OPTIMIZED SECURITY PROTOCOL FOR WIRELESS SENSOR NETWORKS

by

Hasan Tahir

2008-NUST-MSPHD- CSE(E)-16

MS-08 (SE)

Submitted to the Department of Computer Engineering in fulfillment of the requirements for the degree of

MASTER OF SCIENCE
in
SOFTWARE ENGINEERING

Thesis Supervisor

Prof Dr Muhammad Younus Javed

College of Electrical & Mechanical Engineering

National University of Sciences & Technology

2011

TABLE OF CONTENTS

LIST OF TABLES

TABLE **PAGE**

LIST OF FIGURES

LIST OF ABBREVIATIONS

AES	Advanced Encryption Standard
AM	Active Message Handler
CBC	Chain Block Chaining
CFB	Cipher Feedback Mode
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CTR	Counter
DES	Data Encryption Standard
Dest	Destination Address
Destid	Destination ID
ECB	Electronic Code Book
EU ECRYPT	European Union-European Network of Excellence for Cryptology
grp	Group
I/O	Input/Output
ISO	International Standards Organization
IV	Initialization Vector
Len	Length
MAC	Message Authentication Code
NESSIE	New European Schemes for Signatures, Integrity and Encryption
OSP	Optimized Security Protocol
RF	Radio Frequency
RSA	Rivest, Shamir, Adelman
SN	Sensor Node
Src	Source Address
Srcid	Source ID
TinyOS	Tiny Operating System
WSNs	Wireless Sensor Networks
XOR	Exclusive OR

DECLARATION

I hereby declare that I have developed this thesis entirely on the basis of my personal efforts under the guidance of my supervisor Prof Dr Muhammad Younus Javed. All the sources used in this thesis have been cited and the contents of this thesis have not been plagiarized. No portion of the work presented in this thesis has been submitted in support of any application for any other degree of qualification to this or any other university or institute of learning.

_____

Hasan Tahir

ACKNOWLEDGEMENTS

First and foremost I want to thank my advisor Dr. Muhammad Younus. It has been an honor to study under his supervision. I appreciate all his contributions of time, ideas, and funding to make my MS experience both productive and stimulating. I am also thankful for the excellent example he has provided as a successful researcher and professor of computer science.

I would also like to thank the faculty members at the college of E&ME for their constant support during our studies. For this dissertation I would like to thank my committee members Dr. Aasia Khannum, Dr. Saad Rehman and Dr. Arsalan Shaukat for their time, interest, and helpful comments.

Lastly, I would like to thank my family for all their love and encouragement. For my parents who raised me with a love of science and supported me in all my pursuits. And most of all for my loving, supportive and encouraging sister and brother whose faithful support during the final stages of my degree is so appreciated. Thank you it would not have been possible without you.

To my parents, sister and brother…

ABSTRACT

Extensive research has been conducted in the field of wireless sensor networks(WSNs). Both the academia and researchers have envisioned a broad range of applications for WSNs. Many of these applications require varying levels of security. Security is an attribute that is computational and communicational intensive. Severe lack of resources and limited capability has made providing security a challenging task in WSN. Therefore techniques need to be devised that provide security without compromising the limited resources available to WSNs.

This thesis focuses on the design, implementation and analysis of an Optimized Security Protocol (OSP) that fulfills the requirements of confidentiality, authentication and integrity in WSNs. OSP fulfils requirements of high level security without compromising resources. OSP architecture is based on the further optimization of computations in the Rabbit stream cipher and reduction in communication overhead to save sensor's life time.

The proposed OSP has been implemented using power of TinyOS coupled with NesC. In order to evaluate the system, several experiments have been carried out with respect to encryption/ decryption of various data blocks. Evaluation of OSP has been done by conducting a comprehensive efficiency analysis of the proposed architecture. Furthermore, the results of execution time and memory footprint for OSP have been compared to its cryptographic counterparts. The decrease in the memory footprint and execution time proves that OSP is a very viable choice for WSNs. Since OSP provides extensive security features, operates within the optimal ranges for WSNs and outperforms existing security protocol for WSNs therefore this protocol promises to be widely accepted.

Chapter 1

INTRODUCTION

WSNs are being extensively studied for their promising applications and cost effective nature. Researchers and scientists have been able to identify a broad range of environments where WSNs can be applied. Because WSNs are low cost sensing devices it is predicted that in the near future these small sensors will be implemented in places that were not imagined before. Small sensor size and their low price implies that these wireless sensors can be deployed in great numbers and they can sense a broad range of parameters like temperature, vibration, movement and light [1].

Besides the civilian applications researchers have also proposed military applications for WSNs. Many of these applications will require the sensors to be deployed in highly hostile environments where they are highly prone to a large range of security attacks. So far little emphasis was given to the security of WSNs, but as these wireless networks become ready for wide spread adoption and researchers also recognize the insecure nature of wireless mediums research trends have shifted and more emphasis is now being given to the security of WSNs. The industry and the academia acknowledge the fact that wide spread adoption of any wireless technology should not commence without addressing the inherent security issues. In most wireless networks attackers attempt to exploit the minor weaknesses of the system to gain access to whatever resources they can. Although this is also true in WSNs but the limited battery and the possibility of a hostile environment means that a small weakness can be effectively exploited to bring down an entire network [2].

1.1 PROBLEM OVERVIEW

Security in computing and networking is a heavily studied domain and researchers have been successful in proposing protocols and techniques to counter security threats. But when it comes to WSNs conventional security mechanisms cannot be applied owing to

the lack of resources available. WSNs have a very limited memory, power, and computational capability therefore providing security solutions for WSNs is a challenging task. Researchers in the field of security understand that limited energy is the greatest hindrance in the implementation of intricate security protocols. WSNs need lightweight security algorithms that are resilient to attack and can be applied in domains that require varying levels of security. In other words algorithms and protocols are required that provide high levels of security with a minimum memory footprint. Security algorithms need to be designed that are not computational intensive and do not need repeated communications to operate.

1.2 PROJECT OBJECTIVES

The aim of this project is to design a lightweight security protocol for WSNs. A protocol that consumes very little resources and can be widely adopted for its high levels of security. Optimized Security Protocol referred to as OSP will fulfil all the requirements of a security protocol without compromising essential resources. OSP attempts to provide a suite of basic security features i.e. confidentiality, authentication and integrity. Although only confidentiality fulfils the very basic requirements of a security protocol, researchers still recognize the fact that confidentiality is not adequate in ensuring a fool proof system, therefore features for authentication and integrity also need to be incorporated into the OSP. This protocol has been designed using refined, revised and reduced sized packets so that security provision does not become an expensive characteristic for the network. OSP is based on the Rabbit stream cipher because it has been recognized by the eSTREAM project for its promising nature of providing high levels of security in a resource constrained environment.

To completely study the effectiveness of OSP it had to be implemented so that its results can be analyzed. OSP has been implemented using the language NesC in the TinyOS operating system. The simulator used for the deployment of sensors is TOSSIM. All tools and languages used in the project are specialized for use with WSNs.

1.3 THESIS OUTLINE

This thesis is aimed at bringing to light the various security aspects related to WSNs. The aim of this work is to develop an implementable security protocol for resource constrained WSNs. Developing a protocol that fulfils the requirements of high level security yet not compromising the resource demand is a critical task.

The thesis is logically broken down so that each chapter builds on the learning's from the previous chapters. Chapter 2 provides details about WSNs and their resource constrained nature. In this chapter security has also been discussed with particular emphasis on WSN security. Chapter 3 builds on the concepts of security and WSNs to present a detailed architecture of the OSP. The architecture discusses in details the system architecture, packet formats and also possible keying mechanisms. This chapter employs previously discussed concepts to shows how the architecture of OSP can be formally implemented using TinyOS, nesC and TOSSIM. Chapter 4 analyzes OSP with relation to other similar security protocols. The resource consumption results are elaborated with the help of graphs. Finally chapter 5 concludes the thesis and presents future directions for further enhancements and research.

Chapter 2

LITERATURE REVIEW

INTRODUCTION

This chapter is a detailed description of WSNs and security. The chapter begins with an explanation of what WSNs are and their constraints. Then WSNs are discussed in conjuncture with security to explain the security goals of WSNs. Later in the chapter a detailed account of block ciphers and stream ciphers is given. In the end a complete description of the eSTREAM project is given to explain the origin of the Rabbit stream cipher and many other competing ciphers.

2.1 WIRELESS SENSOR NETWORKS

Data Processing is perhaps the largest required functionality of a computation device. Data processing was actually the real motivation behind the development of computer systems. Old fashioned mainframe computers were actually developed to reduce the amount of time that was required to process data. With time these large sized main frame computers became obsolete and smaller desktop computers emerged that could process data much quickly and efficiently. Recently we have seen how computers are being embedded into the environment around us. The sole purpose of embedding these computation devices is to monitor the environment and then perform actions based on the current environmental status.

Fresh developments in Micro-Electro-Mechanical Systems (MEMS) has allowed scientists to develop small sized, low powered and low priced sensors that have the ability of sensing their environment[1][5]. Despite being small these sensors can sense their environment, process the data from the events generated in the environment and then communicate over short distance without any interruption. Since these sensors are small in size and low cost therefore they can be deployed in large numbers thereby forming a wireless network of sensor nodes formally called wireless sensor networks.

Recent research has shown that WSNs can be applied in environments that were previously thought impenetrable. In the near future these small sized sensing devices will be deployed all around us and they will monitor a wide range of parameters. To implement WSNs in such diverse range of applications means we need to develop protocols for WSNs that offer efficiency at the cost of very little resources.

2.2 CHALLENGES OF SENSOR NETWORKING

Researchers have suggested a broad range of applications for WSNs but because of the physical properties of these sensors they suffer from technical issues that need to be addressed before we can employ the power of these smart sensing devices. To fully unleash the power of sensor networks we need to address the issues of limited energy, limited bandwidth, limited hardware and security. Only when these issues [7] are fully addressed can we formally deploy these wireless sensors for our full benefit.

2.2.1 Limited Energy

Fundamentally the sensors of a WSN have been designed to run on battery so they can be deployed in environments that are not physically penetrable. Once a battery has served its

purpose and cannot supply any more power it expires resulting in death of the sensor. A WSN can be composed of thousands of sensors which work in a collaborative fashion. Hence the death of a few of the sensors could render the network useless.

2.2.2 Limited Bandwidth

In WSNs the power required to transmit data is many times the power required to execute an instruction. The reason for this is the complexities involved in the transmission of data. The existing data rate for wireless communications has been restricted to 10-100 Kbits/second. Pottie and Kaiser have shown that the energy required in transmitting one bit over 100 meters in 3 joules. Whereas the same amount of energy can be used to compute around 3 million instructions.

2.2.3 Limited Hardware

In mission critical environments the sensors are required to be small in size so that they are not readily visible. In order to keep the size of a sensor at its minimum the hardware is kept limited. For example the Berkley Mica2 motes possess a small battery, 8 bit CPU that can run at 10MHz, 128KB to 1MB memory and a communication range of less than 50 meters. Researchers have to devise methods and strategies for deploying the sensors within the limited available hardware. The size of the hardware cannot be increased because this would result in expensive hardware that requires further power for processing.

2.2.4 Security

The parameters that are detected in a WSN may seem to be fairly simple but the reality of the fact is that the same data in the wrong hands may prove to be fatal. For example in

battle field monitoring or nuclear power station monitoring, the loss of processed data may prove to be fatal. WSNs are susceptible to a very broad range of attacks. Moreover attackers can use a combination of attacks to truly bring down the entire network.

2.3 SECURITY IN WIRELESS SENSOR NETWORKS

Researchers and scientists have predicted a broad range of applications for WSNs. Many of these applications are highly critical in nature. For example application domains related to military, earth quake monitoring and nuclear power stations will all process data that is very critical in nature. Since WSNs process very fundamental forms of data and then produce results that are highly valuable and crucial in nature therefore these results may be very sought after by adversaries [2,3,4,6].

Recent advances in the field of network security cannot be applied because firstly WSNs have a very unique purpose and design as compared to conventional networks. Secondly, conventional security mechanisms cannot be applied because of the inherent processing and communicational constraints that exist in WSNs. To provide security in WSNs we need to design new protocols and frameworks that are lightweight in nature yet produce high quality results. In simple terms all protocols and frameworks for WSNs need to be optimized for performance, functionality and value added services like security.

2.4 SECURITY GOALS IN WIRELESS SENSOR NETWORKS

To provide security in WSNs we need to address three fundamental security goals namely data confidentiality, data authentication and data integrity.

2.4.1 Data Confidentiality

The data in a sensor network should not be leaked to unintended persons. It should only be accessible by authorized persons. The conventional approach for keeping data protected is to encrypt the data with a secret key which is only possessed by the intended receivers.

2.4.2 Data Authentication

The purpose of authentication is to verify that the parties involved in initiating a communication session are really who they claim to be. Further authentication can be used to ensure that an already established connection is not being interfered by any unwanted party.

2.4.3 Data Integrity

In networks, data integrity refers to the fact that the received data has not been tampered during transit by an attacker. The system must possess the ability to recognize data that has been reordered, modified, deleted, duplicated or inserted.

