

Secure Solution to Data Transfer from Sensor Node to Sink against Aggregator Compromises

TO 7292



DATA ENTERED

Developed By

Sarfraz Azam

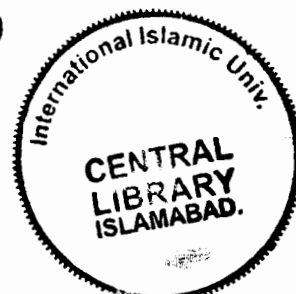
(473-FBAS/MSCS/F08)

Supervised By

Mr. Mata ur Rehman

Department of Computer Science
Faculty of Basic and Applied Sciences,
International Islamic University, Islamabad.

2010



Accession No TH7292

DATA ENTERED

MS
681.2
SAS

- 1- Sensor networks
- 2- Wireless LANs

Department of Computer Science
International Islamic University Islamabad

Dated: -----

Final Approval

This is to certify that we have read the thesis submitted by **Sarfraz Azam**, registration# 473-FBAS/MSCS/F08. It is our judgment that this project is of standard to warrant its acceptance by the International Islamic University, Islamabad, for the Degree of **MS in Computer Science**.

Project Evaluation Committee

External Examiner:

Dr. Abdus Sattar,

Former Director General,
Pakistan Computer Bureau, Islamabad.



Internal Examiner:

Dr. Muhammad Zubair,

Assistant Professor,
Department of Computer Science,
Faculty of Basic and Applied Sciences,
International Islamic University, Islamabad.



Supervisor:

Mr. Mata ur Rehman,

Assistant Professor,
Software Engineering Department,
Faculty of Basic and Applied Sciences,
International Islamic University, Islamabad.



Dedication

I dedicate this research project to my beloved PARENTS

A Dissertation Submitted To
Department of Computer Science,
Faculty of Basic and Applied Sciences,
International Islamic University, Islamabad
As a partial Fulfillment of Requirements for the Award of the
Degree of
MS in Computer Science

Declaration

I hereby declare that this Thesis "**Secure Solution to Data Transfer from Sensor Node to Sink against Aggregator Compromises**" neither as a whole nor as a part has been copied out from any source. It is further declared that I have done this research with the accompanied report entirely on the basis of my personal efforts, under the proficient guidance of my teachers especially my supervisor **Mr. Mata ur Rehman**. If any of the system is proved to be copied out of any source or found to be reproduction of any project from any of the training institute or educational institutions, I shall stand by the consequences.

Sarfraz Azam
Reg # 473-FBAS/MSCS/F08

Acknowledgement

I like to express my gratitude to Allah Almighty the Merciful, the Beneficent and the creator of this universe, for providing me the abilities and knowledge to complete this dissertation. I am heartily thankful to supervisor Mr. Mata ur Rehman and Dr. Amir Qayum from Muhammad Ali Jinnah University, for the evolution of idea to commence this dissertation in the first instance.

I am thankful to Dr. Abid Ali Minhas whose encouragement, able guidance and altruistic support during this dissertation. His support also enables me to develop eloquent understanding of topic. He is the one who actually provided me the platform of Bahria University Islamabad.

I am indebted to Raja Adeel Akthar for his support in the implementation of this project, he guided from scratch to the final level. I am also grateful to him for his understanding and gestures towards me and my project. He consistently helped me to overcome those problems which I really faced during the idea and thesis implementation.

Very special thanks to Dr. Ali Hammad Akbar from (CS Department) UET Lahore for providing me remarkable guidance and university platform in the implementation of my project.

For excellent technical support in this thesis, I would like to pay my thanks to Mr. Adnan Iqbal from NUST and Wasif Tanveer from UET Lahore.

All of my true friends and colleagues support me a lot especially, I would like to express my admiration to Miss Rizwana Manzoor for her quick and extra-ordinary support, without her persistent help, this dissertation would not have been possible.

I would like to mention that my family supports me morally and financially during whole my academic career which enabled me to work dedicatedly especially my father always encouraged me that I remain persistent with my studies.

Sarfraz Azam
Reg # 473-FBAS/MSCS/F08

Project in Brief

Project Title: Secure Solution to Data Transfer from Sensor Node to Sink against Aggregator Compromises

Undertaken By: Sarfraz Azam (473-FBAS/MSCS/F08)

Supervised By: Mata ur Rehman
Assistant Professor
(Software Engineering department)
International Islamic University Islamabad

Start Date: July 2009

Completion Date: September 2010

Tools and technologies: NesC
TOSSIM
TinyViz
PowerTOSSIM
MS Office

Operating System: TinyOS 1.x
Windows XP

System used: Intel(R) Pentium(R) M Processor 1.80 GHz
RAM 1 GB

Abstract

Energy is an igniting issue in the domain of wireless sensor networks. To overcome this issue, the technique of data aggregation is introduced. As the aggregator node collects data from many sensors so it becomes very interesting for compromisers. Unfortunately these aggregator nodes could be under the influence of various attacks. Under such circumstances, we have to make our data more secure, to protect from adversary attacks. Proposed RSS (Re-Sequencing Scheme) provides secure data aggregation. It is based on symmetric key, information dispersal algorithm and multipath routing. Symmetric key is injected in the node at deployment time and it is shared only between the node and sink. Node sense data places, it in buffer then disrupts the data sequence according to symmetric key. It splits data into many shares and routes them on multiple paths. On receiving, these aggregated shares are reconstructed by the sink and arrange the data in order, with the help of symmetric key. The RSS provides authenticity, data availability and protection against eavesdropping, data tampering and denial of service attacks, even in the presence of compromise nodes.

Table of Contents

<u>Contents</u>	<u>Page #</u>
Chapter 1 Introduction	
1. Introduction.....	1
1.1 Wireless Sensor Network	1
1.2 Motivation.....	2
1.3 Problem Domain	3
1.4 Thesis Contribution	3
1.5 Thesis Organization	3
Chapter 2_ Wireless Sensor Network	
2. Wireless Sensor Network	4
2.1 WSN Architecture	5
2.1.1 Anatomy of Sensor Hardware	5
2.1.2 Components of Sensor Node	6
2.1.2.1 Sensing Unit.....	6
2.1.2.2 Processing Unit (Microcontroller).....	6
2.1.2.3 Transceiver	7
2.1.2.4 Power Unit.....	7
2.2 Sensor Node Software	8
2.2.1 Operating System.....	8
2.3 Sensor Node Hardware	8
Chapter 3 Literature Survey	
3. Literature Survey	10
3.1 Different topologies of WSN.....	10
3.2 Aggregator Node	11
3.3 Previous Work	12
3.4 Limitations in Literature Survey.....	19
3.4.1 Limitations on Sensor Side.....	20
3.4.2 Limitations on Aggregator Node	20
3.4.3 Limitations on Paths	20
3.5 Objectives	20
Chapter 4 Problem Analysis	
4. Problem Introduction	21
4.1 Problem Statement.....	22
4.2 Problem Scenarios	22
4.3 Focus of Research.....	24

Chapter 5 Proposed RSS (Re – Sequencing Scheme)	
5. Proposed RSS (Re - Sequencing Scheme).....	25
5.1 Design Goals	25
5.2 Reference Architecture	25
5.3 Network Assumptions	25
5.4 RSS (Re - Sequencing Scheme)	26
5.4.1 Initial Preparation	26
5.4.2 Sensor Node.....	29
5.4.3 Sink Node (On Receiving Shares).....	32
5.5 Scheme Analysis.....	34
Chapter Simulation Details	
6. TinyOS implementation details	40
6.1 TOSSIM Simulator	40
6.2 How to Compile and Run Application in TOSSIM.....	41
6.3 TinyViz.....	42
6.4 NesC Language.....	43
6.5 Simulation Setup.....	45
6.6 Simulations	45
6.6.1 Simulation with TinyViz	46
6.6.2 Simulation with PowerTOSSIM.....	48
Chapter 7 Results	
7. Results.....	50
7.1 Performance Metrics.....	50
7.2 Simulation Results	50
7.3 Comparison With other Schemes	53
Chapter 8 Conclusions and Future Work	
8.1 Conclusions.....	55
8.2 Future Work.....	55
References.....	56
Acronyms.....	59

