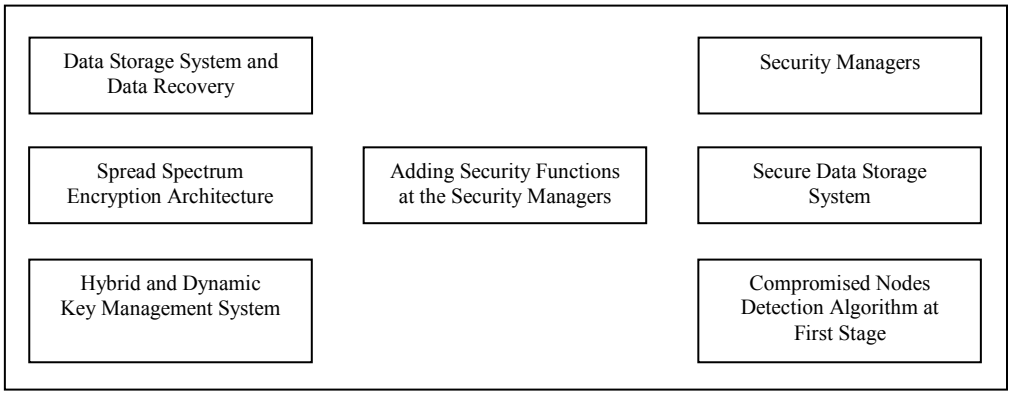


Figure 3.2, SurvSec Security Architecture Components

Figure 3.2 describes SurvSec security architecture components which are security managers for reliable network recovery; secure data storage and recovery, compromised

nodes detection algorithm at the first stage, hybrid and dynamic key management scheme and spread spectrum encryption architecture for post-quantum computing.

The first concern in SurvSec design is to allow a reliable network recovery from base station failure by continuously storing the security-related data of the sensor nodes to enable the new base station to verify the trustworthiness of the deployed sensor nodes.

The second concern in SurvSec design is to maintain the network lifetime where we found that destroying the base station targets the network lifetime as there are nodes but we do not trust them so these nodes are useless.

The third concern is to provide the WSN with distributed security concept by choosing nodes to be security managers. Security managers are responsible for the nodes security issues including gathering security-related data from its downstream nodes and we can add new security functions to the sensor nodes from the security managers such as node certificate to audit the node periodic tasks and to audit its trust level.

The fourth concern is to increase the WSN quality of security service (QoSS) by deploying dynamic security protocol to provide dynamic security for the stored security related data of the sensor nodes.

To implement SurvSec, different components need to be designed to help in performing its functionalities and to ensure its performance.

The security threats must be encoded to lower the storage overheads, and each node should have an ID.

Part of the SurvSec Security Report content is: Node ID, and reported attacks which are the followings: Node Compromise Attack, Revoked Node, Local Intrusion Detection (LID) Cloning Attack, LID Sybil Attack, LID Sinkhole Attack, LID Wormhole Attack, LID Selective Forwarding Attack, Node Outage, Awake Node, Sleep Node, Node Failure, Node Misbehaviour, Selfish Node, Message Corruption, Routing Attacks, Denial-of-Service (DoS) Attack, Security Level, and Re-keying.

Figure 3.3 describes the steps for the reliable network recovery from base station failure.

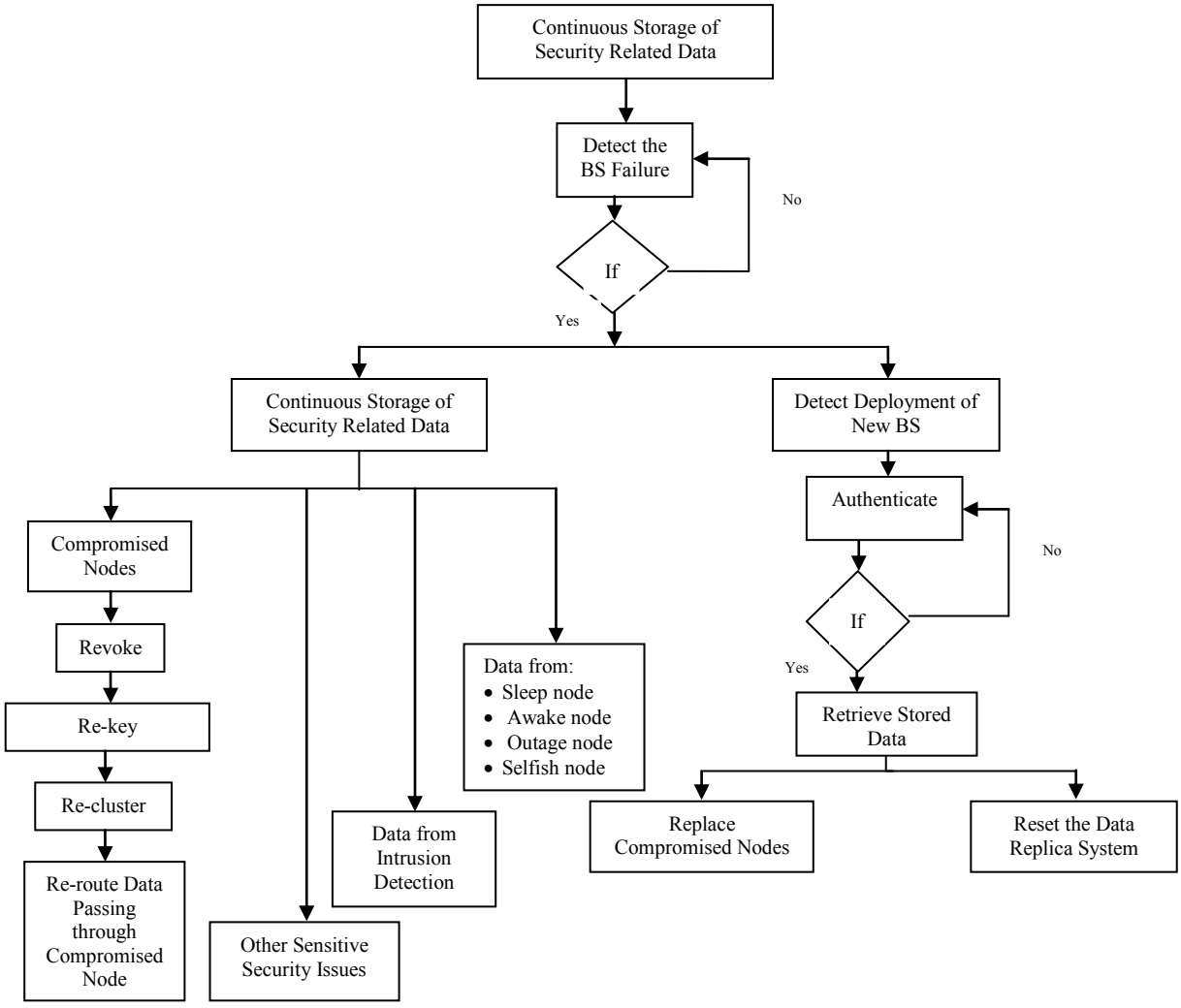


Figure 3.3, SurvSec; Reliable Network Recovery from Base Station Failure

Figure 3.3 describes the functionalities of reliable network recovery from base station failure. First, the protocol continues to store all security-related data of the nodes underneath the security manager. If the last layer of sensor nodes near the base station does not hear the beacon nodes of base station, this means that the base station is failed. Therefore, the protocol detects base station failure and the protocol continues to store all security-related data of nodes underneath the security managers. Also, the protocol will wait for the detection of the deployment of new base station. After the deployment of the new base station, the new base station is authenticated then the stored data is retrieved at the base station from the security managers underneath the base station. Then the

compromised nodes are revoked and the protocol applies re-clustering to reroute data that goes to compromised sensor nodes. After the replacement of compromised sensor nodes the stored data is updated. The followings is the definitions of some attacks:

Cloning attack: it is defined as when an adversary capture legitimate nodes, make clones by copying them, and integrate these clones back into the network.

Sybil attack: it is defined as a malicious node that behaves as if it were a larger number of nodes, for example by impersonating other nodes or simply by claiming false identities.

Sinkhole attack: In sinkhole attacks, the adversary attracts the traffic to a compromised node. The simplest way of creating sinkhole is to place a malicious node where it can attract the most traffic, possibly closer to the base station so that the malicious node could be perceived as a base station.

Wormhole attack: In wormhole attacks an adversary positioned closer to the base station can completely disrupt the traffic by tunnelling messages over a low latency link. Here an adversary convinces the nodes which are multi-hop away that they are closer to the base station. This creates a sinkhole because the adversary on the other side of the sinkhole provides a better route to the base station.

Selective forwarding attack: In selective forwarding attacks malicious nodes simply drop certain messages instead of forwarding every message. Once a malicious node picks up on the messages, it reduces the latency and deceives the neighbouring nodes into viewing it as being on a shorter route. The effectiveness of this attack depends on two factors: the location of the malicious node such that the closer it is to the base station the more traffic it will attract; and the percentage of messages the malicious node drops.

3.6.1 SurvSec Five Phases

SurvSec has five operational phases to ensure its proper functionalities which are the followings:

1- First phase, continuous secure storage of security-related data of sensor nodes,

2- Second phase, BS failure where the last layer nodes near the BS of the hierarchical WSN architecture cannot listen to the BS periodic beacons,

3- Third phase, detection and authentication of the new deployed BS,

4- Fourth phase, reliable network recovery from BS failure and security threats recovery to enable the newly deployed BS to trust the deployed sensor nodes. If the new BS does not trust the network sensor nodes, the network administrator has to follow the order of two expensive solutions:

First; the administrator must test the whole network sensor nodes using software-based attestation to verify their memory contents to detect malicious nodes to revoke them. This solution is expensive in terms of the time and energy required.

Second; if a large number of sensor nodes are found to be malicious, the network administrator needs to deploy a new WSN. This solution is expensive in terms of the time required and WSN money cost.

5- Fifth phase, update / delete the security threats to delete its stored security-related data upon recovery from the security threats.

There is no need to *inform* the network nodes *with BS failure* because of the necessity to continuously store security reports and the continuous sending of security reports update with proactive/reactive methodology.

SurvSec has a *proactive* data storage system in the sense that the sensor nodes at the lower layers send a periodic messages to the sensor nodes at the higher layer and if the higher layer nodes do not receive these messages on the pre-defined times, they send queries to these lower layer nodes. Also, SurvSec has a *reactive* data storage system in the sense that the sensor nodes at the lower layers send a security report update to the sensor nodes at the higher layer upon a detection of security threats.

3.6.2 SurvSec Components

3.6.2.1 SurvSec Hierarchical Security Managers (SM)

The first component of SurvSec is the Security manager which is an ordinary node from the sensor nodes of the hierarchical architecture with the responsibility to provide the network with the distributed security concept. A security manager is responsible for the security-related issues of its downstream sensor nodes until another lower layer of security manager.

In order to enhance the security of wireless sensor networks, SurvSec security architecture specifies the need for Security Managers that acts as data storage for the security data of the network sensor nodes, and Key Distribution Center (KDC). The distributed security managers within the hierarchical architecture of the network avoid single point of failure of the base station. To serve the wireless network, the Security Managers store the security data corresponding to security threats to the network, generate, distribute, renew, revoke, and handle the keys through the interaction with the base station.

In our work, we address the problem of choosing the security managers nodes according to the network size, the number of nodes on each layer and the number of layers within the hierarchical architecture. The security managers of one layer are responsible for its downstream nodes until the security managers on the lower layer of security managers.

The stored security issues are concerned with the data storage of security-related data such as compromised node and cloning node and other security functionalities such as key generation, key distribution, key revocation, sending queries to collect the data from sensor nodes, network intrusion detection system, generating a certificate for each node, and checking the communication links and the routing paths of sensor nodes.

The management of the security managers includes:

- A methodology to choose the security managers.
- Changing of the security managers.
- Network setup for the security managers.
- Frame format of the stored data.

3.6.2.2 SurvSec Hierarchical Secure Data Storage and Recovery System

The second component of SurvSec is the hierarchical secure data storage and recovery system. We must take into considerations certain constraints for the data storage system. The most important constraint is the communication overheads versus the availability of the stored data. The second constraint is the reliability of the recovered data versus the computational complexity which consumes the nodes' precious power. The third

constraint is the probability of nodes failure versus the availability of data replicas. Also, the number of the stored data copies versus the storage overheads. Moreover, the attacks on the data storage systems must be considered. We found that, there are two approaches to be implemented for our system of secure data storage which are the followings:

a. First approach: we can store the security-related data on all nodes of the hierarchical architecture where each node stores the security data of its downstream nodes,
b. Second approach: we need to choose some nodes from the sensor network hierarchical architecture according to the network size, the network number of nodes and the network number of layers. Then, these chosen nodes from the hierarchical architecture will be responsible for the storage of the security-related data in multiple copies.

We choose the second approach. Stored data of security-related data must be encrypted to prevent the attackers from disclosing the network security status after they captured any node and also to prevent the traffic analysis and eavesdropping on the traffic. The used key to secure stored data must be shared between nodes to prevent attackers from reading the security-related data report by compromising a node and using the key on that node. Another issue is that if the attacker knows that his captured node is discovered, he might change this information or compromise other nodes.

SurvSec storage system should have a reliable recovery system for the stored security related data through coding or any other technique.

After the security threat is recovered, its corresponding stored security-related data must be deleted from all of the data replicas and this deletion operation must be ensured to free a memory space from the sensor nodes' memories.

3.6.2.3 SurvSec Compromised Nodes Detection Algorithm

The third component of SurvSec is a new algorithm to detect compromised nodes at the first stage by forming overlapped groups from the network sensor nodes. The nodes of each group will communicate in a closed loop to prevent group of attackers from compromising many legitimate nodes at the same time.

The closed loop communication within the group and the overlapped groups will enforce the attackers to attack the whole groups at once to cover their intrusion. Also, the overlapped groups will prevent the attackers from isolating any number of nodes to compromise them.

3.6.2.4 SurvSec Hybrid and Dynamic Key Management

The fourth component of SurvSec is a new hybrid and dynamic key management scheme to resist compromise node attack, and collusion attacks.

The protocol is location-based therefore; each node is registered at the base station through the security managers with its location and ID. Also, the protocol is dynamic to revoke compromised nodes then apply rekeying and the protocol can add new nodes. The protocol is a hybrid key management to get the advantages of both symmetric key based key management and public key based key management.

3.6.2.5 SurvSec Spread Spectrum Encryption Architecture SSEA

The fifth component of SurvSec is a new encryption architecture which is called spread spectrum encryption architecture to resist quantum computer attacks.

The new encryption architecture is a family of encryption architectures which applies the unpredictability principle for the encryption architecture and this methodology results in developing a barrier between the encryption architecture and the cryptanalysis attacks.

3.7 Summary

The components of SurvSec security architecture provide the WSN with a reliable network recovery from BS failure of surveillance WSN in hostile environment.

The ingredients of SurvSec are secure hierarchical data storage system, security managers for distributed security concept, a new algorithm to early detect compromised nodes at the first stage, SurvSec hybrid and dynamic key management scheme and SurvSec Spread Spectrum Encryption Architecture SSEA for post-quantum computing.

CHAPTER 4

SURVSEC SECURE DATA STORAGE AND RECOVERY SYSTEM

In this chapter, we describe SurvSec reliable network recovery from base station failure using the concept of security managers. We will show that when the security managers store the security-related information from its sensor nodes underneath this will result in reliable network recovery from base station failure. However, we need to securely store the security-related data at the security managers therefore; we apply new dynamic secret sharing protocol to allow the secure storage of data at the security managers.

4.1 Introduction

This chapter is organized as follows: Section 4.2 presents the related work. Section 4.3 describes the assumptions, attacker model and network setup. Section 4.4 describes an overview of our security architecture SurvSec to recover from BS failure with its ingredients. Section 4.5 presents SurvSec data storage system and its analysis. Section 4.6 presents SurvSec data recovery system and its analysis. Section 4.7 presents SurvSec security for the stored data. Section 4.8 presents the simulation results. Finally, Section 4.9 is the summary.

This chapter proposes a novel security architecture called Surveillance Security (SurvSec) for secure and reliable network recovery from BS failure of surveillance WSN. SurvSec relies on a set of sensor nodes which serve as Security Managers for management and storage of the security-related data of all sensor nodes. SurvSec security architecture provides a methodology for choosing and changing the security managers of

the surveillance WSN. SurvSec has four components: (1) Sensor nodes serve as Security Managers, (2) Data Storage System, (3) Data Recovery System, (4) Security for the Data Storage System. Furthermore, both the frame format of the stored data is carefully built and the security threats are encoded to allow minimum overheads for SurvSec security architecture. In this chapter, we provide detailed specifications of SurvSec security architecture along with its security system for secure and reliable network recovery from BS failure. We evaluate our designed security architecture for reliable network recovery from BS failure. Our evaluation shows that the proposed new security architecture can meet all the desired specifications and our analysis shows that the provided Security Managers are capable of network recovery from BS failure.

Wireless sensor networks (WSNs) are deployed in many missions' critical applications such as surveillance [1], and one of the key issues to the success of their mission is security. The general objective of such an application is to alert the control unit in advance to the occurrence of events of interest in hostile regions. The event of interest will vary according to its mission which might be the presence of moving vehicles or target detection or other events where there are several types of sensors such as Vibration, Motion, Tracking, Video, and Infrared sensors which can be used for surveillance applications [2]. With their deployment, various novel security attacks have appeared. The aims of these attacks are usually to compromise nodes, eavesdropping for traffic analysis, destroying the base station (BS) or to disrupt data flow. We believe that, collaborative work of attackers will first launch physical attacks against the BSs of a surveillance WSN including jamming and destruction. Then they will compromise many legitimate nodes to destroy the deployed network security and to cover their unauthorized intrusions.

BS is a critical part of a WSN and an entire WSN can be rendered useless by taking down its BS. Indeed, it is crucial to protect a BS against both software-based and physical attacks. Several intrusion tolerant techniques have been developed to protect a BS against software-based remote attacks such as DoS attacks that flood the BS with packets, and remote spoofing of the BS to misdirect legitimate sensor data [6]. Software-based techniques cannot protect BS against physical attacks. Therefore, some works have

been done to address the problem of protecting a BS against physical attacks through concealing its geographic location in the network [7].

Our focus in this chapter is to address BS failure. We consider a feasible attack towards BS as single point of failure or even towards multiple BSs to render the whole WSN useless and after this attack collaborative work of attackers can compromise many legitimate nodes.

Also, previous works lack both the procedures to ensure network reliability and security during BS failure such as storing then sending reports concerning security threats against nodes to the new BS and the procedure to verify the trustworthiness of the deployed sensor nodes by the new BS; otherwise a new WSN must be re-deployed which has a high cost and it takes time.

To the best of our knowledge, there has not been work done on securing the surveillance WSN during the time between the BS failure and the new mobile BS deployment which is the perfect time for attackers to compromise many nodes then destroy the security of the whole system. Also, there is not any work that describes how the new BS will verify the trustworthiness of the deployed WSN otherwise a new WSN must be deployed. Therefore, for mission critical applications such as surveillance WSN, if the BS fails, we propose to address this problem through employing our new designed security architecture of Surveillance Security (SurvSec) to detect the BS failure, monitor the network sensitive security issues to store security data in multiple replica, and send the stored data to the new BS after it is authenticated. Furthermore, BS failure shows the importance of reporting the monitored security threats to the new BS through securely storing this sensitive data then sending this data during the recovery process to the new BS.

These procedures will result in reliable recovery from such attack. BS failure can be alleviated such as the work discussed in [11] by the use of multiple base stations deployed along the periphery of the field, and allowing each base station to act as a data sink. Multiple BSs failure is an important performance metric which must be considered and it is a serious attack. Therefore, if the BS fails and the network nodes are not trusted by the new BS, the whole network must be redeployed. Re-deploying such mission

critical large surveillance WSN shows the importance of SurvSec security architecture to efficiently recover from BS failure and later on multiple BSs failure by updating the new BS with all the security information that is needed to trust the network nodes thus enabling to achieve reliable network recovery from BS failure.

In this chapter, we present a novel recovery approach from BS failure that includes monitoring the network security issues to store the sensitive data, and send the stored data to the new BS after deployment to enable efficient recovery from BS failure while maintaining the operation of the network. Our motivation is the high probability of BS failure as single point of failure to render the whole network ineffective. Our goal is to design new security architecture SurvSec for reliable network recovery from BS failure of surveillance WSN in hostile environment.

The **contributions** in this chapter can be summarized as:

The **first contribution** is the development of the new security architecture called Surveillance Security (SurvSec) for fast and reliable network recovery from BS failure of surveillance WSN with a hierarchical data storage system.

The **second contribution** is the design of distributed security managers to enable distributed network security and distributed secure storage.

The **third contribution** is a hierarchical data storage and data recovery system for the security data of the sensor nodes.

The **fourth contribution** is a proposed system to secure the stored data for SurvSec security architecture.

4.2 Related Work

In this section, we present a brief overview of the related works such as some previous approaches taken towards enhancing BS security, fault tolerant models, and security protocols in wireless sensor networks.

Because the BS is a single point of failure and all the data is routed towards it, if it failed then the entire network can be disabled. Therefore, there are number of proposed strategies designed for securing the sensor network against the threats that can lead to the

BS failure. These protocols are summarized as location concealing of BS through privacy algorithms [16], relocating the BS [17], using multiple mobile BSs [17], multipath routing to multiple BSs [6], intrusion tolerant software [18], and anti-traffic analysis strategies such as random fake paths to confuse the adversary and random areas of high communication activity [7].

Since BS and nodes are prone to failure due to energy depletion, hardware failure, communication link errors, software attacks and physical attacks, therefore, fault tolerance is one of the critical issues in WSNs. Fault tolerance is defined as the ability of the system to deliver a desired level of functionality in the presence of faults [51, 52].

4.2.1 Fault Management Protocols

All of the fault management protocols lack the procedures for secure and reliable network recovery from BS failure which are important issues for mission critical applications such as surveillance WSN in hostile environment.

4.2.2 Security Protocols

There are different security protocols proposed and implemented for use with wireless sensor networks. In [29], Perrig et al. proposed Security Protocols for Sensor Networks, SPINS, a suite of security protocols optimized for sensor networks. It consists of two secure building blocks SNEP and μTESLA. In [47], Karlof et al. designed the replacement for the unfinished SNEP, known as TinySec..

All of the security protocols lack the procedures for secure and reliable network recovery from BS failure which are important issues for mission critical applications such as surveillance WSN in hostile environment.

4.2.3 Data Storage Categories

At present, many data storage methods have been proposed for sensor networks. These methods can be divided into seven categories as shown in Figure 4.1: centralized data storage [87–90], distributed data storage [91–94, 81], local storage [95], external data storage [96, 97], collaborative work between sensor nodes for storage [98], data-centric storage [99–101] and hierarchical data storage systems [102, 103]. Several comparison studies have been done assessing the communication overheads, storage

overheads, computations overheads [104, 105]. In the next subsections we will discuss these categories and identifying the problem of storing data generated inside a sensor network.

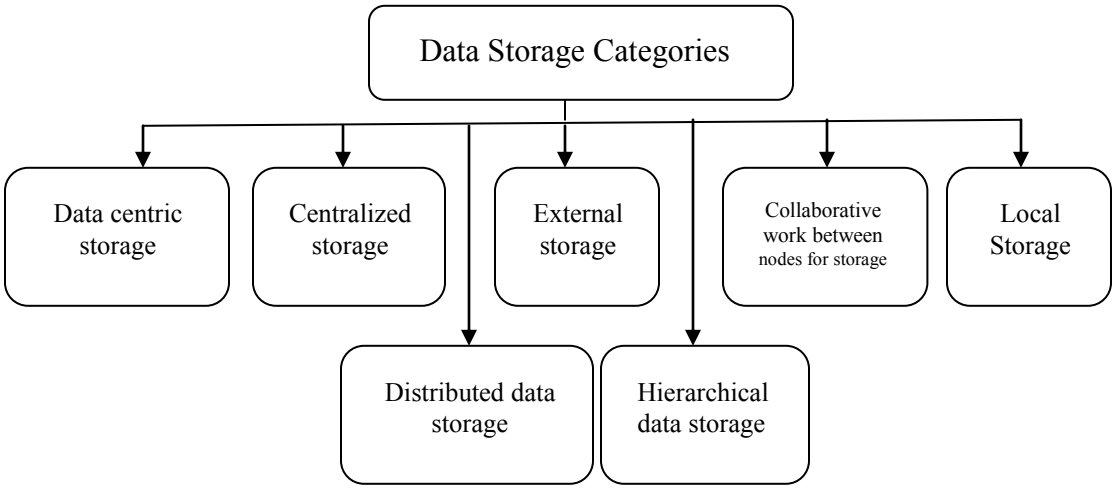


Figure 4.1, Data Storage Categories

4.2.3.1 Local Storage

The data are stored locally in the sensor node which obtained them without any data transmission. Whenever a query is issued by some user, the query has to be flooded to every sensor in the network and each sensor transmits back the qualified local results for the query. This is an expensive approach exactly as the centralized data storage approach when a small fraction of the sensors have qualified data for the query.

In the current implementation of TinyDB [99], events are only signaled on the local node; data is not provided with a fully distributed event propagation system. However, the queries started in response to a local event may be disseminated to other sensor nodes.

4.2.3.2 Collaborative Work between Sensor Nodes for Storage

Cooperative storage systems are mainly designed for sensor networks with disconnected operations where the sensor nodes do not have a connected path to the sink or the base station.

The goal of the cooperative storage systems is to maximize their data storage capacity by appropriately distributing storage utilization and offloading data to external

devices when it is possible. The use of these systems is suitable for a large category of sensor network applications that do not require real-time data access, such as environmental data logging. Such networks generally operate in a disconnected mode. Rather than focusing on multi-hop routing to a base station, the designer of this category wishes to maximize the effective storage capacity of the disconnected sensor network. It accommodates most data, and has the nodes attempt to upload data when the opportunity comes to relieve the network storage.

4.2.3.3 External Storage

The external storage considers an external sink which visits the sensor nodes periodically or up on request. The external storage use data mules which are mobile devices [96]. The data mule is defined as any mobile device that may come in contact with the sensor network islands for the purpose of relieving the stored data.

4.2.3.4 Centralized Storage

The fourth approach is to send the data readings of the sensors to a centralized server or base station where it is stored and processed during the query evaluation. This strategy is suitable for streaming data applications or in scenarios where most of the data generated by the sensors will be used by the query processor. However it is proven to be too costly in communication overheads when the user is only interested in a small fraction of the sensor data [87–90].

Enabling the sharing of sensor data over a common platform is a goal pursued by the SensorBase.org [106] project. SensorBase.org offers a centralized data storage and management system, which provides a uniform and consistent method to “slog” sensor network data. The term ”slog” is a combination of ”sensor” and ”log” reflecting the spirit of sharing information in a blog for sensor nodes. In the centralized case, data are sensed, processed, aggregated and managed at a central location usually the base station.

4.2.3.5 Data–Centric Storage

Data-centric storage, organizes the sensor data into the network using a mapping function. The data-centric approach utilizes a mapping function which maps every data object generated in the network to a sensor called owner based on some

attributes in the data object. The owner is responsible for the storage of the data object and processes locally queries referencing this object.

When a user wishes to query the network, they can send the query only to the owner node responsible for the data relevant to the query through some efficient geographic routing mechanism without flooding the query in the network. The various data-centric systems presented in the literature differ mainly in the mapping function used, which could be a hash function or a tree like structure. A common feature for most of them is that they all require knowledge of the geographic location of the sensors which is not always possible (e.g. if the sensors are not equipped with GPS locator, or if the sensors are located in areas where the GPS systems are blocked such as tunnels).

Examples of data centric storage are the followings:

1- GEM [99] is a data-centric routing and storage system that does not require knowledge of the sensors locations. In GEM, a labeled graph is computed and embedded into the original network topology. The labels assigned to the sensors allow messages to be efficiently routed through the network, while each node only needs to know the labels of its neighbours. GEM utilizes only the leaf nodes to index and store the sensor data, which wastes the storage capacity of the internal nodes in the graph. In addition, GEM does not provide any recovery mechanism for data loss due to node failures and the data size maintained at different sensors cannot be balanced dynamically according to the distribution of the sensor data.

Surprisingly, limited research has been done in the area of reliable data storage for sensor networks. In order to make the Data-Centric Storage (DCS) systems reliable, several works propose to send a special refresh message from the owner node in the network to all nodes which have generated objects stored at the owner.

A GPSR [107] routing protocol is then utilized to return these refresh messages to the owner node with a network perimeter attached. If it is discovered that there is a new node closer to this location than the original owner then the new node will become the owner of the object and will start transmitting refresh messages. This process, however, does not protect the network against data loss due to node failures as the data kept at the failure node will be lost.

2- In [98] the authors propose a Resilient DCS system to achieve scalability and resilience by replicating objects in strategic locations in the network. The idea is to store the object at different replica nodes generated by a hash function. The replica nodes keep exchanging information in order to get a consistent overview of the object generated in the network. This approach, though effective against failures, requires a global view of the network topology along with the position of each sensor, and thus is too expensive or impractical for many sensor network applications.

4.2.3.6 Distributed Data Storage

In the distributed approach, after the sensor node has generated some data, the node stores the data locally or at some designated nodes within the network, instead of immediately forwarding the data to a centralized location out of the network.

For example, a WSN is deployed over a battlefield for military surveillance. The WSN is aimed at providing information services to its authorized users, e.g., soldiers, which frequently move in the field, query the network on demand, and expect real time answers. Distributed data storage approach is used. That is, sensor nodes sense and store various kinds of environment data locally and provide the authorized users the access to the stored data in a distributed manner when queried. In such an application, distributed data storage results in a considerably more robust network as compared to the centralized approach because it may not be feasible to maintain a centralized entity in the hostile environment as the centralized entity itself would become an easy target for attack. Also, if a centralized data storage and access approach is implemented, every query must go through the centralized entity thus the data access delay could be significantly increased; not to mention that the query or data response could be lost due to link failures, traffic congestion, or other reasons, and the result can be devastating.

Therefore, a fault tolerant and compromise-resilient distributed data storage mechanism has to be in position to guarantee the success of such mission critical applications.

WSN security has been extensively studied in recent years with a focus on network communication security. However, distributed data storage security is a fairly new area and has received limited attention so far.

4.2.3.7 Hierarchical Data Storage System

For hierarchical storage systems, all sensor nodes cooperate in storing data into a single database.

The hierarchical data storage system for sensor nodes is based on the constructed tree of the hierarchical architecture. The tree-like structure will have two types of nodes; the normal nodes or forwarding nodes and the special nodes or storage nodes.

There are two main functions in hierarchal data storage system which are the followings:

1- Storage Tree Construction:

In [103], for hierarchical storage, there must be a reliable and load balancing data storage algorithm for sensor nodes. More specifically, this algorithm will have to deal with problems related to the data storage system. The algorithm will use a tree-like structure as network topology for both data storage and in-network routing in hierarchical sensor networks. The algorithm constructs a routing tree that covers all sensor nodes in the network. The algorithm starts out by assigning the base station as the root of the tree and then the root broadcasts the tree construction message to its neighbors asking more sensor nodes to organize into the routing tree.

2- Communications of Nodes in the Tree:

This section presents the mechanism used to communicate between sensor nodes. In [103], in the tree construction phase, each node has several children and a parent in the tree as its neighbours. There are communications links between sensor nodes in the tree as routing paths to the base station and sensor nodes store other routing paths as alternative communications links to the base station.

We need to employ the distributed security concept for a WSN to enable group of sensor nodes to assign a node among them to be a security manager. The security manager stores the security-related data of its downstream sensor nodes for further data recovery in case these sensor nodes experience security threats during periods when base station is not available. The discussion of the security managers will follow in section 4.4.1.

We propose the first security architecture which is called surveillance security (SurvSec) for reliable network recovery from BS failure of surveillance WSN in hostile environment. More specifically, the architecture should allow two main steps. First step is the storage of security data to store the security data and secure the network instead of deploying a new network which comes at a high cost. The second step is the reliable recovery from BS failure by collecting the stored data to the new BS.

4.3 Network Assumptions, and Evaluation Metrics

4.3.1 Network Assumptions

We consider a hierarchical sensor network that is composed of large number of sensor nodes with unique ID and single base station placed in layers where one layer is defined as group of nodes connected to the upper sensor node. The nodes are arranged in clusters and it is assumed they have the ability to detect the compromised nodes. The nodes have Local Intrusion Detection System (LIDS) capable of detecting Cloning attack, Sybil attack and other attacks.

Meanwhile, some nodes continuously store the detected security threats and all other security data related to sensor nodes where these nodes are named security managers. Following the previous works on data storage in WSNs, there are several categories but two main approaches: Centralized data storage [87–90] which is suitable for streaming data applications, and Distributed data storage [91–94, 88] which is suitable for providing information services to the authorized users such as soldiers in the battlefield. Other approaches are the data centric storage systems and those based on the collaborative work between sensor nodes to build the data storage infrastructure systems.

4.3.2 Evaluation Metrics

The evaluation metrics are the followings:

- Low communication overheads.
- Low storage overheads.
- Low recovery overheads.
- High network trustworthiness.

- Small distributed users' table size.

4.4 Overview of SurvSec Security Architecture

In this section, we provide an overview of the SurvSec security architecture. We need to identify the required procedures to store the security-related data which will allow reliable network recovery from base station failure. Also, this section describes the functionalities of sensor nodes selected as security managers to employ the distributed security concept for the sensor network.

SurvSec has a security report and this report content is the security-related data of sensor nodes which are: Node Index, and part of the reported attacks are: Side Channels Attacks, Forward Secrecy Attacks, Node Compromise Attack, Revoked Node, Local Intrusion Detection (LID) Cloning Attack, LID Sybil Attack, LID Sinkhole Attack, LID Wormhole Attack, LID Selective Forwarding Attack, Node Outage, Awake Node, Sleep Node, Node Failure, Node Misbehavior, Selfish Node, Message Corruption, Routing Attacks, Denial-of-Service (DoS) Attacks, Security Level, Re-keying.

4.4.1 Security Managers Setup and Functions

In wireless sensor networks, all the security-related information concerning the sensor nodes must be stored in a distributed manner in some sensor nodes which will be named security managers (SMs). This is to allow the network to be able to verify the trustworthiness of the sensor nodes after security attacks and during all critical situations such as base station failure by retrieving the stored critical information of the security threats such as compromised node attack.

The security managers are responsible for the followings:

1- Storage and management of the security-related data of sensor nodes.

2- Distribution and exchanging of the shared keys between sensor nodes for encryption.

3- Security managers have a very important feature to add to the security of the WSN which is its capability to stop data query from spreading to every sensor node by flooding messages. This feature provide the network with the ability to return data back to the sink from only the security managers where this data is concerning the security-related data of all sensor nodes.

SMs Network Setup and the Methodology used to choose the SMs:

1- The base station has the network topology of all of the sensor nodes and their locations.
2- The base station divides the overall network into divisions of two layers as shown in Figure 4.2 where the overall network is five layers.
3- The base station assigns the first layer of the security managers as the sensor nodes underneath the BS then cluster heads of the first layer sensor nodes. The security manager shares a key with every node of its downstream nodes as shown in chapter 6.
4- The base station assigns the next layers of the security managers after one layer of the nodes underneath the BS and so on. The security manager generates a group key between the security managers and its downstream sensor nodes.
5- The base station changes the security managers from time to time according to the sensor nodes power and the lifetime of the network.

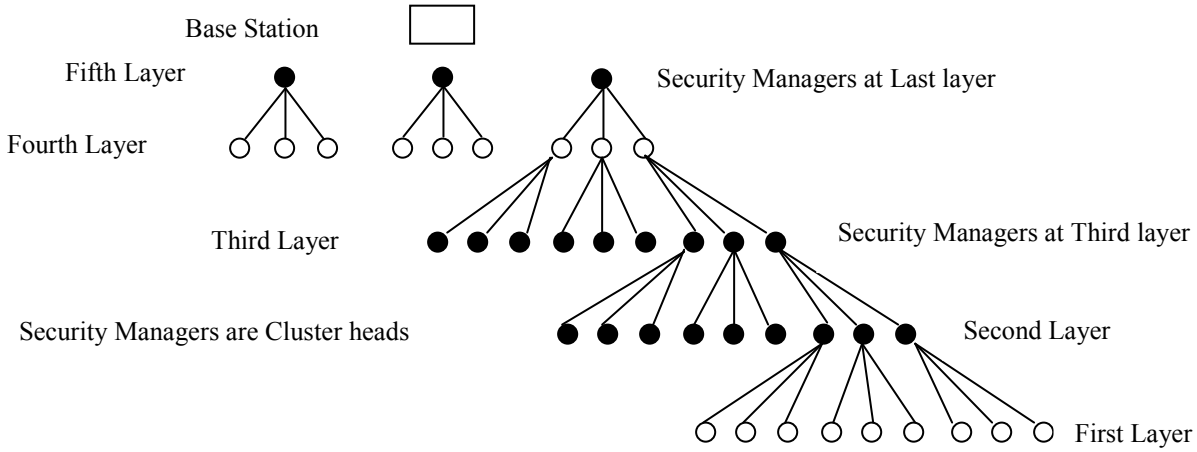


Figure 4.2, Security Managers Network Setup

Security managers will be found every two layers to lower the storage overheads.

4.4.2 Communications of Nodes in the Tree

We have two sensor nodes which are forwarding nodes and security managers' nodes.

The security managers are chosen by the base station. If a security threat takes place at a sensor node, the sensor node will report the security threat to its security manager such as wormhole attack. Also, if a compromised node attack takes place at a sensor node, the sensor node's upper layer node in the hierarchical architecture will report this security threat to the security manager which is responsible for this sensor node.

4.4.3 SurvSec Components:

1- The main component of SurvSec is the hierarchical security managers which are vital to the implementation of the distributed security concept.
2- The second component is the data storage system with a proposed frame format for stored data.
3- The third component is the data recovery system.
4- The fourth component is the security for the data storage system.

4.4.4 Case of Study

Compromised Security Manager:

We proposed a solution to solve the compromised security manager problem or failed security manager by applying re-clustering for this branch of sensor nodes to reroute the data passing through the security manager, revoke the security manager keys, inform the nodes downstream the security manager with the compromised security manager to prevent any node from sending to the security manager and finally, we choose another security manager.

Another solution, we can choose a backup security manager at the same layer of the security manager to take the place of the security manager when it fails.

4.5 SurvSec Data Storage System

In this section, we explain the stored data frame format and the security threats coding where we found that the base station failure is the worst attack scenario because the attacker can compromise many legitimate nodes and the new base station cannot verify the trustworthiness of the deployed sensor nodes of the network. The heart of our

system is founded on the use of encoded attacks, stored data frame format and data recovery system to allow reliable network recovery from the base station failure of Surveillance WSNs. Our security system enables lightweight distributed data storage and recovery system using senor nodes called security managers.

4.5.1 SurvSec Nodes Indexing and Threats Coding

1- Nodes Indexing.

   Each node has a unique node ID. The node ID is stored at the security manager unencrypted to be searched in case of incoming enquiries to investigate the sensor nodes security status. Nodes IDs are loaded at the factory before the sensor nodes join the network.

2- Threats Coding.

   We will build a table to encode each security threat into a determined bits code at that table which is loaded on each sensor node. Each threat has a unique code.

4.5.2 SurvSec Data Storage Frame Format

The stored data frame format shown in Figure 4.3 is the following:

1- Count presents the attack number against the sensor node. We assume that count is 8 bits to enable maximum attacks number of 256 which are the number of attacks to the cluster. This count is made by the SM.

2- Time presents the time of the attack. We assume time is 24 bits to enable 8 bits for the hours, 8 bits for minutes and 8 bits for seconds. This time is 24 bits and it is added at the security managers.

3- Attack ID, We assume that Attack ID is 8 bits to enable maximum attacks of 256 attacks. This attack ID is sent by the node itself.

4- Attacked node ID, We assume Attacked node ID is 16 bits to enable maximum number of 65536 nodes in each branch of sensor nodes. This node ID is sent by the node itself or by monitored nodes.

5- Attacked node reputation, We assume Attacked node reputation is 8 bits to enable 4 reputations levels which are good, medium, over medium and bad. It is added at the security managers.

6- Data replica, We assume Data replica number is 8 bits to enable 256 data replicas within the security managers. This data replica number is added at security managers.

7- Stored record data integrity, We assume Stored record Data Integrity is 32 bits to enable checking the integrity of the stored data records. This data integrity is added at the security managers.

The total stored data at security manager of one monitored sensor node as one record for one attack is 104 bits and it will be increased by 104 bits for each different added attack.

Count, 8 bits	Time, 24 bits
Attack ID, 8 bits	Attacked node ID, 16 bits
Data replica, 8 bits	Attacked node reputation, 8 bits
Stored record data integrity, 32 bits	

Figure 4.3, Data Storage Frame Format

Figure 4.3 describes SurvSec data storage frame format.

4.6 SurvSec Data Recovery System

This section describes our proposed recovery system where we found that we cannot use the erasure coding [109] to recover the error at the 8 bits of the attack ID because the number of used bits for error correction code will largely increase as the attacks increased. Also, the computations required to generate and recover the errors of the encoded attacks will be expensive. Therefore, we proposed to use multiple replicas to ensure the correct query results when investigating the situation of a sensor node security status.

There must be at least three replicas of the stored data for data recovery where each data record specifying a sensor nodes security attacks is stored at least three times at three security managers to allow queries to be sent to two security managers at a time.

The procedures for the stored data recovery system:

1- During the base station failure the sensor nodes send their security reports which include the attacks ID on the sensor nodes along with the data integrity for the stored data frame format of the security-related data to the security managers.
2- The security manager checks the data integrity of the sent data and if the security manager finds an error in the process of verification for the data integrity, the security manager will send two queries to two security managers underneath to ensure the correct result of the attack ID for the reported senor node. If the two results are the same, therefore, there is no problem to accept the result. But, if the two results are different with polluted data integrity, the security manager will send a query to a third security manager to ensure the attack ID.
3- After the authentication process of the newly deployed base station or the recovered base station, the last layer of the security managers will send the security-related data of its downstream sensor nodes to the base station.
4- The base station checks the data integrity of the sent data and if the base station finds an error in the process of verification for the data integrity, the base station will send two queries to two security managers underneath to ensure the correct result of the attack ID for the reported senor node. If the two results are the same, therefore, there is no problem to accept the result. But, if the two results are different with polluted data integrity, the base station will send a query to a third security manager to ensure the attack ID.

4.7 SurvSec Secure Data Storage System

In this section, we describe the dynamic secret sharing concept to generate our proposed distributed users table. This is done to generate a new dynamic secret sharing algorithm which is used to stop eavesdropping on the users that holds the secret shares with the security managers.

4.7.1 Secret Sharing:

This section describes Shamir secret sharing for which Shamir proposed an (t, n) Secret Sharing (SS) scheme [82] based on polynomial interpolation, in which t of n shares of a secret are required to reconstruct the secret [80].

Shamir's Secret Sharing: [81, 110]

The secret k is in Z_p (p is prime, and $p > n$). Each shareholder i is in the set P ($|P| = n$).

All mathematical operations are in the Finite Field Z_p.

To distribute k, select a polynomial $a(x)$ with degree ($m-1$) and constant term k. Generate a share s_i for each i in P with $a(x)$: $s_i = k + \sum_{j=1}^{m-1} a_j i^j$ and s_i is also in Z_p.

To reconstruct k, retrieve m coordinate pairs (i, s_i) of all i in authorized subset B of P ($|B| = m$) and use the pairs in the Lagrange interpolation formula: $k = \sum_{i \in B} b_i s^i$, where $b_i = \prod_{j \in B, j \neq i} \frac{j}{j-i}$.

Example:

1- Building shares:

Suppose that our secret is 1234, $S = 1234$.

We wish to divide the secret into 6 parts ($n = 6$), where any subset of 3 parts ($k = 3$) is sufficient to reconstruct the secret. At random we obtain 2 numbers: 166, 94.

($a_1 = 166$; $a_2 = 94$)

Our polynomial to produce secret shares (points) is therefore:

$f(x) = 1234 + 166x + 94\,x^2$

We construct 6 points from the polynomial:

(1,1494); (2,1942); (3,2578); (4,3402); (5,4414); (6,5614)

We give each participant a different single point (both x and $f(x)$).

2- Reconstruction:

In order to reconstruct the secret any 3 points will be enough.

Let us consider: $(x_0,y_0) = (2,1942)$; $(x_1,y_1) = (4,3402)$; $(x_2,y_2) = (5,4414)$

We will compute Lagrange basis polynomials:

$\ell_0 = (x - x_1 / x_0 - x_1) \,.\, (x - x_2 / x_0 - x_2)$

$$= (x - 4/\ 2 - 4)\ .\ (x - 5/\ 2 - 5) = \frac{1}{6}x^2 - 1\frac{1}{2}x + 3\frac{1}{3}$$

$$\ell_1 = (x - x_0/\ x_1 - x_0)\ .\ (x - x_2/\ x_1 - x_2)$$

$$= (x - 2/\ 4 - 2)\ .\ (x - 5/\ 4 - 5) = -\frac{1}{2}x^2 + 3\frac{1}{2}x - 5$$

$$\ell_2 = (x - x_0/\ x_2 - x_0)\ .\ (x - x_1/\ x_2 - x_1)$$

$$= (x - 2/\ 5 - 2)\ .\ (x - 4/\ 5 - 4) = \frac{1}{3}x^2 - 2x + 2\frac{2}{3}$$

Therefore;

$$f(x) = \sum_{j=0}^{2} y_j\ .\ \ell_j(x)$$

$$= 1942\ .\ (\frac{1}{6}x^2 - 1\frac{1}{2}x + 3\frac{1}{3}) + 3402\ .\ (-\frac{1}{2}x^2 + 3\frac{1}{2}x - 5) + 4414.\ (\frac{1}{3}x^2 - 2x + 2\frac{2}{3})$$

$$= 1234 + 166\,x + 94\,x^2$$

Recall that the secret is the free coefficient, which means that $S = 1234$.

Secret sharing can be divided into four phases as shown in the nest figures of Figure 4.4 until Figure 4.7. These phases are the followings:

1- Shares distribution phase
2- Shares building phase
3- Secret reconstruction phase
4- Shares update phase

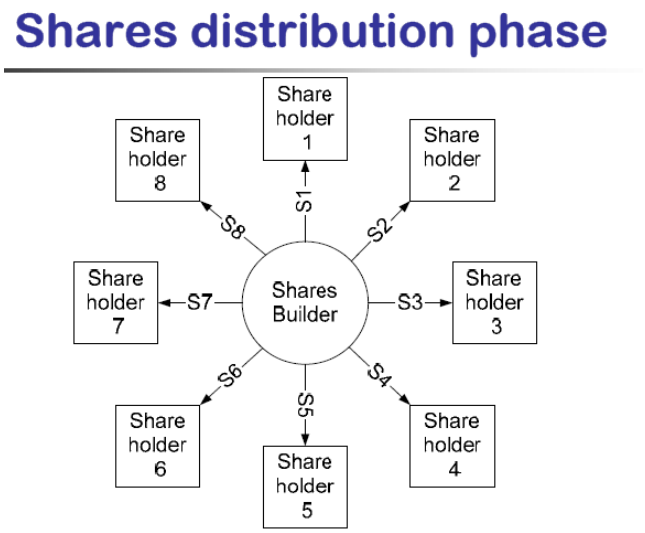


Figure 4.4, Phase 1; Shares Distribution

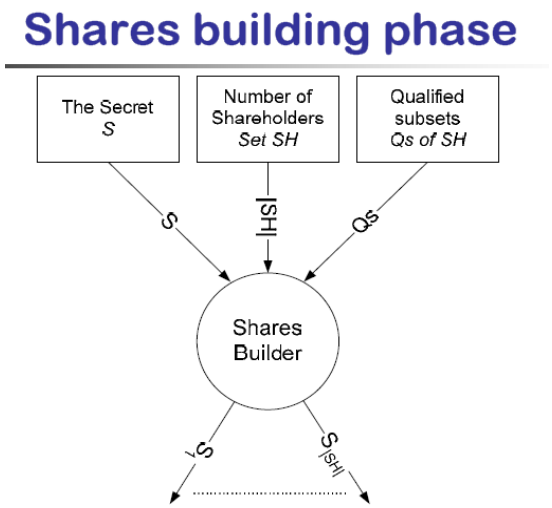


Figure 4.5, Phase 2; Shares Building

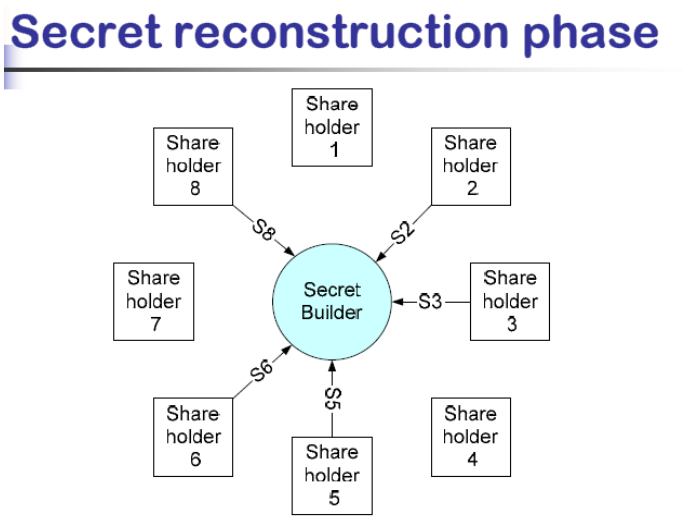


Figure 4.6, Phase 3; Secret Reconstruction

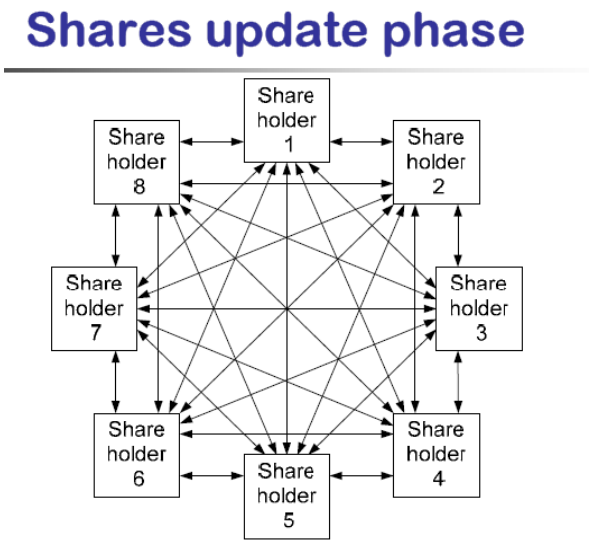


Figure 4.7, Phase 4; Shares Update

4.7.2 Dynamic Secret Sharing:

Secret sharing scheme is a threshold scheme in that without enough shares the secret is information-theoretic secure. There exist many secret sharing schemes. One of them is Shamir's scheme based on polynomial interpolation. Another scheme is the dynamic secret sharing which depends on changing the polynomial and changing the users.