2.5 SECURITY RISKS IN WSNS

WSNs are often deployed in domains that are susceptible to attack by adversaries. Unwanted persons may attempt to eavesdrop, tamper or even modify data that is being communicated. In this section some of the high priority security risks are presented[18,19,20].

2.5.1 Eavesdropping

Since WSN is a wireless based communication network therefore it becomes easier for an attacker to pick up on a data stream. If an attacker can pick up on an inadequately encrypted data stream then it can easily extract data and then draw conclusions based on that data. In a similar scenario an attacker can place listening nodes at significant locations so that the nodes can simply listen to the data being communicated. To protect against such types of attacks an encryption method is needed. Encryption in conventional networks is not a great issue because they possess unlimited processing capabilities [10]. Whereas in WSNs the encryption approach is faced with issues like limited battery, small memory and limited processing capability. To overcome this only symmetric key encryption can be used because it uses less power as compared to asymmetric key. Symmetric keys use two identical keys to perform encryption; therefore a secure key distribution mechanism is also needed for WSNs. Also we must ensure that if a node or a number of nodes are compromised the attacker does not obtain too much information regarding the security scheme and the keys that are being used.

2.5.2 Sensor Node Compromise

In WSNs it is understood that every node can be a potential point of attack. The attackers can also try to insert their own nodes into the network and make the network accept them as authorized nodes. In similar scenarios the attackers can capture legitimate nodes and

use them in attacks like falsification of sensor data, listening sensor data or starting denial of service attacks.

2.5.3 Privacy of Sensed Data

WSNs can form a collaboration of thousands of nodes, where each node is responsible for sensing its environment and then routing the information to a beacon. When all these sensors work together the aggregated data can be quite large. When the data is sent through the entire WSN it becomes very easy for the attacker to access data from almost any point in the wireless network [16]. Furthermore the attacker can infiltrate some of the nodes in the network and hence gain access to a large amount of data. Detecting and overcoming such types of attacks can be difficult because in most cases the attack may not even be apparent. When designing the network one needs to consider that only the essential data is transmitted. For example if the core temperature of a nuclear power generator is required then we may not need to transmit data regarding the time, place, humidity and sensor relative information. Discarding such type of irrelevant data also results in minimization of data transmission costs that occur because of unnecessary data.

2.5.4 Denial of Service (DoS) Attacks

Using the Denial of Service attack an attacker has the ability of rendering its host inoperable. In most cases such type of attacks happen when the attacker tries to overwhelm the network by sending large amounts of malicious data. This technique deprives the sensors from going into their power saving sleep mode. This implies that the sensors consume up a large amount of their precious battery power thereby reducing the

lifespan of the entire network. Once the individual sensors start to fade away the network is first segmented and then later the entire network collapses due to non availability of power. There are many mechanisms to protect against such type of attacks but the creativity with which this attack can be used is limitless.

2.6 SENSOR NETWORK CONSTRAINTS

The major constraints of a typical WSN are its adhoc nature and the low price of the sensors which is visible in the limited power capability, limited battery and limited transmission range. These constraints greatly affect how security is provided to WSNs. The impact of these constraints is discussed below.

2.6.1 Limited Memory Space

WSNs do not possess a large memory as compared to other networking devices. The limited memory size prevents us from storing large and complex algorithms on the sensors. WSNs need simple and optimized algorithms for security and routing.

2.6.2 Limited Power

Limited power in sensors is the toughest challenge in WSNs. Because of this property the security technique cannot be complicated. A very complex security algorithm may provide high level of security but largely at the cost of power consumption. To conserve energy we need to minimize computations and communications. Complicated encryption techniques like public key algorithms and complex authentication mechanisms are avoided because of the inherent power limitation.

2.6.3 Limited Budget

Each sensor node is designed to be very cheap so that the sensors can be deployed in large numbers. Deploying sensors in an unattended fashion makes them very vulnerable to capture and scrutiny. Once a node is captured the attacker can attempt to extract information regarding the security mechanisms. Hence techniques need to be devised that limit the devastating effect in case a node is compromised.

2.7 DATA ENCRYPTION

Many networks need to be made secure even though they do not carry very critical type of data. For example a local area connection in an office may need to be made secure because it provides networking support to the people that work in the office even though the data that is being shared is pretty ordinary. On the other hand there are networks where the data communications in the network needs to be secured because of the criticality of the data. For example a network that may exist in a military environment. To secure WSNs it is very essential to secure the data that is being passed through the network because each sensor has limited resources and the data that is being processed is critical.

To send data securely between two sensor nodes in a WSN the system can encrypt the data either using a Symmetric Key Cryptography (SKC) or Public key Cryptography (PKC). PKC is commonly adopted because of its comprehensive security provision along with the demand for higher resources than SKC. SKC schemes are very light weight in nature hence they are more appropriate for WSNs. The only concern in SKC is the

sharing of keys. If a key is disclosed for some reason then the entire WSN is compromised. If a node-to-node key is utilized then key management becomes difficult. To reap the benefits of both the SKC and PKC a hybrid technique is employed in which asymmetric encryption is used to exchange the secret key between the sending and receiving sensor nodes. Then PKC schemes are applied to transfer data between the sending and receiving sensor nodes. SKC based schemes are of two broad types block ciphers and stream ciphers, which are elaborated in the following section.

2.7.1 Block Ciphers

In block ciphers the plain text is broken down into blocks of *n*-bits. Each block is encrypted one at a time. Most often the block size is kept at 64 or 128 bits. A block cipher encrypts the plaintext by encrypting it *r* times sequentially with a round function. Each round function receives a subkey which is a derivation of the actual key *K*, and performs confusion and diffusions of its inputs. Many similar variances of the block cipher also exist that reinforce the security of the system [30].

2.7.2 Stream Cipher

Unlike block ciphers this method of data encryption uses a bit by bit encryption mechanism. Stream ciphers are composed of two components i.e. a key stream generator and a mixing function. The key stream generator is the central function of the stream cipher while the mixing function is an XOR function. Largely the stream cipher is of two operational modes: Synchronous Stream Ciphers and Self Synchronizing Stream Cipher. In synchronous stream cipher the key stream generator is only dependent upon the shared key for encryption. The shared key is used by the sender for the encryption of outbound

streams. The receiver decrypts the stream using the same shared key. A major disadvantage of this method is that if the key is leaked then the system is compromised. Figure 2.1 illustrates how synchronous stream ciphers operate.

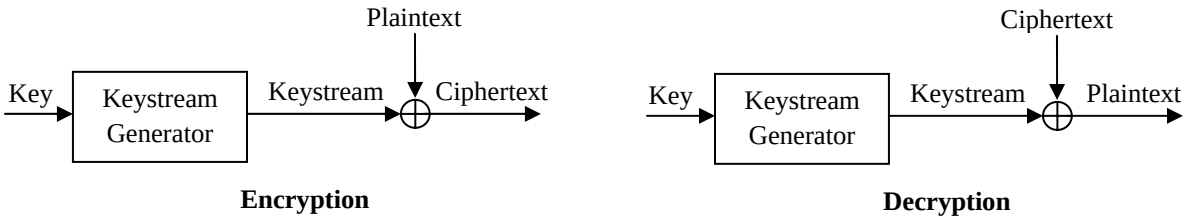


Figure 2.1: Stream Cipher – Synchronizing Mode

In self synchronizing stream cipher the previous states of the cipher bits are given as input to the keystream generator. Figure 2.2 shows the encryption and decryption process taken by the self synchronizing stream cipher.

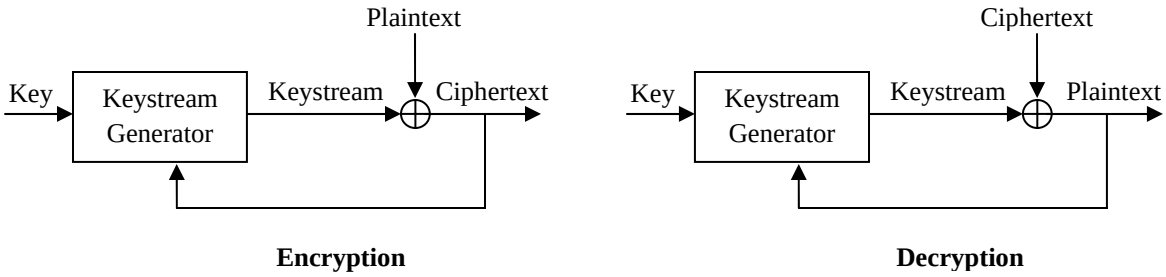


Figure 2.2: Stream Cipher – Self Synchronizing Mode

2.8 PROTOCOL DESIGN OBJECTIVES

Wireless Sensor Networks (WSNs) operate on wireless communication links. It is proven that it is easier to eavesdrop on wireless links because they broadcast their information wirelessly over the network[6]. Owing to this broadcast nature of WSNs, the adversary can easily intercept the transmitted data. Therefore WSNs must deter eavesdropping and

operate within the limited resources available to them. By nature security is an expensive mechanism that requires memory, complex mathematical operations and communications. Unfortunately WSNs have limited memory, processing, communication and power. To provide security in WSNs we need to maintain a delicate balance between security, resource consumption and performance. If this balance is not maintained then the network will either fade away too quickly or it will not provide services as they are intended to be.

The Optimized Security Protocol (OSP) has been developed to provide extended security services along with minimum resource consumption. OSP has been developed keeping in view the best practices required for a protocol to operate in a resource constrained environment [33].

2.8.1 Security

WSNs need to communicate pin point accurate data that can be very sensitive in nature. A sensor must not leak the data it possesses to an unauthorized node. To provide message confidentiality the data is encrypted with a secret key so that even if data is available to the adversary it is not readily understandable. Another benefit of keeping data encrypted with a secret key is that the data can only be decrypted by an entity that possesses the required secret key. To achieve high level security it has been proven that message confidentiality is not sufficient. Therefore authentication and integrity are also required. Using authentication the receiver can verify that the message was actually sent by an authenticated party. Whereas Integrity ensures that the message being communicated has not been tampered with in any way. In the absence of authentication and integrity, messages under transit are prone to cut-and-paste attacks. OSP has been designed to

provide message confidentiality, authentication and integrity while consuming as little resources as possible.

2.8.2 Performance

Owing to the extreme resource limitations in WSNs it is important to provide security without consuming too many resources. Security protocols are needed that provide reasonable security coverage without too much resource demand. An over conservative approach towards resources can limit the level of security available to the network. On the other hand an approach to provide a high level comprehensive security solution may render the network useless because of the intensive demand for resources. Therefore a performance tradeoff is necessary to achieve a satisfactory level of security along with optimal resource consumption. OSP provides high level security with low communication and computation overhead. The security key space is kept moderate to secure the network and also provide extended sensor life.

2.8.3 Ease of Use

WSNs are deployable in a broad range of environments. Each environment has its own requirements and preferences. Therefore OSP has been designed for adaptability and modifiability. OSP can be easily customized i.e. programmers can make adjustments for security and performance. Secondly OSP can be used in combination with any keying mechanism. This provision has been kept because of broadly varying application demands, security and performance.

2.9 SECURITY PRIMITIVES

OSP is designed specifically for WSNs that have to operate in a hostile environment with resource limitations. This section provides an overview of the security primitives that have been employed in the OSP.

2.9.1 Encryption Scheme

Both symmetric and Asymmetric keys have their advantages and disadvantages. Asymmetric key encryption is considered slightly less troublesome because it uses different keys for encryption and decryption. Hence asymmetric keys do not have issues related to secure key sharing. On the other hand since the encryption and decryption keys are different this makes the encryption and decryption process complicated. This aspect of asymmetric keys makes them less efficient compared with symmetric key. A complex encryption and decryption process means cumbersome mathematical calculations, communication overheads and unnecessary drainage of battery power.

Since asymmetric key encryption is computationally expensive therefore a method is used to both safely share the keys and then share data without the overhead. Symmetric key encryption scheme is used to encrypt bulk data. Whereas the asymmetric key encryption is used to encrypt the encryption key which was used for bulk data encryption. This method provides secure key exchange and then allows the sensors to exchange data using symmetric key encryption. Since symmetric key encryption is less computation intensive therefore it is efficient to use symmetric keys for regular communications.

2.9.2 Stream Ciphers

Stream ciphers are designed to process data bit by bit. This aspect of stream ciphers makes them appropriate for environments where large memory blocks do not exist. Stream ciphers provide faster and efficient encryption/decryption as compared to block ciphers.

2.9.3 Initialization Vector

Initialization vectors are used to further strengthen the security of WSNs. Initialization vector is used when two messages are very similar or truly identical. In case of WSNs identical or similar messages are frequently communicated between neighboring nodes. The initialization vectors provide side inputs so that if two identical messages are encrypted then the obtained cipher texts are different. In the absence of an initialization vector the attackers can use various types of attacks to scan for repeating patterns and sequences in the transmitted cipher texts.

2.9.4 Message Authentication Code

Alongside confidentiality, OSP has been designed to provide authentication and integrity. To save essential resources while providing authentication and integrity message authentication code (MAC) has been used. A MAC is a superior algorithm that accepts as input a secret key and a message. The output after performing the MAC is a tag that can be used to verify the authentication and integrity. In simple terms a MAC is a computation performed on a message and the resulting output allows verifiers who possess the secret key, to detect any changes that have been made by attackers to the message while it was in transit.