List of Figures

Figure 1 Wireless Sensor Network Applications.....	1
Figure 2 Overview of Wireless Sensor Network	4
Figure 3 Wireless Sensor Network Architecture	5
Figure 4 Anatomy of a Wireless Sensor node	6
Figure 5 Sensor Node with Solar Panel	8
Figure 6 Varied Types of Sensor Hardware with Specifications	9
Figure 7 One - Hop Model.....	10
Figure 8 Multi - Hop Model	11
Figure 9 Cluster Based Model	11
Figure 10 DataAggregation	12
Figure 11 Secure Aggregation Process of WSN.....	13
Figure 12 Brief Protocol Description	14
Figure 13a Slicing of Data Reading.....	16
Figure 13b Mixing of Data Reading	16
Figure 14 Peer Monitoring Nodes	16
Figure 15 ABBA(A Balls and Bins Approach)	17
Figure 16 Homomorphic Encryption.....	18
Figure 17 Concealed Data Aggregation in Groups.....	19
Figure 18 Shares Transmission on Multiple Paths	19
Figure 19 Multiple Compromised Paths.....	22
Figure 20 Data Reconstruction with “t-1” Shares	23
Figure 21 Map file Containing Symmetric Keys.....	26
Figure 22 Flow Chart Symmetric Key Generation for each Node	27
Figure 23 Injected Symmetric Key.....	29
Figure 24 Sensed Data Buffer.....	29
Figure 25 RSS (Re-Sequencing Scheme) working.....	30
Figure 26 Flow Chart of Sensor Side Process	30
Figure 27 Re-sequencing of Received Buffer	32
Figure 28 Flow Chart of Re-sequencing Process on Sink Side.....	33
Figure 29 ADMA Scenario -1	34
Figure 30 RSS Scenario -1	35
Figure 31 ADMA Scenario -2	35
Figure 32 RSS Scenario -2	36
Figure 33 ADMA Scenario -3	37
Figure 34 RSS Scenario -3	37
Figure 35 Compiling TinyOS Applications.....	42
Figure 36 Screen Shot of TinyViz.....	43
Figure 37 Formation of Component	44

Figure 38 TinyOS Layered Structure	44
Figure 39 Screen Shot of Network Topology.....	45
Figure 40 Screen Shot of RSS (Re – Sequencing Scheme) working	47
Figure 41 Screen Shot of PowerTOSSIM Output File	49
Figure 42 Number of Packets Sent by Individual Node.....	51
Figure 43 Energy Consumption by Individual Node.....	52
Figure 44 Total Energy Consumption by Each Simulation.....	52
Figure 45 Total Energy Overhead	53

List of Tables

Table 1 Simulation Parameters for TinyViz.....	46
Table 2 Input Parameters for PowerTOSSIM	48
Table 3 Simulation Parameters for Power TOSSIM	48
Table 4 Comparison of RSS with other Secure Aggregation Schemes.....	54

Chapter 1
Introduction

1. Introduction

Wireless Sensor Network is an emerging and near future technology of ad-hoc networks. It is an upcoming technology and researchers have taken it challengingly. Due to upcoming technology, it is an attractive topic for researchers and they are contributing to it. It has opened many research areas. Our research work is also a contribution in this area of research. In this chapter, we will briefly discuss the technological aspects of wireless sensor networks.

1.1 Wireless Sensor Network

*“A self organizing network consists of many tiny sensor nodes which communicate wirelessly with each other using radio signals, are operated with battery and can sense, observe etc, is called **Wireless Sensor Networks.**”*

In essence they are developed for surveillance against enemies. Later on their usage became wide spread, and they are being used for domestic, industrial and environmental applications.

They have varied applications in many fields:

- Environmental Monitoring [1]
- Machine Health Monitoring [2]
- Greenhouse Monitoring [3]
- Fleet monitoring [4]
- Fire Rescue [28]

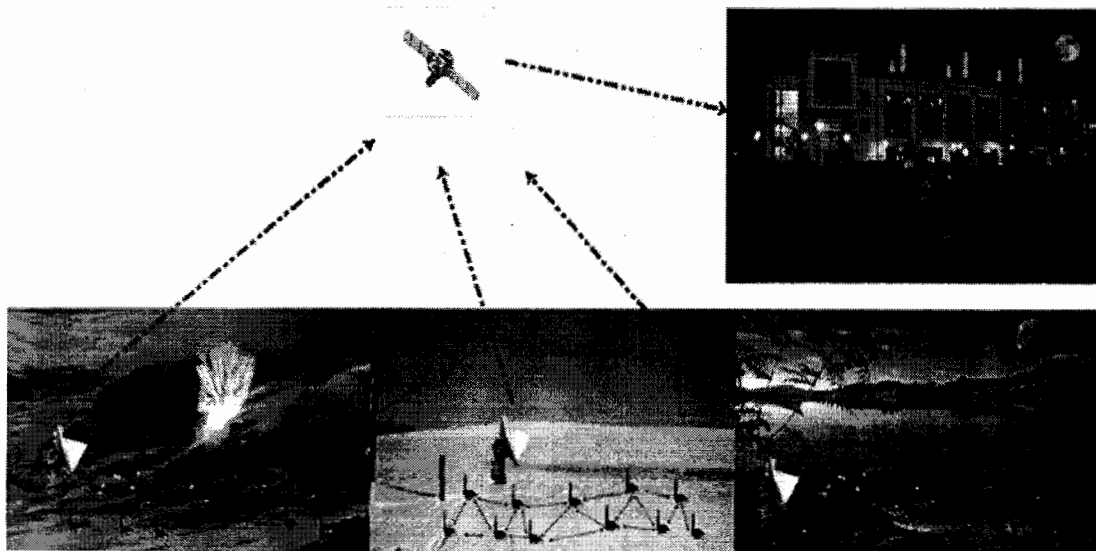


Figure 1 Wireless Sensor Network Applications [5]

WSN has been classified into several research domains which require a lot of research to make them attractive for civilian, military and industrial usage. The above figure1 shows different applications of WSN, namely,

- Sensing
- Energy
- Routing
- Security
- Computation
- Quality of service
- Hardware Development

1.2 Motivation

Sensor nodes are tiny in size, operated with battery, capable of sensing information, performing processing over gathered data, communicating with interconnected nodes with radio signals in the network. As these sensors are battery operated, energy resource is a burning issue relating to them. To overcome this energy issue, many techniques, have been proposed.

Data Aggregation technique is one of the most significant. With it, communication overhead reduces and energy budget prolongs. In this way the life-time of the network increases. Aggregator node collects data from several sensor nodes in the network then performs many mathematical operations over it. Aggregator makes the sensors collected data small or in size and then transmits to the base station. The communication is only performed by the aggregator node and sensor nodes don't use their energy for communication.

As aggregator node receives data from many sensor nodes so it becomes more attractive for adversaries. Instead of compromising many sensor nodes, adversary just attacks the aggregator node which is collecting the information from many sensor nodes. If it is compromised then all data transmission through it becomes risky. Aggregator can be under the influence of many dangerous attacks.

Mostly sensors are used for secret missions; so they transmit secret information which is very important. To protect this information form adversary, is challenging. The basic purpose of the deployment of sensor nodes is to get the important information from where ever these nodes are deployed. Suppose sensors are deployed to measure the temperature of boiler and they are compromised and not providing correct information, the boiler can burst with the increase in temperature. This means that sensor transmitted reading should need high security. If sensor nodes are working properly but there measured information is not reaching the base station accurately, it means that deployed sensors are useless.