1- Dynamic Secret Sharing by Dynamic Polynomials:

The dynamic polynomial can depend on changing the shared secrets. This is done to eliminate the weaknesses of the secret sharing such as eavesdropping to know the shares holders. In 1994, He and Dawson [111] proposed a multistage secret sharing scheme based on the one-way function. By applying successive one-way hash functions, the He-Dawson scheme realized the notion of multi-secret sharing. In 2007, Geng et al. [112] pointed out that the He-Dawson scheme was actually a one-time-use scheme [113] and further proposed a new multi-secret sharing scheme with multi-policy.

2- Dynamic Secret Sharing by Adding New Users:

Dynamic secret sharing [201] can be done by adding new users which is known as multi-level secret sharing. In Multi-Level Secret Sharing, shares have distinct weight (impact) in the secret construction. That is, secret construction requires a smaller number of weightier shares but a large number of lighter shares. Simmons in [114] introduced the disjunctive multi-level access structure. Tassa in [115] introduced the conjunctive multi-level access structure. M. Belenkiy in [116] recently presented a disjunctive multi-level secret sharing scheme. This is the first polynomial-time solution that allows the dealer to add new users dynamically and is by far the most efficient.

4.7.3 Proposed Distributed Users Table:

In this section, we describe our designed dynamic secret sharing which includes dynamic users by changing the shared users in the distributed user's tables. Each SM shares with its downstream sensor nodes multiple shares of secrets to build the used key for the encryption process which is carried out on the SM to securely store the security-related data of sensor nodes. We assume that we use an Arduino Uno Board therefore; the SM or any ordinary sensor node has a memory size of 32 Kbytes.

Adding distributed users table to secret sharing will allow adding and changing of users to enable dynamic users for the secret sharing. This is done to stop eavesdropping during encryption of security-related data.

Our dynamic secret sharing includes reconfigurable distributed users' tables to change the shared users after only two hops.

The total number of nodes around the SM after two hops, which is our bounded limits for multiple hops around the SM, represents the total nodes space which shares with the SM the key space that is used to encrypt the stored data of security-related information.

Therefore, we will have a large group of nodes which can share with the SM the shared secrets with the ability to join new nodes and change other nodes. Furthermore, the members of the distributed users table must be able to deliver the request of the SM to its destination that shares the secret and must be able to deliver the required shared secret from the destination to the SM.

We need to update the distributed users table from time to time depending on the detected compromised nodes. This is done to ensure that there is no compromised sensor node that holds a secret with the security manager sensor node.

The distributed users table has a significant property to add to the system that it allows the dynamic change of the table through reconfiguration of the table where distributed users tables are reconfigurable to change the users and this is done to allow dynamic security.

We need three phases to build the distributed users table: first, the table initialization phase; second, the table establishment phase and third, the table reconfiguration phase.

The initialization phase to build the distributed users table:

1- The BS assigns the security managers and its downstream sensor nodes.
2- The SMs discover its downstream sensor nodes.
3- Each SM shares a group key with its downstream sensor nodes as shown in chapter6.
4- Each SM builds the distributed users table from knowing its downstream sensor nodes.
5- SMs communicate with its shared nodes to share secrets with the SMs in only two hops while the security managers are each two layers and each sensor node can store two distributed users table.
6- Each sensor node will store only two distributed users table which is shared with its security manager.

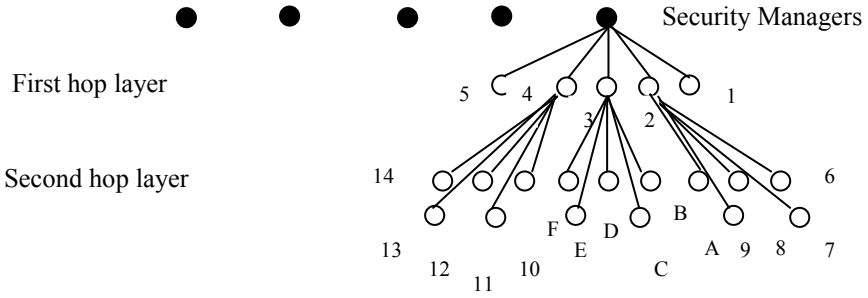


Figure 4.8, Distributed Users' Table Nodes

Figure 4.8 shows the distributed users table construction from the sensor nodes downstream of the security manager with two hops where there are five sensor nodes downstream of the security manager.

First, the SM builds the overall distributed users table and sends it to the BS. Table 4.1 is the overall distributed users table assuming there are five sensor nodes downstream the security managers:

Table 4.1, Overall Distributed Users Table

Index	Count & Reconfigured Count	First hop Node	Second hop Node	Destination that shares the secret
1	0			0X 05,0A,00,15,00
2	1	1	0X 6 - A	0X 06,0B,00,16,00
3	2	2	0X B - F	0X 07,0C,12,00,00
4	3	3	0X 10 - 14	0X 08,0D,00,18,00
5	4	4	0X 15 – 19	0X 09,0E,00,00,1E
6	5	5	0X 1A – 1E	0X 0A,00,15,1A,00

1- The overall distributed users' table is located at the base station where the security manager sends it to the base station and the SM sends the update of the overall distributed users' table to the base station.

2- Initially the count is equal to zero where there are no attacks and the SM shares the secrets with the sensor nodes at the destination of count equal to zero.

3- We assume that each security manager has 5 downstream nodes and we assume we need only 3 shares to reconstruct the secret information which is the key.

4- The count field represents the counted attacks to the sensor nodes downstream of the SM. This field is increased by 5 after it is used. Using the counted attack will result in definite sensor nodes at the destination to share the secret with the SM.

5- The second field is the first hop sensor node.

6- The third field is the second hop sensor nodes.

7- The fourth field is the destination sensor nodes which share the secret with the SM and at the count equal one the first destination is set to hexadecimal value of 06, the second destination is set to hexadecimal value of 0B, the third destination does not share any secrets with the SM, the fourth destination is set to hexadecimal value of 16, and the fifth destination does not share any secrets with the security manager.

Second, the SM builds the distributed users table at the security manager.

Table 4.2 is the distributed users table at the security manager. It explains the data to the first hop sensor node to deliver requests from the security manager to the first hop sensor node and to return requests from the first hop sensor node to the security manager:

Table 4.2, Distributed Users Table at the Security Managers Sensor Nodes

Index	Count & Reconfigured Count	Destination Nodes that shares the secret	Path
1	0	0X 05,0A,00,15,00	1-5, 2-A, 3-0, 4-15, 5-0
2	1	0X 06,0B,00,16,00	1-6, 2-B, 3-0, 4-16, 5-0
3	2	0X 07,0C,12,00,00	1-7, 2-C, 3-12, 4-0, 5-0
4	3	0X 08,0D,00,18,00	1-8, 2-D, 3-0, 4-18, 5-0
5	4	0X 09,0E,00,00,1E	1-9, 2-E, 3-0, 4-0, 5-1E
6	5	0X 0A,00,15,1A,00	1-A, 2-0, 3-15, 4-1A, 5-0

1- We count the sensor nodes downstream of the SM with 2 hops where the SM has 5 downstream sensor nodes and each node has 5 downstream sensor nodes with a total of 30 sensor nodes downstream of the SM.

2- The path column is not used because it is the explanation of the destination column where at the count equal one the first location 0X06 means the destination that holds

the share is the sensor node number 0X6 from the first node at the first hop from the security manager.

3- Also, 0X0B means the destination that holds the share is the sensor node number 0XB from the second node at the first hop from the security manager.
4- Also, 00 means that there is no destination sensor node through the third and fifth node at the first hop from the security manager. This is done because there are only three sensor nodes that shares secrets with the security manager.
5- Also, 0X16 means the destination that holds the share is the sensor node number 0X16 from the fourth node at the first hop from the security manager.
6- Each node at two hops from the SM takes a unique number from 0X00 to 0XFF hexadecimal values.
7- The size of the table at the security manager is 240 bits from 40 bits at each field in the third column where there are five sensor nodes each with 8 bits, and the reconfigured count is 8 bits. Therefore, we have 40 bits multiplied by 5 records and 8 bits multiplied by 5 records with total of 240 bits data.

The establishment phase to build the distributed users table:

First, the SM builds the distributed users table at the first hope sensor nodes.

Table 4.3 is the distributed users table at the first hop sensor nodes. It explains the data to the second hop sensor node to deliver requests from the first hop sensor nodes to the second hop sensor node and return the requests to the security manager:

Table 4.3, Distributed Users Table at the First Hop Sensor Nodes

Index	Node at first hop	Path at second hop
1	1	0X 6 – A
2	2	0X B – F
3	3	0X 10 – 14
4	4	0X 15 – 19
5	5	0X 1A – 1E

The first raw is stored at the first sensor node and the second raw is stored at the second sensor node and so on. The table size is 40 bits at each sensor node where we have five users each with 8 bits.

The path column at second hop first locates where the destination is then it locates the sensor node to that destination. Also, the sensor node at the first hop which is number one takes only raw number one and so on.

Second, the SM builds the distributed users table at the second hop sensor nodes.

Table 4.4 is the distributed users table at the second hop sensor nodes. It explains the data to return requests from the second hop sensor node to the first hop node:

Table 4.4, Distributed Users Table at the Second Hop Sensor Nodes

Index	Node at first hop	Path at second hop
1	1	0X 6 – A
2	2	0X B – F
3	3	0X 10 – 14
4	4	0X 15 – 19
5	5	0X 1A – 1E

The first raw is stored at the first sensor node and the second raw is stored at the second sensor node and so on. The table size is 40 bits at each sensor node where we have five users each with 8 bits.

The path column of second hop only locates where the destination is from second hop sensor node to first hop sensor node. Also, sensor node at the second hop which is number one takes only raw number one and so on.

The reconfiguration phase to build the distributed users table:

The SM compares between the existing sensor nodes in the distributed users table which are fifteen nodes and the overall sensor nodes downstream of the SM which are thirty nodes. The SM adds new sensor nodes to the distributed users table for reconfiguration.

The encryption of security-related data is done using the spread spectrum encryption architecture as shown in chapter 7.

The main contributions for our proposed dynamic secret sharing algorithm with the distributed users table are explained in this section as we proposed a novel idea of a

distributed users table based on the concept of dynamic secret sharing. Our proposed security scheme has the following properties:

(1) It provides dynamic secret sharing with adding and changing of multiple users;
(2) It can limit the damage from compromised sensor nodes since the compromised node can be easily revoked from the distributed users table;
(3) It preserves small size for distributed users table but with high search space for the attacker to decrypt the secure stored data;
(4) It is scalable to large sensor networks due to its lightweight computation and easy key management.
(5) The stored secure data contains the users used for the secret sharing.

4.8 Simulation Results and Performance Analysis

We built an analytical model for the proposed design and we implemented a simulator in MATLAB that can scale to thousands of nodes. In this simulator, sensors can send and receive data from each other's. This data is the security-related data regarding the security reports of sensor nodes. The simulation verifies the correctness and the feasibility of our security architecture. It is our future work to implement SurvSec in some sensor network test beds with all its ingredients. Our simulation scenarios include N nodes distributed randomly. We choose N as 10.000 sensor nodes.

The followings are the built models for simulation:

1- Network setup model for the security managers.
   This model shows the security managers setup during initialization phase of SurvSec.
2- Attacker model.
   This model shows how the attacker can attack the sensor nodes of the network.
3- Changing of security managers' model.
   This model shows how we recluster to change the security managers.
4- Data storage model.
   This model was explained in section 4.5.
5- Data recovery model.

This model was explained in section 4.6.

6- Security model to secure the stored data using the distributed users table.
   This model was explained in section 4.7.
7- Update / Delete security-related data model.
   This model shows how we update the security-related data at the security managers.
8- Network trustworthiness model.
   This model shows how we can trust the network after the BS failure.

4.8.1 Metrics:

The following metrics are considered:

1- Communications overhead: it is defined as the number of queries sent form the sensor node to the SM result from number of attacks then from the SM to other SMs until the last SM at the last layer of sensor nodes near the BS. We need SurvSec to have minimum communications overheads. Figure 4.9 shows that the communications overheads increase as the number of attacks increase.

We assume eight layers sensor network and each attack is at the first layer which will result in eight communication overheads at all layers of the sensor network as shown in Figure 4.9. When an attack occurs at the first layer sensor nodes as shown in Figure 4.2 the attacked sensor node will send a query to its SM with one communication overhead and so on until the BS with total of 8 communications overheads.

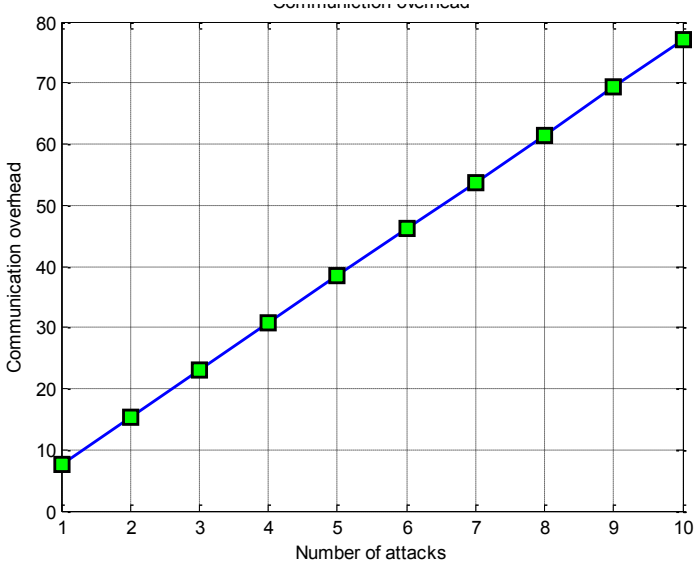


Figure 4.9, Communications Overhead

Communication overheads= $K \times n$.

Where K is (number of layers – the layer of the attack + 1) and n is (number of attacks)

2- Storage overhead: it is defined as the total stored data at the entire security managers' plus the base station which results from number of attacks. Figure 4.10 shows that the storage overheads increase as the number of attacks increase. We assume eight layers sensor network and each attack is at the first layer which will result in storing the data at three security managers and BS with total of 416 bits storage overheads where one attack store 104 bits of security data as shown in Figure 4.10.

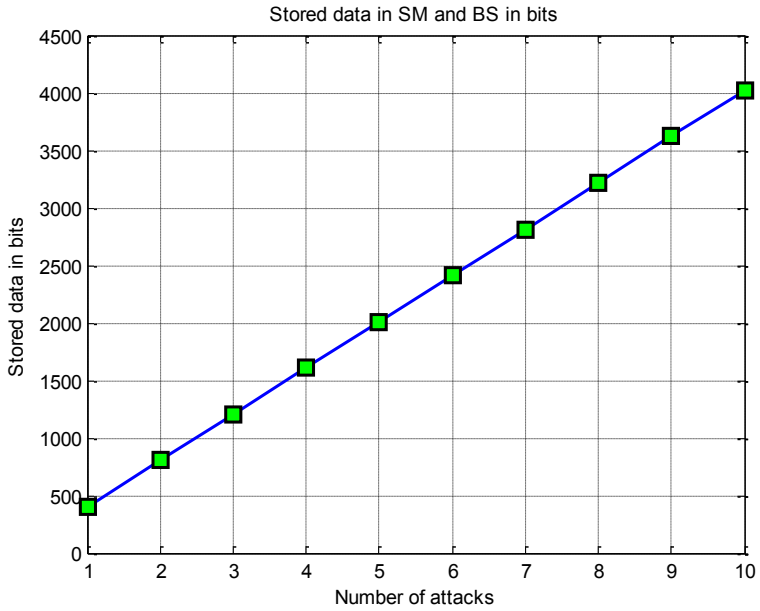


Figure 4.10, Storage Overhead

Data storage overheads = 104 × K × (n+1),

Where K is (number of attacks) and n is the number of security managers storing one copy of the security-related data and we add one because the BS also stores the security-related data.

3- Recovered data overhead: it is defined as the data needed to recover from the attacks at the sensor nodes after the deployment of the new base station. Figure 4.11 shows that the recovered data overheads increase as the number of attacks increase.

We assume eight layers sensor network and one attack can be recovered from 104 bits stored data at the last layer of the security managers near the base station as shown in Figure 4.2. The recovered data overheads can be shown in Figure 4.11.

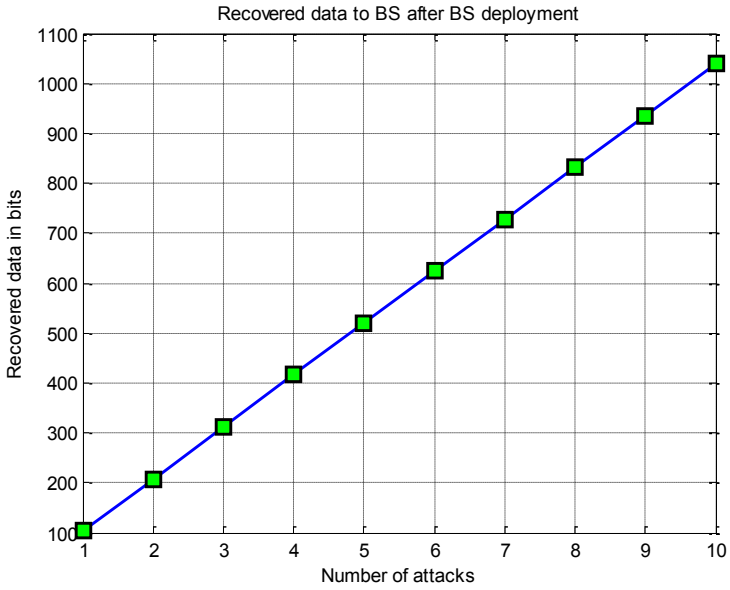


Figure 4.11, Recovered Data to Base Station

Data recovery overheads = 104 × n where n is (number of attacks),

4.8.2 Efficiency:

We now assess the performance of the proposed SurvSec security architecture in terms of the network trustworthiness after the deployment of the new base station and the distributed users' table size. Therefore, first we will analyze the network trustworthiness for our proposed security architecture then we will analyze the distributed users' table size versus the number of nodes in each layer.

1- <u>Network Trustworthiness:</u>

Network trustworthiness is defined as how much percentage the new BS can trust the deployed sensor nodes. The SMs send the security-related information to the new BS therefore; the network trustworthiness is 100% without attacked SMs.

The attacked security managers are critical to the efficiency of SurvSec. Generally speaking, the more attacked security managers the less network trustworthiness.

Figure 4.12 shows the network trustworthiness without any attacks at the security managers while Figure 4.13 shows an increasing rate of attacking the security managers which will result in decreasing the network trustworthiness.

The network trustworthiness is 100% in case there is no attacked security manager and this can be shown in Figure 4.12. Then this network trustworthiness ratio decreases when the security managers are attacked because the security managers as security data senders cannot send their security reports and this can be shown in Figure 4.13.

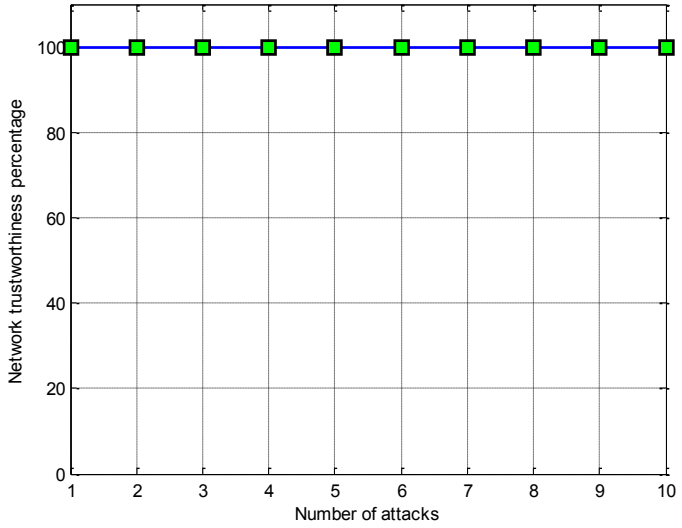


Figure 4.12, Network Trustworthiness without Attacked Security Managers

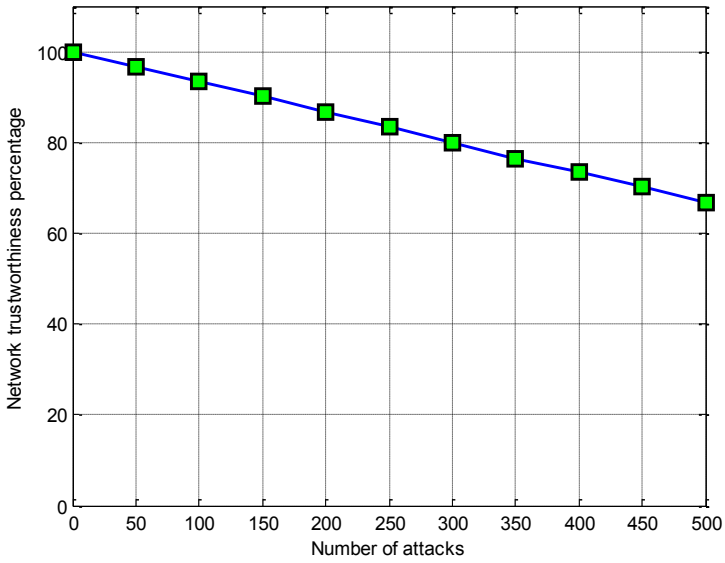


Figure 4.13, Network Trustworthiness with Attacked Security Managers

2- Distributed Users' Table Size:

The distributed users' table is a critical part for SurvSec Security Architecture to enable the delivery of the requests of the SM to its destination, which holds the secret for encryption and delivering the required shared secret from the destination to the SM.

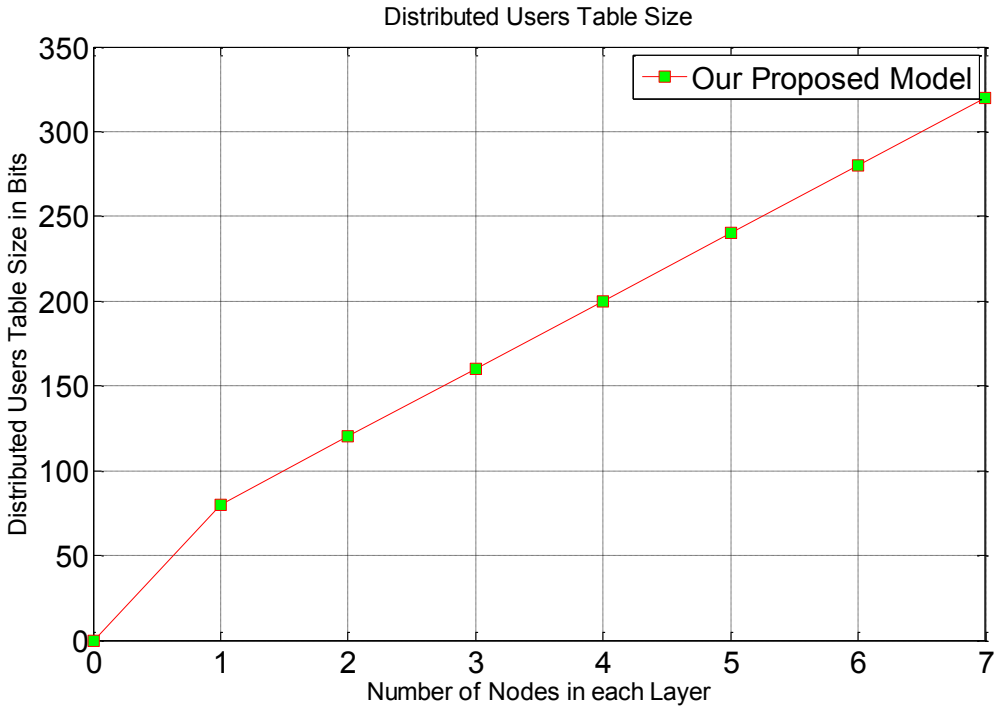


Figure 4.14, Distributed Users Table Size

Distributed users table size = $(n \times 8 \times K) + (8 \times K)$,

Where K is the number of rows in the table and there are five rows in the table and n is the number of sensor nodes in each layer and eight presents the number of bits for each sensor node to store its number.

Figure 4.14 shows the distributed users' table size versus the number of nodes in each layer where the size increases as the number of sensor nodes in each layer increases. From Table 4.2, for three sensor nodes in each layer, the distributed users table size is 160 bits where we have five records each with a count of 8 bits, with a total of 40 bits and 24 bits at each field in the third column where there are three sensor nodes each with 8 bits, with a total of 24 bits multiplied by 5 records added to 40 with total of 160 bits.

From Table 4.2, for four sensor nodes in each layer, the distributed users table size is 200 bits where we have five records each with a count of 8 bits, with total of 40 bits and 32 bits at each field in the third column where there are four sensor nodes each with 8 bits, with a total of 32 bits multiplied by 5 records added to 40 with total of 200 bits. From Table 4.2, for five sensor nodes in each layer, the distributed users table size is 240 bits where we have five records each with a count of 8 bits with total of 40 bits and 40 bits at each field in the third column where there are five sensor nodes each has 8 bits with total of 40 bits multiplied by 5 records added to 40 with total of 240 bits. We found that the distributed users' table size increases with 40 bits for adding one sensor node at each layer.

4.9 Summary

In this chapter, we proposed the first security architecture to achieve secure and reliable network recovery from base station failure. Concretely, we proposed a secure and reliable network recovery from base station failure of surveillance wireless sensor network in hostile environment to improve the security data survival capability in presence of base station failure. We further enhance such scheme by employing distributed security managers and distributed users' table. Our scheme is resilient to base station failure through our designed data storage and recovery systems.

The performance analysis and the simulation results of our proposed hierarchical secure data storage and recovery system provide the WSN with high confidence for secure and reliable network recovery from the base station failure of surveillance WSN in hostile environment.

CHAPTER 5

OVERLAPPED GROUPS TO EARLY DETECT COMPROMISED NODES

In this chapter we describe SurvSec overlapped groups compromised nodes detection algorithm to early detect compromised nodes at the first stage. Node compromise attack is a multi-stage attack which consists of three stages: physically capturing and compromising sensor nodes; redeploying the compromised nodes back to network and compromised sensor nodes rejoining the network. Our work studies how to achieve high resiliency against an increasing number of compromised nodes in large surveillance WSN in hostile environment by collaborative work of attackers at the first stage. Specifically, after sensor nodes are deployed they first build overlapped groups in an Ad Hoc pattern where a group can be any number of nodes. Then, the nodes within the overlapped groups can monitor each other to detect any node compromise attempt.

5.1 Introduction

This chapter is organized as follows: Section 5.2 presents the related work. Section 5.3 describes the network assumptions of the proposed scheme and the threat model. Section 5.4 describes an overview of our security architecture SurvSec for compromised nodes detection at the first stage. Section 5.5 presents the security analysis of the compromised nodes detection algorithm. Section 5.6 presents the performance analysis. Section 5.7 presents the simulation results. Section 5.8 presents comparison with others works. Finally, Section 5.9 is the summary of the chapter.

Surveillance WSNs are deployed in hostile environments such as perimeter, border locations and battlefields to detect unauthorized intrusions. Therefore, Surveillance WSNs are highly vulnerable to collaborative work of attackers to compromise many legitimate nodes. Securing surveillance WSNs is challenging because of low-cost, limited capabilities, resource-constrained sensor nodes. Several protocols have been proposed for detecting compromised nodes. However, some protocols rely on an implicit assumption that compromised node will change its location or its signal strength will be altered after it is compromised; other protocols use alert messages or reputation based trust models which require the nodes misbehavior to discover the compromised nodes. Node compromise attack is a multi-stage attack which consists of three stages: physically capturing and compromising sensor nodes; redeploying the compromised nodes back to network and compromised sensor nodes rejoining the network. Our work studies how to achieve high resiliency against an increasing number of compromised nodes in large surveillance WSN in hostile environment by collaborative work of attackers at the first stage. Specifically, after sensor nodes are deployed they first build overlapped groups in ad hoc pattern where a group can be composed of any number of nodes. Then, the nodes within the overlapped groups can monitor each other to detect any node compromise attempt. We describe the building blocks that can be used to build the protocol for the detection process. Our protocol is designed to be resistant against large number of compromised nodes by collaborative work of attackers. Extensive simulation results are given in section 5.8 to demonstrate the high detection rate of the proposed scheme.

Wireless sensor networks (WSNs) are deployed in many missions' critical applications such as surveillance [1], and one of the key issues to the success of their mission is security. The general objective of such an application is to alert the control unit in advance to the occurrence of events of interest in hostile regions. The event of interest varies according to the mission type which might be the presence of moving vehicles or target detection or other events. There are several types of sensors such as Vibration, Motion, Tracking, Video, and Infrared sensors which can be used for surveillance applications [2]. With their deployment, various novel security attacks have appeared.

The aims of these attacks are usually to compromise nodes, eavesdropping for traffic analysis, destroy base station (BS) or to disrupt data flow. We believe that, collaborative work of attackers will launch compromise nodes attacks against the surveillance WSN to compromise many legitimate nodes and to destroy the deployed network security.

Surveillance WSNs are usually deployed at unattended or hostile environment. Therefore, they are vulnerable to the node compromise attack [117]. A node compromise attack is a three stage attack. In the first stage, the attacker captures some sensor nodes from the network and then compromises these nodes. In the second stage, these compromised nodes are redeployed into the network. In the third stage, the attacker will use these compromised nodes to launch various security attacks. Much work has tackled the node compromise attack [118-128, 182-190]. However, all of them address the node compromise attack either in the second stage based on node redeployment detection [118] or in the third stage based on node misbehaviour detection [119-124, 77]. We believe that group of attackers will launch node compromise attack to jeopardize the whole network in few minutes. Therefore, early detection of node compromise attack can lead to a more effective defense against collaborative work of attackers.

Our focus in this work is to achieve high resiliency against node compromise attack by collaborative work of attackers at the first stage.

To the best of our knowledge, there has not been work done for securing the surveillance WSN at the first stage from collaborative work of attackers to compromise many legitimate nodes at the same time. Therefore, for mission critical applications such as surveillance WSN, we propose to address this problem through employing our new designed overlapped groups-based compromised node detection protocol.

Only two protocols detect compromised nodes at first stage. The first protocol [125] can be easily broken by targeting couple of nodes at the same time and the second protocol [126] has high communication overheads and it is based on the distribution of one key list for all nodes which is not secure if one node is compromised.

Our proposed scheme is based on four algorithms. The first algorithm provides the network with key management. The second algorithm provides the network with secure localization. The third algorithm provides the network with secure clustering. The

fourth algorithm builds overlapped groups from clusters. Each cluster has a security manager SM and a backup security manager BKSM to manage security issues. From the locations of the nodes in the cluster, the nodes can form a group by sending and receiving from their right and left neighbours in the cluster. Each group forms an overlapped group with its neighbour groups. The groups resemble interconnected rings in a chain and if attackers capture one group in the chain, the chain will be cut and its overlapped groups will discover the compromised group. Each node in the cluster sends an encrypted "Hello" message to its neighbours in the cluster every 15 seconds. If a node does not respond to the "Hello" message, this means it is compromised and its neighbours will send to the SM that the node is compromised then to BS and if the SM is compromised, its neighbours will send to the BKSM that the SM is compromised then to the BS.

Our protocol is designed to be resistant against a large number of compromised nodes by collaborative work of attackers. Extensive simulation results are given to demonstrate the high detection rate of the proposed scheme besides the low overheads with high security level for the protocol.

In this chapter, we developed a new overlapped groups-based node compromise detection scheme. Compared with previously reported schemes, the proposed scheme detects the node compromise attack by collaborative work of attackers at the same time in the first stage. Specifically, after sensor nodes are deployed, they first build overlapped groups in an Ad Hoc pattern. The group can be composed of any number of nodes and the nodes are connected in closed loop as shown in Figure 5.2. Then, the nodes within the overlapped groups can monitor each other.

In this chapter, we present a novel node compromise detection scheme against collaborative work of attackers working at the same time in the first stage. Our motivation is the high probability of node compromise attack by collaborative work of attackers to render the whole network ineffective. Our goal is to design new node compromise detection scheme for surveillance WSN in hostile environment.

Contributions of this work can be summarized as:

The **first contribution** is the development of the new security architecture called Surveillance Security (SurvSec) for node compromise detection of surveillance WSN.

The **second contribution** is the formation of overlapped groups to allow each group to monitor its overlapped groups.

The **third contribution** is the early detection of node compromise attack at the first stage.

5.2 Related Work

We need an effective security scheme to identify compromised nodes in a timely manner because compromised nodes in surveillance WSN represent uncovered areas. A node compromise attack involves three stages. From [118-124, 84], the authors proposed many protocols to detect compromised nodes based on location, signal strength, reputation, weighted trust, intrusion detection and MAC layer misbehaviour. However, these approaches are not effective since they can detect compromised nodes on the second or the third stage and they depend on node's misbehaviour or node's location, which means a node may be compromised but behaves well until a programmed time.

In [125] Xiaodong made the first attempt to detect node compromise in the first stage. He described a new couple-based compromised node detection protocol to build couples of sensor nodes in an Ad Hoc pattern to detect node compromise attack at the first stage. The nodes within the same couple can monitor each other. This protocol assumes each sensor node can detect being connected by a programming board during the attack, then the node will send a message to its couple identifying that it is compromised. This protocol cannot be used against collaborative work of attackers to compromise large number of nodes where attackers can collect the couples at the same time. Furthermore, Xiaodong did not explain the path from the couple of the compromised node to the base station to report the compromised node attack where this path is critical to send the message of compromised node attack from the couple to the base station.

In [126], two protocols are proposed and the protocols require high storage overhead for one key list for the whole network, high communication overhead to broadcast "Hello" message to all neighbours then receive the same message from the neighbours, and high energy cost. The two protocols are based on four messages. Each sensor node broadcasts a "Hello" message to his neighbors which receive this message

and reply to it. If the node did not send for three times, it is marked as compromised and the compromised node neighbors flood the network with the node is compromised message. This protocol uses one key list for the whole network which is insecure because if one node is captured, then the key list is known to the attacker and the protocol is no longer secure.

Also, software-based attestation techniques [75, 76, 78, 127, 128] have been proposed to verify the contents of the code running on nodes where the node's free memory space is filled with incompressible random noise known to the attester. These techniques use a challenge-response protocol between a trusted verifier and nodes. A verifier generates a challenge which is a random number and sends it to a suspected node. When receiving this challenge, the node traverses its memory in a pseudorandom fashion and recursively computes a cryptographic checksum over each traversed memory space, and then sends the final checksum to the verifier. The verifier can verify the result since it knows the expected memory image of a legitimate node. Software-based attestation techniques based on the base station as verifier will incur large secure communications overheads with all the nodes for testing the whole network [128] and also the base station could be a single point of failure.

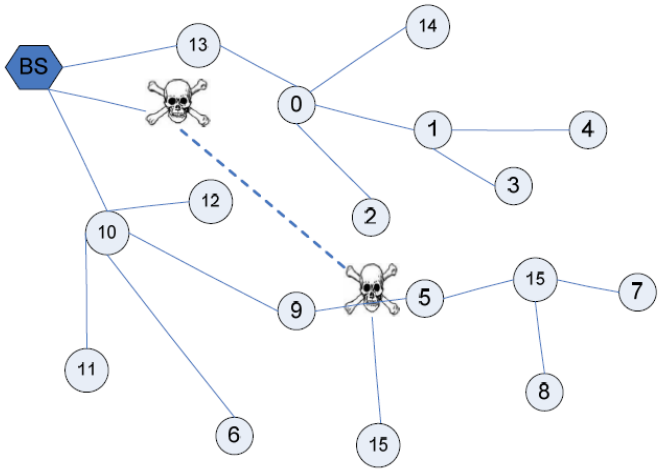


Figure 5.1, Two Attackers Trying to Compromise Sensor Nodes

Figure 5.1 shows two attackers which are trying to compromise sensor nodes at the WSN. For the detection in the second stage: In [118], Song et al. made the first attempt to detect compromise node in the second stage. They assume that an adversary will not be able to

precisely deploy the compromised sensors back into their original positions. Then, the detection of location change will become an indication of a potential node compromise.

For the detection in the third stage: In [117], Carl et al. demonstrate the case in which nodes can be compromised in the third stage and they show exactly what information can be obtained and how it can be used to disrupt, falsify data within, or eavesdrop on sensor networks. They suggest that sensor nodes in hostile environment would be desirable not to respond to the standard on-chip debugging and if a node can detect its own movement by either accelerometers or GPS then it can pre-emptively delete important information stored in SRAM, flash, or anywhere else on the system. Their work implies very high cost for large distributed network.

In [119], Kyasanur and Vaidya propose modifications to IEEE 802.11 MAC protocol to simplify misbehaviour detection. Once the sensor nodes are compromised, they will launch false data injection attack. Thus, several en-route filtering schemes [120, 121] have been proposed to drop the false data en-route before they reach the sink. Nevertheless, these schemes only mitigate the threats. Thus in [122], Ye et al. propose a probabilistic nested marking scheme to locate colluding compromised nodes in false data injection attacks. Recently, several software-based attestation schemes [123, 124] for node compromise detection in sensor networks have been proposed. However, they are not readily applied into regular sensor networks due to several limitations [84]. In [84], Yang et al. present two distributed schemes toward making software-based attestation more practical. In these schemes, neighbours of a suspicious node collaborate in the attestation process to make a joint decision.

5.3 Network Assumptions, Attack Model and Design Goals

In this section, we formulate the network assumptions, the attack model and identifying the design goal.

5.3.1 Network Assumptions

We consider the following assumptions in our network model:

1- The WSN is composed of a base station and a large number of sensor nodes uniformly deployed at a certain area. The base station is a trusted and powerful data collection device which is responsible for collecting the data sensed by sensor nodes. Each sensor node has a unique nonzero identifier and is stationary in a location.

2- The WSN forms overlapped groups. Each group is overlapped with other groups by one sensor node as shown in Figure 5.2.

3- The communication in the network between sensor nodes in the group is formed by a closed loop. Each two groups are overlapped in one sensor node. We assume each sensor node periodically collects the sensed data and reports them to the base station via a predefined routing.

4- Each sensor node can detect being connected by a programming board when the adversaries launch the physical node compromise attack.

5- We consider beacon nodes equipped with a GPS called beacons.

6- We assume sensor nodes are static and some nodes continuously store the detected security threats and all other security data related to nodes where these nodes are SMs. SMs have BKSMs to replace the SMs if they are compromised.

5.3.2 Attack Model

In the attack model, we assume that a group of attackers can capture a large number of sensor nodes at the same time in a local area, reprogram them with malicious code, and redeploy them back into the network using the physical node compromise attack. Specifically, the attackers have two physical attack policies: 1) directly physically attack the sensor node at the sensor node's original position; 2) firstly shut down some sensor nodes and launch physical attack at another place. Also, we assume that there are n sensor nodes in a local area, and the attackers can compromise k sensor nodes at the same time in the local area where k is from 2 to 5 sensor nodes at the same time.

5.3.3 Design Goals

The design goal is to develop an overlapped groups-based detection scheme to early detect node compromise attack. To achieve the design goal, we assume that nodes in the

cluster are connected in a group and each group shares other groups in one sensor node as shown in Figure 5.2. Therefore, when the attackers launch the physical node compromise attack against a group the other groups that share the attacked group will report this attack to the security manager then to the base station.

5.4 Overview of SurvSec Overlapped Groups Security Architecture

The proposed scheme has four phases which are key management phase to distribute keys among nodes, secure localization phase to determine nodes locations, secure clustering phase to choose BKSM to revoke SM if it is compromised, and forming overlapped groups phase for the overlapped groups based compromised nodes detection protocol at first stage. The proposed scheme has four types of sensors: SMs, BKSMs, initiators and sensor nodes.

In this section, we describe the overlapped groups based detection scheme in detail to early detect sensor node compromise attack. Specifically, we will address the node compromise problem in the first stage.

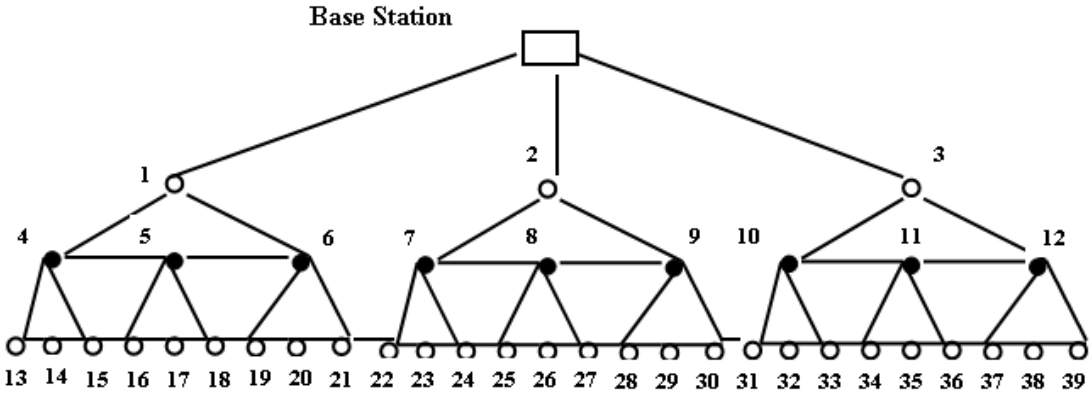


Figure 5.2, SurvSec Overlapped Groups-based Compromised Node Detection Protocol Network Setup for 39 Nodes

Figure 5.2, describes the network setup for the proposed protocol. Black nodes represent the overlapped nodes between the groups. Nodes 1, 4, 5 and 6 will form a group of four nodes. Nodes 2, 7, 8 and 9 will form a group of four nodes. Nodes 3, 10, 11 and 12 will form a group of four nodes. Also, nodes 4, 13, 14 and 15 will form a group of four

nodes. Also, nodes 5, 16, 17 and 18 will form a group of four nodes. Also, nodes 6, 19, 20 and 21 will form a group of four nodes. The group of nodes 1, 4, 5 and 6 is overlapped with three groups in three nodes. Node 15 is connected to node 16 to form overlapped groups at the last layer of groups. Also, node 18 is connected to node 19 to form overlapped groups at the last layer of groups and node 21 is connected to node 22 to form overlapped groups.

5.4.1 Key Management Phase

We propose a novel hybrid and dynamic key management protocol utilizing our novel scheme of certificates shared verification to eliminate the needs for High end Sensor Nodes (HSNs) which have high power for intensive calculation of public key operations. High end Sensor Nodes are the best targets for the attackers in the hostile environment. Our proposed key management scheme has two steps which are: key pre-distribution and key establishment.

The key pre-distribution and the key establishment are discussed in chapter 6. The key management algorithm using initiators is discussed in chapter 6 where it is named algorithm 2 of the key management. We assume that there are nodes named as initiators every predefined number of nodes such as 30, 20 or 10 nodes to start the operation of key management process. Initiator node verifies the certificates of the first two nodes then it sends the certificates of the second two nodes to the verified first two nodes then it sends certificates of the other four nodes to the verified four nodes. Algorithm 2 is efficient in terms of the distribution of power consumption among sensor nodes in the cluster and it can be used with all SMs in their clusters. The nodes under the initiator are ordinary nodes and the nodes under the ordinary nodes are SMs and so on until we reach another initiator.

5.4.2 Secure Localization Phase

A number of secure localization algorithms have been reported. Different researchers have different strategies to categorize them. These strategies can be divided into direct and indirect localization, centralized and distributed localization, range-based and range-free localization, absolute and relative localization. We propose to get the location information to form the group from the followings approach:

The indirect approaches of localization were introduced to overcome some of the drawbacks of the GPS-based direct localization techniques while retaining some of its advantages. In this approach, a small subset of nodes in the network, called the beacon nodes, are equipped with GPS receivers to compute their location. Beacon nodes send beams of signals providing their location to all nodes in their vicinity. Using the transmitted signal containing location information, nodes compute their location. Each node needs three beacon nodes to locate its position.

Our proposed scheme depends on the SM and certificates shared verification where SM shares other nodes the verification process of certificates for secure localization. We assume that each cluster has three beacon nodes. Sensor nodes in the cluster send the beacon nodes certificates to the SM then the SM sends these certificates to its underneath nodes for verification to ensure one verification time for beacon nodes certificates for the whole cluster. This is done because Verification power is 1000 times more than communication power [150]. The SM assures that certificate verification for beacon nodes is done only once for the whole cluster to reduce the power of verification. Each node needs to verify three beacon nodes with a total of 3n verifications but with certificate shared verification this is done once.

The secure localization algorithm is discussed in chapter 6 where it is named algorithm 3 of the key management.

5.4.3 Secure Clustering Phase

SMs can form secure clustering with their nodes underneath and the SM can choose BKSM to replace it if the SM is compromised.

Secure clustering is done to choose BKSM to replace SM if it is compromised. The secure clustering algorithm is discussed in chapter 6 where it is named algorithm 4 of the key management.

5.4.4 Forming Overlapped Groups Phase

Each node in a cluster sends its location to its SM. From the nodes locations at the SM, the SM starts the process to form a group. Assume each cluster has n nodes and the SM builds the overlapped group from the nodes in the cluster as shown in algorithm 5.

Algorithm 5 represents forming a group from n nodes which are labeled from n to n-(n-1). The SM sends a message to its nearest node containing the sequence of sending and receiving messages in the cluster to form a group according to each node neighbours. The SM chooses a group key for the cluster and sends it to all nodes in the cluster.

Algorithm 5: Forming Overlapped Groups

1: SM_n → n, n-(n-1) : { join group_msg }

The SM at layer n sends an encrypted "Hello" message to node n and node n-(n-1) to form the group and the message contains the interconnections of all nodes in the cluster to form the group. The used key is the group key between the SM and the nodes in the cluster. The sent message includes what every node is connected to in the cluster to form a closed loop.

2: n → n-1, SM_n : { join group_msg }

Node n sends an encrypted "Hello" message to node n-1 and SM to complete the process of forming a group. The message contains the interconnections of all nodes in the cluster. Used key is the group key between SM and nodes in the cluster plus one.

3: n-1 → n, n-2 : { join group_msg }

Node n-1 sends an encrypted "Hello" message to node n and node n-2 to complete the process of forming a group. The message contains the interconnections of all nodes in the cluster. The used key is the group key between the SM and nodes in the cluster plus two.

4: n-2 → n-1, n-3 : { join group_msg }

Node n-2 sends an encrypted "Hello" message to node n-1 and the node n-3 to complete the process of forming a group. The message contains the interconnections

of all nodes in the cluster. The used key is the group key between the SM and nodes in the cluster plus three.

5: n-3 → n-2, n-4 : { join group_msg }

Node n-3 sends an encrypted "Hello" message to node n-2 and the node n-4 to complete the process of forming a group. The message contains the interconnections of all nodes in the cluster. The used key is the group key between the SM and nodes in the cluster plus four.

6: n-4 → n-3, n-5 : { join group_msg }

Node n-4 sends an encrypted "Hello" message to node n-3 and the node n-5 to complete the process of forming a group. The message contains the interconnections of all nodes in the cluster. The used key is the group key between the SM and nodes in the cluster plus five.

7: n-5 → n-4, SM_n: { join group_msg }

If number of nodes in the group is 6, Node n-5 sends an encrypted "Hello" message to node n-4 and the SM to complete the process of forming a group. The message contains the interconnections of all nodes in the cluster. The used key is the group key between the SM and nodes in the cluster plus six. This message closes the loop.

Finally, the "Hello" message is sent from one node to two neighbour nodes in the cluster and the two nodes respond to the "Hello" message. If the node is compromised, it will not send the "Hello" message and therefore, the recipient nodes will mark it as compromised and they will send to the SM to revoke that node. If the SM is compromised, its monitored nodes will send to the BKSM to revoke the SM.

1- Our proposed compromised nodes detection scheme is based on the overlapped groups to discover the compromised group. If a node is compromised in a group, it will be detected by its neighbours which will send to the SM that this node is compromised. If a SM is compromised, its neighbour nodes will send to the BKSM to revoke it.
2- Each node sends at the first time with key K then next time with key K+n+1 and next time with key K+2n+1 and so on.
3- Each node sends a "Hello" message and receives two messages from its neighbours in 15 seconds.
4- Each group forms an overlapped group with its upper group and its lower group.

We designed the compromised nodes detection protocol at the first stage such that our network resembles a chain and each cluster in the network forms a group and each group is a ring in the chain and the rings are interconnected therefore, if one ring is compromised, its interconnected rings will discover this.

5.5 Security Analysis

Security analysis of our protocol focuses on resilience to node compromise attack, collusion attack and impersonation attack.

5.5.1 Compromised Node Attack

1- If an attacker compromises one regular node, therefore, the probability of insecure link is $P_{insec} = 1/N$ where N is the number of nodes at the network. For n compromised regular nodes the probability of insecure links is $P_{insec} = n/N$.
2- If the attacker compromises one SM, therefore, the probability of insecure links is $P_{insec} = (n_s + 3) /N$ where n_s is the number of nodes in the cluster of the SM. For n compromised SMs the probability of insecure links is $P_{insec} = n (n_s + 3) /N$.
3- Our proposed key management assumes compromised node detection at the first stage and compromised nodes revocation. Therefore, SM will revoke the regular compromised node and the BKSM will revoke the SM to eliminate the insecure links.

5.5.2 Collusion Attack

Two nodes can collude when they share their keys with each other. Our designed protocol is resistant to collusion attack because each sensor node communicates only with a SM therefore; compromised nodes cannot discover themselves.

5.5.3 Impersonation Attack

Each node has a certificate to join the key management process and to join the network. This prevents the attacker from impersonating any legitimate node. Also, knowing the public key of the SM will not reveal the private key for the SM because this needs the attacker to solve the elliptic curve discrete logarithmic problem ECDLP which is a hard problem.

5.6 Performance Analysis

The performance analysis is measured in computation complexity, communication complexity, storage complexity and setup time. We assume that the network is secure during setup time which depends on number of initiators.

5.6.1 Computation Complexity

The SM generates a group key and sends it encrypted with the shared link key with every node in the cluster to use it in the process of compromised nodes detection. Each sensor node decrypts the message sent with the group key with its shared link key with the SM.

Our scheme has lower computation overhead than the scheme that uses couples to detect compromised nodes at the first stage as this scheme uses public key to decrypt the messages. Our scheme has the same computation overhead compared to the scheme which uses distributed compromised nodes detection at the first stage. Our scheme has low computation overhead to generate the group key and to send it encrypted to all the nodes in the group.

5.6.2 Communication Complexity

Communication complexity is the number and size of packets sent and received by a sensor node. In our protocol, the number of messages sent is one message every 15 seconds and there are two messages received every 15 seconds with a total of three messages sent and received every 15 seconds to establish the compromised nodes detection protocol. Our scheme has lower communication overhead than the other two schemes that detects compromised nodes at the first stage.