2.10 eSTREAM PROJECT

eSTREAM is a multi year project being run by the EU ECRYPT network. The purpose of the project was to identify new stream ciphers that promise to be widely accepted and adopted. The project was initiated because of the disappointment of all six stream ciphers that were submitted to the NESSIE project. The eSTREAM portfolio began establishment in November 2004. After going through three phases of study and analysis the final phase was completed in May 2008. The goal of the project was to identify those stream ciphers that have high throughput in environments with limited resources. Table 2.1 presents the four finalized stream ciphers of the eSTREAM project [23].

Table 2.1: eSTREAM Portfolio Stream Ciphers

eSTREAM Portfolio Stream Ciphers
Rabbit
HC-128
Salsa20
SOSEMANUK

2.10.1 Rabbit Cipher

Rabbit stream cipher was presented to the eSTREAM project in May 2005. Rabbit stream cipher was designed for high performance. The core component is a bitstream generator that can encrypt 128 message bits per iteration. The cipher is composed of a mixing function that is based on arithmetic functions that are available on a modern processor. This mixing function is the actual strength of the cipher[28].

2.10.2 HC-128 Cipher

HC-128 is also a part of the eSTREAM portfolio. The cipher was designed to provide swift bulk data encryptions without compromise in security [25]. The cipher is composed of two secret tables. Each table is designed to hold 1024 32-bit words. For each state update a 32-bit word in each table is updated using a non-linear update function. Both tables are updated after 2048 steps.

2.10.3 Salsa20 Cipher

Salsa20 has been selected for the eSTREAM portfolio because of its superior diffusion mechanisms. The cipher uses pseudorandom functions along with bitwise additions and constant rotations to defeat timing attacks.

2.10.4 SOSEMANUK Cipher

SOSEMANUK is a 128 bit cipher that is part of the eSTREAM portfolio. The cipher has an associated proof of concept but the cipher has not been proved in the finalization of the eSTREAM portfolio. The cipher uses a 128 bit initialization vector. According to the author this cipher is based on the design of SNOW and the Serpent block cipher.

2.11 WSN DEVELOPMENT ENVIRONMENT

TinyOS is an operating system that is event driven yet it possesses a very small memory footprint (instruction and data memory 400 bytes). The operating system supports several platforms and it is open-source environment. TinyOS is supported by a programming language and model (NesC). Programs in TinyOS are built out of components that have specific interfaces for interactions. The components are of two types: Modules and

Configurations. Modules and configurations work in a closely knit fashion. The modules define the application behavior while the configuration is used to connect the components together. Each component has a frame, function and access interfaces. The frame is composed of variables to keep track of internal/ initial states. All the code, commands, events and tasks written in NesC are part of the function [29].

2.12 SUMMARY

In this chapter a background study on WSNs was presented. The presented concepts form the foundations of the project. Details regarding individual sensors, entire network and security in WSNs have been discussed in detail. The eSTREAM project has been studied for its dedications towards providing security in resource constrained environments. Algorithms, techniques and various directions that have been discussed form the foundation of the research work in the project.

Chapter 3

DESIGN & IMPLEMENTATION OF OSP

INTRODUCTION

When designing a security protocol for conventional systems researchers have a very limited set of constraints within which they have to operate. Whereas in the case of WSNs, researchers face the issue of being faced by constraints that are many and severe in nature. WSNs present constraints and limitations that require researchers to rethink the way conventional security works. In a resource constrained environment producing a new design can mean redesigning from the basic foundations. Often packet formats need to be optimized and redundant data needs to be discarded so that essential resources can be conserved.

This chapter presents the architectural design and implementation details of the newly proposed Optimized Security Protocol for WSNs referred to as OSP. This protocol has been designed to meet the highest levels of security. Besides the conventional confidentiality, OSP also provides authentication and integrity. Furthermore, since wireless sensors can be deployed in highly hospitable environments, therefore the importance of providing confidentiality, authentication and integrity in a single protocol increases.

3.1 OSP DESIGN RUDIMENTS

OSP is a security protocol specifically designed for resource constrained WSNs. A security protocol that is designed for a resource constrained but mission critical environment must provide the highest level of security while consuming as little resources as possible. The designed protocol must consume minimum amount of resources while providing high levels of security in a quick and optimal fashion. OSP has been optimized by first eradicating the communication overhead that incurs because of sending separate encrypted packets for the initialization vector. Secondly the packet format has been redesigned to remove redundant bits and to accommodate that data which is actually required by the system. A smarter packet results in efficient encryption/ decryption.

3.1.1 Encryption

Rabbit is one of the qualifying eSTREAM portfolio ciphers. Rabbit has been specially designed for security and performance without influencing limited resources. Rabbit has a lightweight algorithm that consists of a 128 bit key that is expanded into eight state variables and eight counter variables. The key and initialization vector (IV) are plugged into four rounds of the next state function to eradicate any recognizable correlations between the key, IV and plaintext. The next state function is employed to jumble up bits so that no obvious pattern is visible. The keystream generator uses the same components for modification of counter variables and state variables.

In OSP, the encryption process begins by providing the plaintext (PT) to Rabbit. To formally perform encryption, Rabbit is also provided with the symmetric key (K_E) and the initialization vector (IV). When the Rabbit cipher is performed on the plaintext the

output obtained is ciphertext (CT). But at this stage the ciphertext cannot be transmitted; because in its current state, it only fulfills the requirements of message confidentiality, whereas OSP is required to also provide authentication and integrity. These security features can only be provided when a MAC is computed on the ciphertext. To save precious program space OSP has been designed to reuse the Rabbit next state function for computing the MAC. A separate MAC algorithm is not needed because the Rabbit next state function possesses superb diffusion properties and further in this manner a separate footprint for a MAC algorithm is not required. The obtained MAC is embedded into the OSP packet and transmitted along with the CT.

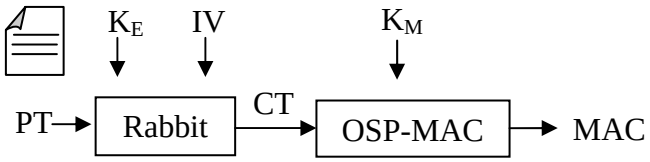


Figure 3.1: Computation of Ciphertext (CT) and MAC in OSP

3.1.2 OSP Packet Format

A security protocol that requires minimum processing is not possible without an optimized packet format. Therefore, the packet format used by OSP has been redesigned to accommodate fields that did not previously exist in the TinyOS packet. The design of the TinyOS packet format included fields that were not very necessary for communication and these fields have been replaced in OSP design by making a more advantageous use of each bit in the packet. Given below is the TinyOS packet format[31].

Dest (2)	AM (1)	Length (1)	Grp (1)	Data (0…29)	CRC (2)

Figure 3.2 The TinyOS Packet Format with Number of Bytes Per Field [31]

The TinyOS packet format contains Grp and CRC fields which are not required by OSP. These fields have been replaced by the source (Src), message counter (Ctr) and the MAC. The Grp is a field that is used in cases when data is transmitted to a group of sensors. Using the Grp field only those sensors can communicate that possess the same Grp. In OSP the Grp field has been replaced by Src field to provide security on perlink basis rather than group basis.

The CRC field in TinyOS packet is replaced by the MAC in OSP. The MAC is a superior algorithm as compared to the CRC, the later can only detect transmission errors that exist because of channel noise whereas MAC detects transmission errors as well as modifications done on the message to check for message integrity. The updated and optimized OSP packet format is as follows:

Dest (2)	AM (1)	Length (1)	Src (2)	Ctr (2)	CT (0...29)	MAC (4)

Figure 3.3: The OSP Packet Format with Number of Bytes Per Field [33]

The header fields in OSP packet namely Dest, AM and Length are kept unencrypted so they are readable at once. If they were kept encrypted then every sensor would first have to decrypt the fields to see if it is the intended destination sensor.

3.1.3 IV Format

To further conserve the limited resources that are available, the design of OSP reuses the fields in its packet for the generation of the IV. A major advantage of this is that separate transmission of IV is not needed. Thus we can save the costs that occur from the creation and transmission of an independent IV. The unencrypted fields of the OSP packet are utilized to generate the IV. All fields of the IV are sent unencrypted. The unique combination of Src and Ctr ensures that a single node can send around 2^{16} packets before a repetition of the IV is observed. Given below is the IV format [33] for OSP:

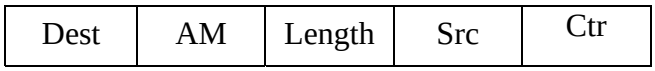


Figure 3.4: IV Format

3.2 ARCHITECTURAL DIAGRAM OF OSP

OSP is a design that is based on reuse so that resources can be conserved. The sender provides the plaintext (PT) to the Rabbit algorithm along with the symmetric key (K_E) and the initialization vector (IV). This produces the ciphertext (CT) which in turn is used to compute the MAC using the MAC encryption key. After the computation of MAC, the required fields of the OSP packet are populated and transmitted to the intended node. This approach to encryption is often referred to as Encrypt-Then-MAC approach [33].

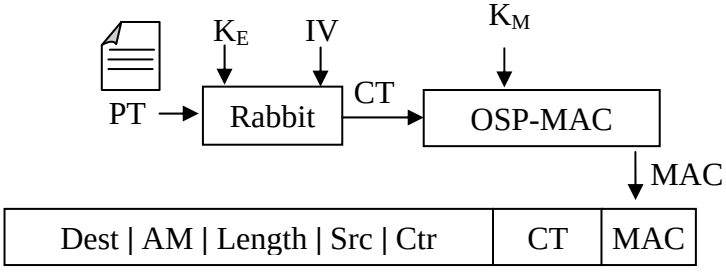


Figure 3.5: Functional Diagram of OSP - Sending Party

When the above packet is received by the receiving node the process of determining if the message is legitimate or not is initiated. The receiving node extracts the CT and uses the agreed K_M to compute an independent MAC. If the received MAC and the MAC computed at the receiving node are both identical then the message is in its true form, else the packet is discarded. If the packet is legitimate then the CT, K_E and the IV are used by the receiving party to recover the plaintext using Rabbit cipher [33].

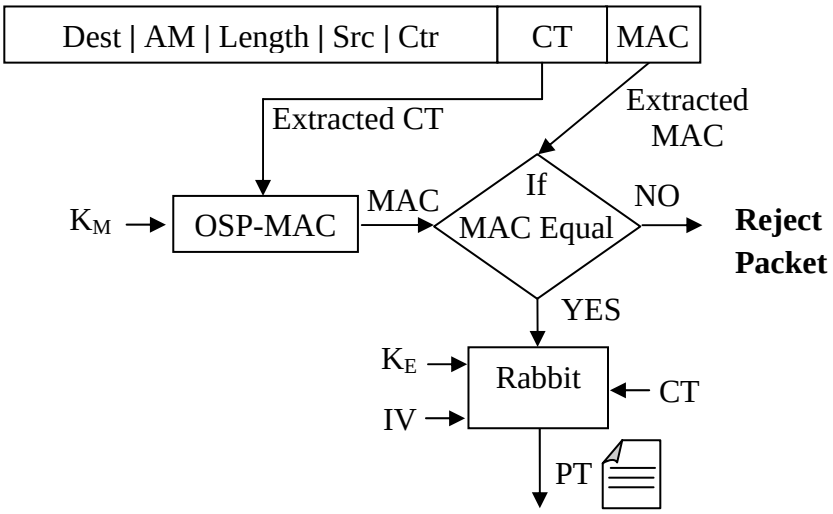


Figure 3.6: Functional Diagram of OSP - Receiving Party

3.3 OSP ALGORITHM

The underlying primitives of Rabbit are secure, so it is possible to build a proof of given notion of security of the encryption MAC algorithm and for OSP.

The ciphertext is formed by encrypting the plaintext using Rabbit under K_E and IV. Further, a 128 bit MAC encryption key K_M, is used to generate 8 subkeys for the OSP-Mac function. The K_M subkeys and IV are further used to generate state and counter variables that are used in the next state function for the generation of MAC keystream.

The generated keystream blocks S_{M0},....,S_{M7} are XORed with the ciphertext blocks that were generated as a result of encryption using K_E to get C_{tM}. The calculated C_{tM} is divided into four equal blocks namely Ct_M[31...0], Ct_M[62...32], Ct_M[95...64], Ct_M[127...96]. These blocks are XORed to generate the OSP-MAC which in turn is verified by the receiving party to check for integrity and authenticity of the transmitted message. The receiving party extracts the ciphertext from the received packet and calculates MAC using the same K_M (MAC key) and initialization vector to see whether the 4bytes MAC received in the packet matches the calculated MAC. Figure 3.7 shows the complete OSP algorithm [28].