It is concluded from the above statements that there should be some scheme or mechanism which makes data secure and protects from adversary. Security of the data is very important and if we make it better then we can better utilize these sensor nodes for our purpose. If security of data will not maintained then the use of all these equipments will not be significant.

1.3 Problem Domain

Energy resource is limited in wireless sensor nodes. In order to prolong the network life-time, we use aggregation technique. Aggregator node sometimes called intermediate node, collects data from several sensors and after aggregation transmits it towards the sink. This data aggregation technique reduces the communication and automatically energy consumption becomes less. Aggregator node has collection of data so it is attractive for attackers. Due to data aggregation, effective attacks can be performed over collected data. These attacks can be eavesdropping, data tampering, replay, denial of service etc. When an aggregator is compromised, the adversary knows, what is going through the traffic and according to its wishes, it can alter the information.

1.4 Thesis Contribution

In this research work a security protection scheme is proposed that protects sensors data from adversary, even when aggregator nodes are compromised. "RSS (Re-Sequencing Scheme)" makes the sensor data protected and it provides secure data aggregation. Our scheme resists against dangerous attacks of adversary and provides protection in such type of situations. Proposed scheme provides "*secure solution to data transfer from sensor node to sink against aggregator compromises*"

1.5 Thesis Organization

Subsequent chapters are organized as follows: Chapter 2 provides a brief introduction to the wireless sensor networks. Chapter 3 describes the literature survey and the related information to the problem domain. Chapter 4 explains the actual identified problem briefly. Chapter 5 provides the proposed solution of the identified problem. Chapter 6 describes the implementation and the simulation details of the proposed solution. Chapter 7 shows the results and comparisons with other renowned techniques of the scheme briefly. Chapter 8 contains the conclusion and future work.

Chapter 2
Wireless Sensor Networks

2. Wireless Sensor Networks

Wireless sensor network basically belongs to ad-hoc networks. As they are ad-hoc, their infrastructure is a little different from other networks. WSN anatomy consists of low cost sensor nodes and a base station which has high capabilities as required by the application. Both can communicate with each other using radio signals. Sensor can sense data, process over the sensed data and then transmit it. Sensor nodes are operated with battery, so their life is directly proportional with energy budget. The decrease in energy resource is the countdown of sensor node life. On other side there is a base station which is independent of this complexity. It has enough resources like energy, memory, fast processing, and internet to communicate with other world. As against this sensor has less memory, less processing power and less energy resource. Due to less energy resource, processing power and memory can't be increased and if we increase energy resource, then the size of the sensor becomes greater and it becomes useless for many applications. Within these limitations, we have to create such technique which makes better utilization of available resources and provides us better services.

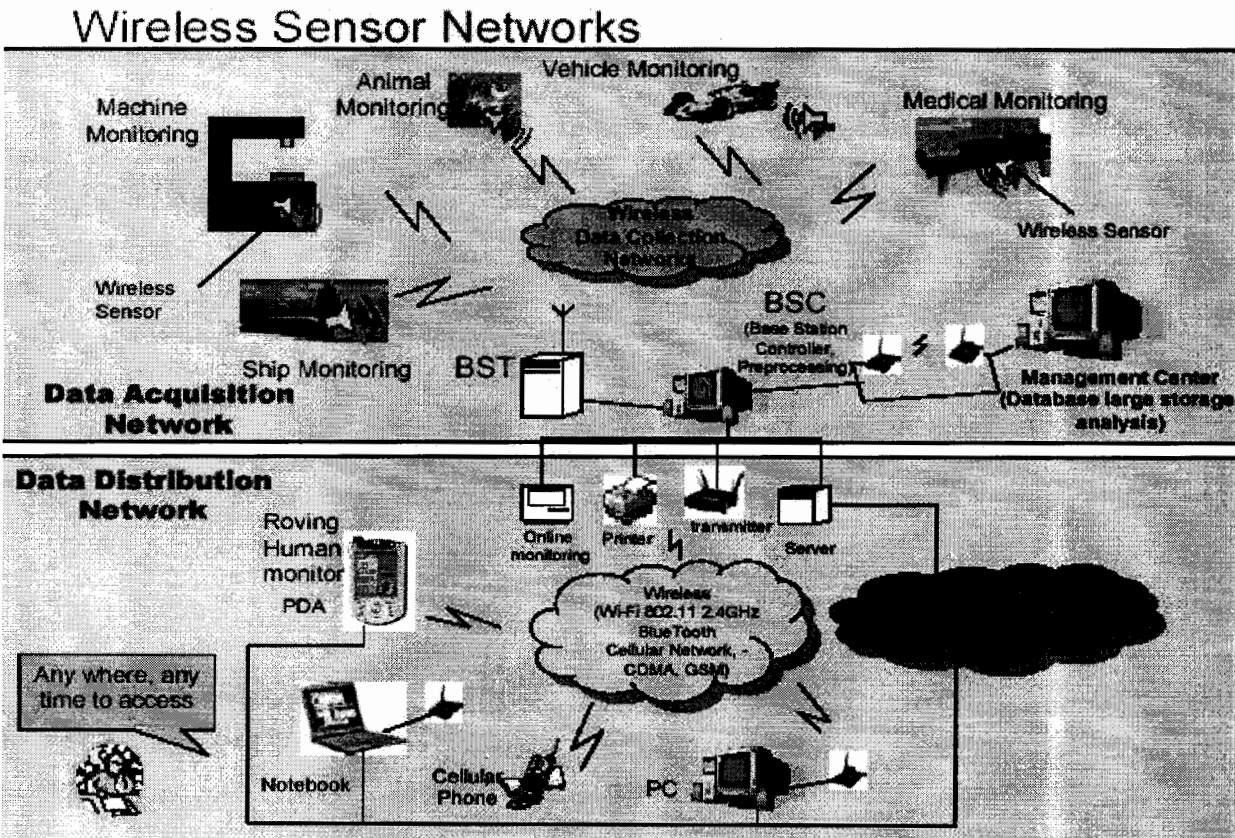


Figure 2 Overview of Wireless Sensor Network [6]

2.1 WSN Architecture

We should have an area of interest where we have to deploy these sensor nodes. The sensor would sense data, do some defined computations over it then transmit it to the sink. On the other hand sink node have strong capabilities; it applies different schemes and mathematical formulas to the received data and extracts useful information from it. In some schemes sensors are programmed to encrypt the sensed information before transmitting but some nodes send information without encrypting. If received data is in encrypted form then different operations are required to extract the information.

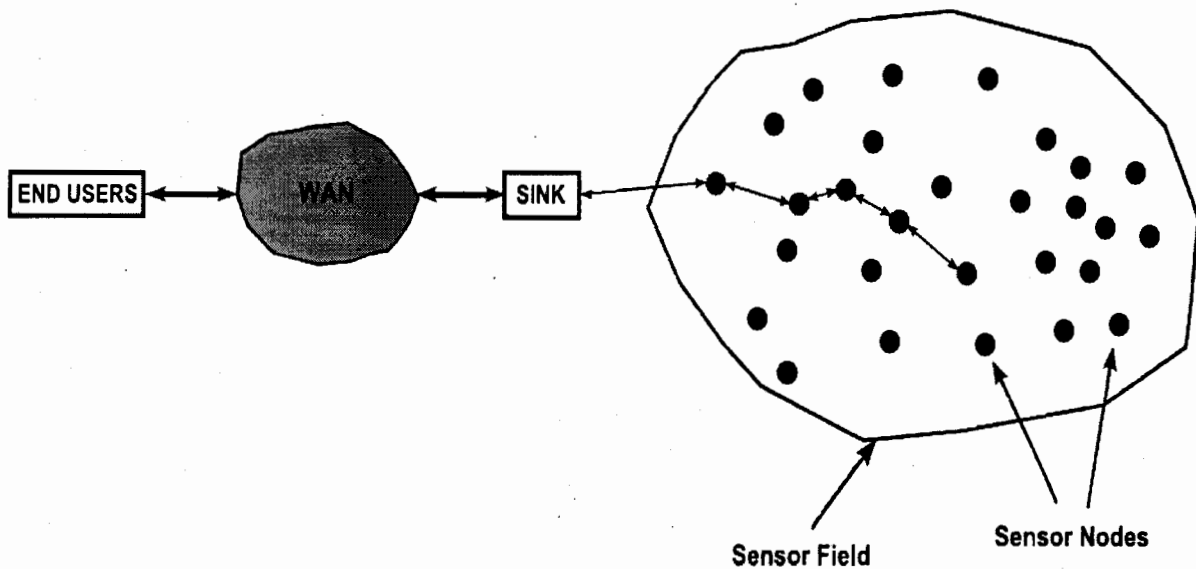


Figure 3 Wireless Sensor Network Architecture [7]

Sometimes the end-user who wants to get the benefit from these deployed sensor nodes is away from the network. An adversary can have access to the network and may want to get information while staying away from the network. In that case sink node uses the wide area network (internet) to communicate with the end user.