5.6.3 Storage Complexity

Storage complexity is the amount of memory units required to store security credentials. Each sensor node stores the group key with the SM and other nodes in the cluster. Our scheme has the same storage overhead as the scheme which uses couples to detect compromised nodes at the first stage but it has lower storage overhead than the scheme which uses distributed compromised nodes detection at first stage. Our scheme stores only one key which is the group key between the SM and the nodes in the group.

5.6.4 Setup Time

Our scheme has a low setup time to achieve the compromised nodes detection at the first stage. The setup time for the network includes the key management time, secure localization, secure clustering and compromised nodes detection at the first stage. For initiators every 10 nodes setup time is 1 min for the whole network.

5.7 Simulation Results

5.7.1 Simulation Environment

We built a model for the proposed design and we implemented a simulator in MATLAB that can scale to thousands of nodes. In this simulator, sensors can send and receive data from each other's. The simulation verifies the correctness and the feasibility of our security architecture. Our simulation scenarios include *n* sensor nodes distributed randomly. We choose *n* as 1000 sensor nodes.

The followings are the built models for simulation:

i. Network setup model for the overlapped groups.

ii. Attackers' model.

iii. Compromised nodes detection protocol.

In the simulations, these parameters are given as follows:

a. The number of sensor nodes *n* is varied from 39 to 1000 sensor nodes.

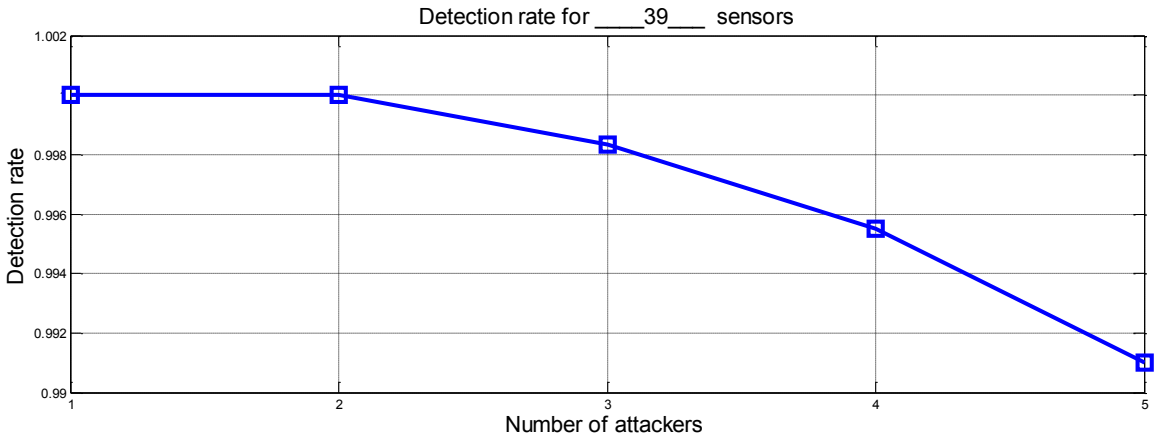
b. The interval of beacon information is set to 15 seconds.

c. The time of an adversary to successfully compromise a sensor node is varied from 30 seconds to 60 seconds.

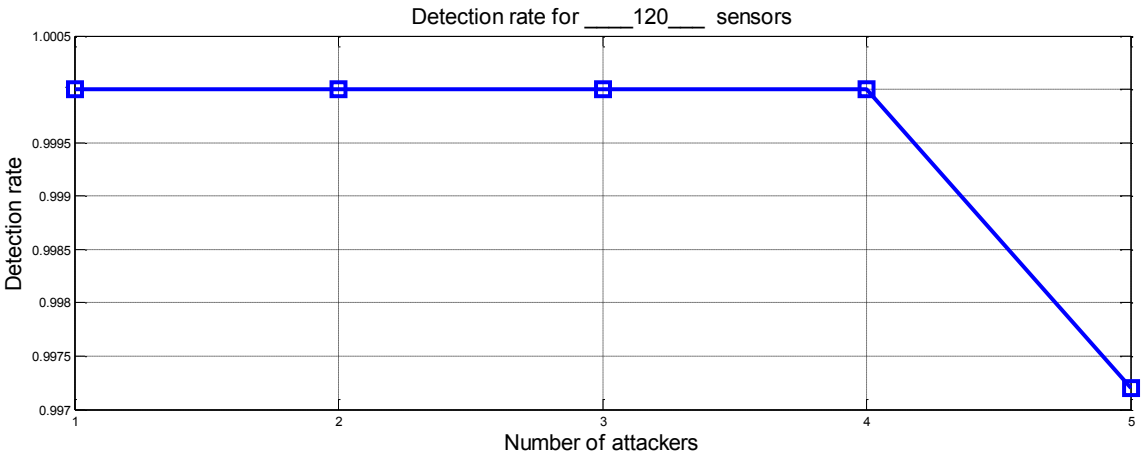
We test the network with different parameters settings. For each case 1000 networks are randomly generated.

5.7.2 Simulation Results

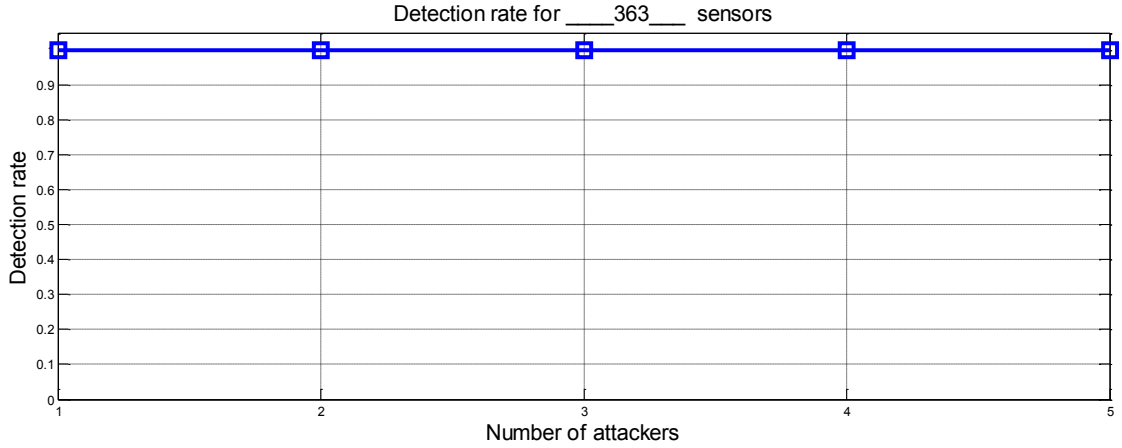
In this section, we evaluate the detection rate under different *n*. The detection rate is defined as the number of detected compromised sensor nodes over all compromised sensor nodes. In Figure 5.3 and 5.4, the parameter *k* is the number of compromised sensor nodes in the network and the parameter α is the percentage of sleep nodes. The x-axis is the number of attackers and the y-axis is the detection rate. We use small number of points at y-axis so that the detection rate will be accurately determined otherwise it is determined as 100% which is not true.



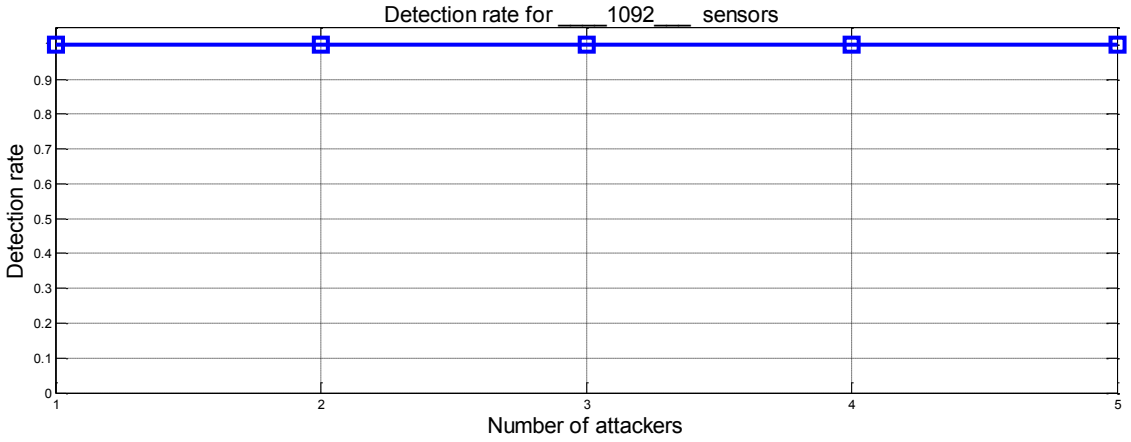
(a) $n = 39$, $k = 5$



(b) $n = 120$, $k = 10$

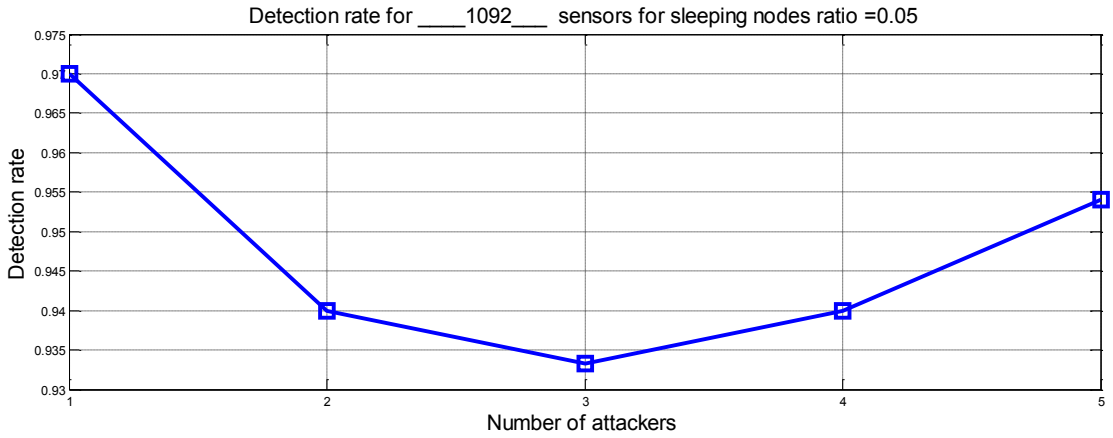


(c) $n = 363$, $k = 15$

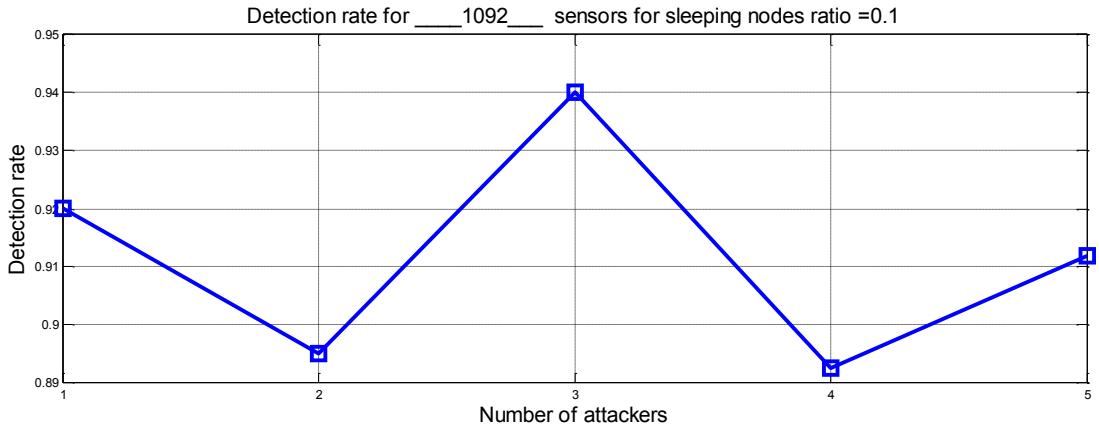


(d) $n = 1092$, $k = 25$

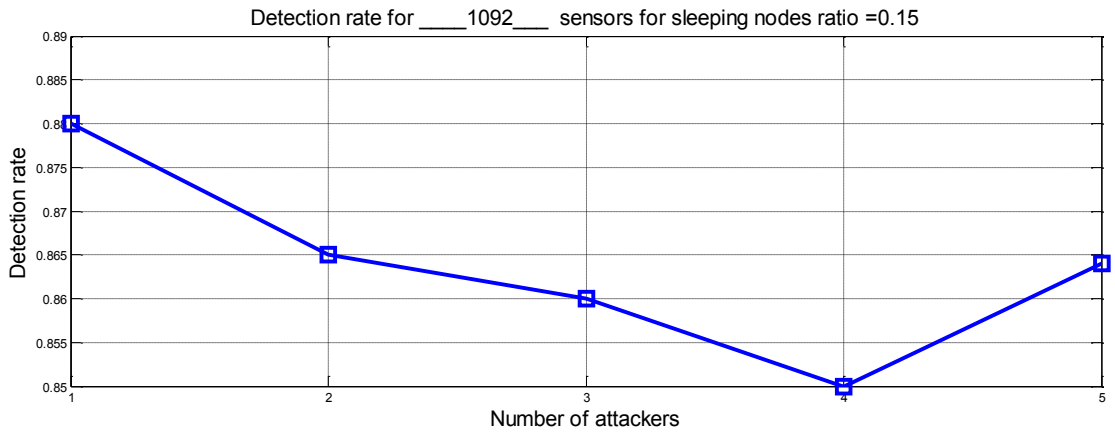
Figure 5.3, Detection Rate Varies with Number of Compromised Nodes under Different n =39, 120, 363, 1092, Interval = 15 Sec.



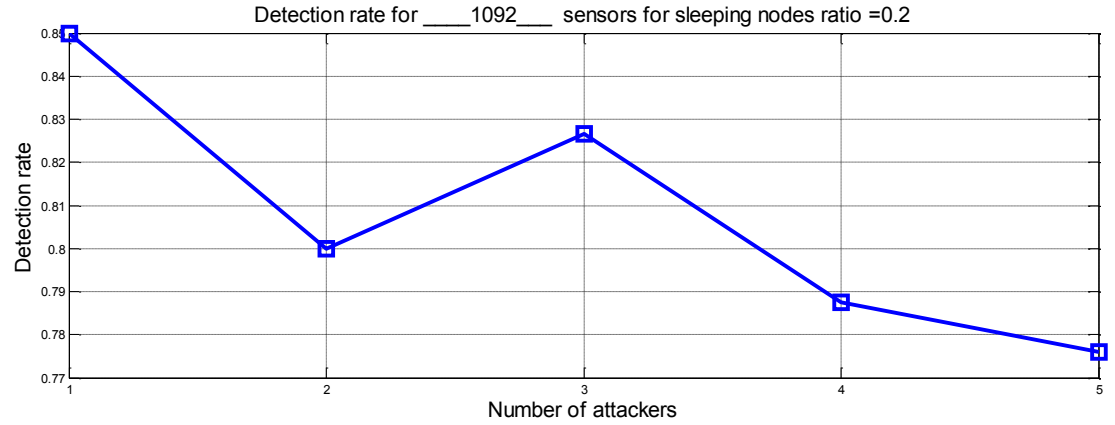
(a) α = 0.05, k = 25



(b) α = 0.10, k = 25



(c) α = 0.15, k = 25



(d) α = 0.20, k = 25

Figure 5.4, Detection Rate Varies with n Under Different α = 0.05, 0.10, 0.15, 0.20, Interval = 15 Sec.

In the proposed adversary model, we assume that an adversary can simultaneously compromise k sensor nodes, where $k<n$.

Thus, we first evaluate the detection rate under different parameters n, k and beacon interval and the results are shown in Figure 5.3. From Figure 5.3, we can see the detection rate does not increase linearly with k. When n = 363 or n = 1092, the detection rate reaches the maximum. Due to this observation, when the number of sensor nodes

increase, we found that the proposed scheme has high resiliency against node compromise attack by collaborative work of attackers at the same time for large hierarchical WSN.

In the above simulation, similar to most previously reported work, we only consider that all sensor nodes are always active. However, in reality, in order to extend the network longevity, sensor nodes need to periodically enter into the sleep mode. In the sleep mode, a sensor node does not send or receive any messages from others. This will result in most power saving to the network. However, the sleep mode provides the attackers with the best chance to compromise many legitimate sleeping sensor nodes while these nodes are not detected as compromised nodes. Assume that all n sensor nodes in a local area follow the same active/sleep schedule and sensor nodes within the groups are scheduling synchronization. For α the percentage of sleep nodes. At the same time, in each period, only α percent sensor nodes are in a sleep mode. With these settings, we run the above simulations again, where α has different values, interval 15 sec and number of sensor nodes n = 1092.

Figure 5.4 shows the detection rate in terms of different parameter α. From the figure, we can see that when α increases, the detection rate will decrease. Thus, this is another trade-off between the detection rate and the network longevity when we choose the proper active/sleep schedule.

5.8 Comparison with Others Works

Now, we compare between our proposed model and previous works that detects compromised nodes at the first stage.

Table 5.1, Comparison between Our Model and Other Models.

	Property	CAT [125]	Distributed Detection [126]	Our Model
1	Detect compromised nodes for group of attackers	No	No	Yes
2	Detection rate	Less than 100%	Near 100%	Near 100%
3	Communication overhead	14 messages every 15 sec for beacon every 2 sec High overhead	At least 6 messages every 15 sec moderate overhead	At least 3 messages every 15 sec low overhead
4	Computation overhead	Low	Low	Low
5	Storage overhead	Low to store one key	High to store key list	Low to store one key
6	Setup time	Low	Low	Lower
7	Power cost	High	High	Low
8	Detection algorithm vulnerabilities if a node is compromised	No	Key list is vulnerable and must be revoked	No

Our proposed model can be used against collaborative work of attackers to compromise a large number of nodes at the same time. Also, the detection rate is near 100%. Our model has low communication overhead, low computation overhead, low storage overhead and low setup time. Our model has low power cost since it sends and receives only three messages every 15 sec which is lower than the other two schemes.

5.9 Summary

In this chapter, we proposed the overlapped groups-based compromised nodes detection scheme to early detect the node compromise attack in the first stage. Concretely, the simulation results showed that by building groups among neighboring sensor nodes in a local area, a physical node compromise attack can be detected immediately. Also, the

simulation results showed that the proposed detection scheme has a high detection rate. This work is an initial work to form overlapped groups for detecting compromise attack at the first stage and we do not expect that the proposed scheme will solve all the problems in the node compromise nodes attack. Our future work will continue to build more overlapped groups to early detect the compromise nodes attack.

We cannot detect compromised nodes during groups' formation and this is a limitation for the proposed system.

We cannot detect compromised nodes that are being compromised wirelessly and this is a limitation to the proposed system.

The third component of SurvSec security architecture is a new compromised nodes detection algorithm at the first stage against collaborative work of group of attackers compromising sensor nodes at the same time.

In this chapter we discussed in details the compromised nodes detection algorithm and evaluated its performance.

CHAPTER 6

SURVSEC HYBRID AND DYNAMIC KEY MANAGEMENT SCHEME

In this chapter, we describe our proposed certificates shared verification key management with hybrid and dynamic key management scheme for SurvSec security architecture. The key management scheme is hybrid to gather the advantages of both symmetric key-based key management and asymmetric key-based key management and to overcome both disadvantages. Also, the key management is dynamic to provide the system with key revocation, rekeying and addition of new sensor nodes. Our designed SurvSec key management is suitable for the hostile environment.

6.1 Introduction

This chapter is organized as follows: Section 6.2 presents the related work. Section 6.3 describes the network assumptions and threat model. Section 6.4 describes the proposed hybrid and dynamic key management scheme along with certificates shared verification process. Section 6.5 presents security analysis of the proposed scheme. Section 6.6 presents the performance analysis. Section 6.7 presents simulation results. Section 6.8 presents the security proof for the proposed model. Section 6.9 presents the comparison with previous works. Finally, Section 6.10 is the summary of the chapter.

Key management is the fundamental security mechanism in WSN which is needed for secure localization, secure clustering, secure data aggregation, secure authenticated broadcasting and secure routing. In this chapter, a novel hybrid and dynamic key management scheme was proposed. This new scheme established secret keys between sensor nodes for SurvSec security architecture with high security level, high performance

and low setup time. Hybrid key management provides high security level in the hostile environment however previous work assumed heterogeneous network utilizes High end Sensor Nodes (HSNs) with high power for high computations of certificates verification. This assumption provides attackers the best chance to destroy the network by targeting the HSNs. Also, HSN is connected to a large number of nodes and there is no backup node for it. In addition, if the attackers target HSNs, then the connectivity and scalability will be affected where these nodes are points of failure. Moreover, previous work did not explain how to revoke compromised HSN. Furthermore, increasing the number of HSNs will increase the network deployment cost. Finally, if HSN is destroyed, nodes cannot have rekeying or addition of new nodes or revocation of compromised nodes. This chapter proposed a hybrid scheme with homogenous network that uses some sensor nodes named as security managers (SMs) with a proposed novel mechanism called certificates shared verification to verify the certificates of group of nodes with distributed computations to overcome the absence of HSNs. This chapter presents an analytical evaluation and extensive simulation. The simulation results showed that at the cost of increasing communication overhead, the certificates shared verification mechanism was developed. Also, simulation results showed that the proposed scheme has lower computation overhead at the SM side and lower setup time than HSN model. Both schemes have the same storage overhead. Location based key management protocols are very efficient methods in terms of key connectivity, storage overhead, improving the security and scalability and localizing attacks. Also, dynamic key management assumes long lived networks with more frequent addition of new nodes thus requiring network rekeying for sustained security and survivability.

In this chapter, we proposed a new hybrid and dynamic key management scheme that establishes secret keys between sensor nodes. The hybrid scheme reduces the high cost public key operations at the sensor side and replaces them with efficient symmetric key based operations.

Hybrid key management combines the advantages of symmetric key and public key and it is the best solution for the hostile environment. Previous researches on hybrid key management have suggested using a heterogeneous network with HSNs and low end

sensor nodes (LSNs), where HSNs are used to perform high power calculations such as certificate verification, exponentiation, elliptic curve scalar multiplications and additions and modular multiplications.

HSNs are the best targets for the attackers to destroy the network where HSN is connected to a large number of nodes. Also, HSN verifies certificates one by one within its connected nodes, which takes long time. Our scheme uses security managers to process a certificate shared verification process in distributed manner, with less time for the same number of nodes, as shown in section 6.6. Destroying HSN in the middle of a branch results in cut communications in the branch. Also, each node underneath HSN needs three certificates verification for beacon nodes which has a high cost for large number of nodes while our scheme assumes beacon nodes certificates verification once for the whole cluster. Finally, if HSN is destroyed, nodes cannot have rekeying or addition of new nodes or revocation of compromised nodes.

The proposed key management scheme has four types of nodes, which are security manager (SM), backup security manager (BKSM), initiator node and sensor nodes. The key management scheme assumes seven phases which are: key pre-distribution, key establishment, secure localization, secure clustering, rekeying, keys revocation and addition of new nodes. The protocol has four algorithms. The first algorithm is used for certificates verification and keys distribution. The second algorithm is used for initiator nodes to initiate the key management process. The third algorithm is used for secure localization. The fourth algorithm is used for secure clustering. Sensor nodes near the BS are the first layer SMs. SMs are located every two layers. First, SMs near the BS verify the certificate of the BS and the BS verifies the certificates of the first layer SMs then they share a symmetric link keys. Second, first layer SMs determine their locations from their neighbour beacon nodes after receiving the neighbour beacon nodes certificates and then send them to BS for verification. Third, SMs broadcast their certificates to their neighbour nodes underneath and these nodes verify the certificate of the SMs. Fourth, neighbour nodes underneath SMs broadcast their certificates to SMs which in turn send these certificates to BS for verification then SMs and neighbour nodes underneath share a symmetric keys. Fifth, neighbour nodes underneath SMs determine

their locations from their neighbour beacon nodes after receiving the neighbour beacon nodes certificates and then send them to SMs then to the BS for verification. Sixth, SMs and their neighbour nodes underneath form secure clustering then SMs select BKSMs according to maximum connectivity between BKSM and sensor nodes in the cluster. Finally, lower layer SMs send the certificates of their neighbour nodes underneath and beacon nodes to higher layer SMs for verification.

Our scheme proposes to deploy an initiator node every predefined number of nodes to start the process of key management in a distributed manner and to finish it in controlled efficient time where these nodes are SMs. These nodes collect the certificates of their underneath nodes for verification and execute our proposed second algorithm. Finally, every initiator node communicates with its higher layer node and its upper layer SM.

In this chapter, we proposed a novel idea of certificates shared verification to avoid using HSN and our scheme has a BKSM for every cluster to replace the SM if it is compromised.

The proposed scheme provided secure clustering algorithm to choose backup security managers (BKSMs). In addition, the proposed scheme can revoke the compromised SM by the BKSM. Moreover, the BKSM will maintain the network scalability and connectivity if the SM is compromised. Furthermore, the proposed scheme provides secure localization algorithm with certificates shared verification to lower computation overheads and to verify certificates of beacon nodes once for the whole cluster. The proposed dynamic key management uses certificates shared verification to reduce computations overheads and setup time for rekeying and addition of new nodes. The proposed scheme uses initiator nodes every predefined number of nodes to start the key management process for its underneath nodes to overcome absence of HSN. The proposed scheme can distribute link keys in less time than the HSN model.

Motivated by insufficient hardware resources, a great deal of research has focused on the symmetric cryptography-based solutions [129-134, 37, 191-200] for light-weight computation. These symmetric-key schemes, however, require complicated key management that may cause large memory and communication overhead. This drawback

has not yet been investigated by experimental work. Recent progress in implementation of elliptic curve cryptograph on sensors [35, 50, 135] proves public key cryptography is now feasible for resource constrained sensors.

In this chapter, we proposed a new hybrid and dynamic key management that uses a hybrid key management scheme in order to establish secret keys between sensor nodes and the scheme is based on the nodes location.

The published symmetric key-based key management protocols and public key based key management protocols are vulnerable to sybil attack and cloning attack. In these attacks, the attacker can steal the identity of the sensor then launch impersonation attack to use it elsewhere in the network. Also, the attacker can copy the certificate of the node beside the public and private keys for cloning attack to join the network with legitimate credentials.

The proposed scheme is a hybrid key management scheme to incorporate the advantages of both the symmetric and asymmetric key management schemes. Also, the proposed scheme is a dynamic key management which will provide the network with rekeying, revocation of compromised sensor nodes and addition of new nodes.

The **contributions** of the proposed design can be summarized as:

1- We designed a homogenous network that utilizes SMs, BKSMs and initiators to implement the distributed security concept instead of using HSNs which is the best target for the attackers.

2- We designed the certificates shared verification mechanism to distribute the high power computations of certificates verification among sensor nodes in the cluster.

3- We designed an integrated key management scheme that combines hybrid key management; and dynamic key management to resist attacks in the hostile environment.

4- We designed a secure localization algorithm that employs the certificates shared verification scheme with low computation overhead through verifying beacon nodes certificates only once for the cluster where previous scheme assumes that each sensor node verifies certificates of three beacon nodes.

5- We designed a secure clustering algorithm that chooses a BKSM to replace and revoke the SM if it is compromised. Also, the BKSM will maintain high connectivity and high scalability if the SM is compromised.
6- We designed the network with low setup time, and low cost compared to network with HSNs. The computation overhead at SM is lower than that at HSN.
7- We designed our key management to be dynamic to provide rekeying, revocation of compromised sensor nodes and addition of new nodes using certificates shared verification.

6.2 Related Work

In this section, we present the related work to our proposed scheme where we will employ our designed hybrid and dynamic key management for SurvSec security architecture.

6.2.1 Static versus Dynamic Key Management

The success of a key management scheme is determined in part by its ability to efficiently survive attacks on highly vulnerable and resource challenged sensor networks. Key management schemes in sensor networks can be classified broadly into dynamic or static solutions based on whether rekeying (update) of administrative keys is enabled post network deployment.

6.2.1.1 Static Key Management Scheme

These schemes assume that once administrative keys are predeployed in the nodes, they will not be changed. Administrative keys are generated prior to deployment, assigned to nodes either randomly or based on some deployment information, and then distributed to nodes. Most static schemes use the overlapping of administrative keys to determine the eligibility of neighbouring nodes to generate a direct pair-wise communication key.

The basic key predistribution scheme was first proposed by Eschenauer and Gligor [25]. It assumes homogeneous nodes that are loaded with keying material and perform the same key management functions. In this scheme k keys are randomly selected by each node out of a large pool of P keys. A major advantage of such scheme is the exclusion of the base station in key management. Another advantage is incurring no post-deployment

communication overhead on sensor nodes. However, successive node captures enable the attacker to reveal keys stored in captured nodes and use them to attack other nodes.

An enhancement of the basic scheme was proposed in [130], in which two nodes can establish a link only if they share q keys. Liu and Ning [136, 137] provided further enhancements by using t-degree bivariate key polynomials. Instead of selecting k keys out of a pool of P simple keys for each node as in the basic Eschenauer and Gligor scheme [25], a key server first randomly generates a pool of P bivariate t-degree polynomials, each of which is uniquely identified by a polynomial ID. The server then chooses a random subset of polynomials and distributes the polynomial shares and polynomial IDs to the sensor nodes. Two nodes can directly communicate only if they can identify at least one polynomial in common by exchanging their polynomial IDs, and using the polynomial-based scheme to compute the pair-wise communication key.

In [137], the authors assume that nodes are deployed in groups; each group might represent a deployment event to a certain location in the deployment field. Individual nodes are assumed to be aware of their group prior to deployment.

6.2.1.2 Dynamic Key Management Scheme

Basically, dynamic key management schemes change administrative keys periodically, or on demand or on detection of node capture. The major advantage of dynamic keying is enhanced network survivability, since any captured key(s) is replaced in a timely manner in a process known as rekeying.

Another advantage of dynamic keying is that it provides better support for network expansion; upon adding new nodes, unlike static keying, which uses a fixed pool of keys, the probability of network capture does not necessarily increase. Both homogeneous and heterogeneous dynamic key management schemes have been proposed in the literature.

The major challenge in dynamic keying is to design a secure yet efficient rekeying mechanism. A proposed solution to this problem is using exclusion-based systems (EBSs); a combinatorial formulation of the group key management problem developed in [138].

Rekeying takes place either periodically or when one or more nodes are captured (or suspected of being captured). A drawback of the basic EBS-based solution is that a small number of nodes may collude and collectively reveal all the network keys.

The application of EBS was first proposed for key management in sensor networks in [139]. In this scheme, nodes were assumed to be anonymous (with no preloaded node ID). The sensor network establishes a coordinate system (or virtual infrastructure) around the base station.

An example of non-EBS dynamic keying schemes is due to Jolly et al. [140] who proposed a key management scheme based on identity-based symmetric keying. The network model involves a base station and several clusters of sensor nodes, each led by a (better equipped) cluster gateway. Rekeying involves the re-establishment of clusters and redistribution of keys.

Although the storage requirement is very affordable, the rekeying procedure is inefficient due to the large number of messages exchanged for key renewals. In addition, they require a centralized key server to play a major role in key management. Since the network model involves three types of nodes: sensor nodes, cluster gateways, and base station with different keying functionalities, this scheme is classified as heterogeneous where no node location or other deployment information is used in key assignment.

In order to address the collusion problem in EBS, Younis et al. proposed SHELL [141]; an EBS-based scheme that performs location-based key assignment to minimize the number of keys revealed by capturing collusion nodes.

6.2.2 Key Management based on Encryption Key

6.2.2.1 Symmetric key-based Key Management Scheme

Symmetric-key based schemes are widely used because these schemes consume less computation time and power than other schemes, which are suitable for the limited resource characteristics of the wireless sensor network. However, the shortages of the symmetric key schemes are also obvious. Different schemes may have different weakness such as security strength (resilience), scalability and connection probability (connectivity). Based on the key distribution, key discovery and key establishment in the schemes, we can divided these schemes into eight categories: entity based key

management schemes [29], pairwise key pre-distribution schemes [130], pure probabilistic-based schemes [28], polynomial-based key pre-distribution schemes [131], matrix- based key pre-distribution schemes [132], tree-based key pre-distribution schemes [133], combinatorial design-based key pre-distribution schemes [134] and exclusion basis systems EBS-based key pre-distribution schemes [40, 138].

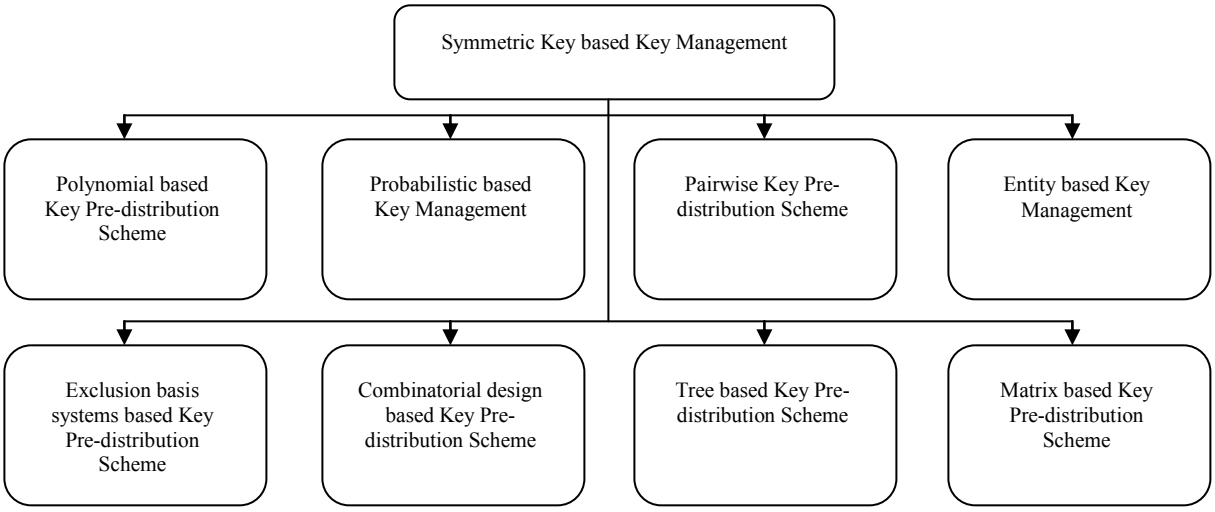


Figure 6.1, Symmetric Key-based Key Management Schemes Categories

Figure 6.1 shows the symmetric key-based key management schemes categories.

The symmetric-key based key management schemes, however, require complicated key management that may result in large memory and communication overhead. Also, symmetric key-based key management schemes are susceptible to man-in-the middle attack, collusion attack, cloning attack and sybil attack as described in chapter 2.

6.2.2.2 Asymmetric key-based Key Management Scheme

The public key-based key management schemes have many advantages such as low communications overhead, low storage overhead, high scalability. It can provide simpler solution with much stronger security strength. Public key solutions were thought to be computationally expensive for wireless sensor network. However, some researchers [142] show that public key schemes are viable on sensor node.

Public key-based schemes have been categorized into three types: RSA-based asymmetric encryption system, ECC-based asymmetric encryption system and ID-based

key agreement schemes. In general, public key schemes have better security strength, scalability and connectivity but it has high computation overhead.

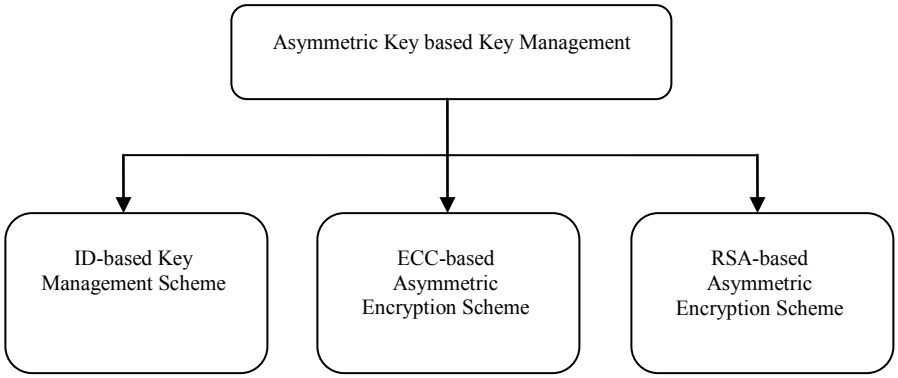


Figure 6.2, Asymmetric Key-based Key Management Schemes Categories

Figure 6.2 shows the asymmetric key-based key management schemes categories.

RSA and elliptic curve cryptography (ECC) are two major public key techniques. Public key technology is widely used in the realm of Internet. On the other hand, some researchers believe that these techniques are too heavy weight for sensor network because of its constrains. However, several research groups (Gura et al. [38]; Watro et al. [143]; Karlofand et al. [47]; Gaubatz et al. [144]) have successfully implemented public-key cryptography in wireless sensor networks. Gura et al. [38] compared the ECC and RSA on small devices. They show that both RSA and elliptic curve cryptography are viable on 8-bit CPU. The relative performance advantage of ECC point multiplication over RSA modular exponentiation increases with the decrease in processor word size and the increase in key size. They also demonstrate that ECC-160 point multiplication outperforms the RSA-1024 private-key operation by an order of magnitude and is within a factor of 2 of the RSA-1024 public-key operation. The asymmetric key-based key management protocols require higher computations than symmetric key-based key management schemes. Also, asymmetric key-based key management schemes are susceptible to cloning attack and sybil attack as described in chapter 2.

6.2.2.3 Hybrid Key Management schemes

Several research groups (Huang et al. [135, 202]; Zhang and Varadharajan [145]) proposed the hybrid key establishment schemes for wireless sensor networks. The motivation is to exploit the difference among the base station, the cluster heads and the sensors, and place the cryptographic burden on the base station or the sensors where the resources are less constrained. Sensors are more computational power and energy resources limited. On the other hand, the base station has much more computational power and other resources. The hybrid key establishment schemes reduce the high computational cost on the sensors by placing them on the base station side. Huang et al. [135] proposed a hybrid authenticated key establishment scheme, which is based on a combination of elliptic curve cryptography (ECC) and symmetric-key operations. The hybrid key establishment protocol reduces the high cost elliptic curve random point scalar multiplications at the sensor side and replaces them with low cost and efficient symmetric-key based operations. Moreover, it authenticates the two identities based on elliptic curve implicit certificates to avoid the typical key management problem in pure symmetric-key based protocols.

Hybrid schemes are suitable for the larger hierarchical wireless sensor network. Hybrid schemes may have advantages of both asymmetric key schemes and symmetric key schemes for larger sensor network. The public key-based key management schemes will make strong security and will become a reality with more research work in the future. The ongoing direction is how to secure the wireless sensor network by combining the cryptographic techniques to provide the best solution for different environment.

6.2.3 Key Management based on Location

Liu et al. propose in [146] LBKs (location-based keys) that relies on location information to achieve key management. The keys are established according to the geographical location of sensor nodes. However, knowing the geographical location of nodes is not guaranteed with random deployment.

Recently researchers have suggested utilizing the location of sensor nodes [39, 46, 147- 149] after node deployment to improve the security and scalability of key management schemes. Location based key management protocols are very efficient

methods in terms of key connectivity and storage overhead. Location-aware key management is resilient against node capture attacks in large-scale sensor networks.

6.3 Network Assumptions and Threat Model

6.3.1 Network Model

We consider a wireless sensor network consisting of a base station, many cluster heads, numerous sensor nodes which are grouped into clusters, beacon sensors equipped with GPS called beacons, and each node has a unique ID. Each node has a unique location. Each cluster is controlled by a cluster head, which can broadcast messages to all sensors in the cluster. The network architecture is depicted in Figure 4.2.

The assumptions of this model are as follows:

1- We assume that sensors are static, so once they are deployed they do not leave their locations.

2- Some nodes continuously store the detected security threats and all other security data related to sensor nodes where these nodes are named security managers. The security managers store the nodes' ID and locations underneath.

3- We assume that the goal of the adversary is to uncover the keys used in the system in order to compromise the network.

4- We assume that our key management scheme is supported by a secure routing protocol such as SAODV [152] which runs with the key management process.

6.3.2 Threat Model

In this chapter, we mainly consider an adversary that tries to uncover the keys of the network and manipulate the system through capturing and compromising some network nodes. No trust assumptions are made on the sensors. When sensors are captured; their memory can be read and erased or tampered with. The cluster heads are not assumed to be tamper proof either. Cluster heads compromise attack includes the uncovering of its keys through collude. Also, the attacker can launch collusion attack.

6.4 Proposed Scheme

The proposed scheme has seven phases which are key pre-distribution phase, key establishment phase, secure localization phase, secure clustering phase, key revocation phase, rekeying phase and add new node phase. The proposed scheme has four types of sensors: SMs, BKSMs initiators and sensor nodes.

6.4.1 Key Pre-distribution Phase:

The key pre-distribution phase consists of acquiring the sensors certificate from the certificate authority CA. ECC is used in this protocol to perform security functions on sensors with limited computing resources. The protocol uses the elliptic curve explicit certificate scheme instead of X.509 because of the resulting low storage overhead, low communication overhead, which is a dominant factor for low bit transmission channels in WSN.

The certificate generation processes for any sensor node U is performed offline before it joins the network. The steps for keys predistribution are the followings:

1- An elliptic curve E defined over $GF(p)$ where p is the characteristic of the base field with suitable coefficients and a base point P of large order n is selected and made public to all users.

2- CA selects a random integer q_{CA} as its static private key, and computes the public key $Q_{CA} = q_{CA} . P$, Where . is point multiplication.

3- To obtain a certificate and private-public key pair, the sensor U randomly selects a key pair (q_U, Q_U) where $Q_U = q_U . P$ and sends Q_U and q_U to CA.

4- CA verifies U's identity and private-public key pair.

5- The explicit certificate for U is the concatenation of CA's public key Q_{CA}, the device identity ID_U, the U public key Q_U and the certification expiration date t_U , i.e., the certificate is (Q_{CA}, ID_U, Q_U, t_U) signed by the CA private key using the Elliptic Curve Digital Signature Algorithm ECDSA where the signature is discussed in section 6.8.

6.4.2 Key Establishment Phase:

Certificates Verification & Keys Distribution

Power of the signature verification for ECDSA is 1000 times more than the power of the signature transmission [150]. Each node in HSN model performs certificate verification four times for three beacon nodes and for HSN certificate. With the same number of certificates verification at each node, we developed our proposed certificates shared verification scheme. Each node in our scheme verifies four certificates only with the cost of increasing the communication overhead with four messages for every node. These verifications are: first verification for SM certificate, two verifications for two nodes underneath that node, and one verification for beacon node certificate. We assume that there are nodes named as security managers (SMs) and these nodes are located every two layers. We assume that there are nodes named as initiators every predefined number of nodes such as 30, 20 or 10 nodes to start the operation of key management process.
We explain our scheme in the form of two algorithms.

Algorithm 1: Certificates Verification and Keys Distribution

1: BS → n : {BS (Q_{CA}, ID_{BS}, Q_{BS}, t_{BS}) }

BS broadcasts its certificate to nodes near BS at layer n and nodes verify certificate of BS. These nodes are SMs. Verification uses ECDSA as discussed in section 6.8.

2: n → BS : {n (Q_{CA}, ID_U, Q_U, t_U) }

The nodes near the BS at layer n send their certificates to the BS and the BS verifies the certificates of these nodes. The verification uses ECDSA.

3: n : selects (k), calculates (d_U), encrypts (d_U)

Each node near BS at layer n selects a k-bit random number c_U of 160 bits to produce its link key contribution with the BS.

Each node at n calculates the value of $d_U = H(c_U \| ID_U)$ where H is a cryptographic

---

hash function. Each node at n encrypts d_U with BS public key Q_{BS}. To encrypt and send a message d_U to BS, d_U must be encoded to a point on the elliptic curve which has x and y to be Pd_U. Each node at n chooses a random positive integer x and produces the ciphertext C_m consisting of the pair of points which are:

$C_m = (x\,P, Pd_U + x\,Q_{BS})$.

4: n → BS : { C_m }

Each node near BS at layer n sends its encrypted link key contribution with the BS which is C_m.

5: BS : decrypts (C_m), selects (k), calculate (d_{BS}), encrypts (d_{BS})

BS decrypts C_m for every node at n. BS multiplies first point in the pair by BS's private key and subtracts result from second point:

$Pd_U + x\,Q_{BS} - q_{BS}\,(x\,P) = Pd_U + x\,(q_{BS}\,P) - q_{BS}\,(x\,P) = Pd_U$.

BS selects a k-bit random number c_{BS} of 160 bits for each node near BS to produce its link key contribution with nodes near BS.

BS calculates the value of $d_{BS} = H(c_{BS} \,||\, ID_{BS})$ for every node near BS where H is a cryptographic hash function.

BS encrypts d_{BS} for every node near BS using symmetric key encryption under key d_U, generating value $y = E_{du}\,(\,ID_{BS} || \,d_{BS})$ where d_U is the x value of Pd_U.

6: BS → n : { y }, {E(K || Nonce)$_K$ encryption of K with nonce}

BS sends y, the encrypted link key contribution of BS, to every node near BS. BS generates the link key with every node near the BS at n by calculating K where K = $d_u \,||\, ID_U \,||\, d_{BS} \,||\, ID_{BS}$ then BS generates the encryption of K along with nonce using

key K which is H to prevent active attacks such as replay attack. BS sends H of every node at n to its participant to achieve correctness.

7: n : decrypts (y), calculates (K)

Every node at n decrypts the received message y using symmetric key encryption under key d_U to obtain the value d_{BS}. Every node at n generates the link key with BS by calculating $K = d_u \,||\, ID_U \,||\, d_{BS} \,||\, ID_{BS}$.

8: n → BS : {z}, {E(K || Nonce)$_K$ encryption of K with nonce}

Every node at n calculates z = encryption of key with nonce using key K and sends z to BS. BS checks if z = encryption of key with nonce using key K to prevent any adversary from applying active attacks such as replay attack. If yes, the link key is established correctly. Otherwise, the protocol is terminated.

9: n → n-1 : {n (Q_{CA}, ID_{SM}, Q_{SM}, t_{SM}) }

Each SM at layer n broadcasts its certificate to nodes at layer n-1 and nodes at n-1 verify the certificate of its SM. Each node at layer n-1 verifies SM certificate.

10: n-1 → n : {n-1 (Q_{CA}, ID_U, Q_U, t_U) }

Each node at layer n-1 sends its certificate to its SM at layer n

10: n → BS : all certificates {n-1 (Q_{CA}, ID_U, Q_U, t_U) }

Every SM at layer n sends the certificates of its nodes at layer n-1 to BS for verification because SM will lose high power and consume large time for verifying certificates of at least four nodes connected to it.

11: BS → n : {valid certificates or invalid certificates}

BS sends to each SM an encrypted message indicating that its certificates from layer

n-1 are valid or not. Then SMs at layer n executes steps from 3 to 8 to share symmetric link keys with nodes at layer n-1.

12: n-1 → n-2 : {n-1 (Q_{CA}, ID_U, Q_U, t_V) }

Every node at layer n-1 sends its certificate to its neighbour node at layer n-2 and the node at layer n-2 verifies the certificate of node at layer n-1. Nodes at layer n-2 are SMs.

13: n-2 → n-1, n : {n-2 (Q_{CA}, ID_{SM}, Q_{SM}, t_{SM}) }

Every node at layer n-2 sends its certificate to its connected node at layer n-1 then to the SM at layer n. The node at layer n-1 verifies the certificate of node at layer n-2 and node at layer n-2 verifies certificate of node at layer n-1.

14: n-2 → n, n-1 : {share link keys }

SM at layer n-2 executes steps from 3 to 8 to share symmetric link keys with node at layer n-1 and SM at layer n.

15: n-2 → n-3 : {n-2 (Q_{CA}, ID_{SM}, Q_{SM}, t_{SM}) }

Every node at layer n-2 which is a SM broadcasts its certificate to nodes at layer n-3 and nodes at n-3 verify the certificate of its SM.

16: n-3 → n-2 : {n-3 (Q_{CA}, ID_U, Q_U, t_U) }

Each node at layer n-3 sends its certificate to its connected SM at layer n-2.

17: n-2 → n : all certificates {n-3 (Q_{CA}, ID_U, Q_U, t_U) }

Every SM at layer n-2 sends the certificates of its nodes at layer n-3 to its SM at layer n for verification.

18: n → n-1 : all certificates {n-3 (Q_{CA}, ID_U, Q_U, t_U) }

SM at layer n sends the certificates of nodes at layer n-3 to its downstream nodes at layer n-1 for verification.

19: n-1 → n : {valid certificates or invalid certificates}

Every node at layer n-1 sends to its SM indicating that the checked certificate from layer n-3 is valid or not.

20: n → n-2 : {valid certificates or invalid certificates}

SM at layer n sends to the SM at layer n-2 indicating that the checked certificates from layer n-3 are valid or not. Then SMs at layer n-2 executes steps from 3 to 8 to share symmetric link keys with nodes at layer n-3. **Finally**, lower layer SMs send certificates of their neighbour nodes underneath to higher layer SMs for verification.

Discussion

The bottleneck of algorithm 1 is the number of the SMs near the BS because if the number of these nodes increases, this will reduce the setup time for the nodes underneath the SMs. Therefore, if the number of SMs near the BS is more than three, SMs near the BS execute algorithm 2.

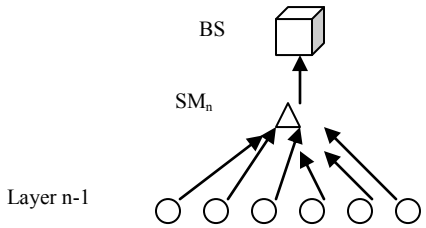


Figure 6.3.a, Certificates Verification for layer n-1

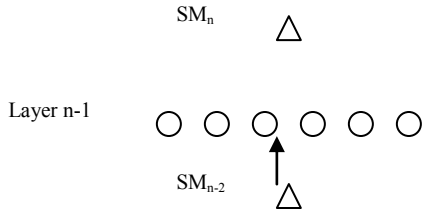


Figure 6.3.b, Certificates Verification for layer n-2

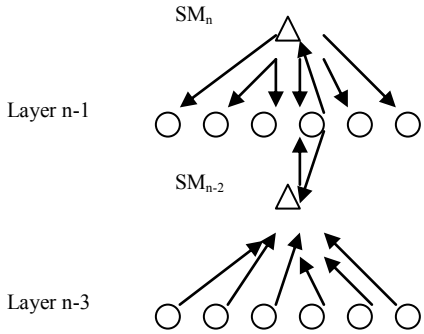


Figure 6.3.c, Certificates Verification for layer n-3

Fig. 6.3 shows the certificates shared verification process in three layers using the first algorithm. Initiator nodes start the process of key management in a distributed manner where these nodes are predetermined every number of nodes such as 30, 20 or 10 nodes. Initiator nodes work as HSN to control the setup time for the key management.