Let Pt denote the plaintext

Let Ct denote the ciphertext generated as a result of K_E

Let K_E denote the encryption key

Let K_M denote the MAC encryption key

$Ct_E = E_{KE}$ (Pt)

{ K0, K1, K2, K3, K4, K5, K6, K7 }=K_M (128 bit)

IV = [DEST | AM | LEN | Src | Ctr]

$C_{tM} = Ct_e \oplus S_{M0}, S_{M1}, S_{M2}, S_{M3}, S_{M4}, S_{M5}, S_{M6}, S_{M7}$

OSP – MAC = Ct_M [31…0] ⊕ Ct_M [63…32] ⊕ Ct_M [95…64] ⊕ Ct_M [127…96]

Figure 3.7: The OSP Algorithm

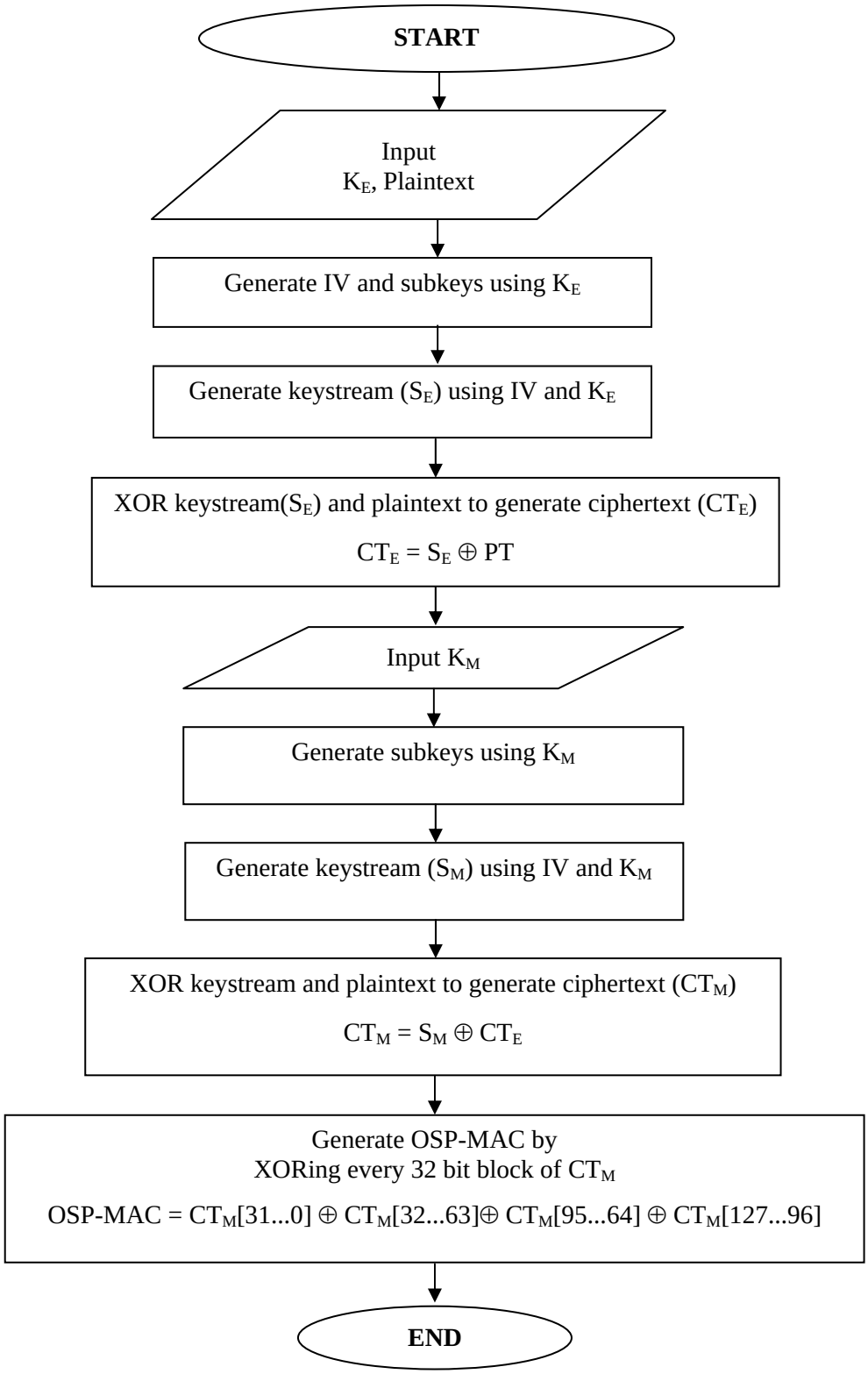
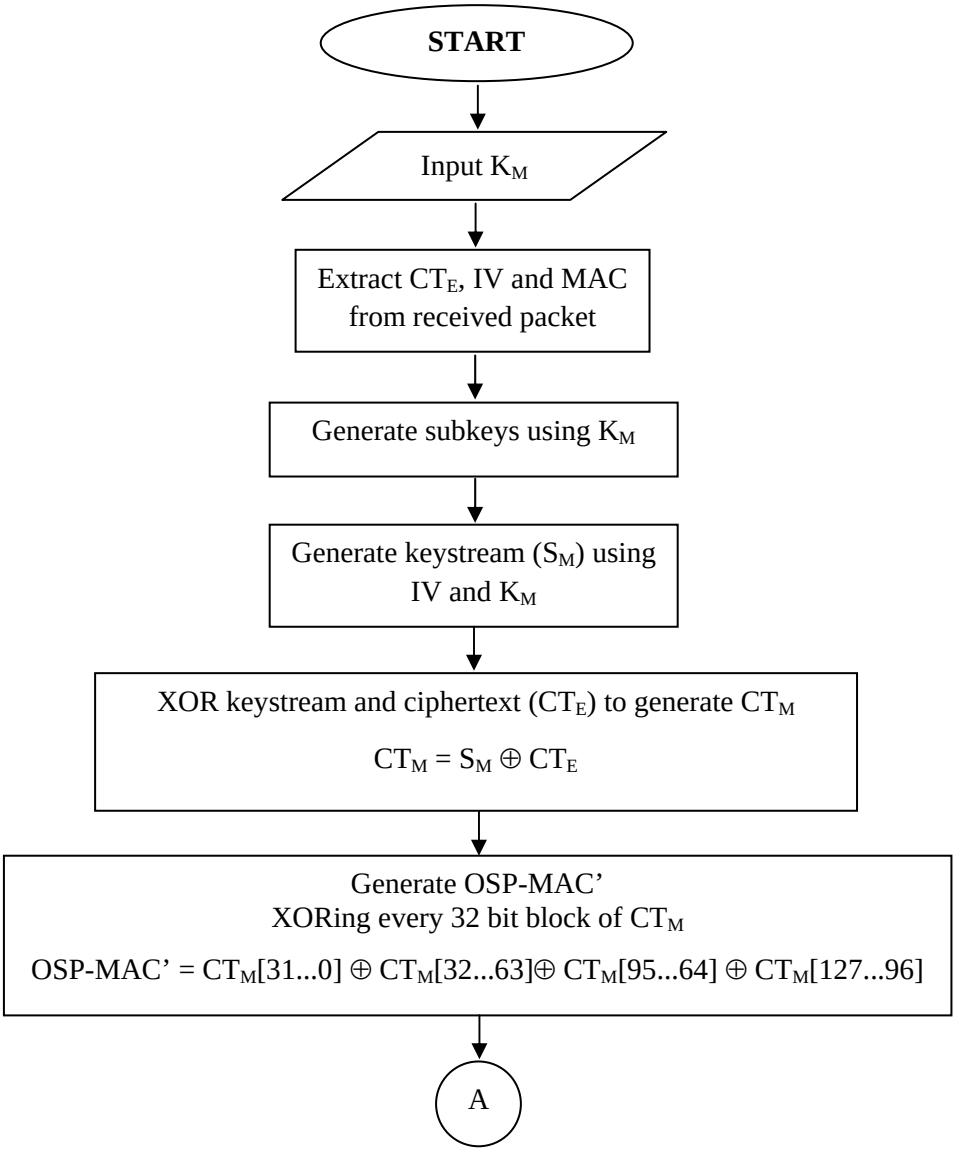


Figure 3.8: Flowchart of OSP – Message Sending Party


START
Input K_M
Extract CT_E, IV and MAC from received packet
Generate subkeys using K_M
Generate keystream (S_M) using IV and K_M
XOR keystream and ciphertext (CT_E) to generate CT_M
$CT_M = S_M \oplus CT_E$
Generate OSP-MAC'
XORing every 32 bit block of CT_M
OSP-MAC' = $CT_M[31...0] \oplus CT_M[32...63] \oplus CT_M[95...64] \oplus CT_M[127...96]$
A

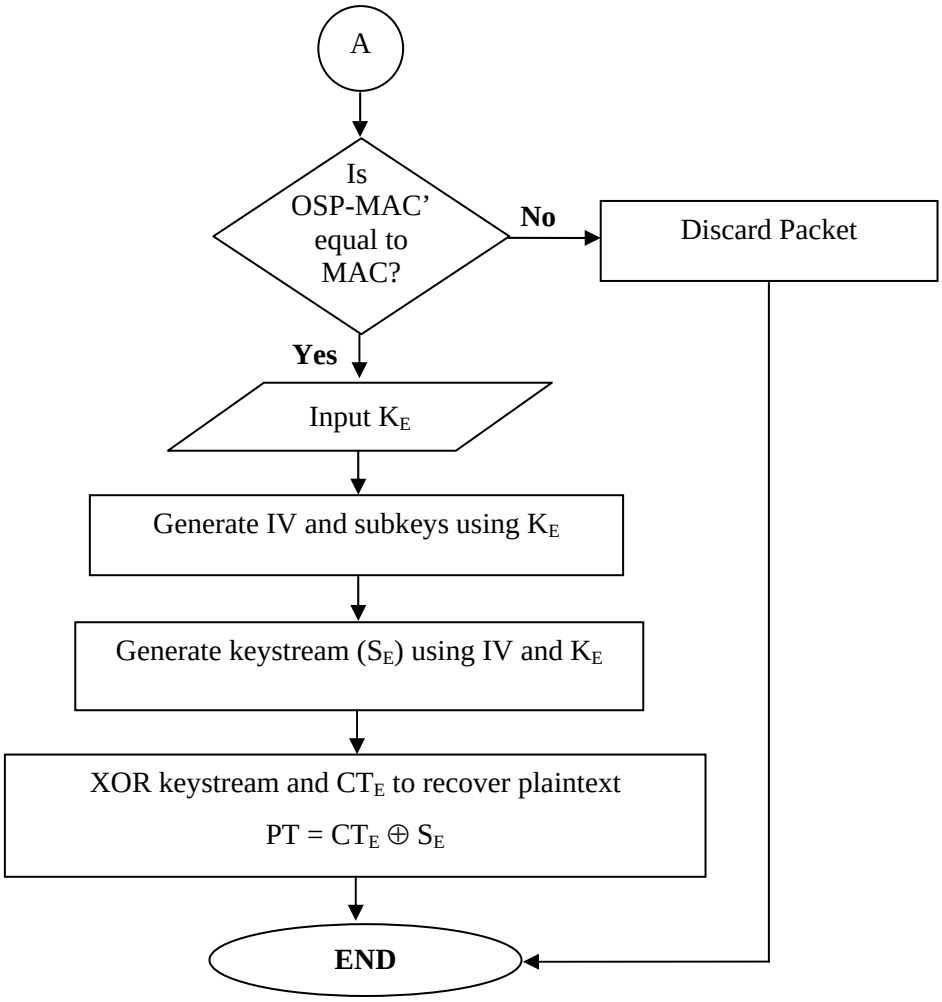


Figure 3.9: Flowchart of OSP – Message Receiving Party

3.4 SIMULATION SETUP

TinyOS is a specialized operating system designed for WSNs. The operating system is a component based platform that provides extensive support to the various requirements of sensor networks [32]. TinyOS has been designed specifically for resource constrained environments and it provides support to the most popular motes such as MICA2, RENE and BTNode.

TinyOS is supported by NesC which is an executable application that fundamentally assembles individual components. The greatest advantage of the component assembly is that the user can exclude all those components that are not being used by the application. Excluding components means reduced code size, application simplicity and elimination of many potential error sources. OSP has been designed to employ the benefits of both TinyOS and NesC. Figure 3.10 shows the ideal component diagram of OSP.

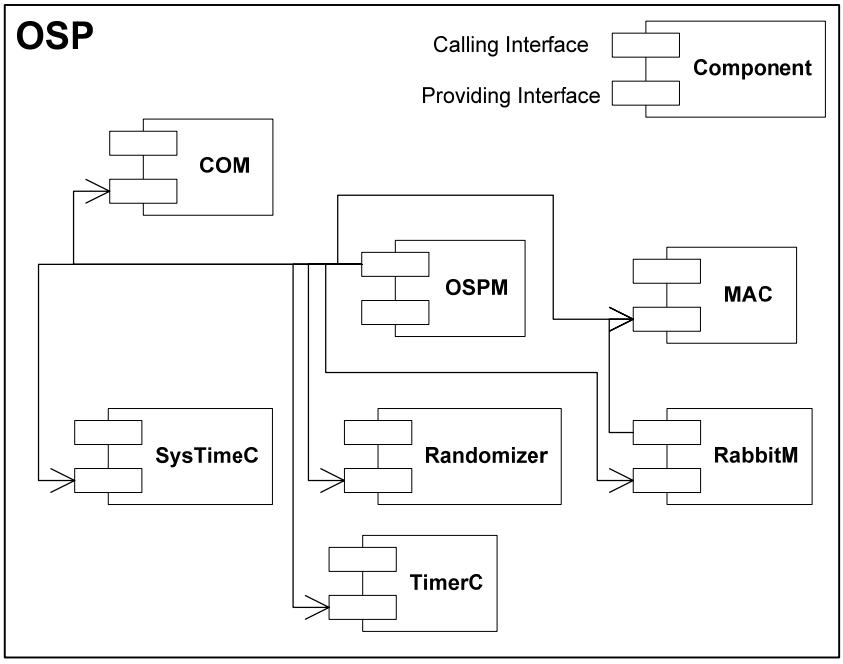


Figure 3.10: Ideal Component Composition Diagram of OSP

Fundamentally, OSP is a collection of components and their interconnection. OSP is composed of the OSPM function that is the calling point of other components. The OSPM component communicates directly with the TimerC which is the default time keeping component. The OSPM also links with the COM component which is the generic communication component. The OSPM is fundamental in nature because it

communicates with SysTimeC, RabbitM and the Randomizer. SysTimeC is used for performing timing calculations. RabbitM component is used to provide Rabbit encryption/decryption and MAC generation modules. The MAC calculations are performed by invoking the services of the MAC component. The Randomizer component is essential for random deployment of sensor nodes in the network.