2.1.1 Anatomy of Sensor Hardware

Sensor nodes are tiny in size and they are basically composed of four basic components [8], which are given below.

1. Power unit
2. Sensing unit
3. Transceiver
4. Processing unit (microcontroller)

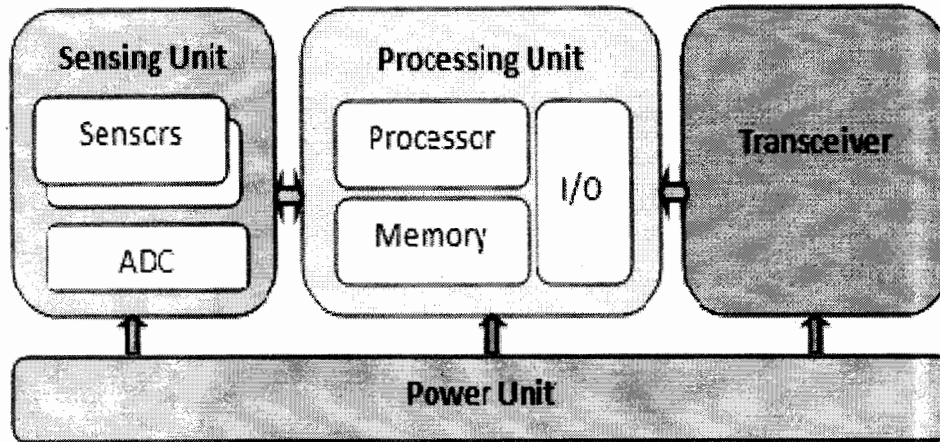


Figure 4 Anatomy of a Wireless Sensor node [8]

2.1.2 Components Sensor Node

We can observe all four basic components of sensor in above figure 4. Now, we will try to explain each component one by one briefly.

2.1.2.1 Sensing Unit

A sensing unit consists of two parts: one is composed of sensors and second is ADC (Analog-to-Digital converter). The first portion sensors consists of one or many sensors. The task of the sensors is to sense the measurable area. Sensors produce measurable information from the deployed area. Information produced by sensors is in the form of analog signals. Here the use of second part ADC starts, whose functionality is to convert the analog signal to digital. Now sensors sense information from the deployed area and transfer analog signal to ADC which then convert this analog signal into digital before transferring to the processing unit for processing. Different types of sensors exist like thermal, infrared, seismic, visual, and magnetic. Sensor nodes are in active and passive modes. Active modes continuously manipulate the environment (probing in the environment e.g. Radar system). In the passive mode they start working when the certain action takes place in the deployed area. They are sometimes omni-directional or unidirectional. A sensing portion has one or more sensors, so they all sense information which can be integrated to form a complete information.

2.1.2.2 Processing Unit (Microcontroller)

The processing unit is composed of three sections processor, memory and I/O (input/output devices). Processor gets digital data from sensing unit and start processing over the digital data. This microcontroller is programmable and use low power. A microcontroller integrates memory provisions, processing and I/O peripherals which reduces additional wiring, hardware, circuit board space and energy. I/O provides the flexibility to integrate different peripherals with the circuit board. One important portion of microcontroller is the memory and two types of memory

exist: one is for “program memory” and second one is for “user memory”. Program memory contains the programming for the sensor. You can say user memory as a buffer for storing data. It is not common for all sensors to include some external memory like flash memory.

External Memory

This memory can be attached with the sensor node to increase its storage capacity. It is attached with I/O. All sensors don't support external memory attachment, so it is not common in all sensors. External memory can be in the form of RAM (Random Access Memory) and flash memory. Flash memory has large data storage capacity and lower cost.

2.1.2.3 Transceiver

This component manages the communication process like connecting sensor nodes with each other, transmitting and receiving data. It uses RF (Radio Frequency) as a communication medium and it is according to 802.15.4 and Zigbee standard. Maximum distance of sensor node communication is 100 meter. It works between the frequency range of 433MHz and 2.4GHz. It has four operational modes. It can sleep, remain idle, transmit and receive. Wireless sensor communication operates over scientific, medical and industrial RF bands and these bands are allowed for unlicensed scenarios. These bands are called ISM bands.

2.1.2.4 Power Unit

Power unit is the backbone of the sensor node. All other units of sensor node cannot work even for one second without this power unit. If sensor node has large energy resource it means sensor node will live for long time. Long time means more sensing, more computation and more communication. The life of the sensor is directly proportional to the energy budget. Among all components of sensor transceiver takes more energy than all because it has to communicate with other sensor nodes for transmitting and receiving data. The process unit takes very less energy for its computation. Energy resource is a great issue for the sensor node. If we increase the size of energy resource then the size of the sensor node increases. While residing inside those limitations we use different techniques to save energy. If we can't increase energy resource, then we have to decrease its usage. This power unit is in many forms: some use single battery, coin battery or AA 1.5V battery cells. There is also the usage of the solar panels but they also require secondary rechargeable batteries for storing the energy prepared by solar cells. When solar panels are attached with the sensor node, its size becomes greater. Solar energy operated sensor nodes are used mostly for environmental monitoring purposes. Figure 5 shows the solar panel used with the sensor node for power generation. Different terminologies are used for power unit, like energy resource, energy budget etc.

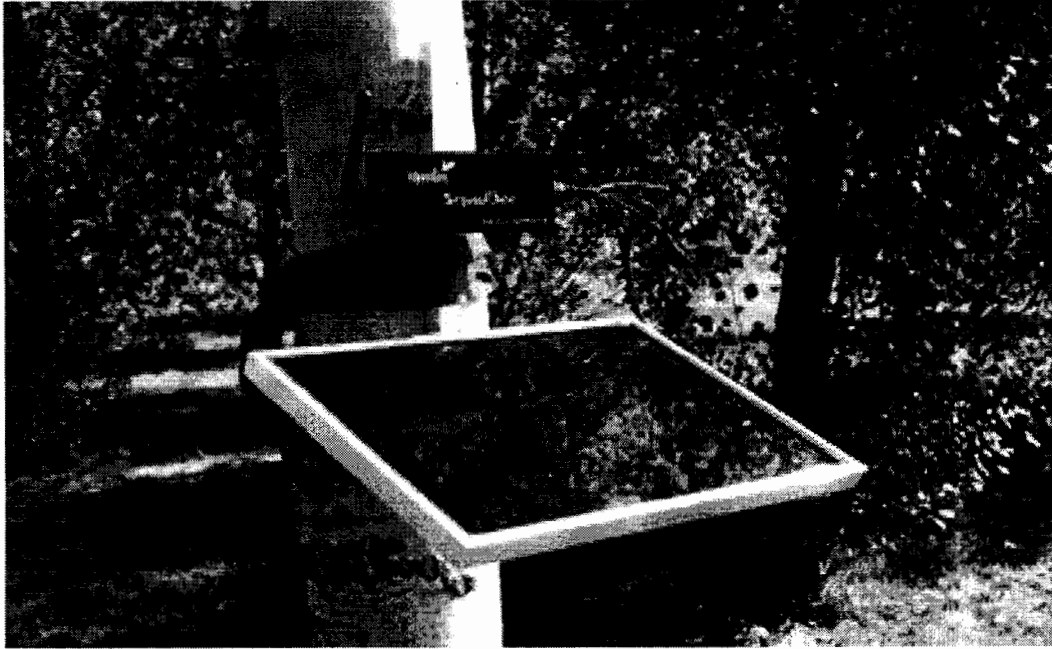


Figure 5 Sensor Node with Solar Panel [9]

2.2 Sensor Node Software

Sensor nodes are hardware devices and they use some software for working. [27] LabVIEW and nesC are used for programming sensor applications. NesC has almost similar syntax to the C language but in binding the syntax becomes different. TOSSIM, NS2, MATLAB and Cooja are the software to perform the simulations.