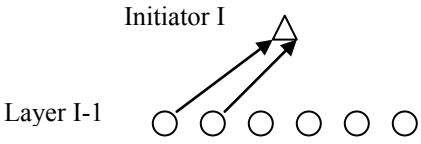


Figure 6.4.a, Certificates Verification using Initiator for 2 nodes

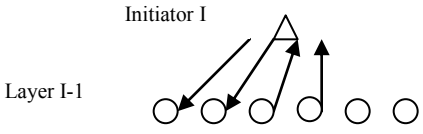


Figure 6.4.b, Certificates Verification using Initiator for 4 nodes

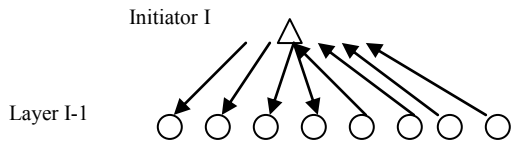


Figure 6.4.c, Certificates Verification using Initiator for 8 nodes

Figure 6.4 shows the certificates shared verification process for one layer using algorithm 2. The SM verifies the certificates of first two nodes then it sends the certificates of the second two nodes to the first two nodes then it sends certificates of other four nodes to the verified four nodes. Algorithm 2 is efficient in terms of the distribution of power consumption among sensor nodes in the cluster and it can be used with all SMs in their clusters. Algorithm 1 provides a high speed for certificates verification but its drawback is that the cluster nodes between an initiator and its upper layer SM are not involved in the process of certificates verification. Therefore, there is a trade-off between high speed certificates verification using algorithm 1 and distributed power consumption using algorithm 2.

Algorithm 2: Initiator nodes to start key management process

1: I → n : { I (Q_{CA}, ID_{SM}, Q_{SM}, t_{SM}) }

Each initiator node broadcasts its certificate to its underneath nodes at layer n to verify it. The nodes at layer n verify the certificate of the initiator.

2: n → I : { n (Q_{CA}, ID_U, Q_U, t_U) }

The initiator node receives the certificates of its underneath nodes for verification. We assume there are n nodes underneath the initiator node. First, the initiator node verifies the certificate of the first two nodes.

3: I → $n_{1,2}$: { share link keys }

The initiator node shares link keys with node 1 and node 2 as steps from 3 to 8 in algorithm 1.

4: I → $n_{1,2}$: { $n_{3,4}$ (Q_{CA}, ID_U, Q_U, t_U) }

The initiator node sends to node 1 and node 2 underneath the certificates of node 3 and node 4 for verification.

5: $n_{1,2}$ → I : { valid certificates or invalid certificates }

Node 1 and node 2 send to the initiator node two messages indicating that certificates of nodes 3 and 4 are valid or not.

6: I → $n_{3,4}$: { share link keys }

The initiator node shares link keys with node 3 and node 4 as steps from 3 to 8 in algorithm 1.

7: I → $n_{1,2,3,4}$: { $n_{5,6,7,8}$ (Q_{CA}, ID_U, Q_U, t_U) }

The initiator node sends to node 1, node 2, node 3 and node 4 underneath the certificates of node 5, node 6, node 7 and node 8 for verification and nodes 1, 2, 3, 4 respond with valid certificate or not.

8: I → $n_{5,6,7,8}$: { share link keys }

The initiator node shares link keys with node 5, node 6, node 7 and node 8 as steps from 3 to 8 in algorithm 1. **Finally**, the process of the initiator continues to verify all of its underneath nodes then its underneath nodes use algorithm 1 to share link keys with their underneath nodes and so on.

---

1. Certificates shared verification between the SM near the BS and the BS needs two messages but it needs four messages between SM at lower layer and SM at upper layer.

2. Each SM establishes a link key with its nodes underneath in ten messages but the SM near the BS establishes a link key with its nodes underneath in eight messages.
3. After the SMs and the sensor nodes establish link keys, they determine their locations using our proposed secure localization scheme with certificates shared verification.

6.4.3 Secure Localization Phase:

A number of secure localization algorithms [151] have been reported. Different researchers have different strategies to categorize them. These strategies can be divided into direct and indirect localization, centralized localization and distributed localization, range-based localization and range-free localization, absolute localization and relative localization.

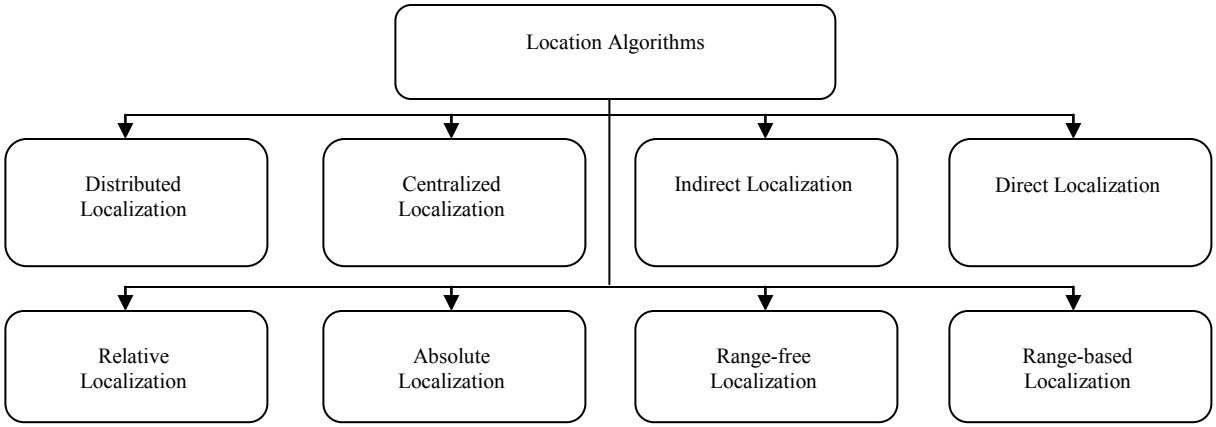


Figure 6.5, Location Algorithms Categories

We propose to get the location information from the followings approach:

The indirect approaches of localization were introduced to overcome some of the drawbacks of the GPS-based direct localization techniques while retaining some of its advantages. In this approach, a small subset of nodes in the network, called the beacon nodes, are equipped with GPS receivers to compute their location. Beacon nodes send beams of signals providing their location to all nodes in their vicinity. Using the

transmitted signal containing location information, nodes compute their location. Each node needs three beacon nodes to locate its position.

Our proposed scheme depends on the SM and certificates shared verification for secure localization. We assume that each cluster has three beacon nodes. Sensor nodes in the cluster send the beacon nodes certificates to the SM then the SM sends these certificates to its upper layer SM for verification to ensure one verification time for beacon nodes certificates for the whole cluster. The upper layer SM sends these certificates to its underneath nodes for verification. Verification power is 1000 times more than communication power.

Algorithm 3: Secure Localization

1: $\text{Beacons}_{1,2,3} \rightarrow \text{SM}_n$: {$\text{Beacons}_{1,2,3}$ (Q_{CA}, ID_B, Q_B, t_B) }

The beacon nodes near BS broadcast their certificates and locations to SMs near BS. We need three beacon nodes to locate the position.

2: $\text{SM}_n \rightarrow$ BS : { $\text{Beacons}_{1,2,3}$ (Q_{CA}, ID_B, Q_B, t_B) }

The SMs near BS at layer n send the certificates of the beacon nodes to BS for verification.

3: BS $\rightarrow$ SM_n : {valid certificates of $\text{Beacons}_{1,2,3}$ }

BS sends to SMs at layer n that beacon nodes certificates are valid.

4: $\text{SM}_n \rightarrow \text{Beacons}_{1,2,3}$: { $\text{Key}_{1,2,3}$ }

Every SM at layer n shares a link key with the three beacon nodes in four steps.

5: SM_n : calculates (x, y) position

Every SM at layer n calculates its position.

6: $\mathbf{Beacons_{1,2,3} \rightarrow n\text{-}1 : \{Beacons_{1,2,3}\ (Q_{CA}, ID_B, Q_B, t_B)\ \}}$

The beacon nodes near BS broadcast their certificates and locations to nodes at layer n-1.

7: $\mathbf{n\text{-}1 \rightarrow SM_n : \{\ Beacons_{1,2,3}\ (Q_{CA}, ID_B, Q_B, t_B)\ \}}$

The nodes at layer n-1send the certificates of beacon nodes to SMs at layer n for verification. If the beacon nodes certificates are previously verified, it is ok but if there are new beacon nodes certificates, then SMs at layer n send the new beacon nodes certificate to BS for verification.

8: $\mathbf{SM_n \rightarrow n\text{-}1 : \{\ Key_{1,2,3}\ \}}$

Every SM at layer n sends its link keys with the beacon nodes to its connected nodes at layer n-1.

9: n-1 : calculates (x, y) position

Every node at layer n-1 calculates its position.

10: $\mathbf{Beacons_{4,5,6} \rightarrow SM_{n\text{-}2} :\{Beacons_{4,5,6}\ (Q_{CA}, ID_B, Q_B, t_B)\ \}}$

The beacon nodes near SMs at layer n-2 broadcast their certificates and locations to SMs at layer n-2.

11: $\mathbf{SM_{n\text{-}2} \rightarrow SM_n : \{\ Beacons_{4,5,6}\ (Q_{CA}, ID_B, Q_B, t_B)\ \}}$

The SMs at layer n-2 send the certificates of the beacon nodes to SMs at layer n for verification.

12: $\mathbf{SM_n \rightarrow n\text{-}1 : \{\ Beacons_{4,5,6}\ (Q_{CA}, ID_B, Q_B, t_B)\ \}}$

The SMs at layer n send the certificates of the beacon nodes to nodes at layer n-1 for verification.

13: n-1 → SM_n : { valid certificates of $Beacons_{4,5,6}$ }

The nodes at layer n-1 send to SMs at layer n that beacon nodes certificates are valid.

14: SM_n → SM_{n-2} : { valid certificates of $Beacons_{4,5,6}$ }

The SMs at layer n send to SMs at layer n-2 that beacon nodes certificates are valid.

15: SM_{n-2} → $Beacons_{4,5,6}$: { $Key_{4,5,6}$ }

Every SM at layer n-2 shares a link key with the three beacon nodes in four steps.

16: SM_{n-2} : calculates (x, y) position

Every SM at layer n-2 calculates its position.

17: $Beacons_{4,5,6}$ → n-3 : {$Beacons_{4,5,6}$ (Q_{CA}, ID_B, Q_B, t_B) }

The beacon nodes near nodes at layer n-3 broadcast their certificates and locations to nodes at layer n-3.

18: n-3 → SM_{n-2} : { $Beacons_{4,5,6}$ (Q_{CA}, ID_B, Q_B, t_B) }

The nodes at layer n-3 send the certificates of beacon nodes to SMs at layer n-2 for verification. If the beacon nodes certificates are previously verified, it is ok but if there are new beacon nodes certificates, then SMs at layer n-2 send the new beacon nodes certificate to SMs at layer n for verification.

19: SM_{n-2} → n-3 : { $Key_{4,5,6}$ }

Every SM at layer n-2 sends its link keys with the beacon nodes to its connected nodes at layer n-3.

20: n-3 : calculates (x, y) position

Every node at layer n-3 calculates its position. **Finally**, lower layer SMs send certificates of beacon nodes to higher layer SMs for verification.

1. Certificates shared verification for beacon nodes certificates between the SM at lower layer and the SM at higher layer will reduce the setup time and reduce computations complexity at the cost of increasing only four messages.
2. Certificates verification for beacon nodes is done only one time at the SM not multiple times at each node underneath the SM to reduce computations complexity.
3. Sensor nodes underneath SM will use the shared keys between the SM and the beacon nodes which will reduce the setup time, computations and storage overhead.
4. After the SMs and the sensor nodes determine their locations, they form secure clustering.

6.4.4 Secure Clustering Phase:

SMs can form secure clustering [153] with their nodes underneath and the SM can choose BKSM to replace it if the SM is compromised.

Algorithm 4: Secure Clustering

1: BS → n : {req SM_msg }

BS sends to nodes near BS at layer n that these nodes are SMs using its shared symmetric key with these nodes.

2: SM_n → n-1 : { adv cluster_msg }

Every SM at layer n sends an encrypted advertise message to nodes at layer n-1 to form a cluster.

3: n-1 → SM_n : { join cluster_msg }

Every node at layer n-1 sends an encrypted message to its SM at layer n to join the cluster.

4: SM_n → n-1 : {choose BKSM }

The SM at layer n chooses BKSM according to maximum connectivity between the

---

BKSM and the nodes in the cluster where BKSM must be connected to all nodes in the cluster.

5: $BKSM_n$ → n-1 : { BKSM (Q_{CA}, ID_{BKSM}, Q_{BKSM}, t_{BKSM}) }

The BKSM sends its certificate to the nodes at layer n-1 where SM at layer n verifies this certificate. Also, the BKSM sends its certificate to its upper layer node to establish a link key with it to reroute data if SM is compromised.

6: n-1 → n-2 : { req SM_msg }

The nodes at layer n-1 send to nodes at layer n-2 an encrypted message that these nodes are SMs.

7: SM_{n-2} → n-3 : { adv cluster_msg }

Every SM at layer n-2 sends an encrypted advertise message to nodes at layer n-3 to form a cluster.

8: n-3 → SM_{n-2} : { join cluster_msg }

Every node at layer n-3 sends an encrypted message to its SM at layer n-2 to join the cluster.

9: SM_{n-2} → n-3 : {choose BKSM }

The SM at layer n-2 chooses BKSM according to maximum connectivity between the BKSM and the nodes in the cluster where BKSM must be connected to all nodes in the cluster.

10: $BKSM_{n-2}$ → n-3 : { BKSM (Q_{CA}, ID_{BKSM}, Q_{BKSM}, t_{BKSM}) }

The BKSM sends its certificate to the nodes at layer n-3 where SM at layer n-2 verifies this certificate. Also, the BKSM sends its certificate to its upper layer node to establish

a link key with it to reroute data if SM is compromised. **Finally**, the steps of forming the secure clustering are performed until the last layer of SM.

1. Our proposed secure clustering scheme assumes a hybrid key management protocol to achieve high security level.
2. Our proposed scheme chooses a BKSM to solve the problem of the compromised SM and to sign the message of revoked SM.
3. Our scheme achieves secure clustering in four messages.

6.4.5 Key Revocation Phase:

The first component of our dynamic-based key management scheme is the keys revocation of the compromised sensor nodes. SurvSec security architecture has a compromised nodes detection algorithm at the first stage to be able to detect compromised nodes but it is discussed in chapter 5.

When a sensor node is compromised by an adversary, all the session keys used by this sensor node will be revoked. The SM will broadcast a revocation message containing the identification of the compromised node to all the nodes underneath. A digital signature is computed over the message by utilizing Elliptic Curve Digital Signature Algorithm ECDSA at [154] with SMs private key. When a node receives the revocation message, it checks the message by verifying the digital signature. This prevents an adversary from sending a fake revocation message. If SM is compromised, it is revoked by the BKSM.

6.4.6 Rekeying Phase:

The second component of our dynamic based key management scheme is rekeying after compromised nodes detection or rekeying can be done periodically. Rekeying is used when the SM is compromised. The BKSM will share a link key with its upper layer SM then the BKSM will use our novel scheme of certificates shared verification with its upper layer SM to verify the certificates of the cluster nodes. Finally, the BKSM will share link keys with its lower SM and its nodes in the cluster.

6.4.7 Addition of New Nodes Phase:

When a new node joins the network, it tries to find its nearest SM by broadcasting a Hello message contains the new node certificate.

To support the addition of new nodes, the SM verifies the certificate of the new nodes using our novel scheme of certificates shared verification.

6.5 Security Analysis

The security analysis of our proposed protocol focuses on the resilience to node compromising attack, and collusion attack.

6.5.1 Compromised Node Attack

1- If an attacker compromises one ordinary node, therefore, the number of insecure link is $P_{insec} = 1 / N$ where N is the number of nodes at the network. For n compromised ordinary nodes, number of insecure links is $P_{insec} = n / N$.

2- If the attacker compromises one SM, therefore, the number of insecure links is $P_{insec} = (n_s + 3) / N$ where n_s is the number of nodes in the cluster of the SM and 3 represents the links with the upper SM, lower SM and SM upper node. For n compromised SMs, the number of insecure links is $P_{insec} = n\ (n_s + 3) / N$.

3- Suppose that in a network of N nodes, there are m SMs and BKSMs. The probability to compromise one SM or one BKSM is P(com) = 2m / N, so the probability of at least *k* nodes from the SMs and BKSMs are captured is:

$$p(k) = 1 - \sum_{i=0}^{k} \binom{n}{i} p(com)^i (1 - p(com))^{n-i} \quad (1)$$

The probability that all SMs and BKSMs are captured is:

$$p(k) = 1 - \sum_{i=0}^{2m} \binom{n}{i} p(com)^i (1 - p(com))^{n-i} \quad (2)$$

4- Our proposed key management assumes compromised node detection at the first stage and compromised nodes revocation. Therefore, the SM will revoke the ordinary compromised node and the BKSM will revoke the SM to eliminate the insecure links.

Node compromising attack refers to the capability of an attacker to inject cloned nodes or false IDs in the network using the key materials it gets from the compromised nodes. Node captures in hostile environments is inevitable. An effective key management scheme should be able to recover from such attacks to be effective. We describe some of the inherent security advantages of utilizing our proposed key management scheme. Then, using the threats identified in section 6.3, we analyze how well our proposed scheme recovers from those attacks. A clustered and hierarchical framework for a WSN with security managers applying distributed security provides many beneficial security properties. Isolation is the primary benefit of a clustered key management scheme. Security managers are responsible for distributing and establishing link keys. Therefore, an attack such as compromised node attack that reveals keys of sensor nodes within one cluster will not impact any other cluster in the network. SurvSec security architecture has compromised node detection algorithm to detect compromised nodes. Security managers are reported with the compromised sensor nodes underneath.

6.5.2 Collusion Attack

Two nodes can collude when they share their keys with each other. Our designed protocol is resistant to collusion attack because each sensor node communicates only with a SM therefore; compromised nodes cannot discover themselves.

Each compromised sensor node will only reveal its link key with the security manager plus its public and private key. Therefore, it is conceivable that when the

compromised sensor nodes collude they will only reveal their keys but this collusion attack will not result in capturing the network. If the compromised sensor node changes its location for launching collusion attack, it will be discovered and revoked. From such a scenario, the adversary is incapable of revealing all encrypted communications in the network. The main idea of our proposed scheme is the location based key management where every node report its ID and location before it join the network to prevent compromised node attack and collusion attack.

6.6 Performance Analysis

The performance analysis is measured in computation complexity, communication complexity, storage complexity and setup time. We assume that the network is secure during setup time which depends on number of initiators.

6.6.1 Computation Complexity

Our proposed hybrid key management scheme using certificates shared verification has much lower computations overhead at SM side rather than computations at HSN in heterogeneous network. For algorithm 1, our scheme assumes each sensor node in each cluster verifies four certificates for the keys distribution and localization which are the certificate of its SM, two certificates from its underneath nodes and one beacon node certificate. SM verifies one certificate which is its upper node. For algorithm 2, our scheme assumes each sensor node in each cluster verifies at most four certificates for the keys distribution and localization which are the certificate of its initiator, two certificates from the nodes of its cluster and one beacon node certificate. Initiator node verifies three certificates which are two certificates from its underneath nodes and one certificate for its upper node.

Each sensor node and SM performs hash two times to generate one link key. The sensor node encrypts its part of the link key with the SM's public key using ECC 160 bits scalar multiplication and addition. Also, the SM decrypts the received message from the sensor node with its private key. The SM encrypts its part of the link key using symmetric key under the key from the sensor node. The sensor node decrypts the message from the

SM using symmetric key. Our scheme has less computation overhead at SM than the scheme uses HSNs at HSN.

In our scheme:

Each node performs at most 4 verifications and shares key with SM or initiator for keys distribution and localization.

The SM or initiator performs at most 3 verifications and shares keys with n nodes for keys distribution and localization where n nodes are ranged from 4 to 8 nodes in the cluster.

In HSN scheme:

Each node performs 4 verifications and shares key with HSN for keys distribution and localization.

The HSN performs n+3 verifications and shares keys with n+3 nodes where n nodes are ranged from 10 to 30 nodes underneath the HSN.

Our scheme has lower computations than the HSN scheme.

6.6.2 Communication Complexity

Communication complexity is the number and size of packets sent and received by a sensor node. In our protocol, the number of messages sent and received to establish a key between one sensor node and a SM is ten messages and we need six messages to establish link key between lower layer SM and upper layer SM. The device ID is 64 bits, expiration time is 64 bits, random number is 160 bits and L the sensor location is 64 bits. The certificate is 56 bytes from 20 bytes CA public key, 8 bytes node ID, 20 bytes node public key and 8 bytes validity time. Our scheme has a higher communication overhead than the HSN model with 4 messages to establish a link key for every node.

In our scheme:

For algorithm 1:

Communication overhead = 6 N_{SM} + 10 m N_{SM},, N_{SM} is number of SMs and m is the number of nodes underneath SM within its cluster.

For algorithm 2:

Communication overhead = I (12 + 8 (m – 2)) + 6 I, I is the number of initiator nodes, m is the number of nodes underneath the initiator. 2 nodes needs 12 messages and other nodes in the cluster need 8 messages and 6 represents the communication between the initiator and its upper node.

For algorithm 1 and 2: Total communication overhead is C_{com}.

$C_{com} = N_{SM} (6 + 10 m) + I (2 + 8 m)$.

We found that the communication overhead for algorithm 2 is lower than communication overhead for algorithm 1.

In HSN scheme:

For one HSN every 30 nodes: communication overhead is C_{com}.

$C_{com} = N_{HSN} (6 + 6 n_0 + 8 n_1 + 10 n_2 + 12 n_3)$. Where N_{HSN} is the number of HSNs, n_0 is the number of first layer nodes underneath the HSN, n_1 is the number of second layer nodes underneath the HSN, n_2 is the number of third layer nodes underneath HSN, n_4 is the number of fourth layer nodes underneath the HSN and 6 represents the communication between the HSN and its upper node.

For one HSN every 20 nodes: communication overhead is C_{com}.

$C_{com} = N_{HSN} (6 + 6 n_0 + 8 n_1 + 10 n_2)$.

For one HSN every 10 nodes: communication overhead is C_{com}.

$C_{com} = N_{HSN} (6 + 6 n_0)$.

Our model has lower communication overhead than the HSN model for one HSN every 30 but our model has higher communication overhead than the HSN model for one HSN every 20 or 10 nodes.

6.6.3 Storage Complexity

Storage complexity is the amount of memory units required to store security credentials. Each sensor node stores its public key, private key, BKSM public key and the link key shared with the SM. The SM stores all of the shared keys with each sensor node underneath plus its public, private key, link key with upper SM, link key with the lower SM and link key with its upper node. Our scheme has the same storage overhead as HSN scheme.

In our scheme:

Total SMs storage overhead = $(N_S+5)\ N_{SM}$, N_S is the number of nodes underneath SM and N_{SM} is the number of security managers.

Sensor nodes storage overhead = $3\ N_S$.

In HSN scheme:

Total HSNs storage overhead = $(N_S+5)\ N_{HSN}$.

Sensor nodes storage overhead = $3\ N_S$.

6.6.4 Setup Time

We assume that verification using ECDSA takes 4 sec [155], share link key takes 1 sec [135] and certificate transmission takes 0.2 sec [135]. The setup time of the share link key is less than the setup time of the reference model [135] because the proposed model has 4 steps where the reference model has 5 steps. The steps of the proposed model are encryption of key with ECC then encryption of key with symmetric key then encryption of key with symmetric key then encryption of key with symmetric key where the 5 steps of the reference model are encryption of key with ECC then encryption of key with symmetric key then key derivation function (KDF) of the key then hash of the key then hash of the key. Hash of the key takes more time than encryption using symmetric key because the hash function of SHA-1 is 80 rounds and AES-128 is 10 rounds. Therefore, we assume that the proposed model share link key time is equal to the share link key of the reference model. The transmission time is dominant factor but on the other hand, the bottleneck will be the certificate verification operation time. Setup time is equal to verification time plus communication time plus share link key time.

In our scheme:

For algorithm 1 the setup time is T.

T = 4S + n X 1S + (5 n + 2) X 0.2S, verification is done in parallel where upper layer SM sends to its underneath nodes the certificates of the nodes underneath its lower layer SM which is n nodes. Therefore, we need one verification time and n times to share link keys and (5n+2) messages to send all certificates to the verifiers and have the result.

For algorithm 2 the setup time is T.

T = m X 4S + n X 1S + (4 + 4 (n - 2)) X 0.2S, verification is done m times, share link keys is done n times and we need number of messages equal to (4 + 4 (n-2)).
The setup time for algorithm 1 is lower than the setup time for algorithm2.

In HSN scheme:

Setup time = n X 4S + n X 1S + 6n X 0.1S, where n is the number of nodes underneath the HSN and 6n is the number of messages between nodes and HSN.
Our proposed scheme with algorithm 1 has much lower setup time than HSN model where we perform parallel verification but HSN model performs sequential verification.
Our proposed scheme with algorithm 2 has a lower setup time than the HSN model where we perform parallel verifications but the HSN model performs sequential verification. Our proposed model combines both algorithm 1 and algorithm 2.

6.6.5 Scalability

In our scheme:

BKSM will replace the SM if it is compromised and this insures high scalability to extend the network.

In HSN scheme:

If a HSN is compromised in a branch, the scalability of the branch cannot be achieved because there is no backup HSN.

6.6.6 Connectivity

In our scheme:

BKSM will replace the SM if it is compromised and this insures high connectivity with its underneath nodes.

In HSN scheme:

If a HSN is compromised in a branch, the connectivity for the nodes underneath the HSN cannot be achieved because there is no backup HSN.

6.7 Simulation Results

In this section, we evaluate the communication overhead, the computations overhead and the network setup time under different number of nodes N for our proposed model and HSN model.

We built our proposed model and HSN model and we implemented a simulator in MATLAB that can scale to thousands of nodes. In this simulator, sensors can send and receive data from each other's. The simulation verifies the correctness and the feasibility of our security architecture. It is our future work to implement SurvSec in some sensor network testbeds with all its ingredients. Our simulation scenarios include N nodes distributed randomly. We choose N 1000, 2000 and 3000 sensor nodes.

In the simulations, these parameters are given as follows:

1- The number of sensor nodes *N* is varied from 1000, 2000 and 3000 sensor nodes.
2- The simulation is done for HSN or initiators every 30 nodes, 20 nodes and 10 nodes.
3- The communication overhead for the security manager to exchange a key with a node is according to algorithm 1 or algorithm 2 or both as shown in section 6.6.

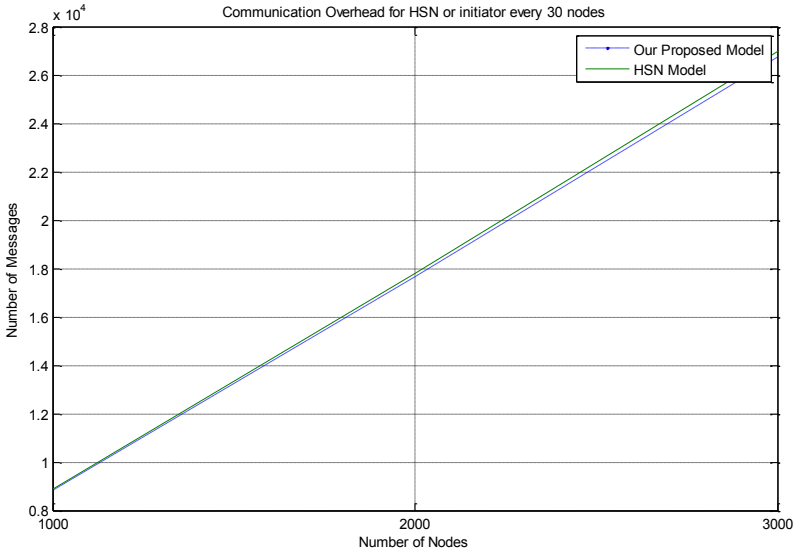


Figure 6.6.a, Communication overhead every HSN or Initiator every 30 nodes

Fig. 6.6.a shows the communication overhead for HSN model and our proposed model for one HSN every 30 nodes and one initiator every 30 nodes. Our proposed model has a lower communication overhead than the HSN model.

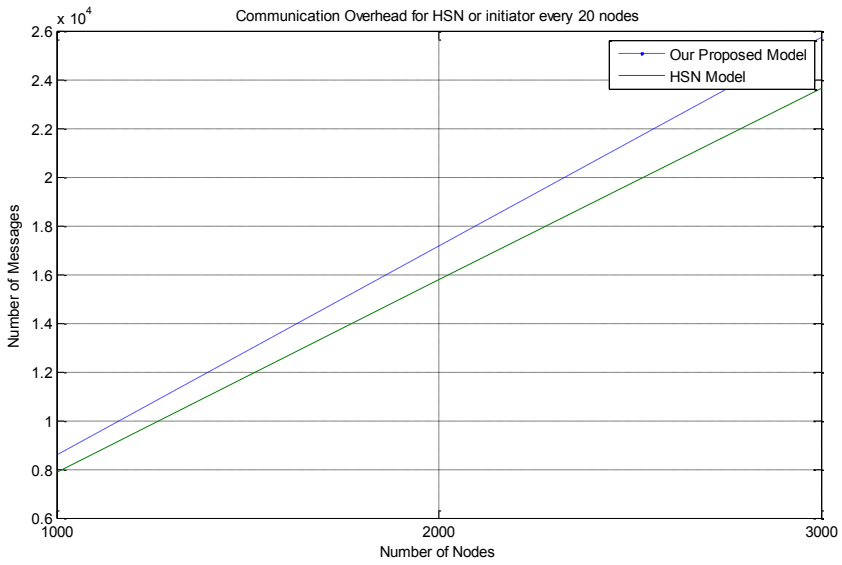


Figure 6.6.b, Communication overhead every HSN or Initiator every 20 nodes

Fig. 6.6.b shows the communication overhead for the HSN model and our proposed model for one HSN every 20 nodes and one initiator every 20 nodes. Our proposed model has higher communication overhead than the HSN model with 10%.

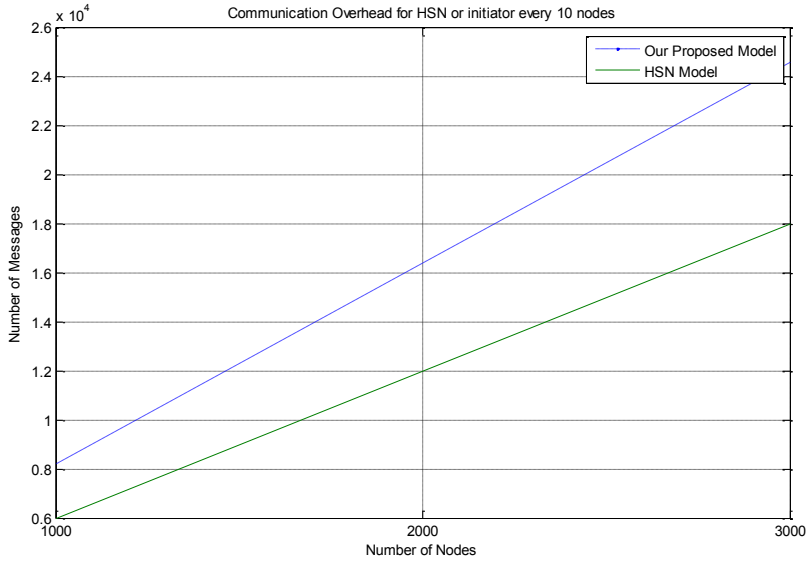


Figure 6.6.c, Communication overhead every HSN or Initiator every 10 nodes

Fig. 6.6.c shows the communication overhead for the HSN model and our proposed model for one HSN every 10 nodes and one initiator every 10 nodes. Our proposed model

has higher communication overhead than HSN model with 20%. We need larger bandwidth to overcome the increasing communication overhead.

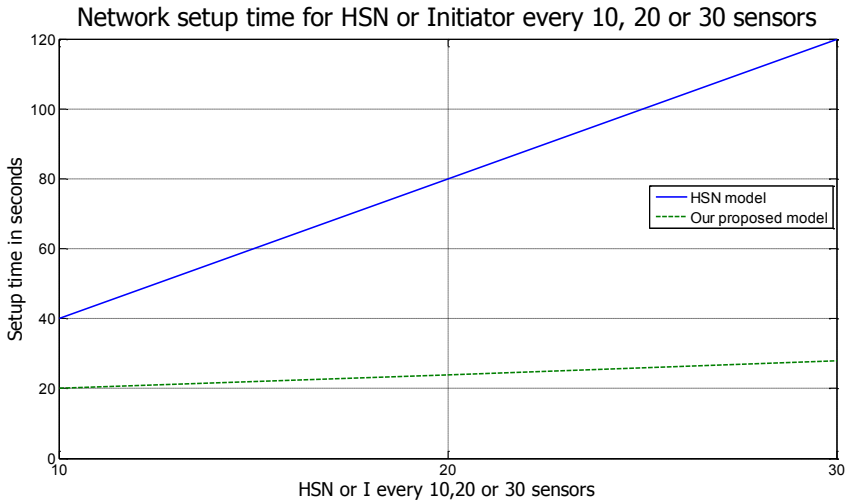


Figure 6.7, Network Setup Time for HSN or Initiator every 30, 20, and 10 nodes

Fig. 6.7 shows the network setup time for the HSN model and our proposed model for one HSN or one initiator every 30 nodes, 20 nodes and 10 nodes. Our proposed model has at least half the network setup time than the HSN model.

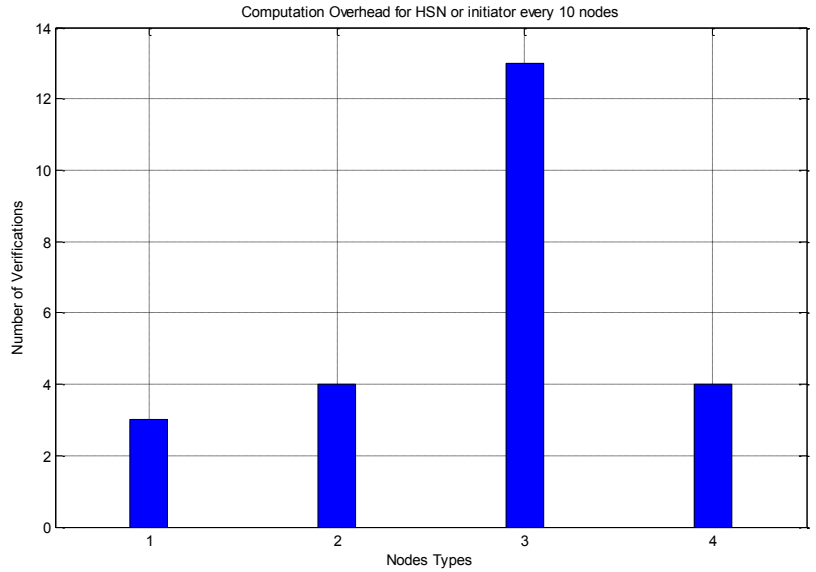


Figure 6.8, Computation Overhead of Certificates Verifications for HSN or Initiator every 10 nodes

Fig. 6.8 shows the computation overhead for certificates verifications for the HSN model and our proposed model for one HSN node or one initiator every 10 nodes. Number 1 at x-axis is the number of certificates verification at the SM which is 3 verifications for key establishment and secure localization. Number 2 at x-axis is the number of certificates verification at every node in our proposed model which is 4 verifications for key establishment and secure localization. Number 3 at x-axis is the number of certificates verification at the HSN which is 13 verifications for key establishment and secure localization. Number 4 at x-axis is the number of certificates verification at every node in HSN model which is 4 verifications for key establishment and secure localization. Our proposed model has lower computation overhead than HSN model. Our scheme has one quarter lower certificates verifications overhead than the HSN model at SM side and one half lower certificates verification overhead in total. Finally, for HSN or Initiators every 10 nodes we increase communication overhead by 20% and we decrease the computation overhead to one half whereas the power of certificates verification using ECDSA is 1000 times more than the power of communication.

6.8 Security Proof

In this section, we describe the security for our proposed key management scheme and introduce two classes of attacks.

Security of our Proposed Scheme

The proposed hybrid key management scheme achieves the correctness and secrecy requirements necessary to provide a distributed key generation protocol based on Elliptic Curve Discreet Logarithmic Problem (ECDLP).

The definition of the security of key management scheme is based on correctness of the key management and the attacks that target the key management scheme during the steps of the key management process. We use ECDSA to sign the certificates. The hybrid key management scheme has two classes of attacks. Existential unforgeability against adaptive chosen message attacks (EUF-CMA) [156] is the strongest security model of

signature scheme where the adversary is allowed to ask the signer to sign any message of its choice adaptively and the adversary can adopt its queries according to previous answers. Finally, the adversary should not provide a new message signature pair with non-negligible advantage. The first class of attacks targets the certificates signature of the nodes. The second class of attacks targets the steps of the keys establishment protocol between SM and any node.

For class 1 attacks, there are two types of adversaries which are more powerful than ordinary adversaries. Type 1 adversary is an uncertified client which wants to impersonate a victim by using public keys along with the identity of the victim. Type 2 adversary is a malicious CA, which wants to sign forged certificates. However, the Type 2 adversary cannot access the corresponding private key of the victim. Moreover, the public key that the Type 1 adversary attacks consists of the public key of CA, the identity and public key of the victim. The public keys that a Type 2 adversary attacks consist of the identities and public keys of a victims, and the public key of CA and the attacker wants to know the private key of the CA to sign new certificates. The system should remain secure under such two types of powerful attacks.

For class 2 attacks, there are two types of attacks which are impersonation attack and replay attack.

Correctness of the Key Management:

The proposed key management is based on the verification of the certificate using ECDSA. Therefore; the correctness of the key management is measured by the correctness of the ECDSA. This section describes the correctness of ECDSA [206].

The CA wants to send a signed certificate to a sensor. At first, the curve parameters (*CURVE, G, n*) must be agreed upon. Also, the field and equation of the curve must be agreed upon. Also, G is a base point of prime order on the curve; and n is the multiplicative order of the point G must be agreed upon.

Each node creates a key pair, consisting of a private key integer d_A, randomly selected in the interval $[1, n-1]$; and a public key curve point $Q_A = d_A \times G$. We use $\times$ to denote elliptic curve point multiplication by a scalar.

We sign the certificate (*C)* by these steps:

1. Calculate $e = HASH(C)$, where HASH is a cryptographic hash function, such as SHA-1.
2. Let Z be the L_n left most bits of e, where L_n is the bit length of the group order n. Z is less than n.
3. Select a random integer k from $[1, n-1]$.
4. Calculate the curve point $(x_1, y_1) = k \times G$.
5. Calculate $t = x_1 (mod\, n)$. If $t = 0$, go back to step 3.
6. Calculate $r = k^{-1}(Z + td_A)(mod\, n)$. If $r = 0$, go back to step 3.
7. The signature is the pair (t, r).

When computing *r*, the string z resulting from $HASH(C)$ shall be converted to an integer. Note that Z can be greater than n but not longer in size.

It is crucial to select different k for different signatures, otherwise the equation in step 6 can be solved for d_A, the private key: Given two signatures (t, r) and (t, r'), employing the same unknown k for different known certificates C and C', an attacker can calculate Z and Z', and since $r - r' = k^{-1}(z - z')$ (all operations in this paragraph are done modulo n the attacker can find $k = \frac{z - z'}{r - r'}$. Since $r = k^{-1}(z + td_A)$, the attacker can now calculate the private key $d_A = \frac{rk - z}{t}$.

Signature verification algorithm

For a node to authenticate the CA signature, the node must have a copy of CA public key curve point Q_{CA}. The signature is the pair (t, r).

A node follows these steps:

1. Verify that *t* and *r* are integers in $[1, n-1]$. If not, the signature is invalid.
2. Calculate $e = HASH(C)$, where HASH is the same function used in the signature generation. Let Z be the L_n left most bits of e.

3. Calculate $w = r^{-1} (mod\ n)$.

4. Calculate $u_1 = zw (mod\ n)$ and $u_2 = tw (mod\ n)$.

5. Calculate the curve point $(x_1, y_1) = u_1 \times G + u_2 \times Q_{CA}$.

6. The signature is valid if $t = x_1 (mod\ n)$, invalid otherwise.

Correctness of the Algorithm

E is the curve point computed in step 5 of verification,

$E = u_1 \times G + u_2 \times Q_{CA}$

From the definition of the public key as $Q_{CA} = d_A \times G$,

$E = u_1 \times G + u_2 d_A \times G$

Because elliptic curve scalar multiplication distributes over addition,

$E = (u_1 + u_2 d_A) \times G$

Expanding the definition of u_1 and u_2 from verification step 4,

$E = (Zs^{-1} + rs^{-1} d_A) \times G$

Collecting the common term s^{-1},

$E = (Z + rd_A) s^{-1} \times G$

$s = k^{-1} (Z + rd_A)$

Expanding the definition of s from signature step 6,

$E = (Z + rd_A)(Z + rd_A)^{-1} (k^{-1})^{-1} \times G$

Since the inverse of an inverse is the original element, and the product of an element's inverse and the element is the identity, we are left with

$E = k \times G$

From the definition of r, this is the verification step.

Class 1 Attacks:

Type 1 attack: uncertified client with public key of the victim.

Theorem 1. We say that our scheme is existentially unforgeable against adaptive chosen message (EUF-CMA) Type 1 attack if no polynomial bounded Type 1 adversary *A* has a non-negligible advantage against the challenger in the following game:

Key Gen: the adversary tries to discover the SM private key from knowing the SM public key and from sending queries to the SM.

Sign messages: the adversary issues queries to the challenger and these queries may be asked adaptively. The challenger responds with the resulting signature to the adversary.

Output: Finally, the type 1 adversary outputs a new signature σ for a message M.

Proof. The adversary *A* wins the game if the output signature is non-trivial and the attacker can produce the private key of the SM. This probability is negligible since ECDSA is secure and in order to get the private key from the public key, the attacker must solve the elliptic curve discrete logarithmic problem ECDLP which is a hard problem. The computation of elliptic curve discrete logarithmic problem is computationally infeasible.

Type 2 attack: uncertified CA with public key of the CA.

Theorem 2. We say that our scheme is existentially unforgeable against adaptive chosen message (EUF-CMA) Type 2 attack if no polynomial bounded Type 2 adversary *A* has a non-negligible advantage against the challenger in the following game:

Key Gen: the adversary tries to discover the CA private key from knowing the CA public key and from sending queries to the CA.

Sign messages: the adversary issues queries to the challenger and these queries may be asked adaptively. The challenger responds with the resulting signature to the adversary.

Output: Finally, the type 2 adversary outputs a new signature σ for a certificate C.

Proof. The adversary *A* wins the game if the output signature is non-trivial and the attacker can produce the private key of the CA. This probability is negligible since ECDSA is secure and in order to get the private key from the public key, the attacker must solve the elliptic curve discrete logarithmic problem ECDLP which is a hard

problem. The computation of elliptic curve discrete logarithmic problem is computationally infeasible.

Discussion. The discrete logarithm problem is as follows: given an element *g* in a finite group *G* and another element *h* in *G*, find an integer *x* such that $g^x = h$.

The ECDSA uses an elliptic curve E over Z_P and a point $P \in E(Zp)$ with order a prime q of size around 160 bits. The signer selects the value $a \in \{1, \ldots, q-1\}$ and computes Q = aP. Its public key is the (p, E, P, q, Q) and his private key a.

To sign a message m having hash value $h(m) \in \{0, \ldots, q-1\}$, he selects a random number $k \in \{1, \ldots, q-1\}$ which is the ephemeral key and computes k P = (x, y) (where x and y are regarded as integer between 0 and p−1). Next, he computes the value r = x mod q and the value $s = k^{-1} (h(m) + a\, r) \bmod q$.

The signature of m is the pair (r, s).

For verification of signature one computes $u_1 = s^{-1} h(m) \bmod q$, the value $u_2 = s^{-1} r \bmod q$, and $u_1 P + u_2 Q = (x_0, y_0)$.

He accepts the signature if and only if $r = x_0 \bmod q$.

The assumption here is that the only way to forge signature is to recover either the secret key a, or the ephemeral key k. Thus, the parameters of the system is chosen in such a way that the computation of discrete logarithms is computationally infeasible, and so a or k is well protected.

Class 2 Attacks:

Type 3 attack: Impersonation attack

The impersonation attack occurs when the attacker tries to impersonate the security manager or an ordinary node.

Theorem 3. It is computationally infeasible for an adversary to impersonate a legitimate node.

Proof. When an adversary wishes to perform impersonation attack to an ordinary node or security manager, he needs to forge the digital signature of the node. We assume that forging the digital signature of the node without obtaining the private key of that node is computationally infeasible in our model.

Type 4 attack: Replay attack

The replay attack occurs when an adversary can intercept the key establishment messages between the security manager and an ordinary node. We focus on the replay attack that can be performed by an adversary to resend a session information request.

Theorem 4. It is computationally infeasible for an adversary to successfully replay an honest node's session formation request.

Proof. Each session formation request include a nonce which acts as a unique one-time session ID to prevent an adversary from replaying the session formation request. When a node receives a duplicated session formation request during the life time of the original session formation request which means it has the same nonce, it ignores the duplicated session formation request.

6.9 Comparison with Others' Works

Now, we compare between our proposed model and HSN model.

Table 6.1, Comparison between Our Model and HSN Model.

	Property	HSN Model [135]	Our Model
1	Computation overhead for key establishment and secure localization	N verification at HSN and 4 verifications at node	3 verifications at SM and 4 verifications at node
2	Storage overhead	3 keys at node (n+5) at HSN	3 keys at node (n+5) at SM
3	Communication overhead for key establishment	6 or 8 or 10 or 12 messages for each node according to HSN every 30 or 20 or 10 nodes	8 messages for algorithm 2 or 10 messages for algorithm 1 for each node
4	Communication overhead for secure localization	No	3 messages from each node to SM and one verification message from SM to each node plus 6 messages for one time verification
5	Computation overhead for secure localization	3n verifications for the cluster	3 verifications for the whole cluster

6	Setup time	n verifications time	parallel verifications executes in 1/n time of HSN model for algorithm1 and n/2 time of HSN model for algorithm 2
7	Scalability	Affected by compromised HSN	High
8	Connectivity	Affected by compromised HSN	High
9	Backup node	No	BKSM
10	Secure localization	High cost at each node for 3 verifications	Low cost for 3 verifications for the whole cluster
11	Rekeying	High cost at HSN	Low cost at SM
12	Addition of new nodes	High cost at HSN	Low cost at SM
13	Probability of insecure links	High with compromised HSN	Low after compromised SM revocation
14	Effect of compromised nodes	No	Affect certificates shared verification
15	Nodes revocation	Cannot revoke HSN	BKSM revokes SM
16	Cost	High	Low

Our proposed scheme distributes certificate verification at nodes underneath the SM rather than verifies certificates at the SM. Also, our scheme verifies beacon nodes certificates once for the whole cluster. Our scheme has higher connectivity and scalability than HSN model. Our scheme can revoke compromised SM through BKSM and has a lower network cost than HSN scheme. Our scheme has a lower network setup time than the HSN scheme and it has same storage overhead. Our scheme has lower computations overhead than the HSN scheme.

6.10 Summary

In this chapter, we proposed the certificates shared verification key management with a novel hybrid and dynamic key management scheme for Wireless Sensor Networks which utilizes Elliptic Curve Cryptography and the symmetric key cryptography. We propose a hybrid authenticated key-establishment protocol, in which we reduce the computation intensive elliptic curve scalar multiplication of a random point at the sensor

side, and use symmetric key cryptographic operations instead. On the other hand, it authenticates the two identities based on elliptic curve implicit certificates, and solves the key distribution and storage problems, which are typical bottlenecks in pure symmetric-key based protocols. The hybrid key establishment protocol has less sensor side computation complexity compared to other public-key based key establishment protocols. We solved the problems of High end Sensor Nodes (HSNs) with the certificates shared verification key management scheme.

In addition, we also design a dynamic key management based on rekeying, keys revocation and addition of new nodes which significantly increase the resiliency of the network to compromised node attack, and collusion attack.

The performance evaluation and security analysis show that our proposed key management scheme has a higher communication overhead than the HSN model, same storage overhead than the HSN model, lower computations overhead than the HSN model and lower setup time than the HSN model. Our scheme provides perfect scalability and connectivity unlike HSN model.

CHAPTER 7

SURVSEC SPREAD SPECTRUM ENCRYPTION ARCHITECTURE FOR POST-QUANTUM COMPUTING

---

*In this chapter, we describe our designed **S**pread **S**pectrum **E**ncryption **A**rchitecture SSEA for SurvSec security architecture to resist quantum computer attacks and linear and differential cryptanalysis attacks. The spread spectrum encryption architecture is a family of three cryptographic architectures. First, SSEA1 is concerned with choosing one encryption algorithm from number of encryption algorithms to encrypt the data where the plaintext enters all the encryption algorithms. Second, SSEA2 is concerned with choosing one subkey out of 16 subkeys at each round of the used encryption algorithm. Third, SSEA3 chooses one algorithm from two encryption algorithms and then choose one subkey out of 16 subkeys at each round then the input for the second algorithm comes from RC4 stream cipher algorithm and the outputs from the two encryption algorithms are XORed. SSEA uses RC4 stream cipher as PRNG to choose one algorithm or one subkey at each round.*

7.1 Introduction

This chapter is organised as follows: in section 7.2, the preliminary information from multiple discipline areas are given. These preliminaries are the hypothesis of the design, the design goals, dynamic security, unpredictability principle, and adaptive security. Section 7.3 discussed the threat model. Section 7.4 presented the existing solutions for symmetric key ciphers to resist QC attacks. Section 7.5 outlined our newly designed key-dependent spread spectrum encryption architecture for SurvSec security

architecture. Section 7.6 explained the proof of security for SSEA3. Section 7.7 discussed the attacks on SSEA3. Section 7.8 compared between our newly designed spread spectrum architecture SSEA3 and the standard block cipher AES-256. Section 7.9 stated the SSEA3 limitations. Finally, section 7.10 is the summary of the chapter.

The fast development towards building a Quantum Computer (QC) increases the consequences of QC attacks and implies high vulnerabilities to symmetric key cipher systems and public key cipher systems. Increasing key length for symmetric key cipher systems to resist QC attacks implies increasing design size of the algorithm which means slowing down the algorithm. Inspired from the unpredictability principle, PRNG is added to the architecture of the symmetric key cipher system to add the unpredictability property to choose which algorithm is used and which subkey is used. **S**pread **S**pectrum **E**ncryption **A**rchitecture (SSEA) is a family of three architectures with a high security level and high speed resistant to QC attacks and linear and differential cryptanalysis attacks. First, SSEA has two or more encryption algorithms and multiple subkeys at each round of the encryption algorithm. SSEA architecture is used to hide which algorithm is used, to hide which subkey is used and to hide the output of the encrypted ciphertext. Second, SSEA security level is increased as the number of subkeys for each round increased or the number of rounds in the algorithm increased or the number of algorithms increased. This model increases the security level where the output from the PRNG is not on the communication channel and the attacker cannot perform analysis on this output. Finally, SSEA3 is chosen as it has the highest speed, the lowest design size and the highest security level over SSEA1, and SSEA2.