3.5 OSP IMPLEMENTATION

Rabbit is a security protocol that is designed for high levels of security and efficiency. Rabbit, by nature is a symmetric synchronous stream cipher that is intended for environments that have limited resources. Rabbit is part of the eSTREAM portfolio [23], because of its promising nature for resource constrained environments hence it is also suitable for WSNs. Rabbit operates by taking a 128-bit secret key as input and generates for each iteration an output block of 128 pseudo-random bits from a combination of the internal state bits. Encryption/ decryption is achieved by performing XOR on the pseudo-random data with the plaintext/ ciphertext. OSP attempts to use Rabbit and an optimized packet format for optimized security provision. In OSP, the plaintext is provided to the Rabbit module along with the encryption key (K_E) and initialization vector. The output obtained from the Rabbit module is the ciphertext. Authentication and integrity of message is verified by further performing computations on the generated ciphertext. In the second phase of its MAC generation cycle, OSP takes a 128 bit MAC key and for each iteration of the Rabbit Next State function, it generates an output block of 128 pseudorandom bits. Further, the MAC of the ciphertext (Ct_E) is generated by XORing the pseudo-random data with the ciphertext blocks to generate Ct_M. The 32 bit OSP-MAC is calculated by XORing every 32 bit ciphertext block within Ct_M

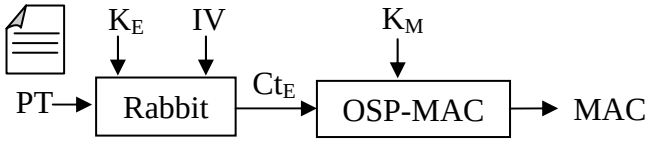


Figure 3.11: OSP Functional Diagram [33]

3.5.1 Inner State

The internal state of the stream cipher consists of 513 bits. These 513 bits are divided between eight 32-bit state variables ranging from X0,...,X7 and eight 32-bit counter variables ranging from C0,...,C7. Figure 3.12 shows the inner state details which are part of the rabbit encryption/decryption and MAC generation module for OSP.

```
Struct t_instance
        {
        unint32_t x[8];
        unint32_t c[8];
        unint32_t carry;
        }
```

Figure 3.12: Basic OSP Structure Definition

3.5.2 Key Setup Scheme

Key setup begins by first expanding the 128-bit key into the eight state and eight counter variables. This expansion is such that there is a one-to-one correspondence between the key, the initial variables and counter variables. Four iterations are performed on the state and counter variables according to the next state function. These four iterations result in reduction of the correlations between the key bits and the internal state variables. Figure 3.13 shows the key setup procedure for the encryption and MAC generation of OSP [18].

The key is divided into subkeys:

K0 = k [15...0] K4 = k [79...64]
K1 = k [31...16] K5 = k [95...80]
K2 = k [47...32] K6 = k [111...96]
K3 = k [63...48] K7 = k [127...112]

The initial state is initialized as follows:

For j=0 to 7
if j is even
Xj = k(j+1 mod 8) || Kj
Cj = k(j+4 mod 8) || K(j+5 mod 8)
else
Xj = k(j+5 mod 8) || K(j+4 mod 8)
Cj = kj || K(j+1 mod 8)

Four iterations of the system are then performed. Each iteration consists of counter updates and the next state function. The counter variables are initialized to:

For j=0 to 7
Cj = Cj ⊕ x(j+4 mod 8)

Figure 3.13: Key Setup Algorithm

The internal state variables and the secret key are passed as parameters to the key setup procedure. The key setup procedure divides the key into eight sub-keys. These eight subkeys are further used in the generation of state and counter variables. All these variables are utilized during the ongoing processing of OSP encryption and MAC calculation. Table 3.1 shows details regarding the key_setup scheme in OSP which is used for encryption/decryption and MAC generation in OSP.

Table 3.1: Function Definition for Key Setup

Key_setup	Performs key setup in OSP		
Input Parameters	Type	Name	Task
	t_instance	*p_instance	Provides a pointer to the internal state
	const char	*p_key	Carries the secret key exchanged.

3.5.3 Initialization Vector

The IV setup scheme functions by modification of the counter state. Using the next state function in figure 3.14 the system performs four iterations so that all state bits are non-linearly dependent on all the IV bits. The modification of the counter by the IV assures that all 2^{64} varying IV will lead to a unique keystreams [18].

C0=C0m ⊕ IV[31..0]
C1=C1m ⊕ (IV[63..48] || IV[31..16])
C2=C2m ⊕ IV[63..32]
C3=C3m ⊕ (IV[47..32] || IV[15..0])
C4=C4m ⊕ IV[31..0]
C5=C5m ⊕ (IV[63..48] || IV[31...16])
C6=C6m ⊕ IV[63..32]
C7=C7m ⊕ (IV[47..32] || IV[15..0])

M refers to master state which is the internal state directly after key setup scheme

Figure 3.14: Initialization Vector Algorithm

After the key setup is complete the resulting inner state is saved as a master state. Then the IV setup is commenced which makes it possible to generate the first key stream block. If re-initialization under a new IV is necessary the IV setup is run on the mater state of the key. The purpose of this is to generate the next keystream block while not having to perform the key setup procedure[14,18].

Table 3.2: Function Definition for IV_setup

IV_setup	Performs IV setup of OSP		
Input Parameters	Type	Name	Task
	t_instance	*p_master_instance	Pointer to master state i.e. internal state after key_setup
	t_instance	*p_instance	Provides a pointer to the internal state
	Char	p_dest	Destination sensor node ID
	Char	p_src	Sending Sensor Node ID
	Const char	*p_iv	Pointer to the initialization Vector IV= Dest | AM | Len | Src | Ctr
	Size_t	iv_size	Size of Initialization Vector(IV)

3.5.4 Next-State Function

OSP is fundamentally based on the Rabbit Next-State function because this is the core function and is used in key setup, keystream generation and MAC generation of OSP. The Next-State function takes eight counter variables as input and after system iteration, counter modification and g-function iteration produces a 128 bit keystream [18].

Counter Modification
Update $C_0,...,C_7$
Update $X_0,...,X_7$

G_func(X_i,C_i)
    a=X&0xFFFF
    b=x>>16
    h=(((a*a)>>17)+(a*b))>>)+b*b
    l=x*x
    result=h⊕l

Using this function, the algorithm updates the inner state as follows:
for j=0 to 7
    Gj=g(Xj,cj)

Figure 3.15: Algorithm for the Next State Function

3.5.5 Extraction Scheme for OSP Encryption/Decryption

Once the key and IV setup are complete, the 128 bit keystream is generated after performing modifications to the state and counter variables. The obtained keystream is then used for future computations. Figure below shows the algorithm for keystream output $S_e[127...0]$.

$S_e[15...0] = X0[15...0] \oplus X5[31...16]$
$S_e[31...16] = X0[31...16] \oplus X3[15...0]$
$S_e[47...32] = X2[15...0] \oplus X7[31...16]$
$S_e[63...48] = X2[31...16] \oplus X5[15...0]$
$S_e[79...64] = X4[15...0] \oplus X1[31...16]$
$S_e[95...80] = X4[31...16] \oplus X7[15...0]$
$S_e[111...96] = X6[15...0] \oplus X5[31...16]$
$S_e[127...112] = X6[31...16] \oplus X1[15...0]$

Figure 3.16: Algorithm for Keystream Generation [18]

3.5.6 Extraction Scheme for OSP MAC

After the key and IV setup are concluded for the MAC key (K_M), the algorithm is iterated in order to produce a 128 bit output block or keystream for computing the MAC. The keystream is then XOR'ed with the ciphertext generated in the encryption/decryption phase of OSP algorithm to generate a Ct_M. Figure 3.17 shows the algorithm for the keystream output $S_M[128...0]$.

S_M [15...0] = X0[15...0] ⊕ X5[31...16]
S_M [31...16] = X0[31...16] ⊕ X3[15...0]
S_M [47...32] = X2[15...0] ⊕ X7[31...16]
S_M [63...48] = X2[31...16] ⊕ X5[15...0]
S_M [79...64] = X4[15...0] ⊕ X1[31...16]
S_M [95...80] = X4[31...16] ⊕ X7[15...0]
S_M [111...96] = X6[15...0] ⊕ X5[31...16]
S_M [127...112] = X6[31...16] ⊕ X1[15...0]

Figure 3.17: Extraction Scheme for OSP MAC [18]

3.5.7 Encryption/Decryption Scheme

Once the keystream bits are obtained they are simply XOR'ed with the plaintext/cipher text to perform encryption/decryption. The encryption/decryption scheme is repeated until all blocks in the message have been encrypted/decrypted. If a case arises where the message size is not a multiple of 128 bits then only the needed amount of least significant bits from the last output block S is used for the last message block M.

Encryption:
$E = M \oplus S_e$
Decryption:
$M' = E \oplus S$

Figure 3.18: Algorithm for Encryption/Decryption

The internal states and variables after processing are passed with the plaintext to the OSP cipher function. Table 3.3 shows the parameters linked with the final stage of encryption/decryption.

Table 3.3: Function Definition for Cipher

Cipher	Performs Ciphering in OSP		
Input Parameters	Type	Name	Task
	t_instance	*p_instance	Provides a pointer to the internal state
	Char	*p_src	Plaintext to be encypted.

3.5.8 MAC Generation Scheme

The generated keystream bits $S_m[128...0]$ are sorted with the ciphertext to generate a stream of pseudo random bits for final MAC calculations shown in figure 3.19.

$Ct_M\ [31...0] = C_t[31...0] \oplus S_M[31...0]$
$Ct_M\ [63...32] = C_t[63...32] \oplus S_M[63...32]$
$Ct_M\ [95...64] = C_t[95...64] \oplus S_M[95...64]$
$Ct_M\ [127...96] = C_t[127...96] \oplus S_M[127...96]$

$OSP\text{-}MAC = Ct_M\ [31...0] \oplus Ct_M[63...32] \oplus Ct_M[95...64] \oplus Ct_M\ [127...96]$

Figure 3.19: MAC Generation Scheme for OSP

3.6 KEYING MECHANISMS FOR OSP

A security protocol can be widely understood if it is elaborated in conjunction with a key management architecture and communication protocol. Key management architectures are designed to handle the complexities related to key creation, key distribution and key revocation. Another purpose of key management architectures is to protect keys from being exploited.

A keying mechanism is chiefly used for the distribution of keys throughout the network. OSP has been designed in such a way that practical implementation is not limited to any particular keying mechanism. Each keying mechanism is unique and each

one has its positive and negative points. Researchers have not been able to prove the effectiveness of any particular keying mechanism because some keying mechanisms provide better functionality in specific environments [16]. It is essential to consider all the keying mechanisms before a final decision is made about the effectiveness of a particular keying mechanism. Essentially a balance needs to be maintained between the amount of resources and the level of security. Selection of the appropriate keying mechanism depends upon the application domain, amount of resources available and the threat model.

Table 3.4: Keying Mechanisms with their Positive and Negative Aspects

Keying Mechanism	Positives	Negatives
Network Wide Key	Low Deployment Complexity	Network Compromise
Per Group Key	Medium Deployment Complexity	Group Compromise
Per Link Key	High Deployment Complexity	Only One Link Compromise

3.6.1 Link Key with OSP

The simplest yet effective mechanism for keying is the per link keying. Two nodes that wish to communicate among themselves maintain a key for communication. This method of keying protects the entire network from an attack in case a key is captured. This implies that an attacker can only decrypt traffic that is addressed to it. Also an attacker can only inject traffic through its immediate neighbours. If a system can detect the location of the attacker then it can be isolated from the rest of the network by denial of service to that particular neighbourhood around the attacker. Based on the architecture of OSP it can be implied that the protocol in conjunction with per link keying will produce

high level of security but with high resource consumption. The resource consumption occurs as a result of complexities owing to key distribution and communication costs.