2.2.1 Operating Systems

There are many operating systems available to implement the applications for wireless sensor network. These operating systems are [27] Contiki, ERIKA Enterprise, Nano-RK, SOS, TinyOS.

2.3 Sensor Node Hardware

As it is a new technology, hardware is not easily available. Some universities have limited hardware for research purpose. The cost of hardware is much high so it is not affordable by individuals. Few types of hardware are given in the figure 6 with their specifications like CPU, power, memory etc. These are used for research in some universities. Due to their high cost all universities cannot afford this hardware.

There are two kinds of sensor nodes. One is the normal sensor node which observes the situation and takes the statistics. Second kind of sensor node is gateway which connects the sensor network with the outside world. The figure below contains both kinds.

Node	CPU	Power	Memory	I/O and Sensors	Radio	Remarks
Special-purpose Sensor Nodes						
Spec 2003	4-8MHz Custom 8-bit	3mW peak 3uW idle	3K RAM	I/O Pads on chip, ADC	50-100Kbps	Full custom silicon, traded RF range and accuracy for low-power operation.
Generic Sensor Nodes						
Rene 1999	ATMEL 8535	.036mW sleep 60mW active	512B RAM 8K Flash	Large expansion connector	10Kbps	Primary TinyOS development platform.
Mica-2 2001	ATMEGA 128	.036mW sleep 60mW active	4K RAM 128K Flash	Large expansion connector	76Kbps	Primary TinyOS development platform.
Telos 2004	Motorola HC508	.001mW sleep 32mW active	4K RAM	USB and Ethernet	250Kbps	Supports IEEE 802.15.4 standard. Allows higher- layer Zigbee standard. 1.8V operation
Mica-Z 2004	ATMEGA 128		4K RAM 128K Flash	Large expansion connector	250Kbps	Supports IEEE 802.15.4 standard. Allows higher- layer Zigbee standard.
High-bandwidth Sensor Nodes						
BT Node 2001	ATMEL Mega 128L 7.328MHz	50mW idle 285mW active	128KB Flash 4KB EEPROM 4KB SRAM	8-channel 10-bit A/D, 2 UARTS Expandable connectors	Bluetooth	Easy connectivity with cell phones. Supports TinyOS. Multihop using multiple radios/nodes.
Imote 1.0 2003	ARM 7TDMI 12- 48MHz	1mW idle 120mW active	64KB SRAM 512KB Flash	UART, USB, GPIO, I ² C, SPI	Bluetooth 1.1	Multihop using scatterness, easy connections to PDAs, phones, TinyOS 1.0, 1.1.
Gateway Nodes						
Stargate 2003	Intel PXA255		64KNSRAM	2 PCMCIA/CF, com ports, Ethernet, USB	Serial connection to sensor network	Flexible I/O and small form factor power management.
Inrync Cerfcube 2003	Intel PXA255		32KB Flash 64KB SRAM	Single CF card, general-purpose I/O		Small form factor, robust industrial support, Linux and Windows CE support.
PC104 nodes	X86 processor		32KB Flash 64KB SRAM	PCI Bus		Embedded Linux or Windows support.

Figure 6 Varied Types of Sensor Hardware with Specifications [26]

Chapter 3
Literature Survey

3. Literature Survey

In this chapter we will discuss the literature which we have studied to gain knowledge of the field. We have studied important and updated knowledge of the field. As it is a new topic for research. Over the past few years a lot of research work has been done in the field of wireless sensor networks and it continues to make more significant breakthrough.

3.1 Different topologies of WSN

Wireless sensor network mostly fall in three categories.

One Hop Model

Sensor nodes directly communicate with the base station. In this model communication is greater because each node is communicating directly with the base station. In this model the energy consumption is more so the network life-time is less.

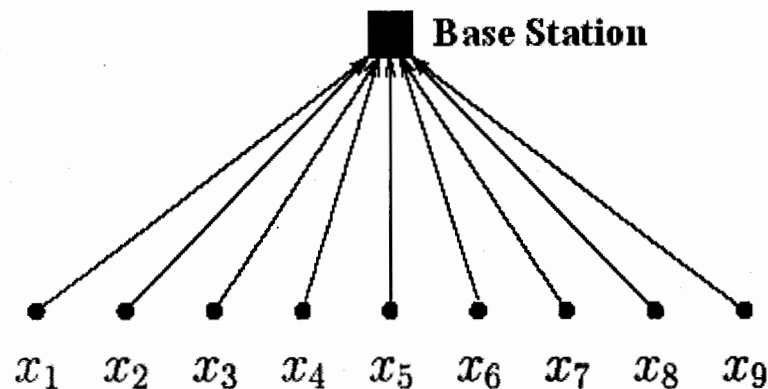


Figure 7 One - Hop Model [11]

Multi Hop Model

In multi hop model sensors communicate with their neighbors and send data to them which further pass on data, which reaches the base station. In this model the use of energy becomes less because each node communicates for lesser duration.

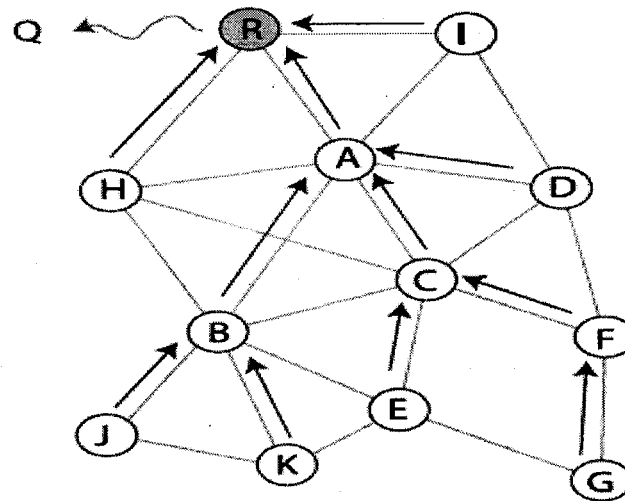


Figure 8 Multi - Hop Model [14]

Cluster Based Model

According to this model, there is a cluster head among a group of nodes. All sensors in that group communicate with this cluster head which further sends data to the base station. It is also less energy consuming.

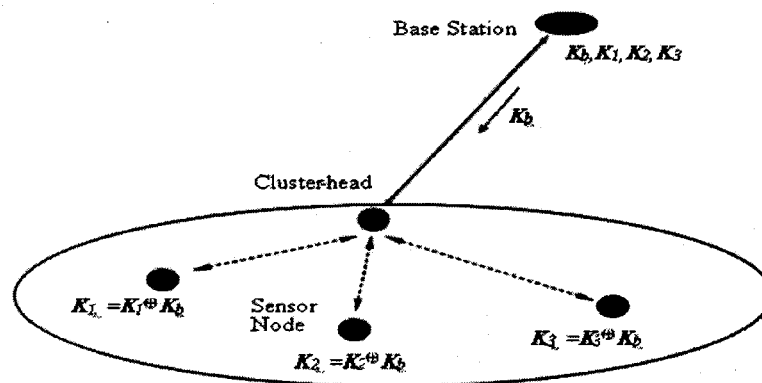


Figure 9 Cluster Based Model [25]

3.2 Aggregator Node

The sensor node called aggregator, collects readings from sensors and has the ability to perform some mathematical operations on the data and then sends to its parent or to the base station. Sometimes it is just like a simple sensor node which aggregates data and also senses like other sensors and integrates its own data in the aggregate. Aggregator node is used to reduce the communication and save energy.