Now, new classification for cryptography has emerged after the formal modern cryptography: Pre-Quantum Computing and Post-Quantum Computing, because quantum computer enables certain problems to be solved efficiently in a short time. We will prove that QC cannot improve on classical methods to solve the unpredictability problem that we based our newly designed architecture on. Even the QC needs to try all the possible combinations to solve unpredictable problem. Quantum computation will have significant impact on symmetric key cipher systems and public key cipher systems.

In 1994, Peter Shor presented quantum algorithms which solve the factoring and the discrete logarithm problems in quantum polynomial time [**157**]. These problems are very difficult in the classical computer model and they provide a basis for the security of the most currently-used public key cryptosystems.

In 1996, Lov Grover developed a quantum algorithm for searching an unsorted database with N entries in $O(\sqrt{N})$ time and using $O(log\ N)$ storage space [**158**], [**159**]. As a result, the brute force attack on symmetric cipher systems can be obtained in only $O(\sqrt{N})$ steps instead of $O(N)$. If a suitably sized quantum computer capable of running Grover's algorithm reliably becomes available, it would reduce a 128-bit key down to 64-bit security, roughly a DES equivalent. This is one of the reasons why AES supports a 256-bit key length. Also, Bennett, Bernstein, Brassard, and Vazirani proved in 1996 that a brute-force key search on a quantum computer cannot be faster than roughly $2^{n/2}$ invocations of the underlying cryptographic algorithm, compared with roughly 2^n in the classical case [**160**]. Thus in the presence of large quantum computers an n-bit key can provide at least $n/2$ bits of security. Quantum brute force is easily defeated by doubling the key length, which has little extra computational cost in ordinary use. This implies that at least a 160-bit symmetric key is required to achieve 80-bit security rating against a quantum computer.

In 2004, the eSTREAM, ECRYPT (European Network of Excellence for Cryptology) Stream Cipher Project, began. This four years effort running from 2004 to 2008 has identified two portfolios of promising new stream ciphers, one for software orientation and the other for hardware orientation. The eSTREAM raised a question, if large QC can be built, how will this influence the symmetric key cryptographic landscape? [**161**].

In [**162**], Akihiro Yamamura and Hirokazu Ishizuka on 2000 were the first to discuss how to attack block cipher algorithms with multiple QCs using Grover's algorithm.

In [**163**], Gilles Piret and François-Xavier Standaert discussed on 2009 the distance between the practical security approach and the actual theoretical security provided by a given cipher. Their experiments illustrated that the provable security

against linear cryptanalysis is not achieved by present design strategies and the relevance of the practical security approach. Finally, they discussed the impossibility to provide provable security of block ciphers against linear cryptanalysis.

Now, the existing proposed solution is to increase the key length. Our newly designed spread spectrum encryption architecture has a significant security level of an exponential gain above all other existing encryption architectures as the number of subkeys increased or the number of rounds increased or the number of algorithms used is increased. Our proposed solution can mitigate linear and differential cryptanalysis attacks to encryption.

First, we reviewed some quantum algorithms, some quantum applications and the advancements to build QC to know what QC can do and what it cannot do. We found that QC is the same as a classical computer when solving the problem of unpredictability to try all possibilities to find the solution.

This background helped us to develop our new key-dependent spread spectrum encryption architecture.

The **contributions** of the chapter can be summarized as follows:

1- We developed a strong barrier for QC which is the unpredictability to find the right subkeys sequence before starting the cryptanalysis.
2- We developed the spread spectrum encryption architecture family.
3- We developed the first encryption architecture characterized by increasing the security level exponentially with increasing the number of subkeys used.
4- We developed the first encryption architecture characterized by increasing the security level exponentially with increasing the number of algorithm rounds.
5- We developed the first encryption architecture characterized by increasing the security level exponentially with increasing the number of algorithms used.

Comparison between our Work and Previous Works:

(1) There has not been any previous work on symmetric key cipher architectures that had a security level growing exponentially with increasing the number of subkeys used or the number of algorithm rounds or the number of algorithms.

(2) Instead of using one subkey at each round, we used 16 subkeys for SSEA2 at each round and the PRNG chooses one subkey from the 16 subkeys at each round.
(3) SSEA is immune to linear and differential cryptanalysis where each plaintext is encrypted with a different algorithm and different subkeys group and therefore, the subkey and the algorithm are not fixed to help for applying the cryptanalysis process.
(4) SSEA has reduced rounds AES-256 where the possible combinations increase as the number of plaintext increases.

7.2 Preliminaries

Most encryption architectures, i.e. using multiple encryption algorithms, are dependent on a fixed architecture therefore; cryptanalysis can be performed over these architectures. In order to mitigate QC attacks and cryptanalysis attacks, we added the property of unpredictability to the encryption architecture as we designed key-dependent encryption architecture.

7.2.1 Hypothesis of the Design

It seems to be very hard to mitigate QC attacks and linear and differential cryptanalysis. Adding unpredictability to encryption architecture through exploring the capabilities of dynamic encryption approaches [**164**, **165**] for cryptography will help to build strong architecture resistant to QC attacks and cryptanalysis. We started our design by assuming that we will use a high speed encryption algorithm and we need to encrypt a short message and a long message.

SSEA2 has a PRNG to choose one subkey form 16 subkeys at each round. The PRNG needs 4 bits output at each round with a total of 4 multiplied by number of rounds of the encryption algorithm. To encrypt a short message such as 100 plaintext blocks, the attacker needs to guess 100 subkeys combinations out of 16 ^ (number of algorithm rounds) as we choose one subkey from 16 subkeys at each round.

To encrypt a long message such as 100,000 plaintext blocks, the attacker needs to guess 100,000 subkeys combinations out of (16 ^ r), where r is the number of algorithm

rounds as we choose one subkey from 16 subkeys at each round. Since the subkeys are dynamic, the attacker cannot perform linear and differential cryptanalysis.

SSEA1 is concerned with choosing one algorithm out of two algorithms each time to encrypt a plaintext. SSEA3 is concerned with choosing one algorithm out of two algorithms each time to encrypt a plaintext then to choose one subkey out of 16 subkeys at each round of the algorithm then the outputs from the two algorithms are XORed where the input to the second algorithm comes from the RC4 stream cipher algorithm.

With the SSEA dynamic encryption mechanism, the dynamic choosing of subkeys and algorithms protects sensitive data from cryptanalysis, which allows only the original sender and authorized receiver to decode the encrypted data packet via the sequence of secret subkeys that they own. Therefore, this protocol overcomes the weakness of fixed key encryption and protects the wireless network against cryptanalysis attacks.

Also, the attacker cannot obtain the output sequence from the PRNG to analyze it; therefore, the attacker must start the cryptanalysis for all possible combinations of the subkeys groups.

7.2.2 Goals of the Design

In this chapter we have three goals to achieve as follows:

1- **Implementable in both Software and Hardware**: The new spread spectrum encryption architecture needs to be able to work perfectly without any constraints from software or hardware perspectives.

2- **Controlling the Security Level:** The spread spectrum encryption architecture increases the security level each time a subkey or algorithm is added to the system. This is the first encryption architecture which has an exponential security gain by increasing the number of subkeys used at each round or increasing the number of rounds in the encryption algorithm or increasing the number of algorithms.

3- **Prevent attacker from applying chosen plaintext ciphertext attack:** The SSEA can prevent the attacker from applying chosen plaintext ciphertext attack because the attacker does not know the plaintext will go to algorithm one or algorithm two also, the attacker has no clue the plaintext is encrypted with which subkeys group.

4- **High Speed Algorithm:** SSEA3 has 3 rounds AES-256 compared to 14 rounds AES-256.

7.2.3 Dynamic Encryption

Dynamic encryption can be achieved by three main categories which are the followings:

1- Key dependent components,

   Key dependent components mean that at the start of the secure session we fill S-Boxes in the encryption algorithm such as Twofish encryption algorithm.

2- Configuration of encryption components to choose one component from multiple components or to choose one encryption algorithm from multiple algorithms or to choose one component from multiple components,

   Configuration of encryption components mean that at the start of the secure session we choose the used S-Box component from multiple S-Boxes or we choose the encryption algorithm from multiple algorithms such as IPSec and SSL.

3- Reconfiguration of encryption components such as S-Boxes.

   Reconfiguration of encryption components mean that the transmitter and receiver have their encryption algorithm on reconfigurable hardware and the encryption algorithm has S-Boxes to be reconfigured.

We added new category to dynamic encryption which is the spread spectrum encryption architecture.

7.2.4 Unpredictability Principle

When a cryptographer is designing a new cipher, its security level may be difficult to establish. The security is an estimation of how difficult it would be to break the cipher without knowing the secret cipher key. Conventionally, it is assumed that the analysis made by the cryptographer and the cryptanalyst is based upon identical information where the cryptanalyst knows the system being used. A key point that we show in this work, is that this condition is necessary. A cryptanalytic break implies that the cryptanalyst has obtained a part of the secrets of the cipher corresponding to the degree of success. This opens the possibility to challenge this fundamental assumption by introducing a construction that will prevent the cryptanalyst from learning the details of the cipher being used.

We conclude that if we use a cipher that includes a general computational process sequence, and keep all the sequence of computations of that process secret, the cryptanalyst will face a problem which he will be unable to solve.

We found that static encryption systems that are deterministic are susceptible to cryptanalysis but dynamic encryption systems need dynamic cryptanalysis process which is an obstacle to cryptanalysis.

The output controlling sequence from the PRNG to choose the subkeys for each round or the used algorithm is unknown to the attacker. Therefore, this provides the spread spectrum encryption architecture with the unpredictability principle where the subkeys and the used encryption algorithms keep changing for every plaintext block. Unpredictability leads to stop the cryptanalysis.

7.2.5 Adaptive Security

We designed the spread spectrum encryption architecture to deploy the adaptive security concept where SSEA can have three security levels from the three architectures of SSEA.

7.3 Threat Model

There are many factors which work together to compromise the security of the symmetric key cipher systems; these are the cryptanalysis techniques, supercomputer, quantum computer, side channel attacks, grid computing, parallel processing, and the special purpose hardware for cryptanalysis such as the COPACOBANA embedded system [**166**]. Therefore, there are increasing demands to design new encryption architecture that is resistant to all these attacks and cryptanalysis attacks.

We suppose that our system adversary is the QC that implements Grover's algorithm [**158**] to find the used key for every ciphertext block. Also, we suppose that a supercomputer is trying to cryptanalyze our proposed system.

Akihiro Yamamura, and Hirokazu Ishizuka on 2000 discussed the quantum cryptanalysis of block ciphers [**161**]. Their algorithm can be applied to compute non-uniformity of distribution between plaintexts, ciphertexts and secret keys of a block cipher.

In [**162**], Gilles Piret and François-Xavier Standaert discussed on 2009 the distance between the practical security approach and the actual theoretical security provided by a

given cipher. Their experiments illustrated that the provable security against linear cryptanalysis is not achieved by present design strategies and the relevance of the practical security approach. Finally, they discussed the impossibility to provide provable security of block ciphers against linear cryptanalysis.

Therefore, we designed our newly key-dependent architecture such that the greater the number of subkeys flows, the higher the security level will be and the greater the number of rounds, the higher the security level will be and the higher the number of algorithms used, the higher the security level will be.

7.4 Existing Works

Existing security systems in (wire/wireless) communications systems or in computer networks relay on a set of encryption algorithms which are secure until cryptanalysts break them. These existing security schemes are vulnerable to cryptanalysis techniques; therefore, there are high demands to provide a barrier between the encryptor unit and the growing attacks from cryptanalysis. In this chapter, our newly designed spread spectrum encryption architecture will be this barrier that can be adopted in security systems to dynamically change the key schedule through using a PRNG and 16 subkeys instead of one subkey at each round.

Today's Existing Encryption Architectures:

1- Survivable security architecture using multiple encryption algorithms such as IPSec and SSL protocols.
2- Cascaded encryption architecture using two or three encryption algorithms.
3- Compression then encryption architecture.
4- Proactive security architecture through frequently changing the key.
5- Using feedback modes of operations for block cipher encryption algorithms.
6- Key-dependent components architecture such as S-Boxes.
7- Stream cipher controlling block cipher key-schedule architecture [**167**].

All the mentioned encryption architectures have a fixed architecture except the key dependent component architecture. We need to apply dynamic encryption to add the

unpredictability property to the encryption algorithms because static encryption is highly vulnerable to cryptanalysis.

Cryptographic experts recommend increasing the symmetric key cipher systems key length to be 256 bits key length to resist the QC upcoming attacks but cryptanalysis is still applicable for static encryption architectures. Therefore, we believe that we need to start developing new encryption architectures that is resistant to QC attacks and cryptanalysis using the same key length but in a different strategy.

7.5 Overview of SSEA

7.5.1 SSEA Family

If we need a barrier between the encryption algorithm and the cryptanalysis, the SSEA is the perfect barrier. If we want a stronger security guarantee, we need to add unpredictability to the cryptosystem and this is done for SSEA. SSEA is a family of three architectures for symmetric key cipher systems. SSEA1 architecture is concerned with choosing one algorithm from multiple algorithms. SSEA2 architecture is concerned with choosing one subkey from multiple subkeys at each round of the block cipher algorithm. SSEA3 architecture is concerned with choosing one algorithm from multiple algorithms, choosing one subkey from multiple subkeys at each round of the block cipher algorithm and masking the output ciphertext with encrypted stream of bits. The three architectures are dynamic and the third one is the strongest one.

7.5.2 SSEA1 Architecture

7.5.2.1 System Components

1- Two AES-256 Encryption algorithms with 7 rounds.

   We use two AES-256 encryption algorithms with two different S-Boxes to solve the synchronization problem between the two algorithms used. Different S-Boxes ensure different algorithms output with the same key.

2- Key schedule.

There are two keys of 256 bits key length. We choose the key schedule of AES-256 to generate all subkeys of the two AES-256 encryption algorithms and the 256 bits seed for the RC4 stream cipher algorithm.

3- RC4 stream cipher algorithm as PRNG.

We use the RC4 stream cipher algorithm as PRNG for the architecture.

7.5.2.2 Encryption

Figure 7.1 shows the SSEA1 architecture which is composed of two AES-256 encryption algorithms with two different S-Boxes such as S1 and S2. Each algorithm has only 7 rounds not 14 rounds this is because the 7 rounds AES-256 needs 2^{32} chosen plaintext ciphertext pairs to break the 7 rounds [**168**]. Each pair has two possibilities to enter algorithm one or algorithm two and 10 pairs has 2^{10} possible combinations. Therefore, 2^{32} pairs has ($2^{\wedge}2^{32}$) possible combinations, which is infeasible for the attacker to try. The PRNG chooses which algorithm is used to encrypt the plaintext. The sequence of PRNG output is not on the communication channel and this fact is the most glamour property of SSEA1 to prevent the attacker from knowing the sequence of using the encryption algorithms. The plaintext enters all the encryption algorithms to stop side channel attack but we choose the output ciphertext according to the PRNG output which is only known to the receiver. For simplicity, SSEA1 has two encryption algorithms and it can have more than two encryption algorithms.

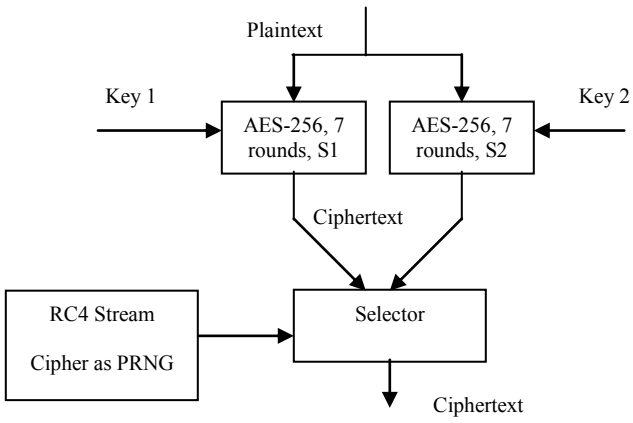


Figure 7.1, SSEA1 Architecture with Two Encryption Algorithms

7.5.2.3 Decryption

The PRNG chooses which algorithm is used to decrypt the ciphertext. The sequence of PRNG output is not on the communication channel to prevent the attacker from knowing the sequence of using the encryption algorithms. The ciphertext enters all the encryption algorithms but we choose the output plaintext according to the PRNG output, which is only known to the receiver. Allowing all encryption algorithms to decrypt will prevent side channel attack.

7.5.2.4 Mathematical Model

For Encryption:

$C_j = \{E_i (P_j)_{Ki}$ under $S_j\}$

The ciphertext is a function of two inputs which are the plaintext and the PRNG output.

C_j is the ciphertext where j = 1 to n and n is the number of plaintexts, E_i is the encryption algorithm and we have two encryption algorithms where i = 1 or 2, P_j is the plaintext, K_i is the key of the encryption algorithm and we have two keys for the two encryption algorithms, S_j is one bit from RC4 stream cipher algorithm as PRNG. S_j selects one algorithm output to be the ciphertext.

For Decryption:

$P_j = \{D_i (C_j)_{Ki}$ under $S_j\}$

D_i is the encryption algorithm and we have two encryption algorithms where i = 1 or 2, S_j selects one algorithm output to be the plaintext.

7.5.2.5 System Analysis

The attacker needs to try all possible combinations to break the system which is (2^P) where P is the number of plaintext blocks. (2^P) is infeasible for the attacker to try if P exceeds 256. This is because the PRNG is not on the communication channel. The algorithm architecture stops linear and differential cryptanalysis because the attacker does not know which algorithm was used to encrypt the plaintext. The attacker needs to try all possible combinations to know the PRNG output sequence.

Finally, SSEA1 with two AES-256 encryption algorithms which have two different S-Boxes is a strong barrier against QC attacks and it has a slightly larger design size than AES-256 because we use only seven rounds of AES-256 and RC4 stream cipher algorithm. SSEA1 has a higher speed than AES-256 and a higher security level. Since the encryption architecture is dynamic, the attacker cannot perform linear and differential cryptanalysis.

7.5.2.6 SSEA1 Advantages

1- The attacker cannot apply known plaintext ciphertext attack or chosen plaintext ciphertext attack to the encryption architecture because the attacker does not know whether the ciphertext came from algorithm one or algorithm two.

2- The attacker cannot apply linear and differential cryptanalysis to the encryption architecture because the attacker does not know whether the ciphertext came from algorithm one or algorithm two.

3- The attacker needs to guess all possible combinations which needs to guess 512 bits key length and to try all possible combinations of using two algorithms and this is impossible.

4- We can use reduced rounds AES-256. Therefore, we use seven rounds AES-256 which needs 2^{32} chosen plaintext ciphertext pairs to break the algorithm with a total of ($2^{2^{32}}$) possible combinations which is infeasible for the attacker to try.

5- SSEA1 has higher speed than AES-256.

6- SSEA1 has higher key length than AES-256 which is 512 bits.

7- SSEA1 Complexity:

The complexity of the system is measured in how many trials the attacker will do to get the right combination of using algorithm one and algorithm two.

For P number of plaintext blocks, the attacker needs to try 2^P trials to get the right combination. If P is larger than 256 then the attacker needs to try 2^256 to know the control sequence of the PRNG.

7.5.2.7 SSEA1 Disadvantages

1- The architecture has larger design size by using RC4 stream cipher algorithm as PRNG and two AES-256 reduced rounds algorithm each of seven rounds.

2- The architecture needs extra synchronization cost to synchronize the two RC4 algorithms at transmitter and receiver.

7.5.2.8 SSEA1 Cryptanalysis

We cannot use AES-128. The attacker can get the 128 bits key from one known ciphertext plaintext pair using QC.

There is no need to use full rounds AES-256. We need 2^{32} known plaintext ciphertext pairs [**168**] to break seven rounds AES-256. These pairs require (2^ 2^{32}) possible combinations which is infeasible for the attacker to try.

7.5.3 SSEA2 Architecture

7.5.3.1 System Components

1- One AES-256 Encryption Algorithm with 7 rounds.

We use AES-256 as the encryption algorithm. We use only seven rounds AES-256. This is because we need 2^{32} chosen plaintext ciphertext pairs to break the seven rounds algorithm with fixed subkeys [**168**]. The subkeys of SSEA2 are not fixed therefore; we can implement only 7 rounds of AES-256. Each round has 16 subkeys and if there are 7 rounds then we have ($16^7 = 2^{28}$) possible subkeys groups for each plaintext.

2- Key schedule.

There is one key of 256 bits key length. The key schedule generates 16 subkeys from the 256 bits key. We choose one subkey from 16 subkeys at each round from the 7 rounds.

The attacker needs to guess ($16^7 = 2^{28}$) subkeys groups' possible combinations to know the sequence of using subkeys. Each plaintext has 2^{28} possible combinations of choosing subkeys groups. The attacker needs to try all possible combinations to break

the system which is (2^{28}^P) where P is the number of plaintext blocks. (2^{28}^P) is infeasible for the attacker to try if P exceeds 9.

The key schedule of AES-256 generates 256 bits seed for the RC4 stream cipher algorithm.

3- RC4 stream cipher algorithm as PRNG.

We use the RC4 stream cipher algorithm as the PRNG at each round to choose one subkey from 16 subkeys.

7.5.3.2 Encryption

Figure 7.2 shows the SSEA2 architecture. The PRNG chooses which subkey is used to encrypt the plaintext. The sequence of PRNG output is not on the communication channel and this fact is the most glamour property of SSEA2 to prevent the attacker from knowing the sequence of using the subkeys. For SSEA2, to encrypt a short message such as 1 plaintext block, the attacker needs to try 2^{28} possible combinations to guess the right subkeys. Also, to encrypt a short message such as 100 plaintext blocks, the attacker needs to try (2^{28} ^ 100) possible combinations to guess the right subkeys. Also, to encrypt a long message such as 100,000 plaintext blocks, the attacker needs to try (2^{28} ^ 100,000) possible combinations to guess the right subkeys.

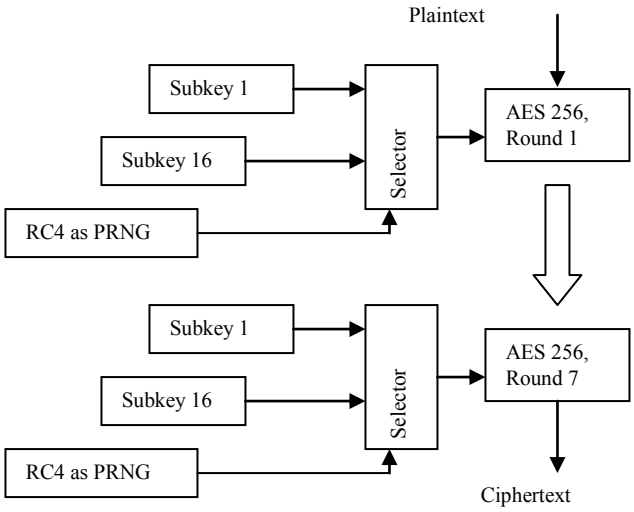


Figure 7.2, SSEA2 Encryption Architecture

7.5.3.3 Decryption

The PRNGs choose which subkey is used to decrypt the ciphertext. The sequence of PRNG output is not on the communication channel to prevent the attacker from knowing the sequence of using the subkeys.

7.5.3.4 Mathematical Model

For Encryption:

$C_j = \{E\ (P_j)_{Ki}$ under $S_j\}$

The ciphertext is a function of three inputs which are the plaintext, subkeys groups and the PRNG output.

C_j is the ciphertext where j = 1 to n and n is the number of plaintexts, E is the encryption algorithm and we have only one encryption algorithm, P_j is the plaintext, K_i is the subkeys generated for the encryption algorithm and we have ($16^7 = 2^{28}$) subkeys groups for 16 subkeys at each round of seven rounds, S_j is four bits from RC4 stream cipher algorithm as PRNG. S_j selects one subkey from 16 subkeys at each round of 7 rounds.

For Decryption:

$P_j = \{D\ (C_j)_{Ki}$ under $S_j\}$

D is the encryption algorithm and we have only one encryption algorithm, S_j selects one subkey from 16 subkeys at each round of 7 rounds.

7.5.3.5 System Analysis

The attacker needs to try all possible combinations of the PRNG for each ciphertext pair to break the system. This is because the PRNG is not on the communication channel. The algorithm architecture stops the linear and differential cryptanalysis because the attacker does not know which subkeys were used to encrypt the plaintext. The attacker needs to try all possible combinations to know the output sequence of the PRNG.

7.5.3.6 SSEA2 Advantages

1- Subkeys are not fixed as they are dynamic.

2- The attacker cannot apply known plaintext ciphertext attack or chosen plaintext ciphertext attack to the encryption architecture because the attacker does not know which subkeys were used to encrypt the plaintext.

3- The attacker cannot apply linear and differential cryptanalysis techniques to the encryption architecture because the attacker does not know the ciphertext is encrypted with which subkeys groups.

4- The attack needs to guess 256 bits key length which is impossible then the attacker needs to try all possible combinations of subkeys groups which is (2^{28} ^ P) where P is number of plaintext blocks.

5- We can use reduced rounds AES-256 with 7 rounds. Therefore, we use 7 rounds AES-256 which needs 2^{32} chosen plaintext ciphertext pairs to break the 7 rounds which needs (2^{28} ^ 2^{32}) possible combinations which is infeasible for the attacker to try.

6- SSEA2 has higher speed than AES-256.

7- The encryption design size is lower than AES-256 where we have AES-256 reduced rounds algorithm with 7 rounds and RC4 stream cipher algorithm.

8- SSEA2 Complexity:

The complexity of the system is measured by how many trials the attacker will do to get the right subkeys at each algorithm.

For one plaintext block, 16 subkeys at each round and seven rounds for the algorithm, the attacker needs to try 16^7 trials to get the right combination.

For P number of plaintext blocks and seven rounds for the algorithm, the attacker needs to try (16^7)^P trials to get the right combination. If P is larger than ten than the attacker needs to try 2^256 to know the control sequence of the PRNG.

7.5.3.7 SSEA2 Disadvantages

1- The architecture needs extra synchronization cost to synchronize the two RC4 algorithms at transmitter and receiver.

2- The architecture needs the 16 subkeys at each round to choose one subkey, which is extra cost for hardware.

7.5.3.8 SSEA2 Cryptanalysis

We cannot use AES-128. The attacker can get the 128 bits key from one known ciphertext plaintext pair using QC.

There is no need to use full rounds AES-256. We need Seven rounds AES-256 which need 2^{32} known plaintext ciphertext pairs [**168**] to break the seven rounds AES-256. These pairs require (2^{28} ^ 2^{32}) possible combinations which is infeasible for the attacker to try.

7.5.4 SSEA3 Architecture:

7.5.4.1 System Components

1- Two AES-256 encryption algorithms with 3 rounds.

We use two AES-256 encryption algorithms with two different S-Boxes to solve the synchronization problem between the two algorithms used. Each algorithm has only 3 rounds of AES-256. Different S-Boxes ensure different algorithms output with same key. The encryption algorithm will keep changing from algorithm one to algorithm two. Each round has 16 subkeys of the 3 rounds. The subkeys are not fixed. The attacker needs to guess ($16^3 = 2^{12}$) subkeys groups' possible combinations to know the sequence of the used subkeys. Each plaintext has (2^{12}) possible combinations of choosing subkeys groups.

Reasons to choose 3 rounds AES-256 for SSEA3:

- One round AES-256 does not achieve unpredictability at subkeys level.
- Two rounds AES-256 do not achieve unpredictability at subkeys level.
- Three rounds AES-256 achieve unpredictability at subkeys level, output level and algorithm level.

- We use double encryption. The ciphertext from the plaintext is encrypted with stream of bits comes from the second algorithm while the input to the second algorithm only known to the receiver and it is not known to the attacker.
- The subkeys are dynamic and they are changing for every plaintext with (2^{12}) possible combinations.
- The architecture is dynamic where the algorithm that encrypts the plaintext is not fixed as we use two encryption algorithms to encrypt the plaintext.
- The attacker cannot perform known plaintext ciphertext attack since the ciphertext is encrypted with unknown input to the attacker.
- The attacker cannot perform man in the middle attack over the ciphertext because the ciphertext is encrypted with unknown input to the attacker.

2- Key schedule.

There are two keys of 256 bits key length. We choose the key schedule of AES-256 to generate 16 subkeys at each round for the 3 rounds of each algorithm. The key schedule of AES-256 generates the 256 bits seed for the RC4 stream cipher algorithm.

3- RC4 stream cipher algorithm as PRNG.

We use RC4 stream cipher algorithm as the PRNG to choose one subkey of the 16 subkeys at each round. The PRNG chooses where the plaintext goes to algorithm 1 or algorithm 2. The output from RC4 stream cipher algorithm is used to enter the algorithm that is not used by the plaintext. The outputs from the two encryption algorithms are XORed.

7.5.4.2 Encryption

Figure 7.3 shows the SSEA3 architecture with two AES-256 encryption algorithms and two session keys for each algorithm. The RC4 stream cipher algorithm chooses which subkey is used to encrypt the plaintext and the RC4 stream cipher algorithm chooses which algorithm will encrypt the plaintext and we marked it as output2. The sequence of PRNG output is not on the communication channel and this fact is the most glamour property of SSEA3 to prevent the attacker from knowing the sequence of using the

subkeys or the sequence of using the encryption algorithms. The output from the RC4 algorithm is encrypted with the second algorithm and we marked it as output1. The output from the two algorithms is XORed and we marked it as output3. For 3 rounds AES-256, we need $((2^{12} \text{ X } 2^{12} \text{ X } 2 \text{ X } 2^{128})^{\wedge}P)$ possible combinations to break the 3 rounds of the algorithm where P is the number of plaintext blocks. The ciphertext is encrypted therefore, the attacker cannot apply known plaintext ciphertext attack and for this reason we use only 3 rounds AES-256.

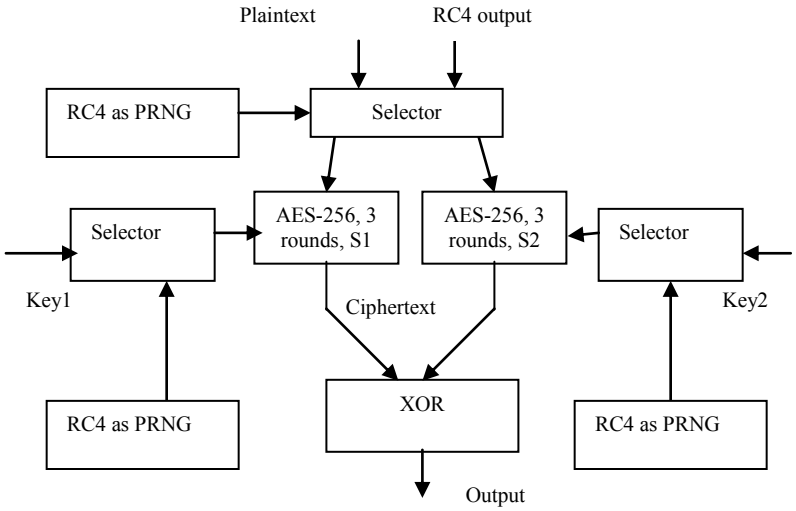


Figure 7.3, SSEA3 Encryption Architecture

7.5.4.3 Decryption

First, we decrypt the output1 from the PRNG and marked it as input1. Second, we perform XOR on the input1 with the output3 from the two encryption algorithms to get the ciphertext of the plaintext which is input2. Third, we decrypt the ciphertext which is input2 with the encryption algorithm that is not used by the RC4 stream cipher algorithm to get the plaintext. The RC4 as PRNG chooses which subkey is used to decrypt the ciphertext. The RC4 as PRNG chooses which algorithm is used to decrypt the ciphertext and which algorithm is used to decrypt the output from the RC4 algorithm. The sequence of PRNG output is not on the communication channel to prevent the attacker from knowing the sequence of using the subkeys or algorithms. The decryption algorithm is

double size the encryption algorithm to allow the decryption speed to be the same as the encryption speed.

7.5.4.4 Mathematical Model

For Encryption:

$C_j = \{E_i (P_j)_{Ki}$ XOR $E_i (RC_j)_{Ki}$ under $S_j\}$

The ciphertext is a function of three inputs which are the plaintext, subkeys groups and the PRNG output.

C_j is the ciphertext where j = 1 to n and n is the number of plaintexts, E_i is the encryption algorithm and we have two encryption algorithms where i = 1 or 2, P_j is the plaintext, K_i is the subkeys generated for the encryption algorithm and we have ($16^3 = 2^{12}$) subkeys groups for 16 subkeys at each round of 3 rounds, S_j is 141 bits from RC4 stream cipher algorithm to encrypt one plaintext block where we need 4 bits to choose one subkey out of 16 subkeys with a total of 12 bits for 3 rounds and one bit to choose one algorithm to encrypt the plaintext and 128 bits to enter the second encryption algorithm. S_j selects one subkey from 16 subkeys at each round of 3 rounds and selects one algorithm to encrypt the plaintext while the other algorithm is used to encrypt 128 bits output from RC4 stream cipher algorithm. RC_j is the 128 bits output from RC4 stream cipher algorithm.

For Decryption:

$RC_j = D_i$ (encrypted $RC_j)_{Ki}$ under S_j

$P_j = \{D_i (RC_j$ XOR $C_j)_{Ki}$ under $S_j\}$

D_i is the encryption algorithm and we have two encryption algorithms, S_j selects one subkey from 16 subkeys at each round of 3 rounds and selects one algorithm to decrypt the plaintext while the other algorithm is used to decrypt the 128 bits output from RC4 stream cipher algorithm.

7.5.4.5 System Analysis

The attacker needs to try all possible combinations of the PRNG for each ciphertext pair and for choosing subkeys groups to break the system. This is because the PRNG is not on the communication channel. The algorithm architecture stops the linear and differential cryptanalysis because the attacker does not know which subkey was used to encrypt the

plaintext. The attacker does not know the 128 bits output from RC4 that enters the second algorithm. The attacker needs to try all possible combinations to know the output sequence of the PRNG. The attacker does not know which algorithm the plaintext went to. This architecture has much higher speed than AES-256 as it has only 3 rounds. Finally, SSEA3 with AES-256 is strong barrier against QC attacks with higher speed than AES-256 full rounds. The ciphertext is encrypted to prevent linear and differential cryptanalysis.

7.5.4.6 SSEA3 Advantages

1- Double encryption.

2- Subkeys are not fixed as they are dynamic.

3- The attacker cannot apply known plaintext ciphertext attack or chosen plaintext ciphertext attack to the encryption architecture because the attacker does not know whether the ciphertext comes from algorithm one or algorithm two and the attacker does not know which subkeys group is used to encrypt the plaintext out of 2^{12} subkeys groups.

4- The attacker cannot apply linear and differential cryptanalysis techniques to the encryption architecture because the attacker does not know whether the ciphertext comes from algorithm one or algorithm two and the attacker does not know which subkeys group is used to encrypt the plaintext out of (2^{12} X 2^{12}) subkeys groups.

5- The attacker needs to guess 512 bits key length and to try all possible combinations of using two algorithms and to try to determine which subkeys group is used out of (2^{12} X 2^{12}) subkeys groups and this is impossible.

6- We can use reduced rounds AES-256. Therefore, we use 3 rounds AES-256. We use 3 rounds because the ciphertext is encrypted and the attacker cannot perform known plaintext ciphertext attacker over SSEA3. SSEA3 needs ((2^{12}X2^{12}X2X 2^{128}) ^ P) possible combinations where P is the number of plaintext blocks which is infeasible for the attacker to try by.

7- The algorithm has higher speed than AES-256.

8- The encryption design size is lower than AES-256 where we have two AES-256 encryption algorithms reduced rounds with 3 rounds.

9- The architecture has higher key length than AES-256 which is 512 bits.

10- SSEA3 Complexity:

The complexity of the system is measured in how many trials the attacker will do to get the right RC4 input to the second algorithm. For one plaintext block, the attacker needs to try 2^128 trials to get the right combination.

For P number of plaintext blocks, the attacker needs to try (2^128)^P trials to get the right combination. If P is larger than one then the attacker needs to try 2^256 to know the control sequence of the PRNG.

7.5.4.7 SSEA3 Disadvantages

1- The architecture needs extra synchronization cost to synchronize the two RC4 algorithms at transmitter and receiver.

2- The architecture needs the 16 subkeys at each round to choose one subkey which comes at extra cost for hardware.

3- The decryption design size is double the encryption design size to allow the decryption speed to be the same as encryption speed but the decryption size is still less than AES-256 full rounds as it has only 12 rounds AES-256.

7.5.4.8 SSEA3 Cryptanalysis

We cannot use AES-128. The attacker can get the 128 bits key using QC.

There is no need to use full rounds AES-256. We use 3 rounds AES-256 which require $((2^{12}X2^{12}X2X2^{128})^P)$ possible combinations where P is the number of plaintext blocks which is infeasible for the attacker to try.

We choose to implement SSEA3 because it has the lowest design size, the highest speed and the highest security level.

7.5.5 AES-256 Components

7.5.5.1 AES-256 Block Cipher Encryption Algorithm.

The AES is a substitution permutation network (SPN) allowing the encryption/ decryption of data by blocks of 128-bits and supporting key lengths of 128, 192 and 256 bits. In the following, we focus on the 256-bits key version. Its internal state, usually represented as a 4×4 matrix of bytes, is updated by iterating through the round structure (10, 12 or 14 times according to the key size whether 128 or 192 or 256 bits respectively). The round is described as four different byte-oriented transformations [**169**].

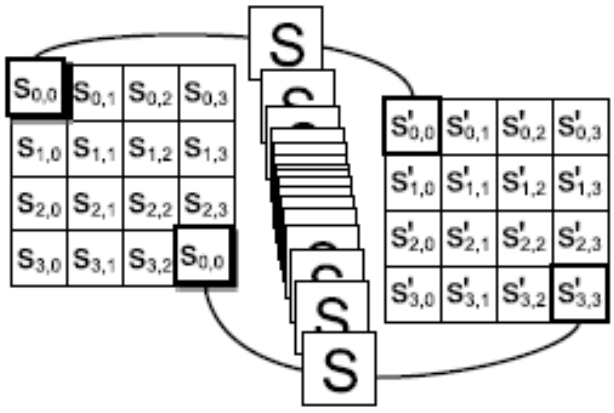


Figure 7.4, BytesSub Transformation [169]

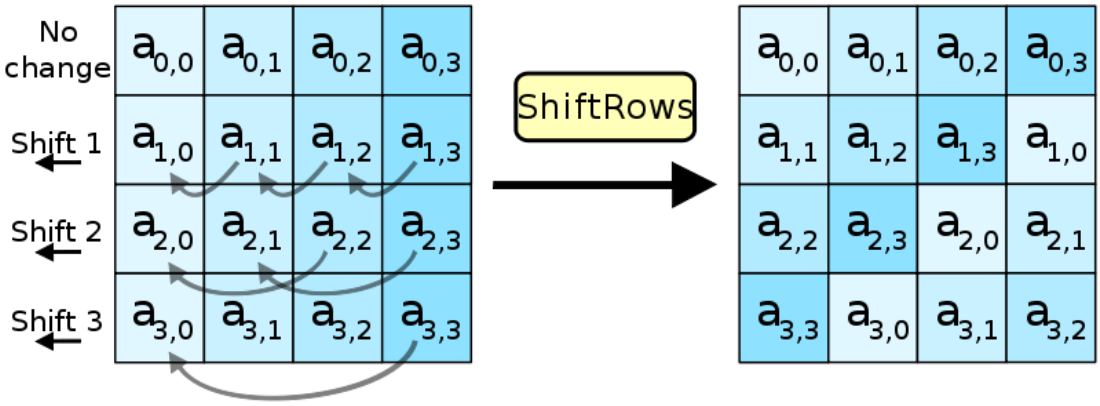


Figure 7.5, ShiftRows Transformation [169]

First, BytesSub introduces the non-linearity by taking, for each byte, the modular inverse in GF(2^8) and then applying an affine transformation. Instead of computing these two

steps distinctly, the full transformation is achieved by passing each byte through an S-Box. We use two different S-Boxes for the two AES-256 encryption algorithms of SSEA1 and SSEA3. ByteSub is shown in Figure 7.4.

Second, ShiftRows modifies the state. It simply consists of a circular left shift of the state's rows by 0, 1, 2 and 3 bytes respectively. ShiftRows is shown in Figure 7.5.

Third, MixColumns applies a linear transformation to the state's columns. Each of them is regarded as a polynomial and is multiplied by a fixed polynomial

$c(x) = 3x^3 + x^2 + x + 2 \pmod{x^4 + 1}$. MixColumns is shown in Figure 7.6.

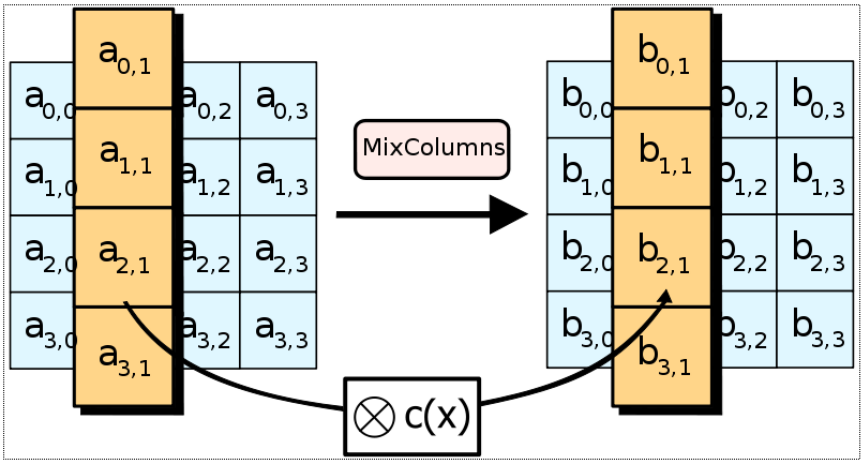


Figure 7.6, MixColumns Transformation [169]

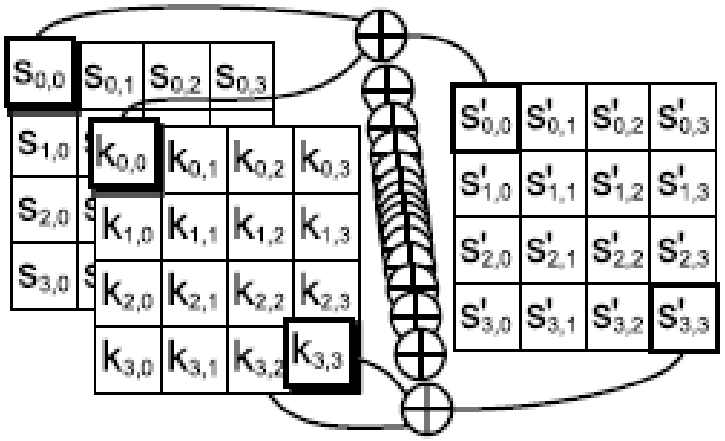


Figure 7.7, AddRoundKey Transform [169]

Finally, the AddRoundKey transform mixes the key with the state. As each subkey has the same size as the state, the combination is performed by a simple bitwise XOR between subkey bytes and their corresponding state bytes as shown in Figure 7. A first key addition is performed before entering the first round, and the last round omits the MixColumns transformation. AddRoundKey is shown in Figure 7.7.

7.5.5.2 Block Cipher Key Schedule.

Prior to the encryption/decryption process, the subkeys have to be generated. The key schedule takes the main key K_0 and expand it for the case of a 256-bit key, where SubWord applies the S-Box to the 32-bit input word, RotWord rotates the word one byte to the left and RC(i) is an 8-bit constant associated to each round i. Key schedule is shown in Figure 7.8.

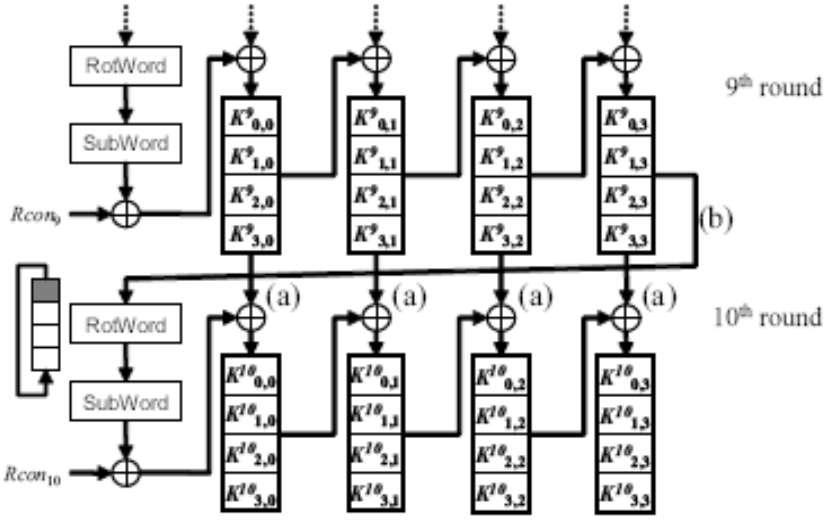


Figure 7.8, AES 256-bit Key Expansion of Two Rounds [169]

7.6 SSEA3 Proof of Security

Our designed SSEA3 applied Kerckhoffs' Principle which stated that "A cipher should be secure when the cryptanalyst knows all details of the enciphering process and deciphering process except the value of the secret key". When evaluating the security of our designed

architecture according to Kerckhoffs' Principle, we found that the cryptanalyst knows everything about the encryption algorithms and the PRNG generates the controlling sequence, except the algorithms secret keys and the PRNG controlling sequence.

Shannon distinguished between two types of security:

- Unconditionally secure - means security against an enemy who has unlimited time and computational resources.
- Computationally secure - means security against an enemy who has a specified limited amount of time and computational resources.

Definition 1:

Let S be the output controlling sequence of PRNG, let A_1 and A_2 be the used block cipher algorithms with n rounds which uses at least 16 subkeys at each round, let L_1 to L_{16} be 16 subkeys of the key, let the controlling sequence at each round from the encryption algorithm chooses one subkey from the 16 subkeys, let the controlling sequence choose which algorithm is used to encrypt the plaintext and let the controlling sequence enters the second algorithm to mask the ciphertext by XORing the ciphertext with the output from the second algorithm. Every plaintext block is encrypted with a different group of n subkeys. Since the output of the PRNG is not on the communication channel and the attacker cannot analyze it therefore,

The PRNG uses a seed to generate the control sequence. The receiver must use the same PRNG with the same seed to generate the same sequence to be able to decrypt the ciphertext. This architecture is a strong barrier for cryptanalysts to break. Therefore, SSEA3 can be used for Post-Quantum Computing to resist QC attacks.

Definition 2:

We can define the computational security as follows [**170**]:

Let $(E;D)$ be an encryption scheme that uses n-bit keys to encrypt $\ell(n)$-length messages.

(E;D) is computationally secure if for every polynomial-time algorithm A: $\{0,1\}^* \rightarrow \{1,0\}$, polynomially bounded ε: $\{0,1\}^* \rightarrow [1,0]$, n, and x_0, $x_1 \in \{0,1\}^{\ell(n)}$, $\left| Pr[A(E_{Un}(x_0)) = 1] - Pr[A(E_{Un}(x_1)) = 1] \right| < \varepsilon(n)$.

Traditional cryptosystem is five tuples (P, C, K, E, D), where P is the plaintext, C is the ciphertext, K is the key space, E is the encryption algorithm, and D is the decryption algorithm.

SSEA is six tuples (P, C, K, E, D, R), where P is the plaintext, C is the ciphertext, K is the key space, E is the encryption algorithm, D is the decryption algorithm, and R is the PRNG seeding.

Theorem 1:

SSEA3 is immune to linear cryptanalysis, differential cryptanalysis and algebraic attacks.

Proof of Theorem 1:

1- Linear Cryptanalysis:

Linear cryptanalysis tries to take advantage of high probability occurrences of linear expressions involving plaintext bits, "ciphertext" bits, and subkey bits.

The basic idea is to approximate the operation of a portion of the cipher with an expression that is linear where the linearity refers to a mod-2 bit-wise operation. Such an expression is of the form:

$$x_1 \oplus x_2 \oplus x_3 \oplus \ldots\ldots \oplus x_n \oplus y_1 \oplus y_2 \oplus y_3 \oplus \ldots\ldots \oplus y_n \equiv 0$$

Where Xi represents the i-th bit of the input $X = [X1, X2, ...]$ and Yj represents the j-th bit of the output $Y = [Y1, Y2, ...]$. This equation is representing the exclusive-OR "sum" of u input bits and v output bits.

The approach in linear cryptanalysis is to determine expressions of the form above which have a high or low probability of occurrence. (No obvious linearity such as above should hold for all input and output values or the cipher would be trivially weak.) If a cipher displays a tendency for equation (1) to hold with high probability or not hold with high probability, this is evidence of the cipher's poor randomization abilities. Consider that if we randomly selected values for $u + v$ bits and placed them into the equation above, the probability that the expression would hold would be exactly 1/2. It is the deviation or bias from the probability of 1/2 for an expression to hold that is exploited in linear

cryptanalysis: the further away that a linear expression is from holding with a probability of 1/2, the better the cryptanalyst is able to apply linear cryptanalysis.

Equation (1) could be equivalently reformulated to have the right side being the sum of a number of subkey bits. However, in (1) as written with the right side of "0", the equation implicitly has subkey bits involved: these bits are fixed but unknown (as they are determined by the key under attack) and implicitly absorbed into the "0" on the right side of equation (1) and the probability p_L that the linear expression holds. If the sum of the involved subkey bits is "0", the bias of (1) will have the same sign (+ or □) as the bias of the expression involving the subkey sum and, if the sum of the involved subkey bits is "1", the bias of (1) will have the opposite sign.

Discussion: Linear cryptanalysis is based on the fact that the algorithm is fixed and the subkeys are fixed where subkeys and the algorithm in SSEA3 keep changing for every plaintext block; therefore, linear cryptanalysis is not applicable.

2- Differential Cryptanalysis:

Differential cryptanalysis exploits the high probability of certain occurrences of plaintext differences and differences into the last round of the cipher. For example, consider a system with input $X = [X1\ X2\ ...\ Xn]$ and output $Y = [Y1\ Y2\ ...\ Yn]$.

In an ideally randomizing cipher, the probability that a particular output difference Δy occurs given a particular input difference Δx is $1/2^n$ where n is the number of bits of x.