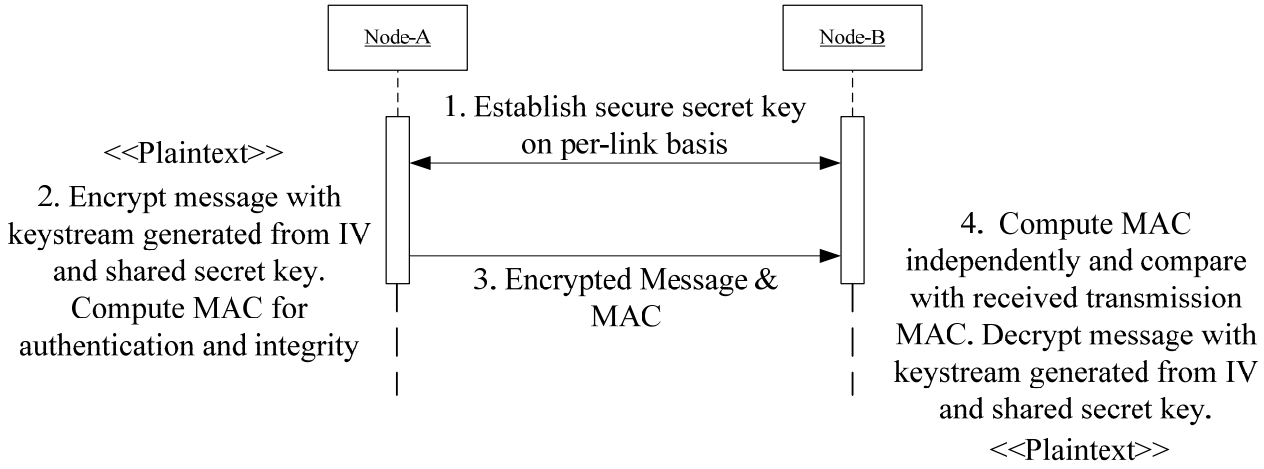


Figure 3.20: Sequence Diagram Depicting the Link Key with OSP

Per link keying should be used in environments where the required level of security is very high. Further the amount of available resources should be very high for utilization of this keying method.

3.6.2 Group keying with OSP

A collaborative approach towards keying is the use of a group key for communication among a large number of nodes. In this approach a single key is used among a group of nodes. If a particular node is captured then the communications among the group are accessible to the attacker. All communications outside the group are not accessible to the attacker. This implies that large group size will result in greater losses if any node is compromised. Therefore moderate group sizes need to be defined. Figure 3.9 shows how a group key is utilized in combination with OSP.

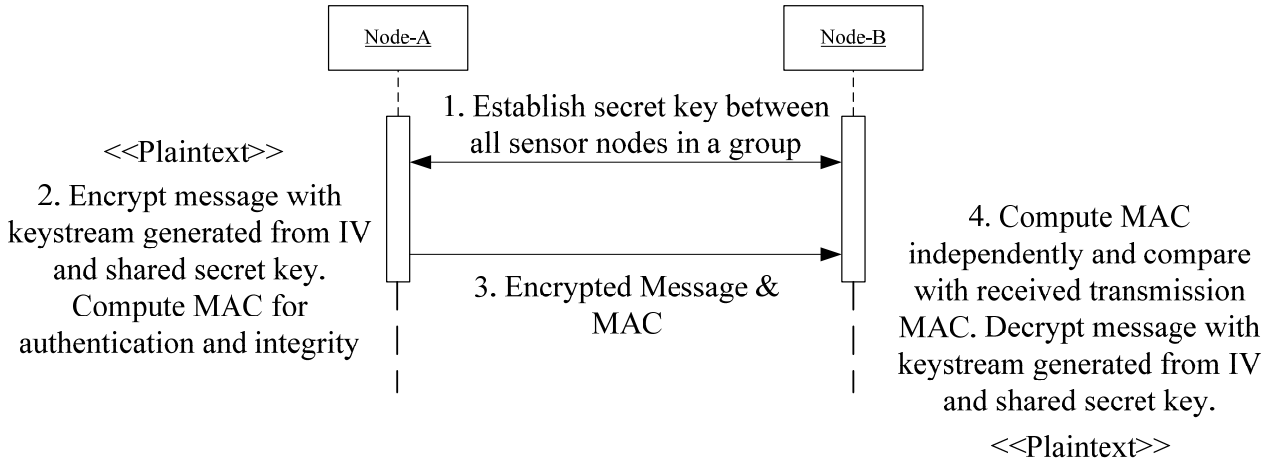


Figure 3.21: Sequence Diagram Depicting the Group Key with OSP

If group keying is used with OSP the level of security will be moderate with average amount of resource consumption. This form of keying should be used in environments where the required level of security is not very high.

3.6.3 Network Wide Keying with OSP

Network wide keying is a method that has very less complexities related to key distribution because a single key is utilized throughout the network. A major advantage of this keying scheme is that any authorized node can communicate within the network. All communications within the network are encrypted but if any single node is compromised then the entire network is at risk. Also network wide keying does not protect against node capture attacks. If any node is captured the attacker can listen to all communications and also inject his own messages into the system. Figure 3.20 shows how a network wide key is utilized in combination with OSP.

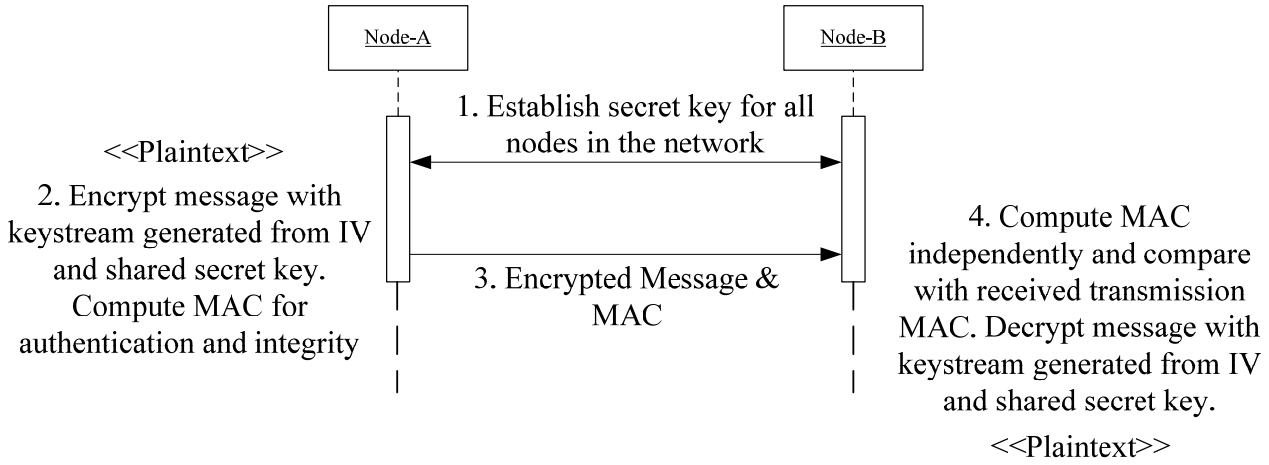


Figure 3.22: Sequence Diagram Depicting the Network Key with OSP

The network wide key should be used in environments where the data being communicated is not very critical and the available resources for security are not very high. If network wide keying is used with OSP the level of security will be low. Even if OSP continues to function without failure it needs to be understood that a single node compromise will result in the compromise of the entire network. Network wide keying will prove very successful in environments where there are no chances of an attack on the network.

3.7 SUMMARY

In this chapter the architecture and implementation of the newly proposed Optimized Security Protocol have been described in detail. OSP is a complete security protocol that fulfils the three most important requirements of security i.e. confidentiality, authentication and integrity. OSP has been designed by using message encryption along with MAC. For the successful operations of this security protocol a new optimized packet

format has been defined. The new packet format results in lower resource consumption due to eradication of redundant data field that exist in the TinyOS packet format.

Chapter 4

TESTING AND EVALUATION OF OSP

INTRODUCTION

OSP is a comprehensive protocol that provides high levels of security for resource constrained WSNs. To finalize the protocol it has to be tested and evaluated in relation to a wide range of parameters. Firstly, since the protocol is designed for a resource constrained environment therefore it has to be evaluated for its ROM/ RAM consumption. Also testing needs to be performed for the operating time of the protocol. OSP cannot be tested and evaluated in total isolation. It has to be tested and evaluated in comparison with other leading security protocols for WSNs.

4.1 OSP TESTING ENVIRONMENT

To formally test and evaluate OSP a proper environment had to be established. This environment provides support for visualization and result analysis.

4.1.1 Simulator

OSP has been tested using TOSSIM which is the default simulator for use with TinyOS With the help of TOSSIM thousands of sensors can be deployed at the same time and TOSSIM provides extended support of analysis of system implementation. TOSSIM is an compiles the code into a native executable that can run directly on a the simulation host.

This feature provides flexibility and thus results in support of thousands of simultaneous nodes. TOSSIM has been extensively compared with physical deployments and the results prove that the simulator is highly comparable and accurate.

The greatest advantage of using TOSSIM is its seamless connectivity with TinyOS. TOSSIM also operates at the network bit granularity level to capture behaviours and interactions that take place in the network.

4.1.2 Visualization Environment

TinyViz is a Java based graphical user interface that extends the functionality of TOSSIM by providing visualization support. TinyViz can be associated to a running application while TOSSIM waits for TinyViz to connect all the simulated sensors. TinyViz also allows its users to trace the execution of TinyOs applications; hence the users can observe the behaviour of their implementation pause and resume fashion.

4.2 OSP FRAMEWORK TESTING

Rigorous testing of OSP is through the various test vectors that provide varying inputs to the system. Fundamentally OSP is designed to take message of any size as input and process it to the implementation. If a message exceeds the standard length then the message is broken down into manageable blocks of size 128 bit. Each test vector is unique but fundamentally operates on three inputs i.e. secret key, initialization vector and plaintext. Varying these three inputs provides varying outputs. The purpose of this testing is to identify abnormal behaviour (if any), and also to identify the efficacy of the entire system.

4.2.1 Test Vector-I

The first test vector which forms input to the OSP framework for testing is shown in table 4.1 and shows the obtained ciphertext when plaintext is encrypted using encryption key and MAC obtained using MAC key.

Table 4.1: Test Vector-I

Test Vector-I	
Encryption Key	[0xa7, 0x92, 0xac, 0xfb, 0x43, 0xdc, 0x15, 0xa5, 0x08,0x60, 0x56, 0x00,0xa7, 0x81,0x70]
MAC Key	[0x67, 0x73, 0xfb, 0x15, 0xeb, 0x9b, 0xff, 0xad, 0xe5, 0xb1, 0x8g, 0x96, 0x02, 0x9c, 0xc3, 0x02]
IV	[0xac, 0x89, 0x00, 0x76, 0x00, 0x56, 0x00, 0x78]
Input Plaintext	[0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00]
Ciphertext	[0xa3, 0x76, 0xd2, 0xc4, 0xa4, 0x01, 0xa7, 0xc2, 0xe2 0x1e, 0x2f, 0x00, 0x2a, 0x36, 0x07, 0xc4]
Recovered Plaintext	[0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00]
MAC	[0x96, 0xfc]

4.2.2 Test Vector-II

The second test vector is designed to test if OSP continues to function with any given encryption key, MAC key, initialization vector and plaintext. Table 4.2 shows the details of the second test vector.

Table 4.2: Test Vector-II

Test Vector-II	
Encryption Key	[0xa1, 0xaf, 0xce, 0x02, 0xfd, 0x5c, 0xeb, 0x9b, 0xff, 0xad, 0xe5, 0x96, 0x0c, 0xda, 0xb1, 0x88]
MAC Key	[0x27, 0xa6, 0xad, 0x96, 0xeb, 0xac, 0xf8, 0xd8, 0x66, 0x03, 0xcb, 0x00, 0xeb, 0xc1, 0x10, 0x00]
IV	[0x00, 0x00, 0x91, 0x67, 0xfe, 0xac, 0xc3, 0x02]
Input Plaintext	[0x16, 0xa1, 0x28, 0x94, 0xd0, 0xeb, 0xe5, 0xaa, 0x84, 0x7f, 0xfe, 0x00, 0x36, 0x91, 0x00, 0xd1]
Ciphertext	[0x14, 0x00, 0x80, 0xa1, 0xf6, 0xd2, 0xdc, 0xa0, 0xe0, 0x87, 0x83, 0xfa, 0x12, 0xd9, 0x02, 0x39]
Recovered Plaintext	[0x16, 0xa1, 0x28, 0x94, 0xd0, 0xeb, 0xe5, 0xaa, 0x84, 0x7f, 0xfe, 0x00, 0x36, 0x91, 0x00, 0xd1]
MAC	[0x88, 0x60]

4.2.3 Test Vector-III

The third test vector is designed to further test OSP with varying encryption key, MAC key, initialization vector and plaintext. Table 4.3 shows the details of the third test vector.