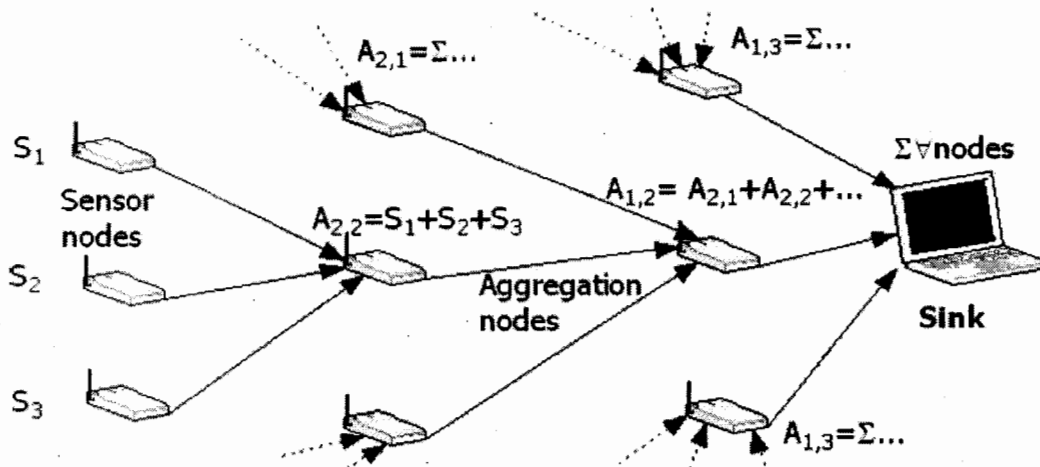


Figure 10 Data Aggregation [24]

3.3 Previous Work

Wireless sensor network have many research areas but we will focus on the security issues of the sensed data and learn how we can protect it from the adversary. For this purpose, we will go through the previous techniques used to secure data. Like other domains of wireless sensor network, the security issue is also very important. Researchers have proposed different type of encryption algorithms, authentication techniques, information consolidating methods and varied forms of security schemes to make the information secure. Securely transfer of information from sensor node to sink is also important for of wireless sensor network.

Lingxuan Hu and David Evas [10] introduce a protocol which provides security during the aggregation of the data and have resistance against the intruders and compromisers. It provides such a mechanism which significantly captures, those nodes which behave badly. This scheme suggests delayed aggregation and delayed authentication. Many messages are broadcasted by the base station towards the leaf nodes. The μ TESLA protocol is used to authenticate those broadcasted messages. A symmetric key is assigned to every sensor at the deployment time which is only shared between the node and the base station. Sensor sends messages to its parent. These messages also include MACs. MACs are calculated using node's temporary key.

$$B \rightarrow E \quad R_B | ID_B | \text{MAC}(R_{Bi}, R_B)$$

Similarly A sends messages to E. Parent node E receive these messages and MACs then store them. Parent node cannot verify the MAC since it uses the temporary key of the leaf. This temporary key is provided to parent later in the verification phase. The parent node waits until the time has elapsed. Now parent node send message to its parent. The message includes the sensor readings, MACs and the computed MAC of the calculated aggregate value.

$$E \rightarrow G \quad R_A | ID_A | \text{MAC}(K_{Ai}, R_A) \mid R_B | ID_B | \text{MAC}(K_{Bi}, R_B) \mid \text{MAC}(K_{Ei}, \text{Aggr}(R_A, R_B))$$

It is not necessary to transmit the computed aggregate; both R_A and R_B are transmitted to G which computes $\text{Aggr}(R_A, R_B)$. Now data reaches to the base station. It can verify it by calculating the MAC of the aggregation then comparing it with the MAC transmitted in the message. This scheme focuses on the integrity of the data but not confidentiality. If leaf node is compromised then there is no way to prevent this. It only provides the solution for integrity but not for confidentiality and data availability. Figure 11 describes the scheme briefly.

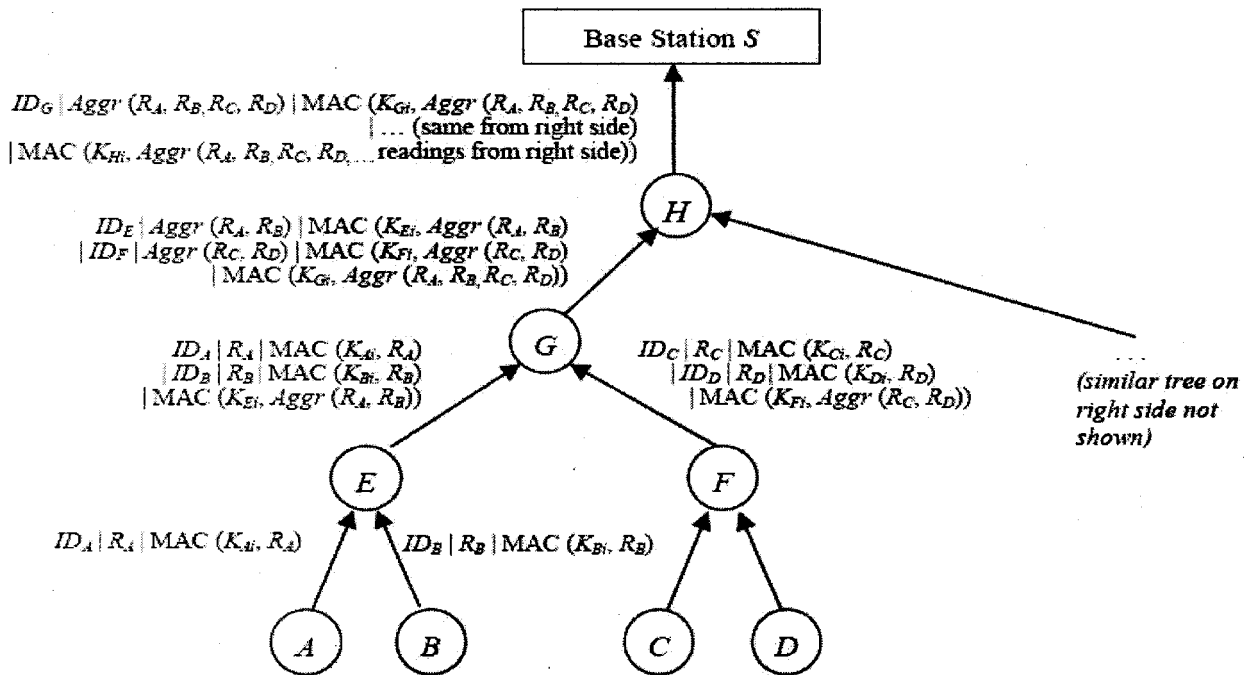


Figure 11 Secure Aggregation process of WSN [10]

Joao Girao et al. [12] introduce the scheme for saving battery power of sensors; the approach was that sense data needs to be consolidated and aggregated on their way to the final destination. Here data aggregation is performed and scheme provides end-to-end data security, even aggregator nodes are not able to read the data. PH (Privacy Homomorphism) is used to encrypt the data so aggregator nodes cannot open it but can perform many mathematical operations over the encrypted data. Public key is placed in the sensor at the deployment time which is known to all the sensors $1 \dots n$. Private Key is only known to the base station which can decrypt the encrypted message.