Differential cryptanalysis seeks to exploit a scenario where a particular output difference Δy occurs given a particular input difference Δx with a very high probability p_D (i.e., much greater than $1/2^n$). The pair $(\Delta x, \Delta y)$ is referred to as a *differential*.

Let two inputs to the system be x' and x'' with the corresponding outputs y' and y'', respectively. The input difference is given by $\Delta x = x' \oplus x''$ where "$\oplus$" represents a bit-wise exclusive-OR of the

n-bit vectors and, hence,

$\Delta x = [\Delta x_1 \Delta x_2 \ldots\ldots \Delta x_n]$

Where $\Delta x_i = x_i' \oplus x_i''$ with x_i' and x_i'' representing the *i*-th bit of x' and x'', respectively.

Similarly, $\Delta y = y' \oplus y''$ is the output difference and

$\Delta y = [\Delta y_1 \Delta y_2 \ldots\ldots \Delta y_n]$

Where $\Delta y_i = y_i' \oplus y_i''$.

Differential cryptanalysis is a chosen plaintext attack, meaning that the attacker is able to select inputs and examine outputs in an attempt to derive the key. For differential cryptanalysis, the attacker will select pairs of inputs, x' and x'', to satisfy a particular Δx, knowing that for that Δx value, a particular Δy value occurs with high probability.

We investigate the construction of a differential $(\Delta x, \Delta y)$ involving plaintext bits as represented by x and the input to the last round of the cipher as represented by y. We shall do this by examining high likely *differential characteristics* where a differential characteristic is a sequence of input and output differences to the rounds so that the output difference from one round corresponds to the input difference for the next round. Using the highly likely differential characteristic gives us the opportunity to exploit information coming into the last round of the cipher to derive bits from the last layer of subkeys.

As with linear cryptanalysis, to construct highly likely differential characteristics, we examine the properties of individual S-boxes and use these properties to determine the complete differential characteristic. Specifically, we consider the input and output differences of the S-boxes in order to determine a high probability difference pair.

Combining S-box difference pairs from round to round so that the nonzero output difference bits from one round correspond to the non-zero input difference bits of the next round, enables us to find a high probability differential consisting of the plaintext difference and the difference of the input to the last round. The subkey bits of the cipher end up disappearing from the difference expression because they are involved in both data sets and, hence, considering their influence on the difference involves exclusive-ORing subkey bits with themselves, the result of which is zero.

Discussion: Differential cryptanalysis is based on the fact that the algorithm is fixed and the subkeys are fixed where subkeys and the algorithm in SSEA3 keep changing for every plaintext block; therefore, differential cryptanalysis is not applicable.

3- Algebraic Attacks:

Typically, an algebraic attack consists of two steps.

1. Collecting step: The cryptanalyst expresses the cipher as a set of simple equations in a number of variables. These variables include bits (or bytes) from the plaintext, ciphertext and the key, and typically also of intermediate computation values and round keys.
2. Solving step: the cryptanalyst uses some data input such as plaintext ciphertext pairs, substitutes these values in the corresponding variables in the set of equations collected in step 1 and tries to solve the resulting set of equations, thereby recovering the key.

It does not come as a big surprise that SSEA3 can be expressed with elegant equations in several ways. Whereas in many other cipher designs the structure is obscured by the addition of many complex operations, in SSEA3 the inner structure is very simple and transparent, clearly facilitating the expression of the cipher as a set of simple equations. The key issue to be judged however is whether equations that look elegant to the mathematician's mind are also simple to solve.

The algebraic equation of SSEA3 has 768 unknown bits of the key which is impossible to solve by equations of plaintext and ciphertext pairs.

Discussion: Algebraic cryptanalysis is based on the fact that the algorithm is fixed and the subkeys are fixed where subkeys and the algorithm in SSEA3 keep changing for every plaintext block; therefore, algebraic attack is not applicable.

7.7 SSEA3 Attacks

We describe different attacks against SSEA3. It is secure against the following attacks:

7.7.1 Attack the PRNG

The adversary has no advantage to learn anything from the PRNG output since its output is not on the communication channel.

7.7.2 Attack the Key Schedule

The attacker cannot apply the related key attack for the SSEA3 because the attacker does not know whether the plaintext is encrypted with algorithm one or algorithm two. Also, the attacker does not know the plaintext is encrypted with which subkeys group. Related key attack contradicts with the design principle of SSEA3 which states that each plaintext chooses one subkey group from 16 subkeys at each round of the encryption algorithm.

7.7.3 Attack Encryption Algorithm using Linear and Differential Cryptanalysis

The linear and differential cryptanalysis assumes that the key is fixed for the encryption process which is not the case for SSEA3. Every plaintext is encrypted with different algorithm and different subkeys group. Therefore, the cryptanalyst cannot apply the linear and differential cryptanalysis over SSEA3.

7.7.4 Quantum Computer Attacks

The quantum computer can perform cryptanalysis on every ciphertext block using Grover's algorithm where the subkeys are fixed which is not the case for SSEA3 where the subkeys are dynamic and the encryption algorithm is dynamic. Key length is 512 bits to stop Grover's quantum algorithm attack.

7.7.5 Supercomputer Attacks

The current fastest supercomputer system is the K computer which is ranked on the TOP500 list as the fastest supercomputer at 8.16 peta FLOPS. It consists of 68,544 SPARC64 VIIIfx CPUs. The system entered service in November 2012 with 864

cabinets. It currently uses 68,544 2.0 GHz 8-core SPARC64 VIIIfx processors for a total of 548,352 cores [**171**].

A supercomputer can perform no more than guessing the sequence of the output controlling sequence of SSEA3. Therefore, supercomputer cannot break the SSEA3.

7.7.6 Attack on Synchronization

If SSEA3 is under miss synchronization attack, the SSEA3 will start with new keys and new seeding for PRNG. The transmitter and receiver must initialize with the same seeding and keys using preamble at the beginning of the secure session.

7.8 Comparison between SSEA3 and Standard AES-256 Block Cipher

Table 7.1, Comparison between AES-256 and SSEA3

No.	Property	AES-256	SSEA3
1	Speed	Speed of 14 Rounds	Speed of 3Rounds
2	Security Level	256 bits Key Length	512 bits key Length
3	No. of Algorithms	One	Two
4	Key Length	256 bits	512 bits
5	Design Size	14 Rounds for encryption and 14 Rounds for Decryption	3 Rounds for each algorithm and RC4 Algorithm and Double Size for Decryption
6	No. of Rounds	14	3
7	RC4 as PRNG	No	Yes
8	Key Schedule	AES Key Schedule	AES Key Schedule
9	No. of Subkeys at each round	1	16
10	Gain	No	Exponential Gain when increasing number of rounds or number of algorithms or number of subkeys at each round

11	Complexity	256 bits	512 bits and $((2^{12}X2^{12}X2X2^{128}) \wedge P)$ possible combinations, P is number of Plaintext blocks
12	Cryptanalysis	Yes	Yes
13	Side Channel Attacks	Yes	No
14	Brute Force Attack	Infeasible for attacker	Infeasible for attacker
15	Synchronization	Simple	Hard
16	QC Attacks	No Grover attack	No Grover attack

7.9 SSEA3 Limitations

Larger Key Length:

SSEA3 uses two keys instead of one key in each session for the two encryption algorithms.

Larger Time for Synchronization:

SSEA3 needs longer time for synchronization than the original AES-256 to setup encryption keys from two keys and decryption keys to synchronize the transmitter and receiver. This time is equal to double times of synchronization for AES-256 with one key.

7.10 Summary

According to the characteristics of SSEA and its advantages, we could conclude that SSEA family of architectures can resist QC attacks and linear and differential cryptanalysis attacks. We choose to use SSEA3 as its security level is higher than the security level of SSEA1 and SSEA2 and SSEA1 and SSEA2 have the same speed but SSEA3 has a higher speed since it has only 3 rounds. SSEA3 decryption needs double

size the encryption design size to allow the encryption speed to be same as the decryption speed.

SSEA security level has exponential gain as the number of rounds increased or the number of subkeys at each round increased or the number of algorithms increased. SSEA is the first encryption algorithm that used the unpredictability principle to add PRNG to the encryption design to hide which algorithm is used to encrypt the data or which subkey is used at each round or to mask the output ciphertext with the encrypted bits stream from RC4 stream cipher algorithm.

The results prove that: the architecture with the advantages of high speed and high security level can be implemented for post-quantum cryptography. The SSEA architecture is a strong barrier for cryptanalysis. Besides, each plaintext block is encrypted with a different algorithm and different subkeys group which is an obstacle for cryptanalysis.

In this chapter, we proposed a new encryption architecture which is called the spread spectrum encryption architecture. This encryption architecture is based on the unpredictability principle where we choose one subkey from 16 subkeys at each round and two algorithms encrypt the plaintext blocks. Our newly designed

SSEA model is easily implemented in both software and hardware. This new encryption architecture will be an essential architecture to the field of post-quantum cryptography.

CHAPTER 8

Hardware Implementation of Reliable Network Recovery from Base Station Failure

---

In this chapter, the design and hardware implementation of the first component of SurvSec security architecture, which is the reliable network recovery from base station failure, is presented. The design and implementation of reliable network recovery from base station failure was implemented on Arduino Uno microcontroller boards, the transceivers used are X-Bee modules 1 mw series 1 and the motion detection sensor used is the X-Band Doppler radar motion detection sensor. The X-Bee transceiver cannot be connected directly to the Arduino Uno microcontroller board; therefore an X-Bee shield card is used to connect between the X-Bee transceiver and the Arduino Uno board. The motion detection sensor is connected to the Arduino Uno board. AES encryption algorithm is implemented on Arduino Uno microcontroller board to encrypt the security reports that are sent from the sensor nodes to the security manager and from the security managers to the new base station. The reliable network recovery from base station failure is hardware implemented to show its validity in real time implementation. Power consumption of the received security report was measured to show that the reliable network recovery from base station failure has low power consumption.

8.1 Introduction

In this chapter, an introduction to hardware implementation for reliable network recovery from base station failure is introduced. Section 8.2 presents the related work, the requirements for the hardware implementation of reliable network recovery from base station failure, the proposed system components, the specifications of the proposed system components and the theory of operations for the proposed system components. Section 8.3 presents the design and implementation of the proposed system. Section 8.4 presents the results and the evaluation metrics. Section 8.5 presents the comparison between our work and previous works. Finally, section 8.6 presents the summary. Appendix A presents the code of the transmitter, receiver, and AES encryption algorithm.

To the best of our knowledge, there is no scheme in the open literature which addresses base station failure. The current security schemes proposed for wireless sensor networks lack the ability of providing reliable network recovery in the case of base station failure. This challenge is quite serious in case of mission critical deployments such as deployments of surveillance wireless sensor networks in hostile environment. In hostile environments, the probability of base station failure is high since, as a single point of failure, it is the target for the adversary. Also, the time and efforts required to destroy a base station is considerably less compared to what is needed to neutralize the actual WSN. **The objectives of the hardware implementation** are first, the low power consumption of transmitted and received security report which is measured and with calculation the battery can work for one year and second, the validity of transmitter and receiver code for reliable network recovery from base station failure.

The threats in the security report content are realized as follows:

1- The code of the attacks and threats are saved in a table and each attack has a unique code in the table.
2- When the node discovers an attack, it tabulates this attack according to the table.
3- We assume that each node has a Local Intrusion Detection (LID) software to detect the attacks.
4- When the attack is realized, the node sends the report to the security manager

This section describes the steps for the hardware implementation as follows:

1- Hardware design of reliable network recovery from base station failure is introduced.
2- Programming of the X-Bee Series 1 Transceivers [203] is done using the X-Bee programmer board.
3- Writing down the code of the transmitter and the code of the receiver on the Arduino Uno microcontroller boards is done.
4- Writing down the code of the AES encryption algorithm on Arduino Uno microcontroller boards is done. The AES is used to encrypt and decrypt the security reports sent from the sensor nodes to the security managers (SMs) and to encrypt and decrypt the sent security reports sent from the security managers to the new BS.
5- Writing down the code of the motion detection sensor is done.
6- Hardware implementation for reliable network recovery from base station failure is done using Arduino Uno microcontroller boards [204], X-Bee shields, X-Bee transceivers, and X-Band Motion detection sensors [205].
7- Debugging the errors on the code is done to achieve the correct code for hardware implementation of reliable network recovery from base station failure.
8- Hardware simulation of the proposed system on Arduino Uno simulator is done.
9- Measurements of the passing current on Arduino Uno board and the power consumption of the sent security report is done.
10- Serial monitor software such as HyperTerminal is used to monitor the data transmitted and received.

To the best of our knowledge, there is no contribution in the open literature addressing the situation a user has to deal with from the time the BS fails to the time the base station is operational again. Also, we have not found any research explaining how the new BS can verify the trustworthiness of the existing sensor nodes. By lacking the ability to verify the trustworthiness of the existing sensor nodes, a user has no choice but to “scrape” the existing deployment and proceed with a new one, despite the deficiencies associated with this choice (e.g. high cost and long duration of having unreliable coverage of the deployed WSN).

This work is addressing this important issue and strives to provide practical answers to this challenging problem. Based on this work, a new security architecture called SurvSec is proposed. SurvSec is capable of maintaining security information even during the BS failure periods. This is accomplished in two steps. The first step is storing the security-related data until the recovery of the BS or the deployment of new BS. The second step is sending the stored data to the recovered BS or the new BS after it is authenticated

Furthermore, BS failure shows the importance of the continuous storage of the security reports of the monitored security threats towards the WSN through securely storing the security-related data of sensor nodes. The stored security-related data will be sent to the new BS during the recovery process. These procedures will result in reliable network recovery from base station failure and also, they will maintain the WSN lifetime where physical attacks towards the BS specifically target the reduction of the WSN lifetime.

Figure 8.1 shows the block diagram of the proposed system. The motion detection sensor is connected to the Arduino Uno microcontroller board. The X-Bee transceiver is connected to the shield card. The shield card is connected to the microcontroller board. The X-Bee module is used to transmit the sensed data from the sensor nodes to the security manager and to transmit the security report from the security manager to the new base station and to transmit the security report from the sensor nodes to the security managers. AES encryption algorithm is implemented on Arduino Uno microcontroller board to securely transmit the security reports and the sensed data from the sensor node transmitter side to the sensor node receiver side.

The receiver sensor node is connected to a PC through serial monitor program which is HyperTerminal program to show the input ciphertext and the output original plaintext. Serial monitor cable with MAX chip is used to monitor the received decrypted data.

The decrypted data is compared with the original encrypted data to show the validity of the hardware implementation of reliable network recovery form base station

failure and to show the correctness of the AES code and the correctness of the transmitter and receiver code.

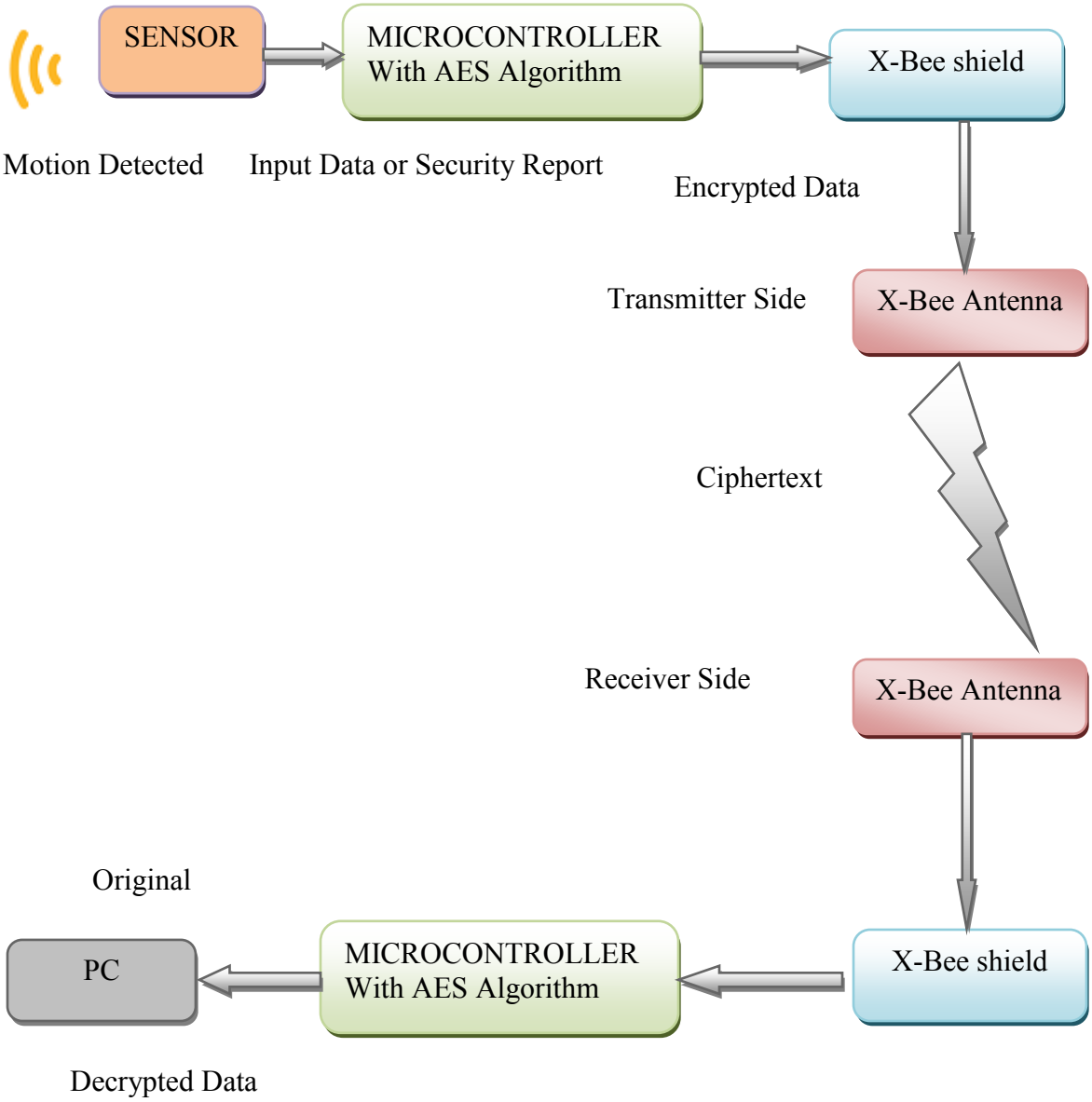


Figure 8.1, The Proposed System Block Diagram

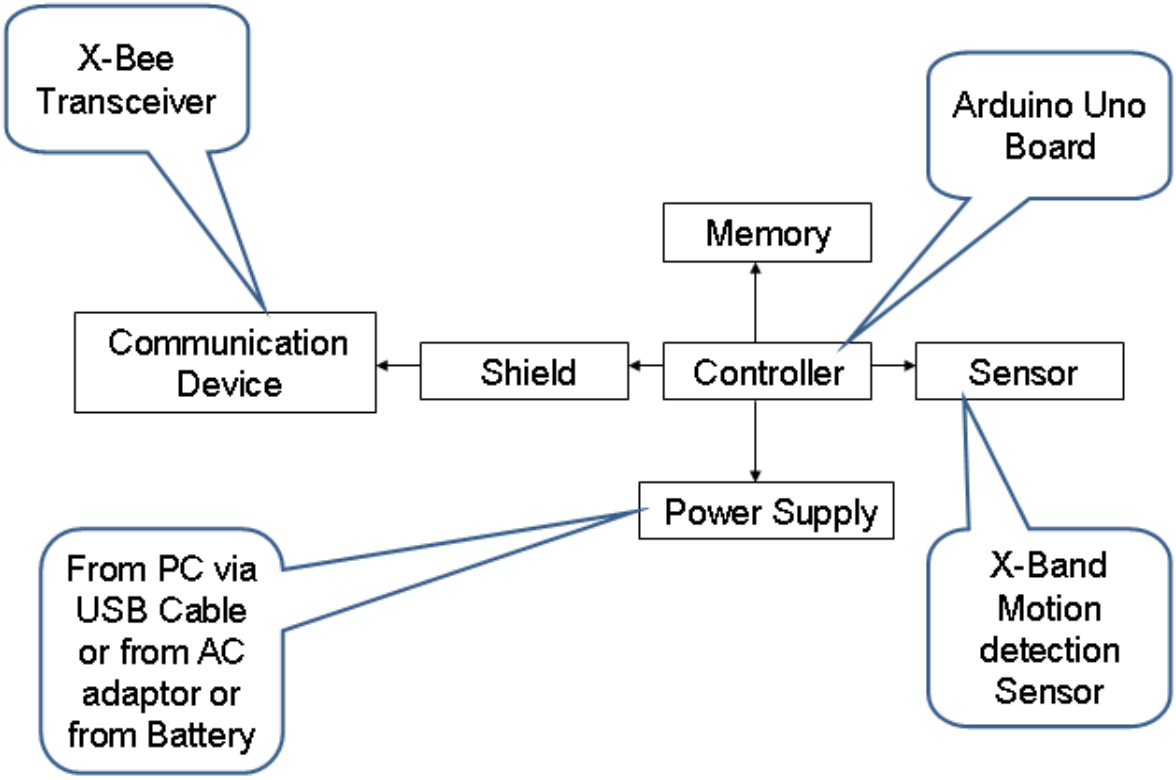


Figure 8.2, Arduino Uno Board Interconnections

Figure 8.2 shows the transmitter side or the receiver side. The Arduino Uno microcontroller board is connected to the shield card. The shield card is connected to the X-Bee Transceiver module. The X-Bee transceiver module is 1 mw Series 1. The Arduino Uno board is powered by PC via USB cable or from AC adaptor or from 9 V Battery. The Arduino Uno board is connected to X-Band Doppler radar motion detection sensor.

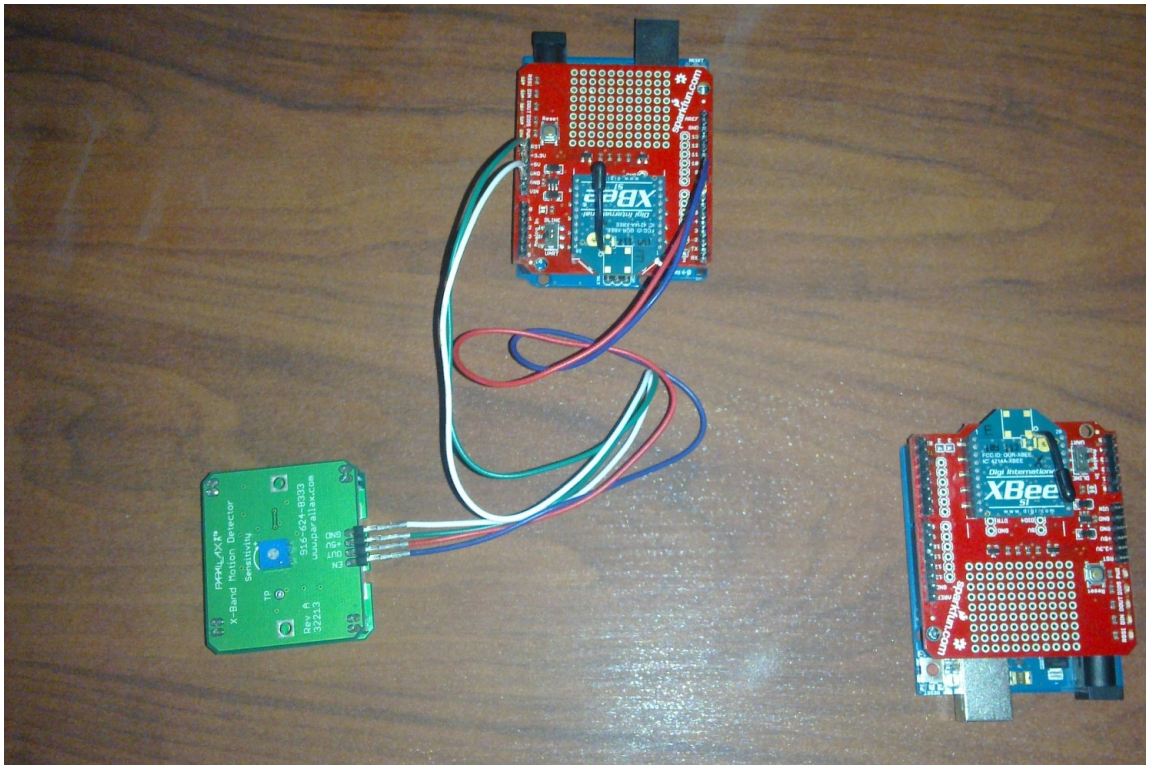


Figure 8.3, Proposed System Transmitter and Receiver

Figure 8.3 shows the transmitter and receiver components. The transmitter is composed of Arduino Uno microcontroller board, Shield card, X-Bee transceiver 1mw series 1 module, X-Band Doppler Radar motion detection sensor and AC adaptor power supply. The receiver is composed of Arduino Uno microcontroller board, Shield card, X-Bee transceiver 1mw series 1 module, AC adaptor power supply, serial monitor cable and MAX chip.

8.2 Proposed System Components

This section introduces the related work, the requirements for hardware implementation, the proposed system components and its specifications and finally the theory of operation for the system components.

8.2.1 Related Work

A surveillance WSN can be hardware implemented using motes or using Arduino Uno microcontroller boards. A sensor node, also known as a mote, is a node in a wireless sensor network that is capable of performing some processing, gathering sensory information and communicating with other connected nodes in the network. A mote is a node but a node is not always a mote. It was chosen to hardware implement the proposed surveillance WSN on Arduino Uno microcontroller boards to fully control all the hardware of the node. Surveillance WSN for battlefield or borders is previously hardware implemented as works discussed in [1, 2, 3, 5].

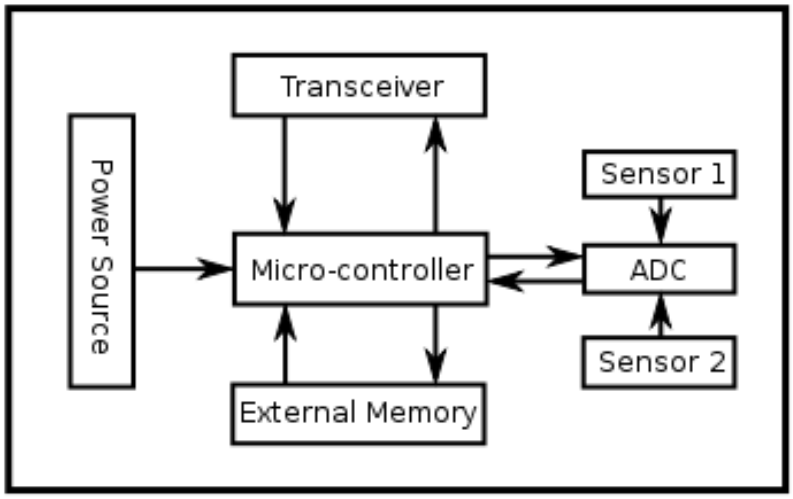


Figure 8.4, The Typical Architecture of the Mote

There is no previous hardware implementation works for reliable network recovery from base station failure. The proposed work is the first work that addresses the reliable network recovery from base station failure.

8.2.2 Requirements for Hardware Implementation

This section summarizes the most important requirements for the hardware implementation of reliable network recovery from base station failure which are the followings:

1. *Processing Power*: the processing power and data storage of WSN nodes are considerably limited and require the use of computational efficient algorithms (for the energy saving purposes) and small software footprints (for the memory saving codes purposes).
2. *Reliability*: it is very important to have the network work without any human involvement/intervention. This is because the whole or part of the network might be located at inaccessible sites where sensor nodes are deployed unattended as well as humans might not have the capacity to identify and respond to very time-sensitive critical messages in due time such as the case of WSN deployed around a very sensitive military-related facility. In this case there is not much margin for failure.
3. *Power Supply*: the energy reserves available to a WSN node are generally very limited derived from 2-3 AAA batteries. Nodes are expected to run for extended periods of time vary from (1 - 2 years) on this internal energy reserve. Thus the design should be energy efficient.
4. *Cost*: the cost of WSN deployments must not be adversely impacted by the inclusion of security services as the cost is often a major factor for WSN technology.
5. *Scalability*: the design of any security architecture composed of security components which must ensure network scalability. Network scalability must be preserved in order to allow for all future expansions of the WSN.

8.2.3 Proposed System Components and their Specifications

This sub-section describes the proposed system components which are nine components and their specifications and features to be used for the hardware implementation of reliable network recovery from base station failure.

Table 8.1, The Proposed System Components

No.	Item	Quantity
1	Max Chip	1
2	Serial Monitor Cable	1
3	X-Bee 1mw Series 1 Module	2
4	Arduino X-Bee Shield	2
5	X-Bee USB Programmer	1
6	X-band Motion Detection sensor	1
7	Arduino UNO Microcontroller Board	2
8	USB cable	2
9	Mini USB cable	1

8.2.3.1 X- Band Doppler Radar Motion Detection Sensor

The X-Band Motion Detection Sensor that is shown in Figure 8.5 is a common ingredient in security systems, automatic lighting, and automatic door openers. It can detect movements in a room, yard, or even on the other side of a wall. It is a Doppler radar sensor that operates in the X-band frequency at 10.525 GHz. It indicates movements with oscillations using its high/low output. Sensitivity is manually adjusted with a potentiometer on the back of the device, offering direct line of sight detection from roughly 8 ft to slightly over 30 ft (2.4 m to 9 m).

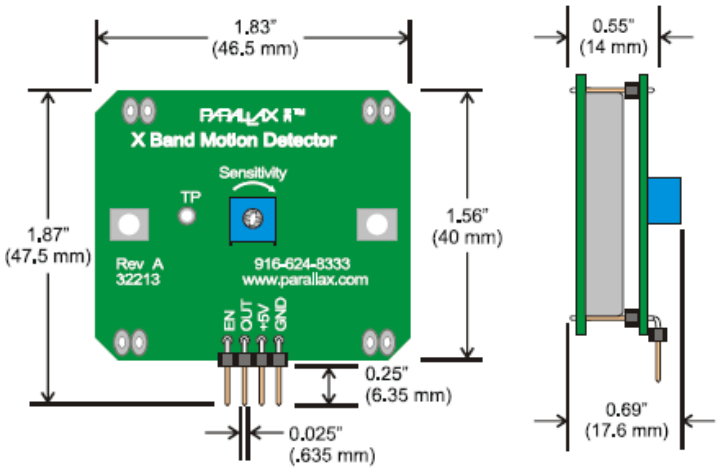


Figure 8.5, The X-Band Motion Detection Sensor Dimensions [205]

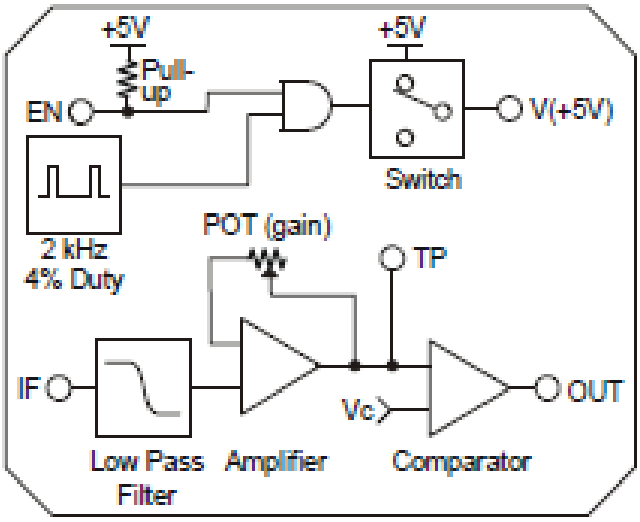


Figure 8.6.a, Control Board [205]

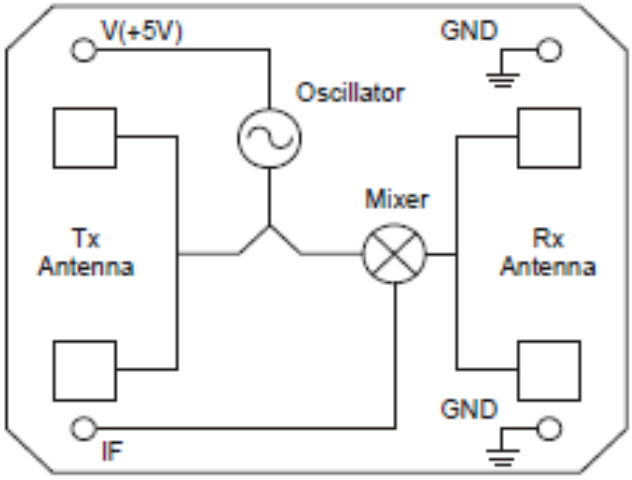


Figure 8.6.b,Antenna PCB [205]

Figure 8.6, The X-Band Motion Detection Sensor Schematic

The X-Band Motion Detection Sensor module is constructed of two boards connected together as shown in Figure 8.6. The two boards are a control board, and the antenna PCB with the Doppler sensor. Its block diagram is shown in Figure 8.6. When the enable pin is either held high or left floating, the control board cycles the Doppler sensor's power at 2 kHz, 4% duty cycle.

The Doppler sensor's 10.525 GHz oscillator signal is routed to the transmit antenna, and also to a mixer diode where it's IF output contains signals with the sum and difference of the transmitted and received frequencies along with components of the original signal and some harmonics. The difference between signal's frequency that results from mixing the outgoing and returning signal frequencies is the important component. It oscillates at a frequency corresponding to how much the returning signal has been either compressed or stretched as a result of the Doppler Effect that an object has on the signal as the object moves toward or away from the sensor.

8.2.3.2 X-Bee 1 mw Series 1 Transceiver

The Digi X-Bee 1 mw Series 1 transceiver 802.15.4 modules are the easiest-to-use, most reliable and cost-effective RF devices. The X-Bee transceiver 802.15.4 modules provide two friendly modes of communication; a simple serial method of

transmit/receive or a framed mode providing advanced features. X-Bees Transceiver is ready to use out of the package, or it can be configured through the X-CTU program utility. The transceiver is controlled by the Arduino Uno microcontroller. These modules can communicate point to point as in Series 1, from one point to multipoint as in Series 2, or in a mesh network as in Series 2.

Figure 8.7, The X-Bee 1 mw Series 1 transceiver 802.15.4 Module [203]

The X-Bee module is chosen according to the comparison between Series 1 and Series 2 modules. We only need to choose an antenna style (chip or wire) and power level (1mw for up to 300ft and 60 mw for up to 1 mile). Our proposed system assumes point to point communication between the two X-Bee transceivers modules. Figure 8.7 shows the X-Bee 1 mw series 1 module.

The two most common RF radios that are available from Digi are the X-Bee Series 1 and the X-Bee Series 2. The Series 1 and Series 2 modules are quite similar, but selection of a module should be based upon application specific needs. All X-Bee radios have the same footprint and for the most part are pin for pin compatible (with a few differences in the placement of ADC/IO lines), but are NOT interoperable. Series 1 and Series 2 use different application profiles, which are unique to each radio family. They can however, use the same RS232 or USB interface boards.

Series 2 X-Bee "ZigBee" modules is the PRO Series 2 ZigBee protocol 63mW with wire antenna. It is good for point-to-point, multipoint and mesh networks. This module is a little more difficult to get going than the Series 1. You must set up a

"coordinator" module so they are not as plug-and-play. Series 2 modules cannot talk to Series 1 modules so if you already have some S1 type X-Bees you may want to stick with them. The S2 modules are not necessarily 'better' than S1 for many projects. They are just different as they use the "ZigBee" wireless stack instead of the 802.15.4. This makes them better for low power usage and advanced users who want a mesh topology (many X-Bees in a spread-out configuration) but they are more difficult to use for basic point-to-point setups.

Table 8-2, Comparison between X-Bee Series 1 and X-Bee Series 2

		X-Bee Series 1	**X-Bee Series 2**
1	Indoor/Urban range	up to 100 ft. (30m)	up to 133 ft. (40m)
2	Outdoor RF line-of-sight range	up to 300 ft. (100m)	up to 400 ft. (120m)
3	Transmit Power Output	1 mw	63 mw
4	RF Data Rate	250 Kbps	250 Kbps
5	Receiver Sensitivity	-92dbm (1% PER)	-98dbm (1% PER)
6	Supply Voltage	2.8 - 3.4 V	2.8 - 3.6 V
7	Transmit Current (typical)	45 mA (@ 3.3 V)	40 mA (@ 3.3 V)
8	Idle/Receive Current (typical)	50 mA (@ 3.3 V)	40 mA (@ 3.3 V)
9	Power-down Current	10 uA	1 uA
10	Frequency	2.4 GHz	2.4 GHz
11	Dimensions	0.0960" x 1.087"	0.0960" x 1.087"
12	Operating Temperature	-40 to 85 C	-40 to 85 C
13	Antenna Options	PCB, Integrated Whip, U.FL, RPSMA	PCB, Integrated Whip, U.FL, RPSMA
14	Network Topologies	Point to point, Star, Mesh (with Digi Mesh firmware)	Point to point, Star, Mesh
15	Number of Channels	16 Direct Sequence Channels	16 Direct Sequence Channels
16	Filtration Options	PAN ID, Channel & Source/Destination	PAN ID, Channel & Source/Destination

802.15.4: X-Bee Series 1 comes standard with 802.15.4 firmware for point-point or star topology. This mature firmware offers ADC (analog-to-digital conversion) inputs, and digital and analog I/O line passing. The 802.15.4 X-Bee is significantly faster than ZigBee; RF latency can generally be calculated in 802.15.4. Throughput is also much higher; a practical maximum throughput is around 80kbps.

ZigBee: X-Bee Series 2 does not offer any 802.15.4-only firmware; it is always running the ZigBee mesh firmware. The ZigBee X-Bee excels in very low-power scenarios, when configured as an End Device, this module has the lowest current draw of any Digi RF product. However, the infrastructure of a ZigBee network is more complex and requires more configurations to fully implement. The main benefit of a ZigBee X-Bee is third-party device support and deep integration with the Device Cloud by Etherios and Digi gateway products.

The shield cards provide several advantages to the X-Bee modules such as friendly standard 0.1 inch pin spacing, mounting holes, and easy-to-solder connections. If you are communicating point-to-point, we still recommend that you always have at least one X-Bee USB programmer so you can easily configure and test each X-Bee module prior to putting it in a point-to-point application.

Features

- Parallax support.
- Wire antenna.
- Cross-compatibility with other 802.15.4 X-Bee modules.
- Low-power sleep modes.
- 100 ft (30 m) indoor/urban range and 300 ft (100 m) outdoor line-of-sight range.
- Configured with API or AT commands, local or over the air 8 digital I/O and 10-bit ADC inputs.
- 802.15.4 Network topology.
- Multiple antenna options.

Applications

- Wireless data acquisition.
- Remote signal beacon for adventure seekers.
- Remote industrial monitoring.
- Lighting control.

Key Specifications

- Up to 115.2 kbps interface data rate.
- 2.4 GHz frequency band.
- Industrial temperature rating (-40C to 85C).
- Transmit power 1 mw (+0 dBm).
- Supply voltage 2.8 - 3.4 V DC; transmit current 100 mA; receive current 50 mA.
- Power-down current <10 uA.

8.2.3.3 X-Bee Programmer

This is a simple to use USB to serial base unit for the X-Bee transceiver. This unit works with all X-Bee modules including the Series 1 and Series 2, standard and Pro version. Plug the X-Bee module into the X-Bee programmer, attach a mini USB cable, and you will have direct access to the serial and programming pins on the X-Bee unit. The X-Bee programmer board is shown in Figure 8.8. The X-Bee programmer is used with the X-CTU software to program the X-Bee modules as coordinator and end device.

Figure 8.8, The X-Bee Programmer [203]

8.2.3.4 X-CTU Program

The X-CTU program is used to program the X-Bee modules after connecting the programmer board with the PC through the Mini USB cable. First, we select the Com Port at which the X- Bee module is located. Second, we press Test/Query bottom to assure the right assignment to the module.

The X-CTU program has a Modem Configuration button to program the X-Bee modules. The library of the X-CTU software should be updated before the programming takes place. There are two methodologies to program the X-Bee modules, first methodology is through AT commands and the second methodology is through the API commands.

We choose to program the X-Bee modules through the AT commands. Figure 8.9 shows the interface of the X-CTU program.

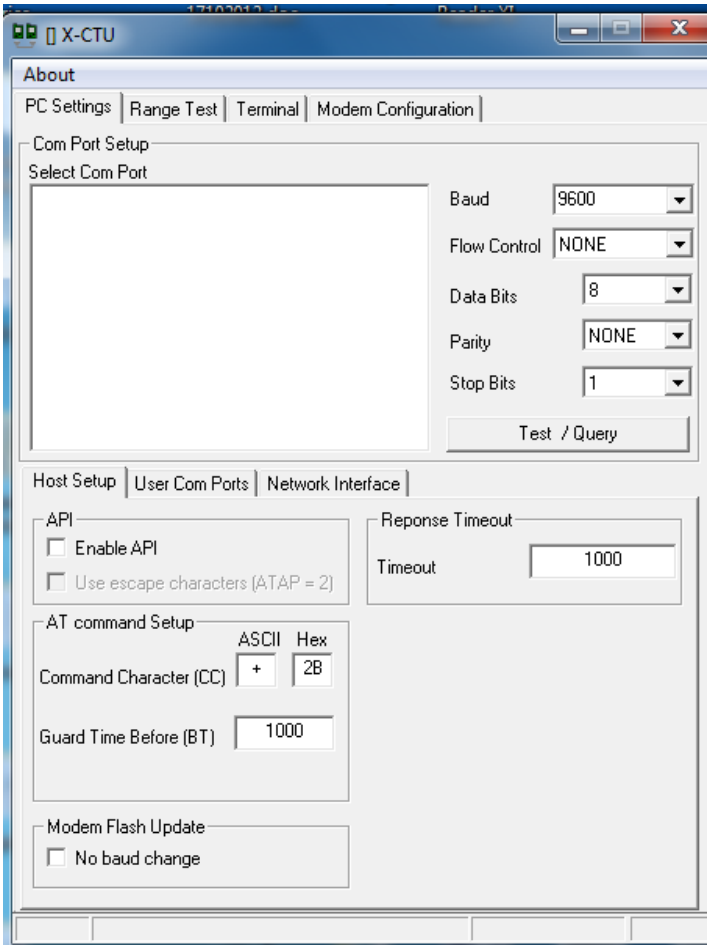


Figure 8.9, The X-CTU Program used to Program the X-Bee Modules

8.2.3.5 Arduino Uno Microcontroller Board

The Arduino Uno board is a microcontroller board based on the Atmel microcontroller ATmega328 as shown in Figure 8.10. It has 14 digital input/output pins where 6 pins can be used as Pulse Width Modulation (PWM) outputs, 6 pins can be used as analog inputs, one pin can be used as a 16 MHz crystal oscillator, one pin can be used as a reset button, a USB connection, and a power jack. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

Arduino Uno

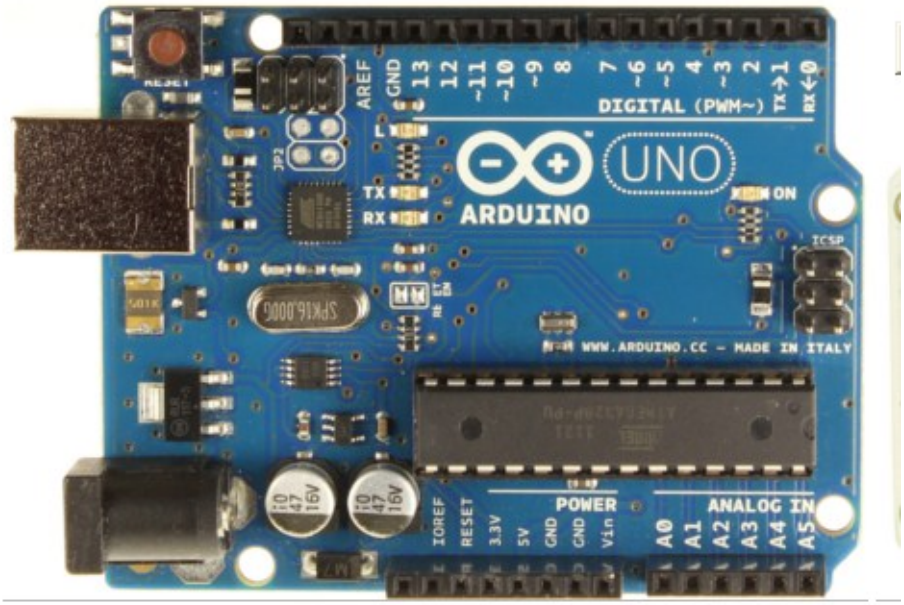


Figure 8.10, The Arduino Uno Microcontroller Board [204]

Features

- Microcontroller: ATmega328
- Operating Voltage: 5V
- Input Voltage (recommended): 7-12V
- Input Voltage (limits): 6-20V
- Digital I/O Pins: 14 (of which 6 provide PWM output)
- Analog Input Pins: 6
- DC Current per I/O Pin: 40 mA
- DC Current for 3.3V Pin:50 mA
- Flash Memory: 32 KB (ATmega328) of which 0.5 KB used by boot loader
- SRAM: 2 KB (ATmega328)
- EEPROM: 1 KB (ATmega328)
- Clock Speed: 16 MHz

8.2.3.6 Arduino Uno Software

The Arduino Uno software is a software program that runs C programming language that is used to write down the code on the Arduino Uno microcontroller board. The code refers to the code of the motion detection sensor, the code of the transmitter, the code of the receiver and the code of the AES encryption algorithm. Figure 8.11 shows the interface of Arduino Uno software.

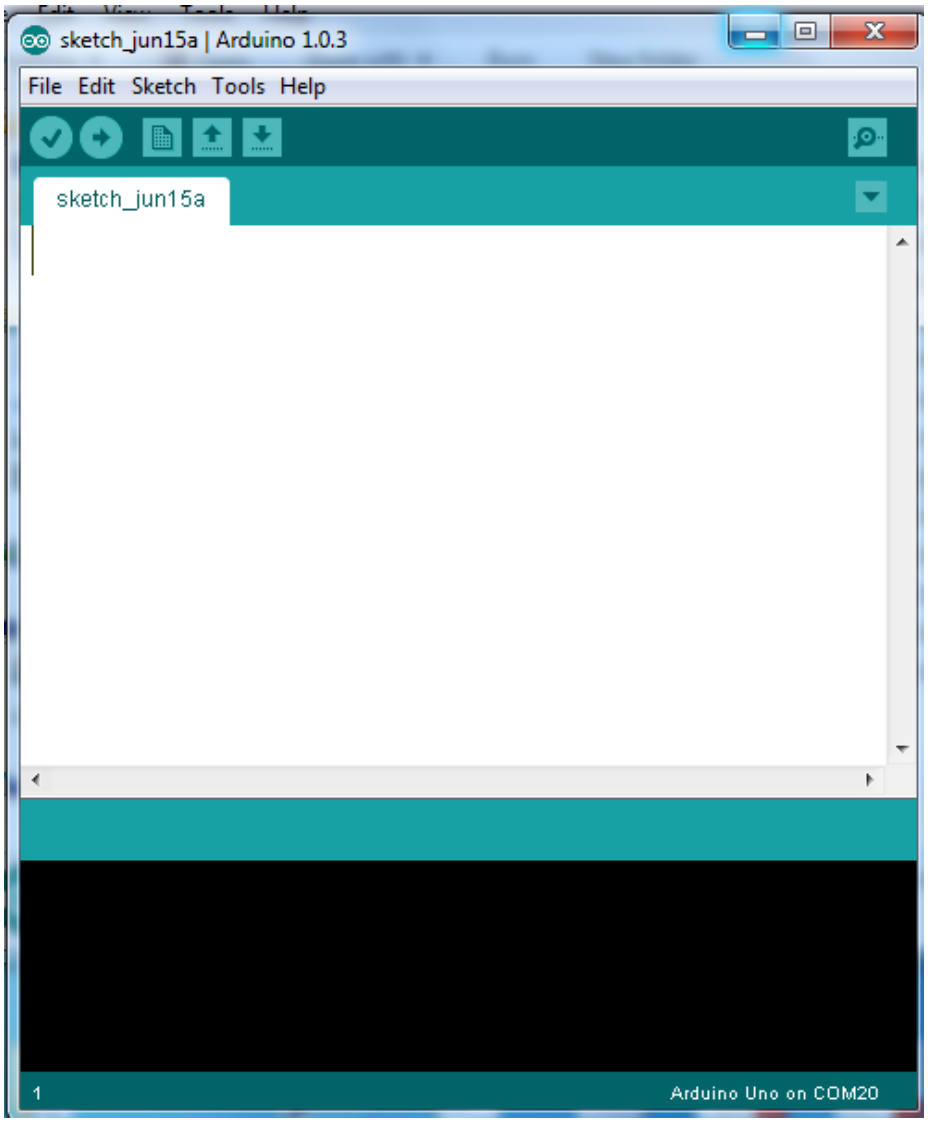


Figure 8.11, The Arduino Uno Software

After writing down the C code, it's verified, debugged, and then uploaded to the microcontroller to test its validation.

8.2.3.7 X-Bee Shield Card

The serial pins (DIN and DOUT) of the X-Bee Transceiver are connected to the Arduino Uno microcontroller board through the shield card as shown in Figure 8.12 which allows the programmer to select a connection to either the UART pins (D0, D1) or any digital pins on the Arduino Uno microcontroller board (D2 and D3 default). Power is taken from the 5V pin of the Arduino Uno board and regulated on-board to 3.3VDC before being supplied to the X-Bee Transceiver. The shield also takes care of level shifting on the DIN and DOUT pins of the X-Bee Transceiver. In the latest revision the diode level shifter is replaced with a more robust MOSFET level shifter.

The board also includes LEDs to indicate power and activity on DIN, DOUT, RSSI, and DIO5 pins of the X-Bee Transceiver. The Arduino Uno board reset button is brought out on the shield, and a 9x11 grid of 0.1" holes are available for prototyping. The shield does not come with headers installed.

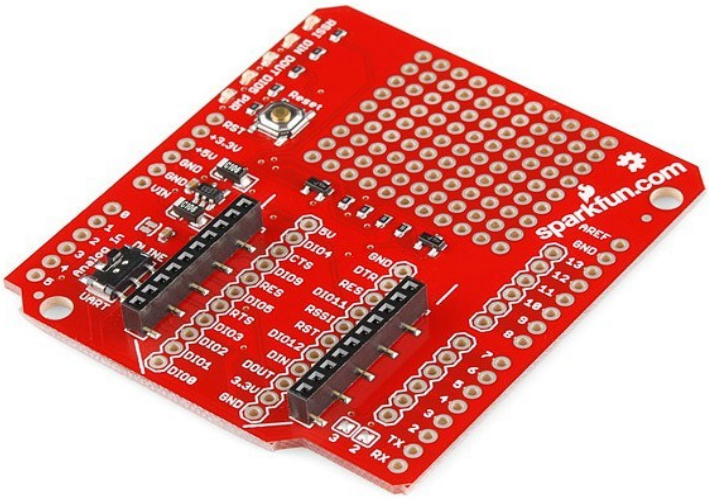


Figure 8.12, The X-Bee Shield Card [204]

Features

- Mounts directly onto your Arduino.
- DIN and DOUT pins of X-Bee can be connected to either the UART pins or any digital pin on the Arduino Uno board (D2 and D3 default).
- 3.3V power regulation and MOSFET level shifting on-board.
- 9x11 grid of 0.1" spaced prototyping holes.
- Reset button brought out to shield.
- Power, DIN, DOUT, RSSI and DIO5 indicator LEDs.