Table 4.3: Test Vector-III

Test Vector-III	
Encryption Key	[0x27, 0xa6, 0xad, 0x96, 0xeb, 0xac, 0xf8, 0xd8, 0x66, 0x03, 0xcb, 0x00, 0xeb, 0xc1, 0x10, 0x00]
MAC Key	[0xa7, 0x92, 0xac, 0xfb, 0x43, 0xdc, 0x15, 0xa5, 0x08,0x60, 0x56,

	0x00,0xa7, 0x81,0x70]
IV	[0x92, 0x10, 0x01, 0xd8, 0xc6, 0xf5, 0x5b, 0x1c]
Input Plaintext	[0x89, 0xfc, 0xf6, 0x00, 0x1f, 0x96, 0x95, 0x0c, 0x3d, 0xb1, 0x88, 0x60, 0x02, 0xfd, 0xd4, 0x12]
Ciphertext	[0xff, 0x6c, 0xe8, 0x58, 0x75, 0x84, 0x70, 0xe1, 0x68, 0x9a, 0xe7, 0xc4, 0xf2, 0x12, 0x03, 0x01]
Recovered Plaintext	[0x89, 0xfc, 0xf6, 0x00, 0x1f, 0x96, 0x95, 0x0c, 0x3d, 0xb1, 0x88, 0x60, 0x02, 0xfd, 0xd4, 0x12]
MAC	[0x01, 0xa7]

4.2.4 Test Vector-IV

The fourth test vector reveals the effectiveness of OSP in processing various inputs. The results of test vector-IV prove that OSP can readily take any encryption/ MAC key, initialization vector and plaintext to produces the correct parameters. Table 4.4 shows the details of the third test vector. From all four test vectors one can readily conclude the effectiveness and correctness of OSP.

Table 4.4: Test Vector-IV

Test Vector-IV	
Encryption Key	[0xaa, 0x91, 0xfe, 0xe9, 0xcd, 0x3b, 0xfe, 0x02, 0x00, 0xcd, 0xe5, 0x3e, 0x0c, 0xd4, 0xb6, 0x99]
MAC Key	[0xa1, 0xaf, 0xce, 0x02, 0xfd, 0x5c, 0xeb, 0x9b, 0xff, 0xad, 0xe5, 0x96, 0x0c, 0xda, 0xb1, 0x88]
IV	[0xff, 0xbd, 0xf0, 0xac, 0xad, 0x74, 0x3a, 0xb8]

Input Plaintext	[0x0d, 0xcd, 0x32, 0x9b, 0xe0, 0x15, 0xd1, 0xd4, 0x36, 0xd7, 0xd6, 0x7f, 0x00, 0x58, 0x2e, 0x9b]
Ciphertext	[0xdf, 0x04, 0xae, 0x03, 0x16, 0x2f, 0x01, 0x97, 0xdf, 0x02, 0xc0, 0x35, 0x12, 0x27, 0xae, 0x97]
Recovered Plaintext	[0x89, 0xfc, 0xf6, 0x00, 0x1f, 0x96, 0x95, 0x0c, 0x3d, 0xb1, 0x88, 0x60, 0x02, 0xfd, 0xd4, 0x12]
MAC	[0xc0, 0x87]

4.3 MEMORY FOOTPRINT ANALYSIS

The analysis of a security protocol for resource constrained environement cannot be completed with a formal memory analysis. The memory of OSP has been measured by using TinyOS in collaboration with TOSSIM. TOSSIM is used to compile the code and analyze the code for any errors. If the compilation is completed without any errors then TOSSIM displays the amount of memory consumed. The memory footprint analysis of OSP is only presented in the form of ROM and RAM consumption.

4.3.1 ROM Consumption Study

To perform in depth analysis of the ROM consumption of OSP, a comparison of no security and OSP is presented. When mica nodes in a WSN communicate without OSP they consume 7076 bytes of ROM. When communication is based on our newly proposed OSP, three essential features of security namely confidentiality, authentication and integrity are provided at the total cost of 12470 bytes in ROM. This consumed ROM is well inside the maximum limit of ROM available to conventional sensor nodes. It is interesting to note that OSP is very lean as compared to other rival protocols like Dragon

MAC and RC4. When OSP is used to provide only encryption it easily outperforms both RC4 and Dragon. Dragon MAC [24] provides only authentication and integrity but consumes 18900 bytes thus giving a difference of 6430 bytes (18900-12470). As shown in figure 4.1 RC4 is a very costly protocol because it consumes more RAM as compared to OSP and provides only encryption. Whereas OSP consumes less number of bytes and provides confidentiality, authentication and integrity all in a single suite.

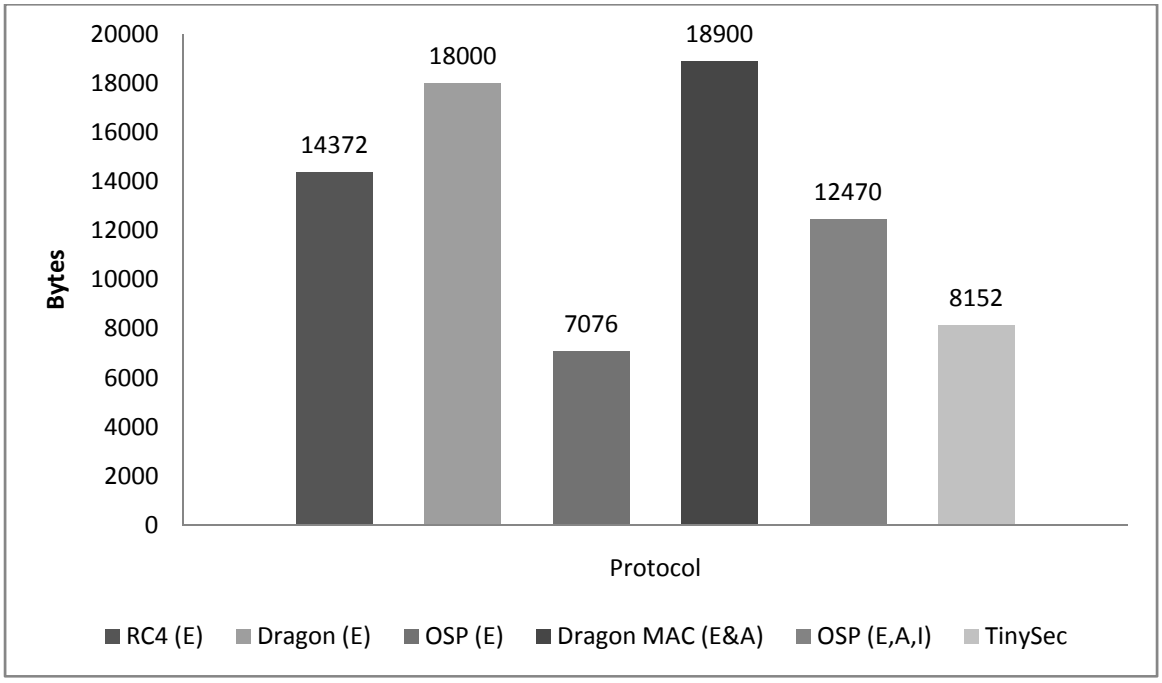


Figure 4.1: Comparison Graph for ROM Consumptions

Table 4.5 shows the ROM consumption results for OSP and other popular security protocols for WSNs. It is evident from the readings that OSP requires very little ROM as compared to its rival protocols. For instance the complete OSP requires less ROM than RC4 and Dragon but provides higher number of security features.

Table 4.5: ROM Consumptions for Competing Protocols

Description	ROM (Bytes)
RC4 (Encryption Only)	14372
Dragon (Encryption Only)	18000
OSP (Encryption Only)	7076
Dragon MAC (Encryption & Authentication)	18900
OSP (Encryption, Authentication, Integrity)	12470
TinySec	8152

4.3.2 RAM Consumption Study

To conduct comprehensive RAM consumption study of OSP the protocol is compared with other popular rival protocols like RC4 (Encryption), Dragon (Encryption), Dragon MAC (Encryption and Authentication). A comparison of OSP with other security protocols is necessary to understand the stability and efficiency of this new security suite. When discussing RAM consumptions OSP ranks slightly higher as compared with other similar protocols. The difference in RAM consumption is minor and is almost negligible. OSP can provide encryption consuming almost the same number of bytes as the encryption only Dragon protocol. When OSP provides encryption, authentication and integrity it consumes only 302 bytes more than the Dragon MAC which only provides encryption and authentication. Since OSP provides more features as compared to Dragon

MAC while consuming 302 extra bytes (1284-982) therefore the protocol can be considered efficient in terms of both ROM and RAM consumption [24]. Figure 4.2 shows the RAM consumption graph for OSP and other competing protocols.

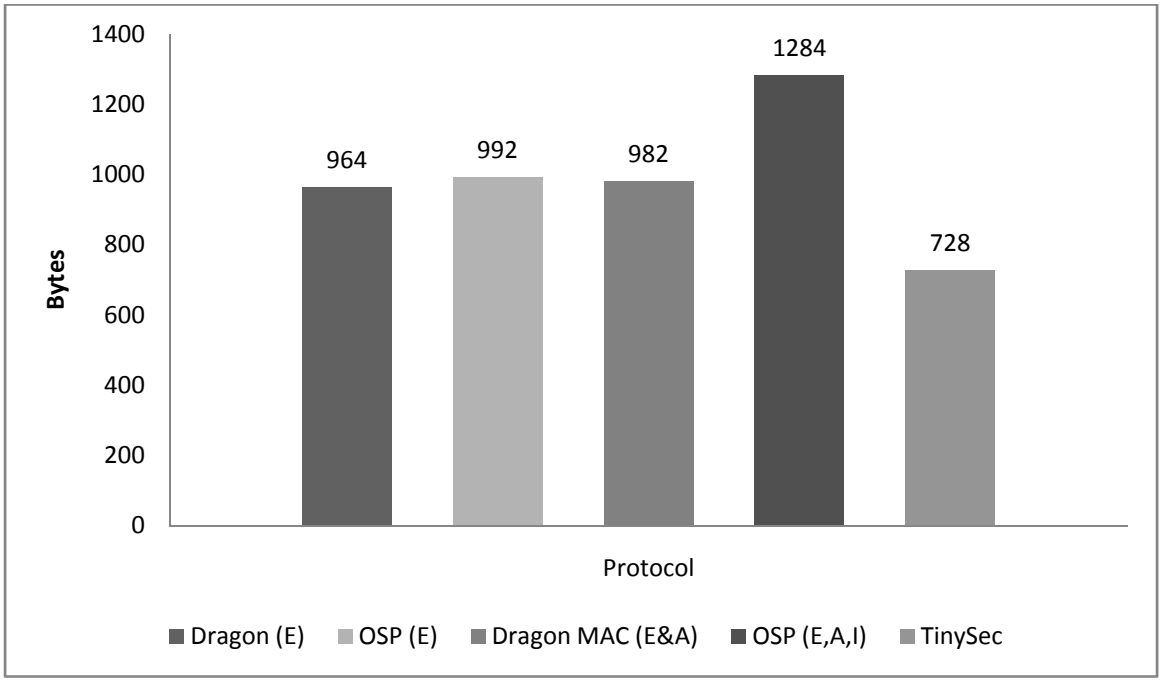


Figure 4.2: Comparison Graph for RAM Consumptions

OSP has a slightly elevated level of RAM requirement. Even though the RAM requirement may seem higher as compared to rival protocols but still it is well in range of the RAM available to WSNs. Table 4.6 shows the RAM requirements in bytes of various high level protocols for WSNs.

Table 4.6: RAM Consumptions for Competing Protocols

Description	RAM (Bytes)
Dragon (Encryption Only)	964

OSP (Encryption Only)	992
Dragon MAC (Encryption & Authentication)	982
OSP (Encryption, Authentication, Integrity)	1284
TinySec	728

4.4 PROCESSING TIME

This section presents a detailed study of OSP in terms of processing time. Since OSP is developed using TinyOS and TOSSIM therefore the processing time is calculated by the internal timers of the simulation environment. Even though OSP is very efficient in terms of memory footprint, the protocol needs to be analyzed for processing time. The results of processing time ensure freshness of the data communicated along with the guarantee that a sensor can communicate a packet with another sensor within a certain time range.

To formally study the processing time of OSP the processing time with encryption only has to be extracted so that one can determine the overall time OSP takes to operate. The time taken by OSP to provide only encryption is 39μs. When OSP is used to provide encryption, authentication and integrity in a single suite then it requires a time of 153.76μs. Figure 4.3 shows the graph of operating times for OSP and other competing protocols.