Initially all sensors are in the sleep mode. When sink initiates a process which requires a subset of sensors to complete the particular task, only that subset of sensor becomes active and senses that task. Sense data is split into secret shares and then encrypted by using, PH (Privacy Homomorphism) encryption transformation scheme and public key. Now the encrypted data is transmitted towards the aggregator node, which receives data from its sensors and performs allowed mathematical operations over the encrypted data. It calculates the aggregate of the all

received encrypted data and transmits the calculated aggregate towards the base station. It receives transmitted aggregate, decrypt it and drive original data from it.

It reduces the computation cost at the aggregator nodes by performing addition and shifting the division operations to the more powerful sink. With such a policy, aggregator node life time increases to some extent. In this scheme the data confidentiality and integrity are maintained due to the use of PH.

Pawan Jadia and Anish Mathuria [12] presented a protocol which gives surety of data confidentiality and data integrity. Data is encrypted and is not available for reading on aggregator nodes. There is no need to store the data at aggregator nodes for authentication which results in reduction of delays. This protocol is not appropriate for some queries like Min, Max. It uses one hop and two hop pair wise keys. One hop key is between node and its parent and two hop key is between node and grandparent. Similarly two MACs are calculated with these keys, first with one hop and second with two hop.

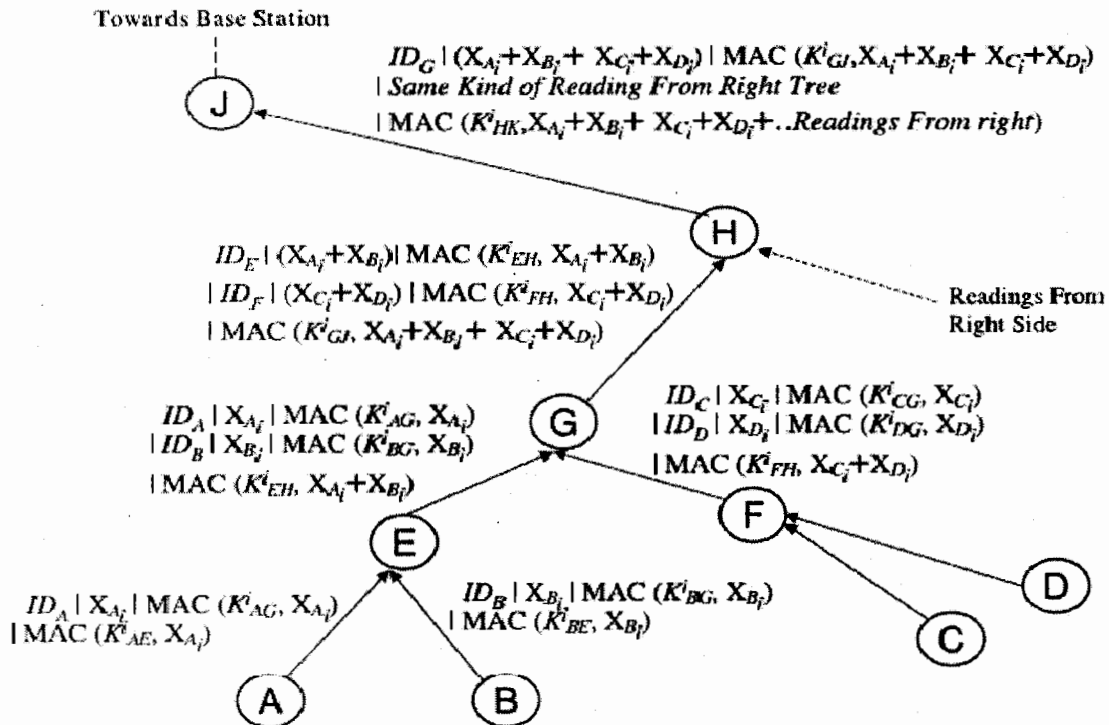


Figure 12 Brief Protocol Description [12]

Query is injected by the base station towards the leaf nodes. On getting query sensor increments its counter value by one. Base station will not enter next query until, it got reaction of all the nodes. This process is for synchronization of the network. Leaf node encrypts reading attaches its ID and two MACs and then transmits the message towards the parent. Parent computes the aggregate of received messages and also calculates MAC with two hop pair-wise key.

Intermediate node transmits towards its parent and so on, to the base station. The base station receives the final encrypted aggregated value. It can check the authenticity of the received messages. It will subtract the encryption keys shared with the leaf nodes to get the aggregated data in the plaintext. It provides confidentiality also integrity and reduces the communication required between sensor and sink. In above figure 12 protocol's brief description is given. Only attacks involving intermediate nodes are considered. Protocol is weak against collusion resistance attack.

Claude Castelluccia et al. [13] proposed a provable and secure additively homomorphic stream cipher, which allows encrypted data aggregation without decrypting it on aggregator nodes. In this statistical values can be computed like mean, variance and standard deviation on the encrypted data. Basically this scheme replaces the exclusive OR with modular addition. In this data privacy is achieved and data is protected from eavesdropping attacks. This is a mixture of cheap encryption technique with simple aggregation method to prepare an efficient mechanism of aggregation of data. With data all nodes send their ID to sink, in this way sink comes to know the non-responding nodes. If network is unreliable, it will create an overhead.

Haowen Chan et al. [14] introduce an algorithm which is secure and provable for hierarchal aggregation. It guarantees to capture any manipulation by adversary. It is based on calculating SUM aggregation and forces the adversary to commit for choosing it an intermediate aggregation result. Then it independently checks which nodes are contributed in the aggregate and their contribution is incorporated. It is assumed that all sensor nodes have ID, hash function, symmetric key decryption and encryption. This consists of three phases query dissemination, aggregation commit and result checking.

Wenbo He et al. [15] introduce two protocols which focus additive data aggregation. One is CPDA (Cluster-Based Private data Aggregation) which has tree parts, formation of cluster, within the cluster formation of results and data aggregation. Second scheme is SMART (Slice-Mix Aggregate). It also consists of three step slicing, mixing and aggregation. In this random key mechanism is used. The distribution of the key has three parts key pre-distribution, shared-key discovery and path-key establishment. The goal of our research work is to bridge the gap between collaborative data collection by wireless sensor networks and data privacy. Figure 13a shows the slicing and in figure 13b mixing is shown.

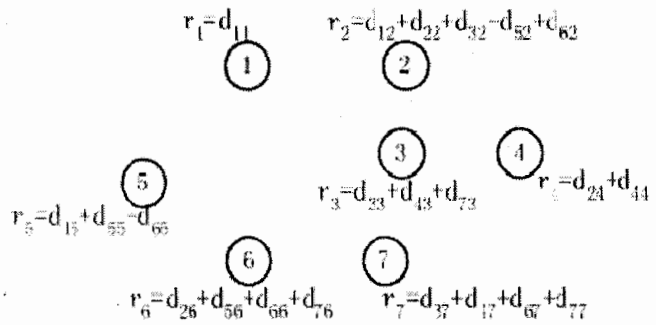
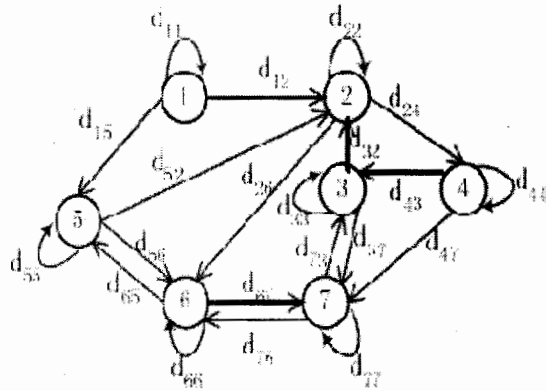


Figure 13a Slicing of Data Reading [15]

Figure 13b Mixing of Data Reading [15]

Roberto Di Pietro et al. [16] introduce a mechanism which provides confidentiality and integrity of the aggregated data. It detects data tampering and highly resilient to node failures. Mainly it is based on delayed aggregation and peer monitoring. Independent nodes close to aggregator are peer monitors. Node encrypts data using homomorphic encryption then this data is aggregated and verified with peer monitors. If it is correct then transferred to sink, else it raises the alarm for the base station about the misbehavior of aggregator. If peer monitors are compromised then there is no way to authenticate the data. The peer monitoring nodes are shown in the figure 14.