8.2.3.8 Arduino Uno Board Power Supply

The Arduino Uno Board is powered by three ways, the first way is by the PC through USB cable, the second way is by AC Adaptor and the third way is by 9 V Battery.

In this work an AC Adaptor was chosen to power the Arduino Uno board.

8.2.3.9 Serial Monitor Cable with MAX Chip

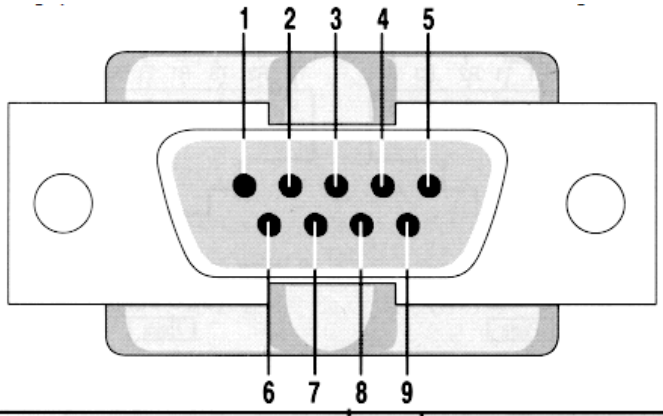
Serial monitor cable has two terminals one of them is USB port and the other port is serial port. It is connected between the USB port of the computer and the MAX chip. The MAX chip converts between 12 V from the computer to 5 V at the Arduino Uno board. The MAX chip is connected to the receiver board at the Arduino Uno board.

Features and Benefits of Serial Monitor Cable

- Compatible with USB 2.0
- 12 Mbps USB data rate
- 921.6 Kbps maximum baud rate for super fast data transmission
- Drivers provided for Windows
- DB9 male connector for RS-232
- LEDs for indicating USB and TxD/RxD activity

Figure 8.13 shows the USB Serial monitor cable. Figure 8.14 shows the Serial port connection.

Figure 8.13, The Serial Monitor Cable [204]



Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

Figure 8.14, The Serial Port [204]

8.2.3.10 HyperTerminal Program

The HyperTerminal program is the software that is used with the serial monitor cable and MAX chip to monitor the output decrypted ciphertext from the receiver. HyperTerminal is a serial monitor software with windows platform only.

HyperTerminal is used with the COM Port that is connected with the Arduino Uno Receiver board to monitor the output from the receiver board and compare it with the input plaintext at the transmitter side. Figure 8.15 shows the interface of HyperTerminal program.

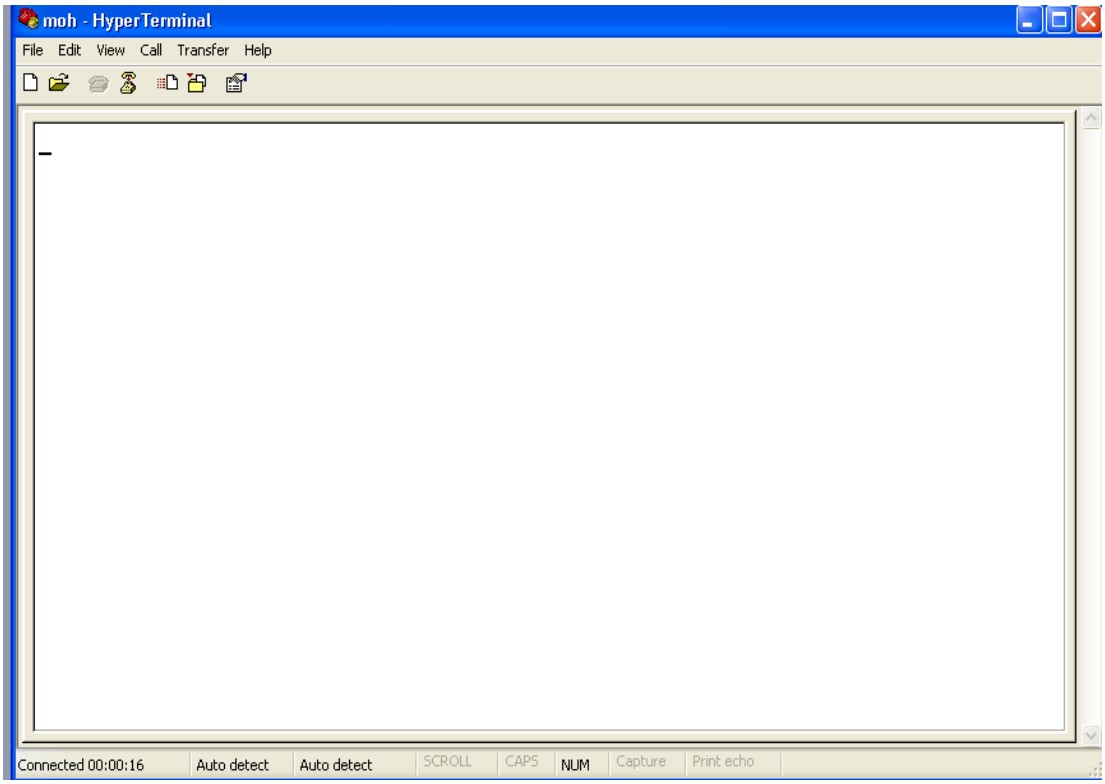


Figure 8.15, The HyperTerminal Serial Monitor Program

8.2.4 Theory of Operation for the Proposed System Components

This sub-section describes the theories of operation that are used in the proposed system to allow correct configuration and programming to the proposed components. These theories of operation are concerned with the Doppler Radar Motion Detection Sensor, the Arduino Uno microcontroller board, and the HyperTerminal software.

8.2.4.1 Theory of Operation for Motion Detection Sensor

Motion in the detection area causes oscillations at the module OUT pin, which can be detected by a microcontroller. The front of the device is the antenna PCB, a printed circuit board surface with a module that transmits and receives antennas which is shown in Figure 8.16.The device should be oriented so that this surface faces the detection area. When the enable (EN) pin is held high or left floating, the device takes brief, periodic, low power Doppler radar measurements. The frequency of the high/low signals the output (OUT) pin transmits corresponds to the speed of the motion.

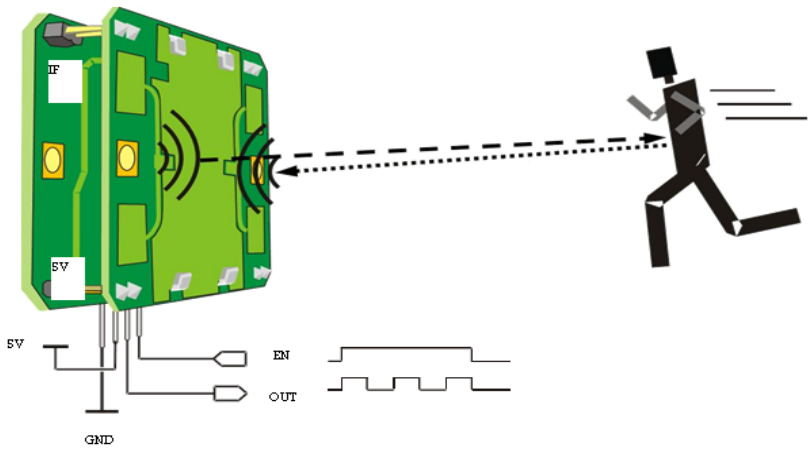


Figure 8.16, Motion Detection Antenna [205]

The X-Band Motion detection range can be adjusted by turning the Sensitivity potentiometer on the back of the device. The "motion/no motion" threshold constant 'Move Threshold' in each example program can also be adjusted, which allows it to ignore small or brief, quick motions. The Larger Move Threshold values require more motion (larger values in the 'cycles' variable) to trigger a detection; smaller values require less motion to trigger a detection. The device's sensitivity may vary with different kinds of walls and window blinds. For example, direct line-of-sight detection can be adjusted from approximately 8 to slightly over 30 ft (2.4 m to 9 m). The sensor will still

be effective through walls and windows, but not through conductive metals, and testing for the conditions under which it will be used is recommended.

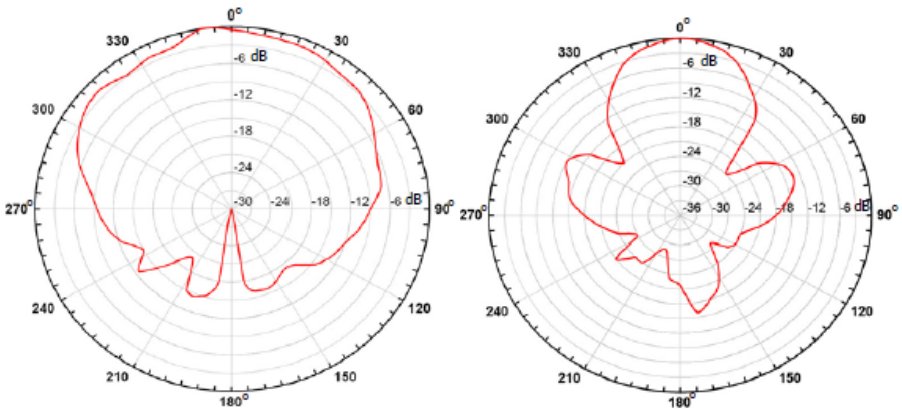
The device's sensitivity also varies with the object's angle, which in turn varies with the antenna radiation plots in Figure 8.17. It is most sensitive to objects directly in front of the antenna PCB which corresponds to an angle of 0°as shown in Figure 8.17.

Sensitivity is maximized in the areas where the plot is between the two outermost circles. This is the region between 0 dB (full power signal) and -3 dB (half power signal).

For example, on the horizontal plane, the antenna's radiation pattern stays above half power from approximately 300 degrees to just over 45 degrees.

In the vertical plane, the half-power beam width ranges from about 340 to 20 degrees. Keep in mind that angles corresponding to larger –dB values (closer to the center of a graph) indicate that the object will have to be closer to the device for it to detect motion.

So, the sensor will still detect motion at angles outside these half-power beam widths, just at a closer range.



Horizontal **Vertical**

Figure 8.17, Motion Detection Sensor Antenna Radiation Pattern [205]

The Doppler sensor's IF terminal passes the signal to the control board, where a low pass filter removes all the higher frequency signals and leaves behind the difference signal. This signal passes through an amplifier whose gain can be adjusted by the control board's potentiometer, and can be monitored by probing the test point, which is a plated

hole labeled TP. The TP signal passes though a comparator, which transforms the signal with the difference frequency to a high/low digital output. The difference signal's frequency is related to the component of the object's speed toward or away from the sensor by this equation:

$$F_d = 2V\frac{F_t}{c}\cos\theta$$

Where:

F_d = Difference frequency (sometimes referred to as Doppler frequency).

V = Velocity of the target.

F_t = Transmit frequency.

c = Speed of light at 3 × 108 m/s.

θ = Motion direction angle deviation from perpendicular to the antenna PCB.

In contrast to speed guns, which use a wave guide to direct the antenna radiation pattern, the X-Band Motion Detector's antenna has a wide radiation pattern to convert velocities from multiple points to oscillations that notify the microcontroller that movement was detected. This device is designed to detect motion, not to determine speed of a moving object. However, an application may still use a rearranged version of the F_d equation to determine speed provided everything else in the detection area stays still. The test code for the X-Band Motion Detector is explained in Appendix A.

8.2.4.2 Theory of Operation for the Arduino Uno Microcontroller Board

The Arduino Uno is a microcontroller board based on the ATmega328 microcontroller. Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It is intended for designers, hobbyists, and anyone interested in creating interactive objects or environments.

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software running on a computer.

The boards can be built by hand or purchased preassembled; the software can be downloaded for free. The hardware reference designs (CAD files) are available under an open-source license; and are free to be adapted according to the needs.

Steps to start with the Arduino Uno board:

1- Get the Arduino Uno board and the USB cable.
2- Download the free Arduino Software.
3- Connect the board to the PC.
4- Install the driver of the board.
5- Launch the Arduino Application.
6- Open the blink example.
7- Select your board.
8- Select your serial port.
9- Upload the program.

8.2.4.3 Theory of Operation for HyperTerminal Program

HyperTerminal is a program that you can use to connect hardware to the PC. HyperTerminal (also known as Hyper Term) is a communications and terminal emulation program that comes with the Windows operating system, beginning with Windows 98.

HyperTerminal can be used to set up a dial-up connection to another computer through the internal modem using Telnet or to access a bulletin board service (BBS) in another computer.

It can also be used to set up a connection for data transfer between two computers (such as your desktop computer and a portable computer) using the serial ports. HyperTerminal can be used for serial-port control of external devices or systems such as

scientific instruments, robots, or radio communications stations. HyperTerminal can also be used as a troubleshooting tool when setting up and using a modem. Commands can be sent through HyperTerminal to make sure that the modem is connected properly.
The Arduino Uno board is interfaced to a HyperTerminal session by following the next steps:

1- Open HyperTerminal.
2- Create a connection called "Arduino".
3- Select the proper port (the Arduino is set to COM20 on my machine).
4- Configure the port: Set the baud rate to 9600, the bits to 8, parity to none, stop bits to 1, and flow control to none.

You are done. Only what the Arduino sends back will be printed on screen.

8.3 Design and Implementation of the Proposed System

In this section, the design and implementation of the proposed system is introduced. This section is concerned with the programming of the microcontroller board, the motion detection sensor program, programming the microcontroller at the transmitter side with the transmitter code, programming the microcontroller at the receiver side with the receiver code, programming the microcontroller at the transmitter side and the receiver side with the AES encryption algorithm, configuring the X-Bee transmitter using X-CTU software, configuring the X-Bee receiver using X-CTU software and finally providing the interconnections between the serial monitor cable and the Arduino Uno board at the receiver.

8.3.1 Security Report Content

We assume that the security report content is the following fields: count which is the number of threats at the sensor node and it is 8 bits, the time which is 24 bits, attacked node ID which is 16 bits, node reputation which is 8 bits, replica number which is 8 bits, attack ID which is 8 bits and data integrity which is 32 bits. The size of the security report for one attack is 104 bits.

Table 8.3, Security Report Content

Field	First Field Count	Second Field Time	Third Field Attack ID	Fourth Field Attacked Node ID	Fifth Field Reputation	Sixth Field Replica Number	Seventh Field Data Integrity
Data Size	8 bits	24 bits	8 bits	16 bits	8 bits	8 bits	32 bits

8.3.2 Programming the Arduino Uno Microcontroller

The Arduino Uno board is a microcontroller board based on the ATmega328 microcontroller. Two Arduino Uno boards are used; one of them is used at the transmitter side and the other is used at the receiver side. The board can be powered by connecting it to a PC through USB cable or by connecting it to an AC adaptor or by connecting it to a 9 V battery.

The Arduino Uno board at the transmitter side is connected to X-Band motion detection sensor, shield card and the X-Bee 1 mw Series 1 Transceiver is connected to the shield card.

The Arduino Uno board at the transmitter has three codes; the motion detection sensor code, the transmitter code and the AES encryption algorithm code.

The Arduino Uno board at the transmitter has two inputs; the input from the motion detection sensor and the input from the security report which is 104 bits for each threat.

The Arduino Uno board at the receiver side is connected to shield card and the X-Bee 1 mw Series 1 Transceiver is connected to the shield card. The receiver is connected to a PC through serial monitor cable.

The Arduino Uno board at the receiver has two codes; the receiver code and the AES encryption algorithm code.

The Arduino Uno board at the receiver has two outputs; the output represents that a motion is detected and the output from the security report which is 104 bits for each threat.

8.3.3 Programming the Microcontroller with the Motion Detection Sensor Code

The X-band motion detection sensor as shown in Figure 8.19 has four legs which are connected to the Arduino Uno microcontroller board as shown in Figure 8.18 according to the motion detection program in Appendix A;

- Pin GND at Sensor ⟹ Pin GND at Arduino board
- Pin ENABLE at Sensor ⟹ Pin # 8 at Arduino board
- Pin OUT at Sensor ⟹ Pin # 7 at Arduino board
- Pin 5 V at Sensor ⟹ Pin 5 V at Arduino board

Figure 8.18, Interconnections between Arduino Uno Board and Motion Sensor

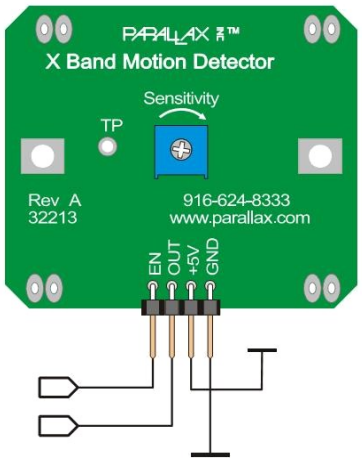


Figure 8.19, Motion Detection Sensor Connection with Arduino Uno Board [205]

The sensor is controlled by the Arduino Uno microcontroller board. The X-band motion detection sensor uses the Doppler Effect which means that it sends a band of

frequencies towards the watched area and receives these frequencies again to calculate the shift between the transmitted and received frequencies due to a motion in the surveillance area. After collecting the motion data, the sensor now will pass these data to the microcontroller which in turn will encrypt the messages using the AES algorithm.
The code that controls the motion detection sensor on the Arduino Uno microcontroller board is shown in Appendix A.

8.3.4 Programming the Microcontroller with the Transmitter Program

The transmitter collects the sensed data from the motion detection sensor and the security report that contains the threats where the fields of the security report are the following: count which is 8 bits; time which is 24 bits; attacked node ID which is 16 bits; node reputation which is 8 bits; replica number which is 8 bits; data integrity which is 32 bits and attack ID which is 8 bits. The total security report for one threat is 8 + 24 + 16 + 8 + 8 + 32 + 8 = 104 bits

Two problems were encountered at the transmitter. The first problem is that the transmitter sends the data in ASCI format. The second problem is that the transmitter does not have a start frame.

The two problems are solved. The first problem is solved by allowing the transmitter to send in ASCI format then the receiver changes the input data from ASCI format to binary format to return the original plaintext.

The second problem is solved by inserting a start frame before the transmission. This start frame is “A” frame. At the receiver side, the receiver starts to receive with “A” frame then the sent data.

The transmitter encrypts the sensed data or the security report with AES encryption algorithm to send the data encrypted.

The receiver receives the data in ASCI format then converts it to binary format then decrypts the data using AES encryption algorithm.
The code of the transmitter is shown in Appendix A.

8.3.5 Programming the Microcontroller with the Receiver Program

The receiver which is a security manager received the sent data but at first the receiver received the "A" frame which is the start frame from the transmitter.

The receiver converts the received data ASCI format to binary format then the receiver decrypts the received data using the AES encryption algorithm.

The code of the receiver is in Appendix A.

8.3.6 Programming the Microcontroller with AES Encryption Algorithm

The code of the AES encryption algorithm at the transmitter and receiver is shown in Appendix A.

8.3.7 Programming X-Bee Transceiver with Programmer Board and X-CTU Program

Both the X-Bee transmitter and the X-Bee receiver are programmed using the programmer board which is connected to the PC through the Mini USB cable and the X-CTU program.

8.3.7.1 Programming the X-Bee Transmitter

The transmitter X-Bee will be programmed to be a coordinator X-Bee with Personal Area Network of 1111 and the destination addresses high and low will be programmed from the back of the receiver X-Bee module using X-CTU program as shown in Figure 8.20.

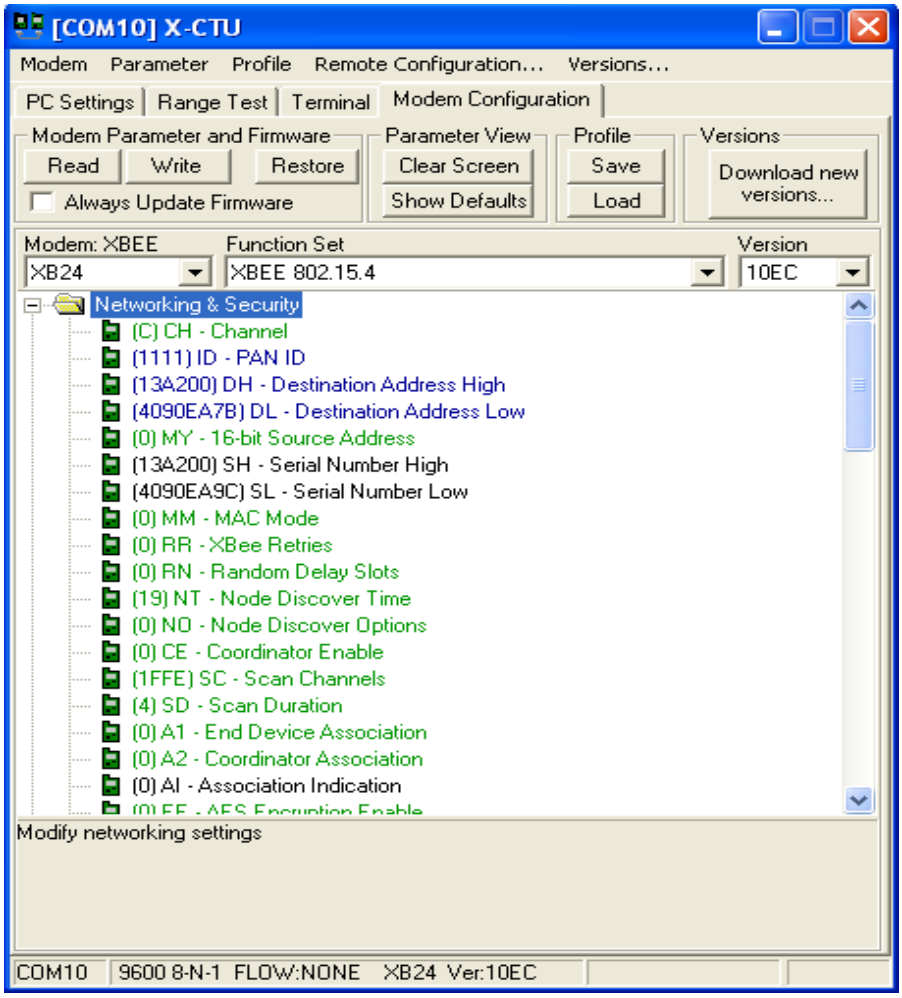


Figure 8.20, X-Bee Transmitter as Coordinator

8.3.7.2 Programming the X-Bee Receiver

The receiver X-Bee will be programmed to be an End device X-Bee with Personal Area Network of 1111 and the destination addresses high and low from will be programmed from the back of the transmitter X-Bee module using X-CTU program as shown in Figure 8.21.

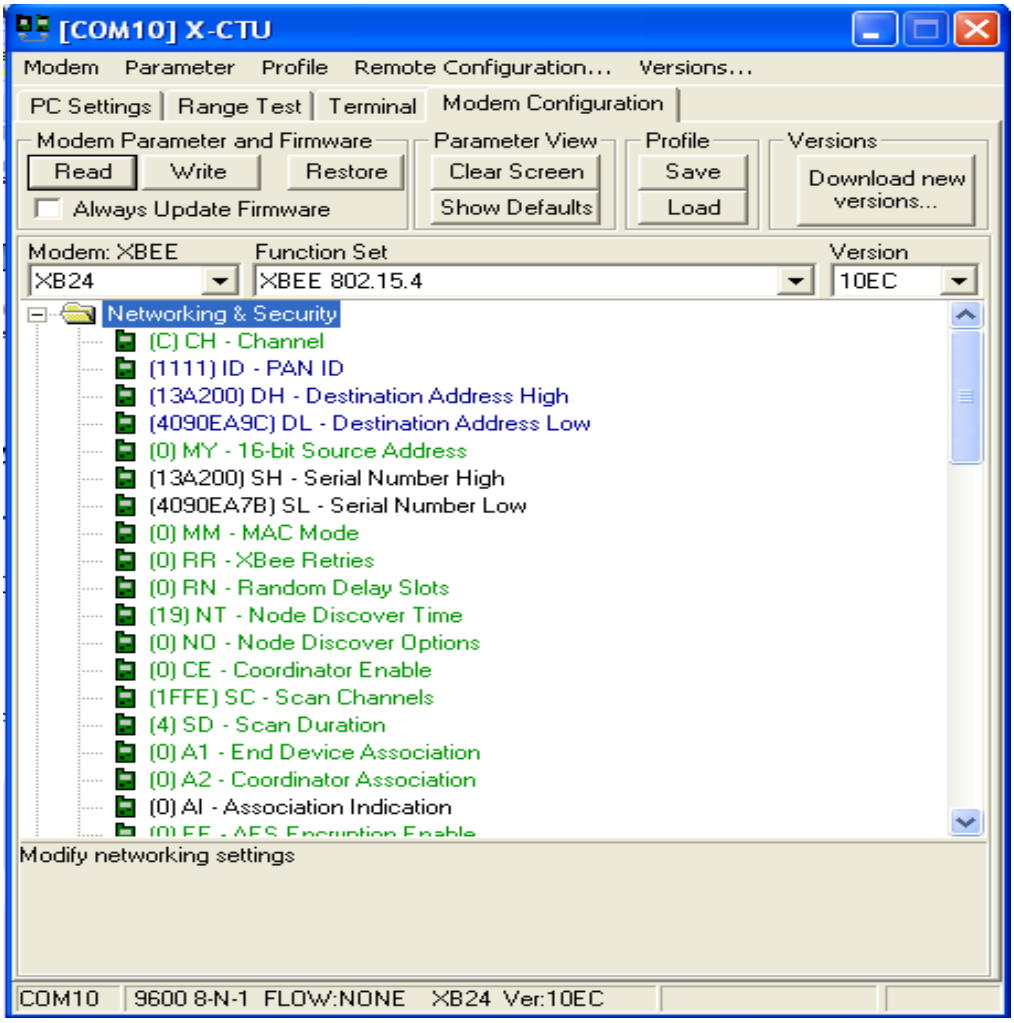


Figure 8.21, X-Bee Receiver as End Device

8.3.8 Connection of Serial Monitor Cable and MAX Chip with the Arduino Uno Board

The serial monitor cable is connected to the MAX chip and the MAX chip is connected to the Arduino Uno microcontroller board as shown in Figure 8.22; The MAX chip converts from 12 Volt at the computer side to 5 Volt at the Arduino Uno board.

- Pin GND at MAX Chip ⟹ Pin GND at Arduino board
- Pin TX at MAX Chip ⟹ Pin # 11 at Arduino board
- Pin 5 V at MAX Chip ⟹ Pin 5 V at Arduino board

Figure 8.22, Interconnections between Arduino Board and Serial Monitor Cable

8.4 Results and Evaluation Metrics

This section shows the evaluation metrics for the proposed system and the results from the proposed system.

8.4.1 Evaluation Metrics

1- Security Report Size

The security report size is 104 bits for each threat.

2- Passing Current at the Receiver from the Security Report

The current is measured from the USB port at the receiver side. The USB port connects between the Arduino Uno board and the PC.

3- Power Consumption at the Receiver from the Security Report

The power consumption at the receiver side from the security report is the multiplication of the passing current and the input voltage.

8.4.2 Results

The results of the hardware implementation for reliable network recovery from base station failure is shown in two folds; the measurements of the passing current at the Arduino Uno microcontroller board at the receiver side and the measurements of the power consumption at the receiver side for the received security report.

8.4.2.1 Measurements of Passing Current at the Receiver from the Security Report

The passing current is measured at the receiver from the USB port. The passing current is 100 mA. Figure 8.23 shows the measurement of passing current at the USB port.

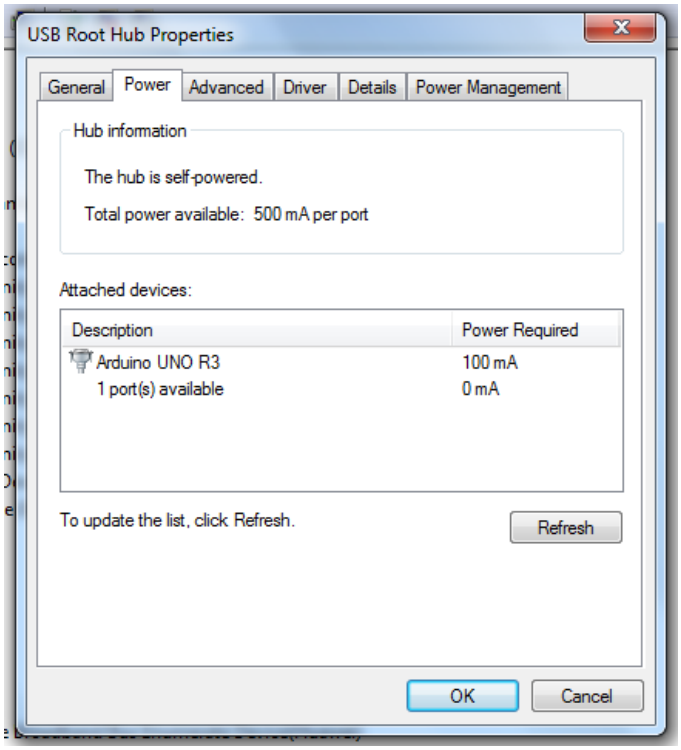


Figure 8.23, Measurement of the Passing Current at Receiver

8.4.2.2 Measurements of Power Consumption at the Receiver from the Security Report

The power consumption is the multiplication of the passing current and the input voltage. The measured power consumption is at receiver which is the security manager.

The sent data for one threat is 104 bits. The power consumption is 5 V multiplied by 100 mA which is equal to 500 mw. For 1 second, the energy consumption is 500 mille Joule.

We test the proposed model with 1A.Hour battery. The battery can remain for 1A.Hour/100mA which is equal to 10 Hours. For large network, we assume that the total threats reported at a security manager are 25 threats in a day therefore; we can send the threats in 25 Seconds for one day. The transmitted data current is variable according to

the distance between the sensor node which is transmitter and the security manager which is the receiver. The transmitted data current is from 100 mA to 300 mA. For the received data at the security manager, the battery can remain for 10H/25 Seconds/day which is equal to 360X4 days. If the transmitted data power consumption is three times the power consumption of the received data, therefore; the battery can remain for 360 days which is low power consumption. The transmitter can send four threats in one second which is the maximum transmitted data from the transmitter.

8.4.2.3 Plaintext Input Data to Transmitter

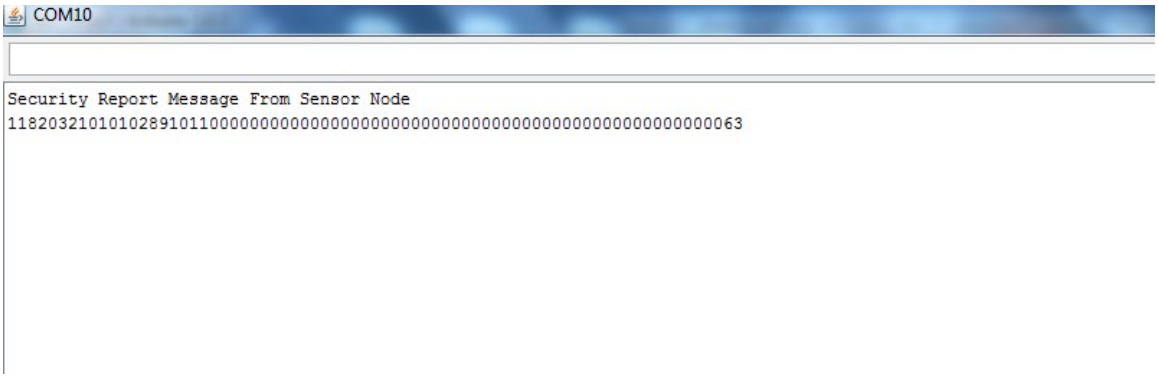


Figure 8.24, Security Report Content Input to Transmitter

Figure 8.24 shows the security report content which is 1 then 18 then 20 then 32 then 10 then 10 then 10 then 2 then 8 then 9 then 10 then 11 where 1 is the count, 18 is the hours and 20 is the minutes and 32 is the seconds, 10 is the attack ID which is Sybil attack, 10 is the attacked node ID, 10 is the replica number, 2 is the reputation of the node and 8 then 9 then 10 then 11 are the data integrity of the security report.

8.4.2.4 Ciphertext Output Data from Transmitter

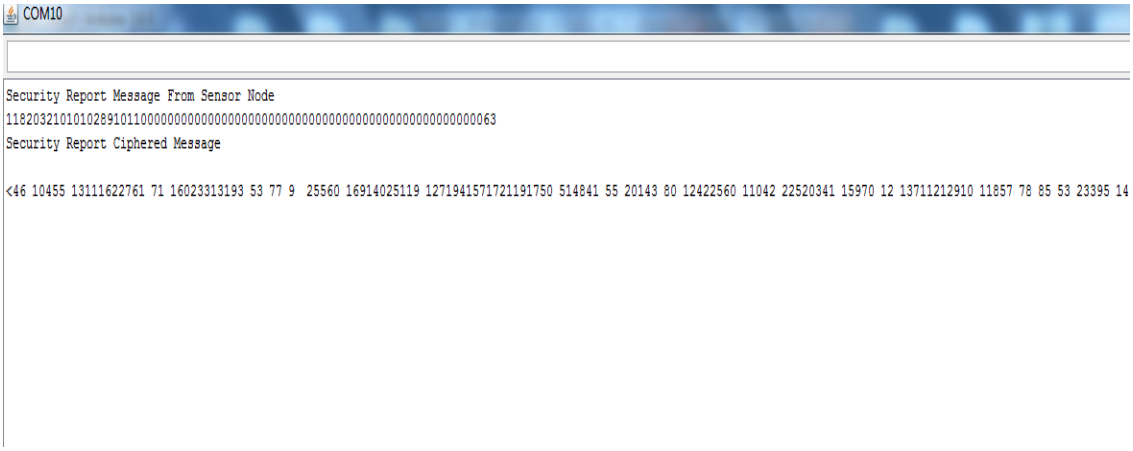


Figure 8.25, Ciphertext Data Output from Transmitter

Figure 8.25 shows the security report ciphered message which is the ciphertext output from the transmitter.

8.4.2.5 Data at Receiver Output

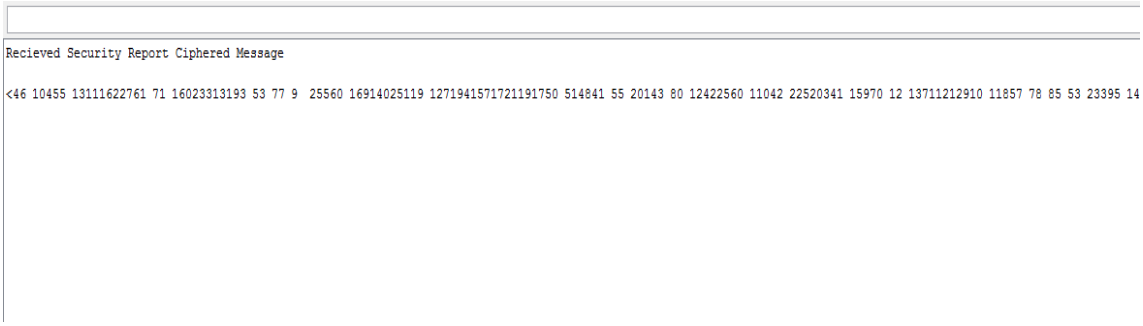


Figure 8.26, Ciphered Security Report Input to Receiver

Figure 8.26 shows the input ciphered security report at the receiver which is equal to the ciphered security report at the output of the transmitter.

Figure 8.27, Security Report Output at Receiver

Figure 8.27 shows the decrypted security report at the receiver output which is equal to the input security report at the transmitter input.

8.5 Comparison between our Work and Previous Works

To the best of our knowledge, there is no scheme in the open literature that addresses the base station failure. The current security schemes proposed for wireless sensor networks lack the ability to provide reliable network recovery in the case of base station failure.

The power consumption of the received security report is 500 mw which is low power consumption. This enables the security manager to receive and send security reports without affecting the lifetime of the security manager.

8.6 Summary

The design and hardware implementation of reliable network recovery from base station failure was implemented on Arduino Uno microcontroller boards. The Transceiver used is X-Bee 1 mw series 1 module. The motion detection sensor is X-Band Doppler Radar motion detection sensor. The X-Bee transceivers are programmed using programmer board and X-CTU program. The output data was monitored using serial monitor cable and HyperTerminal Program. The code of the transmitter, the code of the receiver and the AES encryption algorithm code were done on Arduino Software as shown in Appendix A. The power measurements of the received security report at the security manager show that the reliable network recovery from base station failure has low power consumption.

CHAPTER 9

CONCLUSION and FUTURE WORK

Wireless sensor networks are a unique class of mobile Ad Hoc network consisting of tiny low-cost resource constrained devices that have the ability to sense their environment, to aggregate and to send the data to a destination. The deployment nature and limitations of the nodes resources as well as the wireless communication channel make sensor networks susceptible to a variety of new attacks in addition to the attacks which occur in mobile Ad Hoc networks. Deployment of sensor networks has been envisioned in many sensitive applications such as military operations and health care. Despite advances in miniaturization and other developments in sensor networks occurring at a very fast pace, security within sensor networks requires great effort.

Traditional security measures require heavy communication and computational resources which are beyond the resource constrains of sensor nodes. In this research, it has been argued that cryptographically complex security solutions for sensor networks are not viable for many reasons: firstly, the energy, memory and transmission range limitations; secondly, the wireless channel limitations; thirdly, the deployment nature of sensor nodes being left unattended after deployment; and fourthly, the need to keep costs low to enable dense deployment. Instead, sensor networks need a balanced and comprehensive solution, which is efficient, effective and has low security overheads. Bearing these factors in mind, a novel security framework for wireless sensor networks has been proposed.

Comment on the impact that a node is assigned to be a security manager:

1- Security managers store the security reports.
2- Security managers store the distributed users tables for the new dynamic secret sharing algorithm.
3- Security managers start the new compromised nodes detection algorithm.
4- Security managers start and perform the key management process.
5- Security managers are every two layers to reduce the stored data at the security managers.

In chapter four, we proposed the first security architecture to achieve secure and reliable network recovery from base station failure. Concretely, we proposed a secure and reliable network recovery from base station failure of surveillance wireless sensor network in hostile environment to improve the security data survival capability in the presence of base station failure. We further enhance such scheme by employing distributed security managers and distributed users' table. Our scheme is resilient to base station failure through our designed data storage and recovery systems.

The performance analysis and the simulation results of our proposed hierarchical secure data storage and recovery system provide the WSN with high confidence for secure and reliable network recovery from the base station failure of surveillance WSN in hostile environment.

In chapter five, we proposed the overlapped groups-based compromised nodes detection scheme to early detect the node compromise attack in the first stage. Concretely, the simulation results showed that by building groups among neighboring sensor nodes in a local area, physical node compromise attack can be detected immediately. Also, the simulation results showed that the proposed detection scheme has high detection rate.

The third component of SurvSec security architecture is a new compromised nodes detection algorithm at the first stage against collaborative work of group of attackers compromising sensor nodes at the same time.

The performance analysis and the simulation results of our proposed overlapped groups based compromised nodes detection algorithm provide the WSN with high confidence for early detection of compromised nodes.

In chapter six, we proposed a novel hybrid and dynamic key management scheme for Wireless Sensor Networks which utilizes Elliptic Curve Cryptography and the symmetric key cryptography. We proposed a hybrid authenticated key-establishment protocol, in which we reduce the computation intensive elliptic curve scalar multiplication of a random point at the sensor side, and use symmetric key cryptographic operations instead. On the other hand, it authenticates the two identities based on elliptic curve implicit certificates, and solves the key distribution and storage problems, which are typical bottlenecks in pure symmetric key-based protocols. The hybrid key

establishment protocol has less sensor side computation complexity compared to other public-key based key establishment protocols.

In addition, we designed a dynamic key management based on rekeying, keys revocation and addition of new nodes which significantly increase the resiliency of the network to compromised node attack, and collusion attack. The performance evaluation and security analysis show that our proposed key management scheme has good communication overhead, storage overhead, computations overhead and it provides perfect scalability and resiliency against node capture.

In chapter seven, we proposed a new encryption architecture which is called the spread spectrum encryption architecture. This encryption architecture is based on the unpredictability principle where we choose one algorithm from two algorithms or one subkey from 16 subkeys at each round and the output from the two algorithms is XORed. Our newly designed SSEA3 model is easily implemented in both software and hardware. This new encryption architecture will be an essential architecture to the field of post-quantum cryptography.

The results proved that the architecture with the advantages of low design cost, and strong security level can be implemented for post-quantum cryptography. The SSEA3 is a strong barrier for cryptanalysis. Besides, each plaintext block is encrypted with a different algorithm and different subkeys group which is an obstacle for cryptanalysis. SSEA3 has high speed as it has only 3 rounds AES-256.

The discovery of (possibly currently non-existing) methods to break the technique (if they exist as such) remains an open problem and possible future work.

In chapter eight, we hardware implemented the reliable network recovery from base station failure using Arduino Uno Microcontroller Boards.

Comment on why we use MATLAB to simulate the new architecture not OPNET:

We simulate the new security architecture using MATLAB not OPNET because we are working on the application layer not on the physical layer or data link layer or MAC layer or network layer and the new protocol is consisting of messages.

Summary of Contributions:

The **contributions** of this thesis are:

1. Security Managers for Reliable Network Recovery from Base Station Failure.
2. Secure Data Storage for Reliable Network Recovery from Base Station Failure
3. SurvSec Overlapped Groups to Early Detect Compromised Nodes.
4. SurvSec Hybrid and Dynamic Key Management Scheme.
5. SurvSec Spread Spectrum Encryption Architecture for Post Quantum Computing.
6. Hardware implementation of reliable network recovery from base station failure.

Comment on the draw backs of SurvSec Security architecture:

1. SurvSec security architecture needs public key cryptography for post-quantum computing.
2. SurvSec security architecture needs protection from routing attacks.

Comparison between SurvSec and other Security Architectures:

There are 6 security schemes in WSN which are the followings:

1- Survivable WSN [176] which allows WSN to work under attacks.

2- Security protocols such as TinySec [47], SPINS [29], LEAP [30], SM [48], Zigbee [49], TinyECC [50], Minisec [172], SenSec [175], LSec [173], and LiSP [174] which provide the data with confidentiality, authentication, integrity and freshness.

3- Cross Layers Security architecture such as Intelligent Security Agent ISA [177] which optimize the security architecture and discard all redundancies. We found that the introduction of excessive and uncontrolled interactions can break the design of the system, hindering its usefulness and longevity therefore, cross layers security architecture is more difficult to develop and maintain, as there may be some new dependencies that must be taken into account.

4- Adaptive security schemes such as Flexisec [178] to allow different security levels for different levels of attacks.

5- Intrusion Detection System IDS [179] which has very high implementation cost.

6- SSL [180] and IPSec [181]. They do not secure all attacks where SSL at transport layer and IPSec at network layer.

Table 9.1, Comparison between SurvSec and other Security Protocols

	SurvSec	TinySec [47]	SPINS [29]	MiniSec [172]	LEAP [30]	SM [48]	Zigbee [49]	TinyECC [50]	LSec [173]	SenSec [175]	LiSP [174]
Confidentiality	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No	Yes
Confidentiality for Post QC	Yes	No	No	No	No	No	No	No	No	No	No
Authentication	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Data integrity	No	No	Yes	No	No	No	No	No	No	No	Yes
Data freshness	No	No	Yes	Yes	No	No	No	No	No	No	No
Recovery from BS failure	Yes	No	No	No	No	No	No	No	No	No	No
Distributed security	Yes	No	No	No	No	No	No	No	No	No	No
Security related data storage	Yes	No	No	No	No	No	No	No	No	No	No
Compromised nodes detection by attackers at the first stage	Yes	No	No	No	No	No	No	No	No	No	No
Key management	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	No	No
Hybrid and dynamic key management	Yes	No	No	No	No	No	No	No	No	No	No

All of the above mentioned security schemes do not solve the problem of reliable network recovery from base station failure.

Table 9.1 compares between different security protocols where SurvSec is the only security protocol for post-quantum computing. Also, SurvSec is the only security architecture for reliable network recovery from base station failure. Furthermore, SurvSec is the only protocol which utilizes the distributed security concept for WSN. Moreover, SurvSec has hybrid and dynamic key management system. Finally, SurvSec is the only

protocol which securely stores the security information of sensor nodes in surveillance WSN.

SurvSec Overall Storage Overhead:

Every sensor node in the network except the security managers stores 7 keys:

1- Public key.
2- Public key of SM and BKSM.
3- Private key,
4- Two keys for symmetric key encryption each of 256 bits,
5- Group key for compromised nodes detection algorithm.

Every security manager (SM) placed every two layers with six nodes underneath stores 18 keys:

1- Public key.
2- Private key,
3- Two keys for symmetric key encryption for each node underneath the SM with a total of 12 keys for six nodes underneath the SM each key of 256 bits,
4- Three keys for its upper layer SM, lower layer SM and its upper layer node,
5- Group key for compromised nodes detection algorithm.

Finally, the security manager is every two layers to lower storage overheads.

Future work:

We are heading towards a future of wide scale usage of wireless senor networks where wireless sensor networks will have high connectivity and have the ability to deliver dense communications at very low cost.

For future research we propose extending this security framework to include trust establishment and trust management in sensor networks. Besides this we have an interest in exploring and solving security issues in multimedia and biometric security, cyber security and information assurance, protection against identity theft, and forensic computing.

To address these unique security concerns, it would be imperative to study the adjacent technological advances in distributed systems, ubiquitous computing, broadband wireless communication, nanofabrication and bio-systems.

Also, we propose a future research towards reliable network recovery from multiple base station failure of surveillance WSN in hostile environment.

Furthermore, we propose a future research towards secure multipath routing for surveillance WSN in hostile environment.

Moreover, the most glamour future research will be the development of spread spectrum encryption architecture over Elliptic Curve Cryptography (ECC) which will result in public key cryptography for post-quantum computing.

Security architecture is not limited to our design but it is very big such as the followings:

1- Secure routing.

2- Secure synchronization.

3- Secure power management.

4- Reputation system.

5- Secure data aggregation.

6- Swarm protocols for routing and security.

7- Broadcasting authentication.

8- Digital signature.

Finally, our future work is to simulate our designed SurvSec security architecture with OPNET. Then we will add the other security components of the security architecture.

References

[1] Mahmood Ali, Annette Böhm, and Magnus Jonsson, "*Wireless Sensor Networks for Surveillance Applications – A Comparative Survey of MAC Protocols*", The Fourth International Conference on Wireless and Mobile Communications, **IEEE** 2008.

[2] Tatiana Bokareva, Wen Hu, Salil Kanhere, Branko Ristic, Neil Gordon, Travis Bessell, Mark Rutten and Sanjay Jha, "*Wireless Sensor Networks for Battlefield Surveillance*", Proceedings of The Land Warfare Conference (LWC), October 2006.

[3] Mario Lopez-Ramos, Jérémie Leguay, and Vania Conan, "*Designing a Novel SOA Architecture for Security and Surveillance WSNs with COT*S", International Conference on Mobile Ad-hoc and Sensor Systems 2007, **IEEE** 2007.

[4] Lin Gu, Dong Jia, Pascal Vicaire, Ting Yan, Liqian Luo, Ajay Tirumala, Qing Cao, Tian He, John A. Stankovic, Tarek Abdelzaher, and Bruce H. Krogh, "*Lightweight Detection and Classification for Wireless Sensor Networks in Realistic Environments*", Proceedings of the 3rd international conference on Embedded networked sensor systems 2005, **ACM** 2005.

[5] Tian He, Sudha Krishnamurthy, John A. Stankovic, Tarek Abdelzaher, Liqian Luo, Radu Stoleru, Ting Yan, Lin Gu, Jonathan Hui, and Bruce Krogh, "*Energy-Efficient Surveillance System Using Wireless Sensor Networks*", MobiSYS'04, June 6–9, 2004, Boston, Massachusetts, USA, **ACM** 2004.

[6] Jing Deng, Richard Han, and Shivakant Mishra, "*Intrusion Tolerance and Anti-Traffic Analysis Strategies For Wireless Sensor Networks*", Proceedings of the International Conference on Dependable Systems and Networks DSN 2004, **IEEE** 2004.

[7] Jing Deng, Richard Han, and Shivakant Mishra, "*Countermeasures Against Traffic Analysis Attacks in Wireless Sensor Networks*", Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks SECURECOMM 2005, Pp 113 – 126, **IEEE** 2005.

[8] Jing Deng, Richard Han, and Shivakant Mishra, "*Enhancing Base Station Security in Wireless Sensor Networks*", Technical Report CU-CS-951-03, Department of Computer Science, University of Colorado, 2003.

[9] Soo Kim, Jeong-Gil Ko, Jongwon Yoon and Heejo Lee, "*Multiple-Objective Metric for Placing Multiple Base Stations in Wireless Sensor Networks*", Proc. of International Symposium on Wireless Pervasive Computing (ISWPC) 2007, **IEEE** 2007.

[10] Stefan Ransom, Dennis Pfisterer, and Stefan Fischer, "Comprehensible *Security Synthesis for Wireless Sensor Networks*", Proceedings of the 3rd international workshop on Middleware for sensor networks, **ACM** 2008.

[11] Shashidhar Rao Gandham, Milind Dawande, Ravi Prakash and S. Venkatesan, "*Energy Efficient Schemes for Wireless Sensor Networks with Multiple Mobile Base Stations*", GLOBECOM 2003, **IEEE** 2003.

[12] Ertan Onur, Cem Ersoy and Hakan Deliç, "*Quality of Deployment in Surveillance Wireless Sensor Networks*", International Journal of Wireless Information Networks, Volume 12, Number 1, July 2005, pp. 61-67, **Springer** 2005.

[13] Ting Yan, Tian He, and John A. Stankovic, "*Differentiated Surveillance for Sensor Networks*", SenSys'03, November 5–7, 2003, **ACM** 2003.

[14] T.Kavitha, and D.Sridharan, "*Security Vulnerabilities In Wireless Sensor Networks: A Survey*", Journal of Information Assurance and Security 5 (2010) pp. 31-44, 2010.

[15] Michael Winkler, Klaus-Dieter Tuchs, Kester Hughes, and Graeme Barclay, "*Theoretical and Practical aspects of Military Wireless Sensor Networks*", Journal of Telecommunications and Information Technology, 2008.