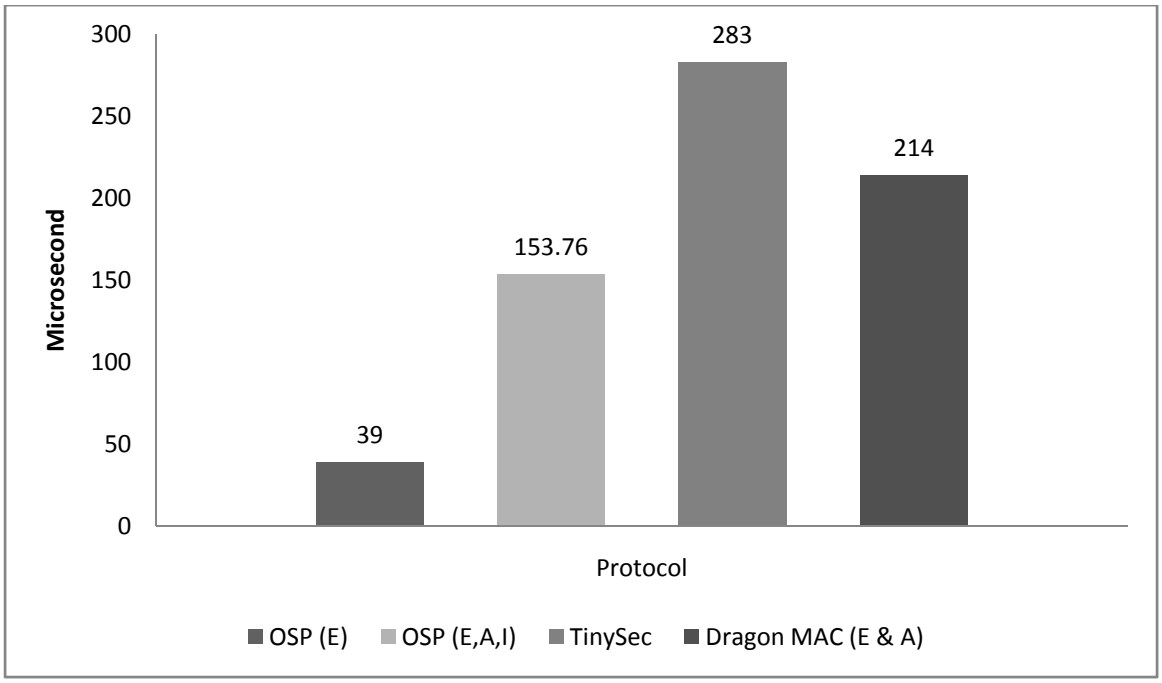


Figure 4.3: Comparison Graph for Operating Time (μs)

The complete OSP suite requires less operating time as compared to TinySec which is well renowned for its efficiency and low resource requirements. Table 4.7 shows the processing times of OSP and other similar protocols.

Table 4.7: Processing Time (μs) for Competing Protocols

Description	Time (μs)
OSP (Encryption Only)	39
OSP (Encryption, Authentication, Integrity)	153.76
TinySec	283
Dragon MAC (Encryption & Authentication)	214

4.5 SECURITY RESILIANCE ANALYSIS

OSP is a comprehensive security solution for resource constrained WSNs. OSP has been designed with new packet formats. The purpose of this exercise is to change the way we view security in WSNs. Even though OSP is very light weight in nature but still there is no compromise in the level of security provision. OSP provides encryption, authentication and integrity without the requirement for resources. The key size of 128 bit ensures that the chances of guessing the key is next to impossible. The superior diffusion properties of Rabbit acts as a deterrent to all known attacks for WSNs [28]. Another benefit of the diffusion rounds translates into less need for rekeying in the network.

OSP has been designed with an IV that attempts to reuse fields from the OSP packet format. The IV has been designed to further prevent the chances of an attacker finding patterns among the transmitted messages. The IV has the SRC and CTR field. These fields assist in making two messages differ from each other thereby reducing the chances of a message repetition. Also OSP has a MAC function that assists in determining if the message is authentic and the sender is really who he claims to be. Another big advantage of using a MAC is that any message that has been affected due to channel noise is detected and then discarded. OSP has been designed to operate with the Rabbit stream cipher because of its recognized security characteristics and minimal resource consumption.

4.6 SUMMARY

To fully test OSP it has been evaluated from various aspects like ROM/ RAM consumption, processing time and security provision. The results have been obtained by

using TinyOS, TOSSIM and TinyViz. Extensive testing of OSP has shown that this new protocol is very stable and requires little resources to operate. OSP has been evaluated in comparison with other popular protocols like TinySec, Dragon MAC and RC4. The results demonstrate that OSP has all the properties that are required by a highly acknowledged security protocol for WSNs.

Chapter 5

CONCLUSIONS AND FUTURE WORK

Researchers have been successful in promising wonderful applications for the field of WSNs. Both researchers and the academia acknowledge the fact that security provision for WSNs is necessary if they are to be widely deployed. What they mean by security is the provision of confidentiality along with advanced features that cater to authentication and integrity. Until recently all research endeavours were directed towards conserving essential resources. This is the precise reason why security provision was considered an expensive attribute/ feature. Latest research in the field of WSN security has been able to promise high levels of security without compromise in resource consumption. This work is geared towards the development of an Optimized Security Protocol - OSP that can be widely acknowledged and implemented.

5.1 CONCLUSIONS

In this project an attempt has been made to develop a new protocol that provides three features namely confidentiality, authentication and integrity without compromising the limited resources available to sensor nodes. The purpose of OSP is to provide a complete security suite that prevents unlawful access and modification of data and at the same time prevents node impersonation by an attacker. The protocol is based on an optimized packet format that is derived from the TinyOS packet format. The newly proposed packet format removes redundant fields and also attempts to reuse existing fields from the

TinyOS packet to reduce the packet size and also create space for handling advanced security mechanisms like MAC. The newly proposed OSP packet is fully supported by a new and unique architecture that can operate with any key exchange mechanism. This keying independent architecture ensures greater flexibility with ease of implementation.

To prove the efficiency of OSP it has been implemented in TinyOS using the TOSSIM simulator. OSP has been tested using TinyOS so that it can be fully deployed on physical sensor nodes. The compilation results of OSP prove that this new protocol occupies very little ROM and RAM while it operates. This implies that when OSP will be physically implemented it will not exhaust the available resources of the sensors. Another attribute of OSP is its very optimal operating time. OSP performs quick communications because of its smaller and optimized packet size. Also OSP does not require unnecessary/ repeated sensor-to-sensor communication thereby conserving time and other essential resources.

5.2 FUTURE WORK

Although OSP has been fully tested and is considered complete in all respect, but there is still room for implementing and testing the protocol on real sensor test bed. The true behaviour of a protocol can be truly tested if it is implemented on real sensor nodes.

OSP has demonstrated good results in terms of memory requirements and time requirements but still there is room for optimization. The algorithm and processing can be tweaked and adjusted for improved results and performance.

OSP is fundamentally a security protocol based on the Rabbit stream cipher hence there is ample space for research in combining the OSP architecture and packet format with a

stream cipher other than Rabbit. Also OSP can be extended to propose and implement a protocol that attempts to incorporate OSP with different Routing protocols.

To further strengthen the security of the entire system OSP can be designed such that it supports security features related to availability of sensor nodes. If this last feature is incorporated into OSP then it will result in a truly outstanding protocol that completely addresses all the security concerns.

BIBLIOGRAPHY

[1] M. Saraogi, "Security in Wireless Sensor Networks," *http://www.cs.utk.edu/~saraogi/594paper.pdf*, Spring 2005.

[2] A. Perrig, J. Stankovic, and D. Wagner, "Security in Wireless Sensor Networks," *Communications of ACM*, vol. 47, no. 6, pp. 53–57, June 2004.

[3] H. Jason, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System Architecture directions for Networked Sensors," *ACM SIGPLAN Notices*, vol. 35, no.11, pp.93-104, November 2000.

[4] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A survey," *Computer Networks*, vol. 38, no. 9, pp 393-422, March 2002.

[5] F. Akyildiz, T. Melodia, K. Kaushik and R. Chowdhry, "A survey on Wireless multimedia Sensor Networks," *Communications of ACM*, vol. 51, no. 4, pp 921-960, March 2007.

[6] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on Sensor Networks," *IEEE Communications Magazine*, vol. 40, no.8, pp. 102–114, August 2002.

[7] J. Deng, R. Han, and S. Mishra "Security, privacy, and fault tolerance in Wireless Sensor Networks," book chapter in *Wireless Sensor Networks*, pp. 215-234: Artech House, July 2005.

[8] S. Duri, M. Gruteser, X. Liu, P. Moskowitz, R. Perez, M. Singh, and J. Tang, "Framework for security and privacy in automotive telematics," in

Proc. 2nd ACM International Workshop on Mobile Commerce, pp. 25-32, Georgia, USA, 2000.

[9] M. Ilyas and I. Maheoub, *Handbook of Sensor Networks: Compact Wireless & Wired Sensing System*, New York: CRC Press, July 2004.

[10] K. Sreenath, L.Lewis and O. Popa, "Simultaneous Adaptive Localization of a Wireless Sensor Network," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 11, no.2, pp. 14-28, April 2007.

[11] P. Levis, N. Lee, M. Welsh and D. Culler "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications," *in Proc. of the first Int. Conf. on Embedded Networked Sensor Systems*, pp. 126-137, California, USA, 2003.

[12] M. Neufeld, A. Jain, D. Grunwald "Nsclick: Bridging Network Simulation and Deployment" *in Proc. of the fifth Int. Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems*, pp. 126-137, Georgia, USA, 2003.

[13] L. Girod, J. Elson, A. Cerpa, T. Stathopoulos, N. Ramanathan, and D. Estrin. "EmStar: A software environment for developing and deploying heterogeneous sensor actuator networks," *ACM transactions of Sensor Networks*, vol. 3, no.3, article no. 13, 2007.

[14] T. Zia and A. Zomaya, "Security Issues in Wireless Sensor Networks," *in Proc. of Int. Conf. on Systems and Networks Communications*, vol. 11, no. 9, pp. 40-45, October 2006.

[15] L. Zhou and Z. Haas, "Securing ad hoc networks," *IEEE Network Magazine*, vol. 13, no.6, pp. 24-30, December 1999.

[16] H. Chan, A. Perrig, and D. Song “Random key pre-distribution schemes for sensor networks,” *in Proc. of the 2003 IEEE Symposium on Security and Privacy*, pp. 197-202. IEEE Computer Society, 2003.

[17] W. Du, J. Deng, Y. S. Han, and P. K. Varshney “A pairwise key pre-distribution scheme for wireless sensor networks,” *in Proc. of the tenth ACM Conf. on Computer and communications security*, pp 42–51, NY, USA, 2003.

[18] H. Chan and A. Perrig “Security and privacy in sensor networks” *IEEE Computer Magazine*, pp 103–105, 2003.

[19] A. Perrig, J. Stankovic, and D. Wagner “Security in wireless sensor networks,” *ACM Communications*, vol. 47, no.6, pp. 53–57, 2004.

[20] A. Menezes, *Handbook of Applied Cryptography*, pp. 353-409: CRC Press, 1996.

[21] B. Schneier, *Applied Cryptography*, second ed.: John Wiley & Sons, Inc. 1996.

[22] Y. Law, J. Doumen, and P. Hartel “Survey and benchmark of block ciphers for wireless sensor networks,” *ACM transactions on Sensor Networks*, vol. 2, no. 1, pp. 65-93, February 2006.

[23] “Ecrypt Stream Ciphers,” *http://www.ecrypt.eu.org/*, May 2008.

[24] S.Y Lim, C.C. Pu, H. T. Lim and H. J. Lee. “Dragon-MAC: Securing Wireless Sensor Networks with Authenticated Encryption” May 2007

[25] H. Wu, "Stream Cipher HC-256," *Lecture Notes in Computer Science, Fast Software Encryption*, 3017/2004, pp. 226-244, July 2004.

[26] A. Biryukov, "A New 128-bit Key Stream Cipher LEX," *Lecture Notes in Computer Science, Selected Areas in Cryptography*, 4356/2007, pp 67-75, September 2007.

[27] L. Batina, S. Kumar, J. Lano, K. Lemke, N. Mentens, C. Paar, B. Preneel, K. Sakiyama and I. Verbauwhede, "Testing Framework for eSTREAM Profile II Candidates," *www.ecrypt.eu.org/stream/papersdir/2006/014.pdf*

[28] M. Boesgaard, M. Vesterager, T. Christensen and E. Zenner, "The Stream Cipher Rabbit," *Lecture Notes in Computer Science, Fast Software Encryption*, vol. 2887/2003, pp. 307-329, February 2004.

[29] D. Gay, P. Levis, D. Culler and E. Brewer, "NESC 1.1 Language Reference Manual," *http://www.tinyos.net/tinyos-1.x/doc/nesc/ref.pdf*, May 2003.

[30] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J.D. Tygar, "SPINS: Security protocols for Sensor Networks," *ACM Transactions on Wireless Networks*, vol. 8, no. 5, pp. 521-534, 2002.

[31] C. Karlof, N. Sastry and D. Wagner, "TinySec: A Link Layer Security Architecture for Wireless Sensor Networks," Proceedings of the second ACM Conference on Embedded Networked Sensor Systems," pp. 162-175, Baltimore, USA, November 2004.

[32] C. Fok, "TinyOS Tutorial," *http://www.princeton.edu/~wolf/EE CS579/imotes/tos_tutorial.pdf*, 2004.

[33] H. Tahir, R. Tahir and M. Y. Javed "OSP – Optimized Security Protocol for Wireless Sensor Networks" Published IEEE International Confrence on Security Science and Technology ICCSST 2011, Chonqing, China, January 21-23, 2011.

RELATED RESEARCH PUBLICATIONS

H. Tahir, M. Y. Javed, R. Tahir “Design and Implementation of an Optimized Security Protocol for Wireless Sensor Networks”. Journal of Computing Volume 3 , Issue 7, July 2011.

H. Tahir, M. Y. Javed “OSP – Optimized Security Protocol for Wireless Sensor Networks”. 2011 International Conference Security Science and Technology - ICSST” Chongqing China, 21-23 January 2011.

H. Tahir, M. Y. Javed “Service Guarantees in Wireless Sensor Networks”. IEEE International Conference on Emerging Technologies ICET 2009. Islamabad. pp 433-436.