[16] Xinfeng Li, Xiaoyuan Wang, Nan Zheng, Zhiguo Wan, and Ming Gu, "*Enhanced Location Privacy Protection of Base Station in Wireless Sensor Networks*", 2009 Fifth International Conference on Mobile Ad-hoc and Sensor Networks, **IEEE** 2009.

[17] Eylem Ekici, Yaoyao Gu, and Doruk Bozdag, "*Mobility-Based Communication in Wireless Sensor Networks*", IEEE Communications Magazine, July 2006, **IEEE** 2006.

[18] Jing Deng, Richard Han, and Shivakant Mishra, "*INSENS: Intrusion-Tolerant Routing for Wireless Sensor Networks*", Computer Communications, Volume 29, Issue 2, January 2006, pp. 216-230, **ACM** 2006.

[19]Sushil Kumar Jain, and Kumkum Garg, "*A Hybrid Model of Defense Techniques against Base Station Jamming Attack in Wireless Sensor Networks*", Proceedings of the 2009 First International Conference on Computational Intelligence, Communication Systems and Networks, pp. 102-107, **IEEE** 2009.

[20]Ying-Hong Wang, Hung-Jen Mao, Chih-Hsiao Tsai, and Chih-Chieh Chuang, "*HMRP: Hierarchy-Based Multipath Routing Protocol for Wireless Sensor Networks*", LNCS 3823, pp. 452 – 459, **Springer** 2005.

[21]Subhas Chandra, Mukhopadhyay, and Yueh-Min Huang, "*Sensors: Advancements in Modeling Design Issues Fabrication and Practical Applications*", **Springer**-Verlag: Heidelberg, Germany, 2008.

[22]Kuldeep Yadav , Kalpana Sharma, and Mrinal Ghose, "*Wireless Sensor Networks Security: A New Approach*", In Proceedings of 16th International Conference on Advanced Computing and Communication, ADCOM 2008.

[23]M.J.Carmel, Mary Belinda, C.Suresh, and Gnana Dhas, "*A Study of Security in Wireless Sensor Networks*", MASAUM Journal Of Reviews and Surveys, Volume 1, Issue 1, September 2009.

[24] Yang Xiao, Venkata Krishna Rayi, Bo Sun, Xiaojiang Du, Fei Hu, and Michael Galloway, "*A survey of Key Management Schemes in Wireless Sensor Networks*", Computer Communications 30 (2007) 2314–2341, **ELSEVIER** 2007.

[25] L. Eschenauer, and V.D. Gligor, "*A Key Management Scheme for Distributed Sensor Networks*", Proceedings of the 9th ACM Conference on Computer and Communication Security, **ACM** 2002.

[26] H. Chan, A. Perrig, and D. Song, "*Random Key Pre-distribution Schemes for Sensor Networks*", Proceedings of the IEEE Symposium on Security and Privacy, pp. 197–213, **IEEE** 2003.

[27] D. Liu, and P. Ning, "*Establishing Pair-wise Keys in Distributed Sensor Networks*", Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS '03) (2003) pp. 52–61, **ACM** 2003.

[28] W. Du, J. Deng, Y.S. Han, P.K. and Varshney, "*A Pair-wise Key Pre-distribution Scheme for Wireless Sensor Networks*", Proceedings of the 10th ACM Conference on Computer and Communications (SecurityCCS'03) (2003) pp. 42–51, **ACM** 2003.

[29] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar, "*SPINS: Security Protocols for Sensor Networks*", Proceedings of ACM MOBICOM (2001), **ACM** 2001.

[30 S. Zhu, S. Setia, and S. Jajodia, "*LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks*", Proceedings of The10th ACM Conference on Computer and Communications Security (CCS '03), Washington D.C., October, **ACM** 2003.

[31] W. Du, J. Deng, Y.S. Han, S. Chen, and P.K. Varshney, "*A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge*", Proceedings of INFOCOM 2004, **IEEE** 2004.

[32] X. Du, Y. Xiao, M. Guizani, and H.H. Chen, "*An Effective Key Management Scheme for Heterogeneous Sensor Networks*", Ad Hoc Networks, vol. 5, pp. 24–34, **Elsevier** 2007.

[33] X. Du, M. Guizani, Y. Xiao, S. Ci, and H.H. Chen, "*A Routing-Driven Elliptic Curve Cryptography Based Key Management Scheme for Heterogeneous Sensor Networks*", IEEE Transactions on Wireless Communications, **IEEE** 2009.

[34] D. Malan, M. Welsh, and M.D. Smith, "*A Public-Key Infrastructure for Key Distribution in TinyOS based on Elliptic Curve Cryptography*", Proceedings of 1st IEEE International Conference Communications and Networks (SECON), October 2004, **IEEE** 2004.

[35] N. Gura, A. Patel, A. Wander, H. Eberle, and S.C. Shantz, "*Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs*", Proceedings of the 6th International Workshop on Cryptographic Hardware and Embedded Systems, Boston, Massachusetts, August 2004.

[36] A.S. Wander, N. Gura, and H. Eberle, "*Energy Analysis of Public-key Cryptography for Wireless Sensor Networks*", Proceedings of the 3rd IEEE International Conference on Pervasive Computing and Communications (PERCOM), **IEEE** 2005.

[37] M. Eltoweissy, M. Moharrum, and R. Mukkamala, "*Dynamic Key Management in Sensor Networks*", IEEE Communications Magazine 2006, pp. 122–130, **IEEE** 2006.

[38] F. Anjum, "*Location Dependent Key Management using Random Key Pre-distribution in Sensor Networks*", Proceedings of WiSe'06.

[39] M.F. Younis, K. Ghumman, and M. Eltoweissy, "*Location-aware Combinatorial Key Management Scheme for Clustered Sensor Networks*", IEEE Transactions on Parallel and Distributed Systems 2006, pp. 865–882, **IEEE** 2006.

[40] Michael Chorzempa , Jung-Min Park , and Mohamed Eltoweissy, "*Key Management for Long-lived Sensor Networks in Hostile Environments*", Computer Communications 30 (2007) 1964-1979, **ELSEVIER** 2007.

[41] Y. Cheng and D. Agrawal, "*An Improved Key Distribution Mechanism for Large-Scale Hierarchical Wireless Sensor Networks*" Ad-Hoc Networks, pp. 35–48, **Elsevier** 2007.

[42] D. Huang, M. Mehta, D. Medhi, and L. Harn, "*Location-aware Key Management Scheme for Wireless Sensor Networks*", 2nd ACM workshop on Security of Ad-Hoc and Sensor Networks SASN 04, pp. 29–42, **ACM** 2004.

[43] Olfa Gaddour, Anis Koubaa and Mohamed Abid, "*SeGCom: A Secure Group Communication Mechanism in Cluster-Tree Wireless Sensor Networks*", **IEEE** 2009.

[44] L. Zhang, Z. Hu, Y. Li, and X. Tang, "*Grouping-based Clustering Routing Protocol in Wireless Sensor Networks*", Wireless Communications, Networking and Mobile Computing, Wicom, pp. 2452–2455, 2007.

[45] L. Li, J. Li, L. Tie, and J. Pan, "*Ackds: An Authenticated Combinatorial Key Distribution Scheme for Wireless Sensor Networks*", the software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007, SNPD, pp. 262–267, 2007.

[46] Cungang Yang, Celia Li, and Jie Xiao, "*Location-based design for secure and efficient wireless sensor networks*", Computer Networks 52 (2008) 3119-3129, **ELSEVIER** 2008.

[47] Karlof, C., Sastry, N., and Wagner, "TinySec: A Link Layer Security Architecture for Wireless Sensor Networks", Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, pp. 162 – 175, **ACM** 2004.

[48] Heo, J., and Hong, C.S. "Efficient and Authenticated Key Agreement Mechanism in Low-Rate WPAN Environment", International Symposium on Wireless Pervasive Computing 2006, Phuket, Thailand 16 – 18 January 2006, IEEE 2006.

[49] ZigBee Alliance (2006) ZigBee Security Specification Overview [online], available: http://www.zigbee.org/en/events/documents/december2005_open_house_presentations/zigbee_security_layer_technical_overview.pdf.

[50] Ning, P., "TinyECC: Elliptic Curve Cryptography for Sensor Networks [online], available: http://discovery.csc.ncsu.edu/software/TinyECC/.

[51] Hai Liu, Amiya Nayak, and Ivan Stojmenovi, "*Fault-Tolerant Algorithms/Protocols in Wireless Sensor Networks*", Guide to Wireless Sensor Networks, Computer Communications, **Springer** 2009.

[52] S. Chessa, and P. Maestrini, "*Fault Recovery Mechanism in Single-Hop Sensor Networks*", Computer Communications 28 (2005) 1877–1886, **Elsevier** 2005.

[53] C.-C. Shen, C. Srisathapornphat, and C. Jaikaeo, "*An Adaptive Management Architecture for Ad-Hoc Networks*", IEEE Communication Magazine, Vol. 41, pp. 108–115, **IEEE** 2003.

[54] R. Badonnel, R. State, and O. Festor, "*Management of Mobile Ad-Hoc Networks: Information Model and Probe-based Architecture*", International Journal of Network Management, Vol. 15, No. 5, pp. 335–347, 2005.

[55] W. Chen, N. Jain, and S. Singh, "*ANMP: Ad-Hoc Network Management Protocol*", IEEE JSAC, Vol. 17, No. 8, pp. 1506–1531, **IEEE** 1999.

[56] J. Zhao, R. Govindan, and D. Estrin, "*Computing Aggregates for Monitoring Wireless Sensor Networks*", In Proceedings of SNPA, 2003.

[57] N. Ramanathan, E. Kohler, and D. Estrin, "*Towards a Debugging System for Sensor Networks*" International Journal of Network Management, Vol. 15, pp. 223–234, 2005.

[58] D. Starobinski, "*Network Observation System* (NOSY)", http://nislab.bu.edu/nislab/projects/wsn_testbed/nosy.html.

[59] G. Tolle and D. Culler, "*Design of an Application-cooperative Management System for Wireless Sensor Networks*", In Proceedings of EWSN, 2005.

[60] J. Lim, D. Kiskis, and K. Shin, "*Aglet: Modular Coordination and Management Framework*", EECS, University of Michigan, Ann Arbor.

[61] J. Lim, D. Kiskis, and K. Shin, "*System Support for Management of Networked Low-Power Sensors*", In Proceedings of IEEE/IFIP NOMS, **IEEE** 2006.

[62] L. B. Ruiz, J. M. Nogueira, and A. A. F. Loureiro, "*MANNA: A Management Architecture for Wireless Sensor Networks*", In IEEE Commmunications Magazine, Vol. 41, No. 41, pp. 116–125, **IEEE** 2003.

[63] L. B. Ruiz, I. G. Siqueira, L. B. e Oliveira, H. C. Wong, J. M. S. Nogueira, and A. A. F. Loureiro, "*Fault Management in Event-driven Wireless Sensor Networks*", MSWiM '04: Proceedings of the 7th ACM international symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, **ACM** 2004.

[64] W. L. Lee, A. Datta, and R. Cardell-Oliver, "*WinMS: Wireless Sensor Network-Management System, An Adaptive Policy-Based Management for Wireless Sensor Networks*", School of Computer Science & Software Engineering, The University of Western Australia, CSSE Technical Report UWA-CSSE-06-001, June 2006.

[65] M. M. Alam, M. Mamun-Or-Rashid, and C. S. Hong, "*WSNMP: A Network Management Protocol for Wireless Sensor Networks*", in 10th International Conference on Advanced Communication Technology, (ICACT'08) vol. 1, pp. 742-747, 2008.

[66] B. Deb, S. Bhatnagar, and B. Nath, "*Wireless Sensor Networks Management*", http://www.research.rutgers.edu/_bdeb/sensornetworks.html, 2005.

[67] Muhammad Z Khan, Madjid Merabti, and Bob Askwith, "*Design Considerations for Fault Management in Wireless Sensor Networks*", 2009.

[68] K. Liu, M. Li, Y. Liu, M. Li, Z. Guo, and F. Hong, "*Passive Diagnosis for Wireless Sensor Networks*", Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems, Sensys'08, pp. 113-126, **ACM** 2008,.

[69] S. Jessica, B. Dirk, and D. Glenn, "*Efficient Tracing of Failed Nodes in Sensor Networks*", Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications, Atlanta, Georgia, USA, pp. 122-130, **ACM** 2002,.

[70] G. Venkataraman, S. Emmanuel, and S. Thambipillai, "*A Cluster- Based Approach to Fault Detection and Recovery in Wireless Sensor Networks*", in 4th International Symposium on Wireless Communication Systems, ISWCS'07. , pp. 35-39, 2007.

[71] C. Yao-Chung, L. Zhi-Sheng, and C. Jiann-Liang, "*Cluster based Self-organization Management Protocols for Wireless Sensor Networks*", IEEE Transactions on Consumer Electronics, vol. 52, pp. 75-80, **IEEE** 2006.

[72] Bin Zhang, and Guohui Li, "*Analysis of Network Management Protocols in Wireless Sensor Network*", 2008 International Conference on Multi Media and Information Technology, **IEEE** 2008.

[73] Nithya Ramanathan, Kevin Chang, Rahul Kapur, Lewis Girod, Eddie Kohler, and Deborah Estrin, "*Sympathy for the Sensor Network Debugger*", SenSys'05, November 2–4, 2005, **ACM** 2005.

[74] Mark Shaneck, Karthikeyan Mahadevan, Vishal Kher, and Yongdae Kim, "*Remote Software-Based Attestation for Wireless Sensors*", The Lecture Notes in Computer Science, pp. 27-41, **Springer** 2005.

[75] Taejoon Park, and Kang G. Shin, "*Soft Tamper-Proofing via Program Integrity Verification in Wireless Sensor Networks*", IEEE Transactions on Mobile Computing, Vol. 4, No. 3, May/June 2005, **IEEE** 2005.

[76] Xiaojiang Du, "*Detection of Compromised Sensor Nodes in Heterogeneous Sensor Networks*", **IEEE** "ICC" 2008.

[77] Yi Yang, Xinran Wang, Sencun Zhu, and Guohong Cao, "*Distributed Software-based Attestation for Node Compromise Detection in Sensor Networks*", Proceedings of the 26th IEEE International Symposium on Reliable Distributed Systems, **IEEE** 2007.

[78] Tamer AbuHmed, Nandinbold Nyamaa, and DaeHun Nyang, "*Software-Based Remote Code Attestation in Wireless Sensor Network*", **IEEE** "GLOBECOM" 2009.

[79] Han-Yu Lin and Yi-Shiung Yeh, "*Dynamic Multi-Secret Sharing Scheme*", International Journal of Contemporary Mathematical Sciences, Vol. 3, No. 1, pp. 37 – 42, 2008.

[80] Qian Wang, Kui Ren, Wenjing Lou, and Yanchao Zhang, "*Dependable and Secure Sensor Data Storage with Dynamic Integrity Assurance*", **IEEE** "INFOCOM" 2009.

[81] Wei Ren, Yi Ren, and Hui Zhang, "*HybridS: A scheme for Secure Distributed Data Storage in WSNs*",International Conference on Embedded and Ubiqtous Computing, **IEEE** 2008.

[82] R. Rivest, and Adi Shamir, "*How to Share a Secret*", **ACM** 1979.

[83] Xiaojiang Du, and Hsiao–Hwa Chen, "*Security in Wireless Sensor Networks*", August 2008, **IEEE** Wireless Communications.

[84] Taeshik Shon, Bonhyun Koo, Hyohyun Choi, and Yongsuk Park, "*Security Architecture for IEEE 802.15.4-based Wireless Sensor Network*", ISWPC'09: Proceedings of the 4th International Conference on Wireless Pervasive Computing, **IEEE** 2009.

[85] Meng-Yen Hsieh, Yueh-Min Huang, and Han-Chieh Chao, "*Adaptive Security Design with Malicious Node Detection in Cluster-based Sensor Networks*", Computer Communications 30 (2007), **Elsevier** 2007.

[86] Anelia Mitseva, Efthimia Aivaloglou, Maria Marchitti, Neeli Rashmi Prasad, Charalabos Skianis, Stefanos Gritzalis, Adrian Waller, Tim Baugé, and Sarah Pennington, "*Towards Adaptive Security for Convergent Wireless Sensor Networks in Beyond 3G Environments*", Wireless Communications and Mobile Computing, **Wiley InterScience**, 2008.

[87] P. Bonnet, J. Gehrke, and P. Seshadri, "Towards Sensor Database Systems", In the Proceedings of the Second International Conference on Mobile Data Management, pp. 3–14, **Springer** 2001.

[88] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. Tag, "A Tiny Aggregation Service for Ad-Hoc Sensor Networks", In the Proceedings of the 5th Symposium on Operating Systems Design and Implementation OSDI, **ACM** 2002.

[89] Y. Yao and J. Gehrke, "Query Processing in Sensor Networks", In the Proceedings of Conference of Innovative Data Systems Research CIDR, **IEEE** 2004.

[90] M. Sharaf, J. Beaver, A. Labrinidis, and P. Chrysanthis, "TiNA: A scheme for Temporal Coherency-aware in-Network Aggregation", In Proceedings of the 3rd ACM International Workshop on Data Engineering for Wireless and Mobile Access MobiDE, **ACM** 2003.

[91] Abhishek Parakh and Subhash Kak, "A Distributed Data Storage Scheme for Sensor Networks", MobiSec 2009.

[92] Norbert Siegmund, Marko Rosenmuller, Guido Moritz, Gunter Saake, and Dirk Timmermann, "Towards Robust Data Storage in Wireless Sensor Networks", the **IETE** Journal 2009.

[93] R. D. Pietro, L. V. Mancini, C. Soriente, A. Spognardi, and G. Tsudik, "Data Survival in Unattended Sensor Networks", In 6th Annual International Conference on Pervasive Computing and Communications (PerCom '08), **IEEE** 2008.

[94] N. Subramanian, C. Yang, and W. Zhang, "Securing Distributed Data Storage and Retrieval in Sensor Networks", In International Conference on Pervasive and Mobile Computing (PerCom 2007), **Elsevier** 2007.

[95] S. R. Madden, M. J. Franklin, J. M. Hellerstein and W. Hong, "*TinyDB: An Acquisitional Query Processing System for Sensor Networks*", **ACM** Transactions on Database Systems, March 2005, Vol.30, pp.122-173.

[96] T. Liu, C. M. Sadler, P. Zhang, and M. Martonosi, "*Implementing Software on Resource-constrained Mobile Sensors: Experiences with Impala and Zebranet*", MobiSys, 2004.

[97] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin, and P. Corke, "*Data Collection, Storage, and Retrieval with an Underwater Sensor Network*", SenSys, 2005.

[98] Liqian Luo, Chengdu Huang, Tarek Abdelzaher and John Stankovic, "*EnviroStore: A Cooperative Storage System for Disconnected Operation in Sensor Networks*", INFOCOM, **IEEE** 2007.

[99] J. Newsome and D. Song., "*GEM: Graph Embedding for Routing and Data-centric Storage in Sensor Networks without Geographic Information*", Proceedings of 1st International Conference on Embedded Networked Sensor, pp. 76–88, **ACM** 2003.

[100] Cheng Tien Ee, Sylvia Ratnasamy, and Scott Shenker, "*Practical Data-centric Storage*", Proceedings of the 3rd Conference on Networked Systems Design and Implementation, **ACM** 2006.

[101] Mohamed Aly, Anandha Gopalan, Jerry Zhao, and Adel M. Youssef, "*STDCS: A Spatio-Temporal Data-Centric Storage Scheme for Real-Time Sensornet Applications*", Proceedings of the 6th International Conference on AD-HOC Networks and Wireless Networks, **IEEE** 2008.

[102] Bo Sheng, Qun Li, and Weizhen Mao, "*Data Storage Placement in Sensor Networks*", MobiHoc'06, May 22–25, **ACM** 2006.

[103] Song Lin, Benjamin Arai, and Dimitrios Gunopulos, "*Reliable Hierarchical Data Storage in Sensor Networks*", 19th International Conference on Scientific and Statistical Database Management (SSDBM 2007), **IEEE** 2007.

[104] S. Ratnasamy, B, Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin and F. Yu, "*Data-Centric Storage in Sensornets with GHT, a Geographic Hash Table*", Mobile Networks Applications, August 2003, Vol.8, pp.427-442.

[105] Kai Xing, Xiuzhen Cheng and Jiang Li, "Location-Centric Storage for Sensor Networks", IEEE International Conference on Mobile Ad-Hoc and Sensor Systems, **IEEE** 2005.

[106] K. Chang, N. Yau, M. Hansen, and D. Estrin, "SensorBase.org: A Centralized Repository to slog Sensor Network Data," in Proceedings the 2nd IEEE International Conference on Distributed Computing in Sensor Systems, June 2006.

[107] B. Karp and H. T. Kung, "*GPSR: Greedy Perimeter Stateless Routing for Wireless Networks*", In MobiCom, pp. 243–254, 2000.

[108] J. C. Abhishek Ghose, and Jens Grossklags, "*Resilient Data-centric Storage in Wireless Ad-Hoc Sensor Networks*", Proceedings of Mobile Data Management, pp. 45–62, 2003.

[109] S.Reed and G.Solomon, "Polynomial Codes over Certain Finite", IEEE 1960.

[110] Wei Ren, Junge Zhao, and Yi Ren, "MSS: A Multi-level Data Placement Scheme for Data Survival in Wireless Sensor Networks", **IEEE** 2009.

[111] J. He, and E. Dawson, "Multistage Secret Sharing Based on One-Way Function", Electronics Letters, 30 (19) (1994) 1591-1592.

[112] Y.J. Geng, X.H. Fan, and F. Hong, "A New Multi-secret Sharing Scheme with Multi-policy", The 9th International Conference on Advanced Communication Technology, Vol. 3, 2007, pp. 1515-1517.

[113] W.A. Jackson, K. M. Martin, and C. M. O'Keefe, "On Sharing Many Secrets", Advances in Cryptology − ASIACRYPT'94, **Springer**-Verlag, 1994, pp.42-54.

[114] G. J. Simmons, "How to (Really) Share a Secret," in the Proceedings of CRYPTO88, 1988, pp. 390–448.

[115] T. Tassa, "Hierarchical Threshold Secret Sharing," in the Proceedings of TCC04, 2004.

[116] M. Belenkiy, "Disjunctive Mmulti-level Secret Sharing," Cryptology ePrint Archive, Report 2008/018, 2008, http://eprint.iacr.org/.

[117] C. Hartung, J. Balasalle, and R. Han, "Node compromise in sensor networks: the need for secure systems," in Technical Report CU-CS- 990-05, Dept. of Comp Sci, Univ of Colorado at Boulder, Jan 2005.

[118] H. Song, L. Xie, S. Zhu, and G. Cao, "Sensor node compromise detection: the location perspective," in IWCMC'07, Honolulu, Hawaii, USA, Aug. 2007.

[119] P. Kyasanur and H. Vaidya, "Detection and handling of mac layer misbehavior in wireless networks," in IEEE DSN, 2003.

[120] S. Zhu, S. Setia, S. Jajodia, and P. Ning, "An interleaved hop-by hop authentication scheme for filtering of injected false data in sensor networks," in IEEE Symposium on Security and Privacy'04, 2004.

[121] H. Yang, F. Ye, Y. Yuan, S. Lu, and W. Arbaugh, "Toward resilient security in wireless sensor networks," in ACM MobiHoc'05, 2005.

[122] F. Ye, H. Yang, and Z. Liu, "Catching moles in sensor networks," in IEEE ICDCS'07, Jun, 2007.

[123] A. Seshadri, M. Luk, E. Shi, A. Perrig, L. van Doorn, and P. Khosla, “Pioneer: verifying integrity and guaranteeing execution of code on legacy platforms,” in SOSP, Oct. 2005.

[124] D. Spinellis, “Reflection as a mechanism for software integrity verfication,” in ACM Trans. Inf. Syst. Secu., Vol, 3, No, 1, 2000.

[125] Xiaodong Lin, “CAT: Building Couples to Early Detect Node Compromise Attack in Wireless Sensor Networks”, IEEE "GLOBECOM" 2009.

[126] Wei Ding, Yingbing Yu, and Sumanth Yenduri, “Distributed First Stage Detection for Node Capture”, IEEE Globecom 2010.

[127] Jun-Won Ho, Matthew Wright, and Sajal K. Das, “ZoneTrust: Fast Zone-Based Node Compromise Detection and Revocation in Sensor Networks Using Sequential Analysis”, 2009 28th IEEE International Symposium on Reliable Distributed Systems, IEEE 2009.

[128] J. Deng, R. Han, and S. Mishra, “Secure Code Distribution in Dynamically Programmable Wireless Sensor Networks”, In Proc. International Conference on Information Processing in Sensor Networks, pp. 292–300, ACM 2006.

[129] H. Chan and A. Perrig. “PIKE: Peer Intermediaries for Key Establishment in Sensor Networks”, INFOCOM, 2005.

[130] Chan H, and Perrig A, “Random key predistribution schemes for sensor networks”. In: Proceedings of the 2003 IEEE symposium on security and privacy, May 2003. pp. 197–213.

[131] Liu D, and Ning P. “Establishing pairwise keys in distributed sensor networks”. In: Proceedings of 10th ACM conference on computer and communications security (CCS03). 2003. pp. 41–7.

[132] Yu Z, and Guan Y. “A Robust group-based key management scheme for wireless sensor networks”. In: Proceedings of IEEE wireless communications and networking conference (WCNC 2005), New Orleans, LA USA. IEEE Press; 2005. pp. 13–7.

[133] Lee J, and Stinson DR. “Deterministic key predistribution schemes for distributed sensor networks”. In: Proceedings of ACM symposium on applied computing 2004, Lecture notes in computer science, vol. 3357, 2005, Waterloo, Canada, 2004. p. 294–307.

[134] Camtepe SA, and Yener B. “Combinatorial design of key distribution mechanisms for wireless sensor networks”. IEEE/ACM Transactions on Networking (TON) 2007;15(2):346–358.

[135] Qiang Huang, Johnas Cukier, Hisashi Kobayashi, Bede Liu and Jinyun Zhang, “Fast authenticated key establishment protocols for self-organizing sensor networks”, WSNA '03 Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications

[136] D. Liu and P. Ning, “Improving Key Pre-Distribution with Deployment Knowledge in Static Sensor Networks,” ACM Trans. Sensor Networks, 2005, pp 204–39.

[137] D. Liu, P. Ning, and W. Du, “Group-Based Key Pre-Distribution in Wireless Sensor Networks,” Proc. 2005 ACM Wksp. Wireless Security (WiSec 2005), Sept. 2005, pp.11–20.

[138] M. Eltoweissy et al., “Combinatorial Optimization of Key Management in Group Communications,” J. Network and Sys. Mgmt., Special Issue on Network Security, Mar. 2004, p. 332b.

[139] M. Eltoweissy et al., “Group Key Management Scheme for Large-Scale Wireless Sensor Network” Ad Hoc Networks, 2005, pp.796-802.

[140] G. Jolly et al., “A Low-Energy Key Management Protocol for Wireless Sensor Networks,” IEEE 2003, p. 335.

[141] M. Younis, K. Ghumman, and M. Eltoweissy, “Location aware Combinatorial Key Management Scheme for Clustered Sensor Networks,” to appear, IEEE Trans. Parallel and Distrib. Sys., 2006.

[142] DuW, Wang R, and Ning P. “An efficient scheme for authenticating public keys in sensor networks”. MobiHoc, 2005. pp. 58–67.

[143] Watro R, Kong D, Cuti S, Gardiner C, Lynn C, and Kruus P. “Tinypk: securing sensor networks with public key technology”. In: Proceedings of the 2nd ACM workshop on security of ad hoc and sensor networks (SASN 04). New York, NY, USA: ACM Press; 2004. p. 59–64.

[144] Gaubatz G, Kaps J-P, and Sunar B. “Public key cryptography in sensor networks”. In: 1st European workshop on security in ad-hoc and sensor networks (ESAS 2004), 2004.

[145] Zhang J, and Varadharajan V. “Group-based Wireless Sensor Network Security Scheme”. In: The fourth international conference on wireless and mobile communica- tions (ICWMC 2008), July 2008.

[146] D. Liu and 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, pp. 72–82, October 2003.

[147] Katerina Simonova, Alan C. H., Ling, X., and Sean Wang, “Location-aware Key Predistribution Scheme for Wide Area Wireless Sensor Networks”, SASN’06, ACM 2006.

[148] Yanchao Zhang, Wei Liu, Wenjing Lou and Yuguang Fang, “Securing Sensor Networks with Location-Based Keys”, IEEE 2005.

[149] Chunguang Ma, Guining Geng, Huiqiang Wang, and Guang Yang, “Location-aware and secret share based dynamic key management scheme for WSN”, Networks Security, Wireless Communications and Trusted Computing Conference, IEEE 2009.

[150] Krzysztof Piotrowski, Peter Langendoerfer and Steffen Peter, “How Public Key Cryptography Influences Wireless Sensor Node Lifetime”, Proceedings of the fourth ACM workshop on Security of Ad-Hoc and Sensor Networks, ACM 2006, pp. 169-176.

[151] C. Savarese, J. Rabay and K. Langendoen. “Robust Positioning Algorithms for Distributed Ad-Hoc Wireless Sensor Networks”. USENIX Technical Annual Conference, Monterey, CA, June 2002.

[152] Mohd Anuar Jaafar, and Zuriati Ahmad Zukarnain, “Performance Comparisons of AODV, Secure AODV and Adaptive Secure AODV Routing Protocols in Free Attack Simulation Environment”, European Journal of Scientific Research, ISSN 1450-216X Vol.32 No.3 (2009), pp.430-443.

[153] Leonardo B. Oliveira, Hao C. Wong, M. Bern, Ricardo Dahab, and A. A. F. Loureiro, “SecLEACH – A Random Key Distribution Solution for Securing Clustered Sensor Networks”, Proceedings of the Fifth IEEE International Symposium on Network Computing and Applications NCA 06, IEEE 2006, pp. 145-154.

[154] Chunguang Ma, Guining Geng, Huiqiang Wang, and Guang Yang, “Location-aware and secret share based dynamic key management scheme for WSN”, Networks Security, Wireless Communications and Trusted Computing Conference, IEEE 2009, April, pp. 770-773.

[155] Erik Dahmen and Christoph Krau, “Short Hash-Based Signatures for Wireless Sensor Networks”, 8th International Conference on Cryptology and Network Security, ACM 2009, pp. 463-476.

[156] Di Pietro, L. V. Mancini, and A. Mei, "Efficient and resilient key discovery based on pseudo-random key pre-deployment", IEEE Workshop on Wireless, Mobile, and Ad Hoc Networks, April 2004, pp. 2132-2140.

[157] Peter W. Shor, “Algorithms for Quantum Computation: Discrete Logarithms and Factoring”, In IEEE Symposium on Foundations of Computer Science, pages 124–134, 1994.

[158] Lov K. Grover, “A Fast Quantum Mechanical Algorithm for Database Search”, Proceedings, STOC 1996, Philadelphia PA, USA, pages 212-219.

[159] Lov Grover, “Quantum Computers can Search Arbitrarily Large Databases by a Single Query”, Phys. Rev., Letter 79, 4709-4712, 1997.

[160] Bennett, Bernstein, Brassard, and Vazirani, “The strengths and weaknesses of quantum computation”, SIAM Journal on Computing 26(5): 1510-1523, 1997.

[161] Steve Babbage, Christophe De Canni`ere, Anne Canteaut, Carlos Cid, Henri Gilbert, Thomas Johansson, Matthew Parker, Bart Preneel, Vincent Rijmen and Matthew Robshaw, "The eSTREAM Portfolio Final Report", April 15, 2008.

[162] Akihiro Yamamura and Hirokazu Ishizuka, "Quantum Cryptanalysis of Block Ciphers", Communications Research Laboratory, Nukui-Kitamachi Koganei, Tokyo, Japan, Pages 35-43, 2000.

[163] Gilles Piret and François-Xavier Standaert, "Provable security of block ciphers against linear cryptanalysis: a mission impossible?", Springer (LNCS 2008) 50:325–338, 2009.

[164] Hamdy S. Soliman and Mohammed Omari, "Application of Synchronous Dynamic Encryption System in Mobile Wireless Domains", Proceedings of the 1ST ACM international workshop on Quality of service & security in wireless and mobile networks, Montreal, Quebec, Canada, Pages: 24 – 30, 2005.

[165] Bo Dömstedt, and Jesper Jansson, "The Theory of Dynamic Encryption, a New Approach to Cryptography", Dept. of Computer Science, Lund University, Lund, Sweden, 2000.

[166] Tim S. Kumar, C. Paar, J. Pelzl, G. Pfeiffer, and M. Schimmler, "Breaking Ciphers with COPACOBANA A Cost-Optimized Parallel Code Breaker". In Cryptographic Hardware and Embedded Systems, CHES 2006, Proceedings of the 8th International Workshop, Yokohama, Japan, LNCS, Springer-Verlag, October 10-13, 2006.

[167] Sandy Harris, "Exploring Cipher space: Combining stream ciphers and block ciphers", eprint, IACR, November, 2008.

[168] http://csrc.nist.gov/archive/aes/round2/conf3/papers/04-slucks.pdf

[169] National Institute of Standards and Technology. Advanced Encryption Standard (AES). Federal Information Processing Standards Publications FIPS 197 (November 2001) http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[170] Peeter Laud, “Semantics and Program Analysis of Computationally Secure Information Flow”, Lecture Notes in Computer Science, 2001, Volume 2028, 2001, pp. 77-91.

[171] Mark Hachman, Japan 'K Computer' on Top of TOP500 Supercomputer List, November 14, 2011, PC Magazine.

[172] M. Luk, G. Mezzour, A. Perrig, and V. Gligor. “Minisec: A secure sensor network communication architecture”. In Proceedings of IEEE International Conference on Information Processing in Sensor Networks (IPSN), April 2007.

[173] Riaz Ahmed Shaikh, Sungyoung Lee, Mohammad A. U. Khan, and Young Jae Song, “LSec: Lightweight Security Protocol for Distributed Wireless Sensor Network”, PWC 2006, pp. 367-377.

[174] Taejoon Park and Kang G. Shin, "LiSP: A Lightweight Security Protocol for Wireless Sensor Networks," ACM Transactions on Embedded Computing Systems, vol. 3, no. 3, August 2004.

[175] Tieyan Li, Hongjun Wu, Xinkai Wang, Feng Bao; “SenSec Design, I^2 R Sensor Network Flagship Project”; Technical Report TR v1.0.

[176] Yi Qian, Kejie Lu and Tipper, D., “A design for secure and survivable wireless sensor networks”, IEEE 2007, Volume: 14, Issue: 5, pp. 30-37.

[177] Idrees Sarhan Gawdan1, Chee-Onn Chow, Tanveer A. Zia and Qusay, I. Gawdan, “Cross-layer based security solutions for wireless sensor networks”, International Journal of the Physical Sciences Vol. 6(17), pp. 4245-4254, 2 September, 2011.

[178] Devesh C. Jinwala, Dhiren R. Patel and Kankar S.Dasgupta, “Configurable Link Layer Security Architecture for Wireless Sensor Networks”, Proceedings of the World Congress on Engineering 2008 Vol I, WCE 2008, July 2 - 4, 2008, London, U.K.

[179] Khanafer M., Guennoun M., and Mouftah H.T., “Intrusion Detection System for WSN-Based Intelligent Transportation Systems”, GLOBECOM 2010, IEEE 2010.

[180] Wooyoung Jung; Sungmin Hong; Minkeun Ha; Young-Joo Kim; and Daeyoung Kim, "SSL-Based Lightweight Security of IP-Based Wireless Sensor Networks", International Conference on Workshop of Advanced Information Networking and Applications 2009, WAINA '09, IEEE 2009.

[181] Granjal J., Silva R., Monteiro E., Sa Silva J.,and Boavida F., "Why is IPSec a viable option for wireless sensor networks", 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, 2008. MASS 2008, IEEE 2008.

[182] Xinyu Jin ; Putthapipat, P. ; Deng Pan ; Pissinou, N. ; Makki, S.K., "Unpredictable Software-based Attestation Solution for node compromise detection in mobile WSN", GLOBECOM Workshop, 6-10 Dec. 2010, pp. 2059 – 2064, IEEE 2010.

[183] Jokhio, S.H. ; Jokhio, I.A. ; Kemp, A.H., "Node capture attack detection and defence in wireless sensor networks", Wireless Sensor Systems, Journal IET (Volume: 2 , Issue: 3), September 2012, pp. 161 – 169, IEEE 2012.

[184] Misra, S. ; Krishna, P.V. ; Abraham, K.I., "Energy efficient learning solution for intrusion detection in Wireless Sensor Networks", Second International Conference on Communication Systems and Networks (COMSNETS) 2010, 5-9 Jan. 2010, pp. 1-6, IEEE 2010.

[185] Yan Guoqiang ; Duan Weijun ; Ma Chao ; Huang Liang, "RSSI vector attack detection method for wireless sensor networks", 3rd International Conference on Communication Software and Networks (ICCSN) 2011, 27-29 May 2011, pp. 229 – 232, IEEE 2011.

[186] Li, B. ; Doss, R. ; Batten, L.M. ; Schott, W., "Fast Recovery from Node Compromise in Wireless Sensor Networks", 3rd International Conference on New Technologies, Mobility and Security (NTMS) 2009, 20-23 Dec. 2009, pp.1 – 6, IEEE 2010.

[187] Chun-ming Rong ; Eggen, S. ; Hong-bing Cheng, "A novel intrusion detection algorithm for wireless sensor networks", 2nd International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology (Wireless VITAE), Feb. 28 -March 3 2011, pp. 1 – 7, IEEE 2011.

[188] Bharathi, M.V. ; Tanguturi, R.C. ; Jayakumar, C. ; Selvamani, K., "Node capture attack in Wireless Sensor Network: A survey", International Conference on Computational Intelligence & Computing Research (ICCIC), 18-20 Dec. 2012, pp. 1-3, IEEE 2013.

[189] FathiNavid, A.H. ; Aghababa, A.B., "A Protocol for Intrusion Detection Based on Learning Automata in Forwarding Packets for Distributed Wireless Sensor Networks", International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 10-12 Oct. 2012, pp. 373 – 380, IEEE 2012.

[190] Livani, M.A. ; Abadi, M., "Distributed PCA-based anomaly detection in wireless sensor networks", International Conference on Internet Technology and Secured Transactions (ICITST) 2010, 8-11 Nov. 2010, pp. 1-8, IEEE 2010.

[191] Tufail, A. ; Ki-Hyung Kim, "A backbone assisted hybrid key management scheme for WSN", International Conference on Information Society (i-Society) 2011, 27-29 June 2011, pp. 86 – 91, IEEE 2011.

[192] Zhang Min-qing ; Fu Wen-Hua ; Li De-Long, "A new key management scheme based on secret information for WSN", 3rd International Conference on Communication Software and Networks (ICCSN) 2011, 27-29 May 2011, pp. 518 – 521, IEEE 2011.

[193] Xiaopeng Cui ; Yongping Zhang, "A Key Management Scheme Based on Cluster Radiation Matrix in WSN", International Conference on Computer Science and Electronics Engineering (ICCSEE) 2012, 23-25 March 2012, pp. 719 – 722, IEEE 2012.

[194] Rahman, M. ; Sampalli, S. ; Hussain, S., "A robust pair-wise and group key management protocol for wireless sensor network", GLOBECOM Workshops 2010, 6-10 Dec. 2010, pp. 1528 – 1532, IEEE 2010

[195] Alagheband, M.R. ; Aref, M.R., "Dynamic and secure key management model for hierarchical heterogeneous sensor networks", Information Security, IET Journal (Volume:6 , Issue: 4), Dec. 2012, pp. 271 – 280, IEEE 2013.

[196] Jia Hu ; Enjian Bai ; Yang Yang, "A novel key management scheme for hierarchical wireless sensor networks", 12th IEEE International Conference on Communication Technology (ICCT) 2010, 11-14 Nov. 2010, pp. 526 – 529, IEEE 2010.

[197] Poornima, A.S. ; Amberker, B.B., "Logical ring based key management for clustered sensor networks with changing cluster head", International Conference on Signal Processing and Communications (SPCOM) 2010, 18-21 July 2010, pp. 1 – 5, IEEE 2010.

[198] Ruj, S. ; Nayak, A. ; Stojmenovic, I., "Fully secure pairwise and triple key distribution in wireless sensor networks using combinatorial designs", Proceedings of INFOCOM 2011, 10-15 April 2011, pp. 326 – 330, IEEE 2011.

[199] Na Ruan ; Yizhi Ren ; Hori, Y. ; Sakurai, K., "Performance Analysis of Key Management Schemes in Wireless Sensor Network Using Analytic Hierarchy Process", 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom) 2011, 16-18 Nov. 2011, pp. 1739 – 1744, IEEE 2011.

[200] Ruj, S. ; Nayak, A. ; Stojmenovic, I., "Pairwise and Triple Key Distribution in Wireless Sensor Networks with Applications", IEEE Transactions on Computers (Volume:PP , Issue: 99), 12 June 2012, IEEE 2012.

[201] Chunguang Ma ; Guining Geng ; Huiqiang Wang ; Guang Yang, "Location-Aware and Secret Share Based Dynamic Key Management Scheme for Wireless Sensor Networks", International Conference on Networks Security, Wireless Communications and Trusted Computing, NSWCTC '09, 25-26 April 2009, pp. 770 – 773, IEEE 2010.

[202] Yi Gu ; Qishi Wu ; Xiaoshan Cai ; Bond, J., "On efficient deployment of high-end sensors in large-scale Heterogeneous WSNs", 6th International Conference on Mobile Adhoc and Sensor Systems MASS '09, 12-15 Oct. 2009,pp. 912 – 917, IEEE 2010.

[203] http://www.digi.com/xbee/

[204] http://arduino.cc/en/Main/arduinoBoardUno

[205]http://www.parallax.com/Store/Sensors/ObjectDetection/tabid/176/CategoryID/51/List/0/SortField/0/Level/a/catpageindex/2/Default.aspx

[206] http://en.wikipedia.org/wiki/Elliptic_Curve_DSA

[207] Ken Peffers, Tuure Tuunanen, Marcus Rothenberger and Samir Chatterjee, "A Design Science Research Methodology for Information Systems Research", Journal of Management Information System, Volume 24, Issue 3, 2008 Pp. 45-77.

Appendix A

The appendix contains four codes: the transmitter code, the receiver code, the AES encryption code and the motion detection sensor code.

Transmitter code:

```
#include <AES.h>

AES aes ;

byte key[] =

{

  0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00,

} ;

byte plain[] =

{

1,      18,     20,     32,     10,     10,     10,     2,

8,      9,      10,     11,     0,      0,      0,      0,

//0,    0,      0,      0,      0,      0,      0,      0,

//0,    0,      0,      0,      0,      0,      0,      0,

//0,    0,      0,      0,      0,      0,      0,      0,

//0,    0,      0,      0,      0,      0,      0,      0,

//0,    0,      0,      0,      0,      0,      0,      0,

//0,    0,      0,      0,      0,      0,      0,      63
```

```
};

byte my_iv[] =
{
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x01,
} ;

byte cipher [4*N_BLOCK] ;
byte check [4*N_BLOCK] ;
char strval[4];
void setup ()
{
  Serial.begin (9600) ;
}
void loop ()
{
    byte i,j;
    prekey (128, 1) ;
    Serial.print("<") ;
    for(i = 0; i < 16; i++)
    {
      itoa((int)cipher[i], strval, 10);
```

```
     strval[3] = 0;
    for(j = 0; j < 3; j++)
    {
      Serial.print(strval[j]);
      delay(10);
    }
   }
   delay(1000);
}
void prekey (int bits, int blocks)
{
  byte iv [N_BLOCK] ;
  long t0 = micros () ;
  byte succ = aes.set_key (key, bits) ;
  if (blocks == 1)
    succ = aes.encrypt (plain, cipher) ;
  else
  {
    for (byte i = 0 ; i < 16 ; i++)
      iv[i] = my_iv[i] ;
    succ = aes.cbc_encrypt (plain, cipher, blocks, iv) ;
  }
}
```

Receiver code:

```
#include <SoftwareSerial.h>
#include <AES.h>
byte key[] =
{
  0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00,
} ;
byte plain1[16];
byte my_iv[] =
{
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x01,
} ;
char recieve_buffer[64*3];
byte cipher [4*N_BLOCK] ;
byte check [4*N_BLOCK] ;
char strValue[4];
```

```
SoftwareSerial mySerial(10, 11); // RX, TX
int Rx_flage = 0;
void setup()
{
 Serial.begin(9600);
 mySerial.begin(9600);
 mySerial.write("Im ready");
 mySerial.println();
}
void loop()
{
 int i;
 if(Serial.available())
 {
  byte del = Serial.read();
  if(del == '<')
  {
   for(i = 0; i < 16 * 3; i++)
   {
    while(!Serial.available());// wait on rx flag to be asserted
    recieve_buffer[i] = Serial.read();
   }
   Rx_flage = 1;
```

```
  }
}
if(Rx_flage == 1)
{
  int aa;
  char tmpstr[4];
  Rx_flage = 0;
  mySerial.println();
  for(int j = 0; j < 64; j++)
  {
    strValue[0] = recieve_buffer[ (j*3) ];
    strValue[1] = recieve_buffer[ (j*3) + 1];
    strValue[2] = recieve_buffer[ (j*3) + 2];
    strValue[3] = 0;
    aa = (byte)atoi(strValue);
    cipher[j] = (byte)aa;
  }
  Serial.println("Recieved Security Report Ciphered Message");
  Serial.println("");
  for (int z=0; z<16; z++){
    Serial.print(cipher[z]);
  }
  prekey(128, 1) ;
```

```
    Serial.println();
    Serial.println("Recieved Decrypted Security Report Message");
    Serial.println();
    for(int j = 0; j < 16; j++)
    {
      itoa(plain[j], tmpstr, 10);
      mySerial.write(tmpstr); mySerial.write(" ");
      Serial.print(plain1[j]); Serial.print(" , ");
    }
    mySerial.write("\n");
    mySerial.write("\n");
    Serial.println();
  }
}
void prekey (int bits, int blocks)
{
  byte iv [N_BLOCK] ;
  long t0 = micros () ;
  byte succ = aes.set_key (key, bits) ;
  long t1 = micros()-t0 ;
  t0 = micros () ;
  if (blocks == 1)
    succ = aes.decrypt (cipher, plain1) ;
```

```
  else
  {
    for (byte i = 0 ; i < 16 ; i++)
      iv[i] = my_iv[i] ;
    succ = aes.cbc_decrypt (cipher, check, blocks, iv) ;
  }
}
```

AES code:

```
#include <AES.h>

AES aes ;

byte key[] =

{

  0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00,

} ;

byte plain[] =

{

  0xf3, 0x44, 0x81, 0xec, 0x3c, 0xc6, 0x27, 0xba, 0xcd, 0x5d, 0xc3, 0xfb, 0x08, 0xf2,

0x73, 0xe6

  0x12, 0x23, 0x45, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00,

  0xC0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00,

  0xE0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00,

  0xF0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xab, 0xcd, 0x00, 0x00,

0xde, 0xad,

} ;
```

```
byte my_iv[] =
{
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x01,
} ;
byte cipher [4*N_BLOCK] ;
byte check [4*N_BLOCK] ;
void loop ()
{}
void setup ()
{
  Serial.begin (57600) ;
  Serial.print ("testng mode") ;
  prekey_test () ;
  otfly_test () ;
  otfly_test256 () ;
}
void prekey (int bits, int blocks)
{
  byte iv [N_BLOCK] ;
  long t0 = micros () ;
  byte succ = aes.set_key (key, bits) ;
  long t1 = micros()-t0 ;
```

```
  Serial.print ("set_key ") ; Serial.print (bits) ; Serial.print (" ->") ; Serial.print ((int) succ)
;
  Serial.print (" took ") ; Serial.print (t1) ; Serial.println ("us") ;
  t0 = micros () ;
  if (blocks == 1)
    succ = aes.encrypt (plain, cipher) ;
  else
  {
    for (byte i = 0 ; i < 16 ; i++)
      iv[i] = my_iv[i] ;
    succ = aes.cbc_encrypt (plain, cipher, blocks, iv) ;
  }
  t1 = micros () - t0 ;
  Serial.print ("encrypt ") ; Serial.print ((int) succ) ;
  Serial.print (" took ") ; Serial.print (t1) ; Serial.println ("us") ;
  t0 = micros () ;
  if (blocks == 1)
    succ = aes.decrypt (cipher, plain) ;
  else
  {
    for (byte i = 0 ; i < 16 ; i++)
      iv[i] = my_iv[i] ;
    succ = aes.cbc_decrypt (cipher, check, blocks, iv) ;
```

```
}
t1 = micros () - t0 ;
Serial.print ("decrypt ") ; Serial.print ((int) succ) ;
Serial.print (" took ") ; Serial.print (t1) ; Serial.println ("us") ;
byte i;
 Serial.println ();
for (i = 0 ; i < 64 ; i++)
{
   byte val =  plain[i];
   Serial.print (val>>4, HEX) ; Serial.print (val&15, HEX) ; Serial.print (" ") ;
}
 Serial.println ();
for (i = 0 ; i < 64 ; i++)
{
   byte val =  cipher[i];
   Serial.print (val>>4, HEX) ; Serial.print (val&15, HEX) ; Serial.print (" ") ;
}
 Serial.println ();
for (i = 0 ; i < 64 ; i++)
{
   byte val =  check[i];
   Serial.print (val>>4, HEX) ; Serial.print (val&15, HEX) ; Serial.print (" ") ;
}
```

```
  Serial.println ();
 for (byte ph = 0 ; ph < (blocks == 1 ? 3 : 4) ; ph++)
 {
  for (byte i = 0 ; i < (ph < 3 ? blocks*N_BLOCK : N_BLOCK) ; i++)
  {
   byte val = ph == 0 ? plain[i] : ph == 1 ? cipher[i] : ph == 2 ? check[i] : iv[i] ;
   Serial.print (val>>4, HEX) ; Serial.print (val&15, HEX) ; Serial.print (" ") ;
  }
  Serial.println () ;
 }
}
void prekey_test ()
{
 prekey (128, 4) ;
 prekey (192, 3) ;
 prekey (256, 2) ;
 prekey (128, 1) ;
 prekey (192, 1) ;
 prekey (256, 1) ;
}
```

Motion Detection Sensor Code:

```
int en_pin = 8;
int pin = 7;
unsigned long duration;
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(en_pin, OUTPUT);
  digitalWrite(en_pin, 1);
  pinMode(pin, INPUT);

}

void loop() {
  // read the input pin:
  duration = pulseIn(pin, HIGH);

  //int buttonState = digitalRead(pushButton);
  // print out the state of the button:
  if(duration > 30000)
  {
```

```
    Serial.print("motion detected, ");
    Serial.println(duration);
  }
  delay(1);        // delay in between reads for stability
}
```