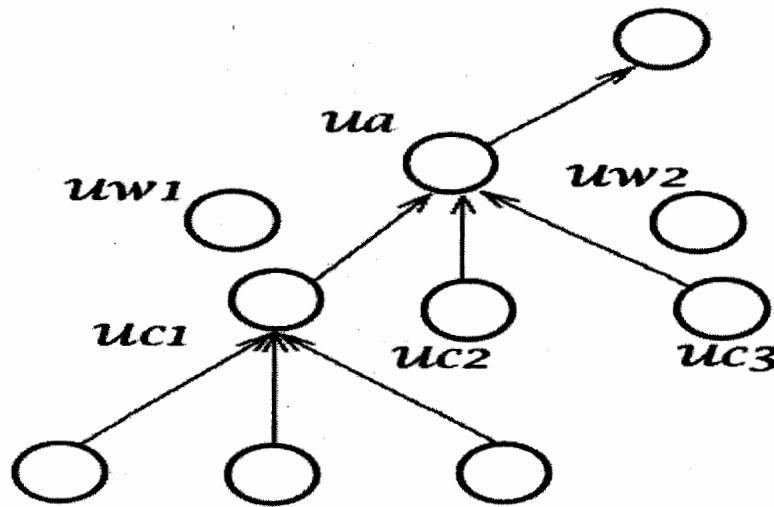


Figure 14 Peer Monitoring Nodes [16]

Claude Castelluccia and Claudio Soriente [17] introduce a protocol which provides integrity, privacy, robustness, less energy consumption, versatility and computation. It uses very cheap cryptographic tools, when protect from internal and external attacks. Base station sends query to network, each node has bucket and there are many slots in the bucket. Sensor data readings are placed in the slots, each slot has slot key then checksum is performed which is also known to sink. Bucket is finally sent to the parent node which adds the received buckets slot by slot. When base station receives these buckets, it subtracts slot keys and get the buckets data. It computes the functions when data is being traversed in the network and avoids sending every measurement to sink. In above figure 15, we have shown the ABBA scheme briefly.

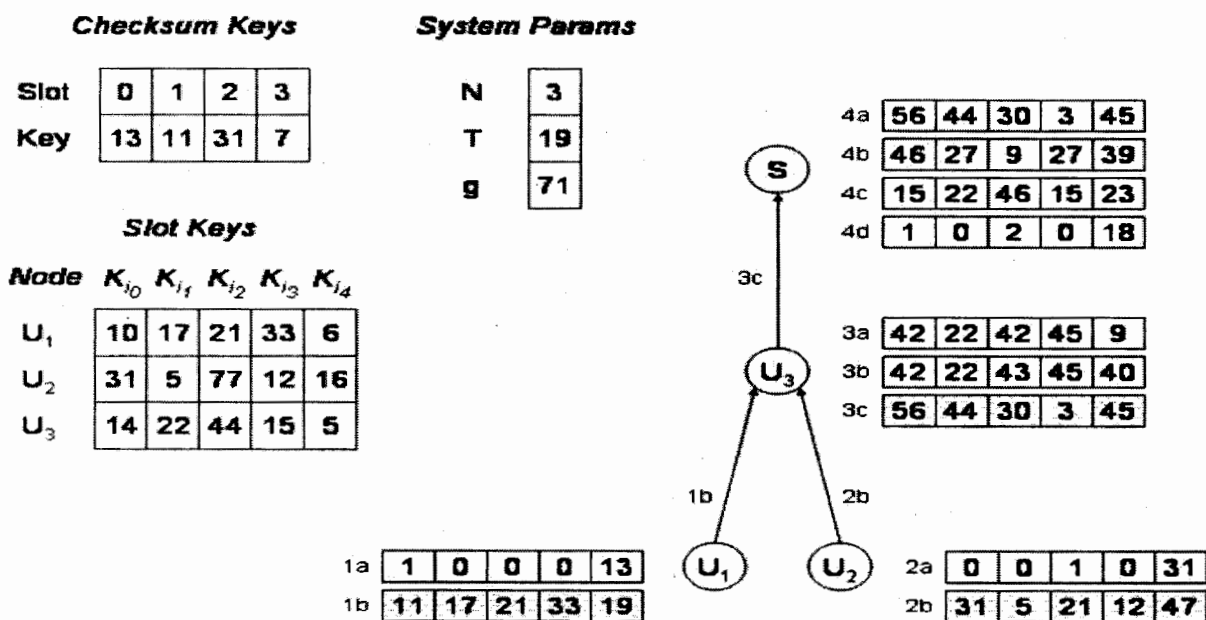


Figure 15 ABBA (A Balls and Bins Approach) [17]

Julia Albath and Sanjay Madria [18] introduce an algorithm which uses homomorphic encryption scheme and adaptive digital signature for achieving integrity, confidentiality and availability. Two digital signatures can't be aggregated because they are not homomorphic so it uses adaptive digital signature technique. The working of scheme is shown in the figure 16.

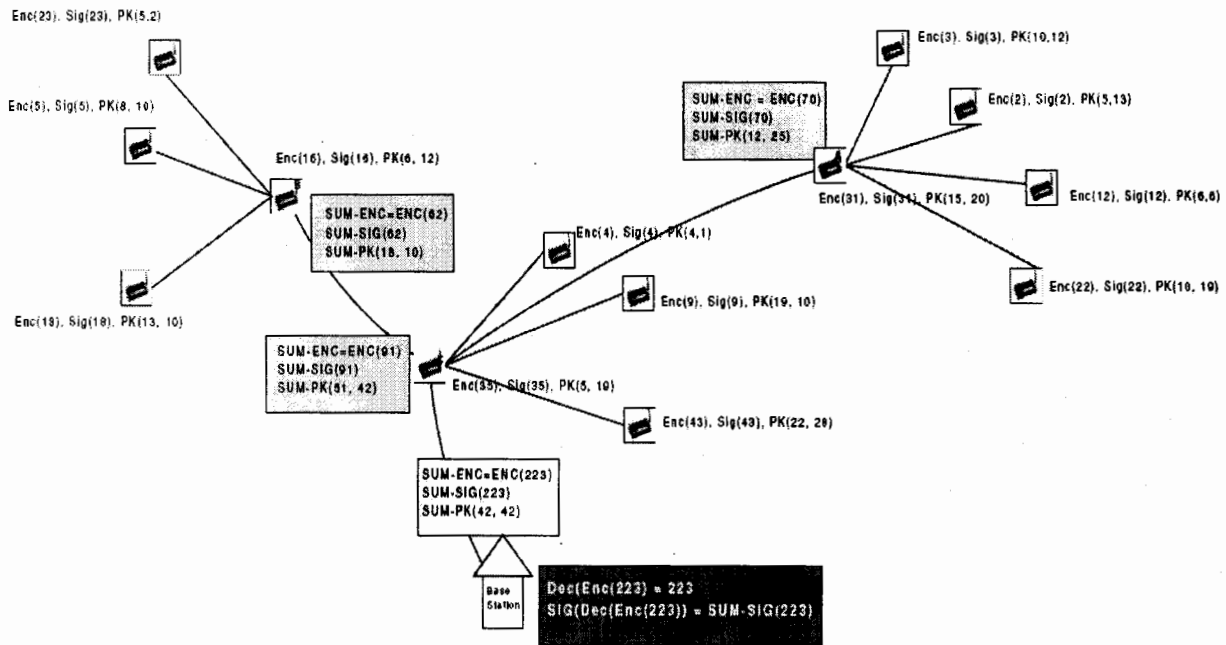


Figure 16 Homomorphic Encryption [18]

Public keys are used by the sensors for encryption and private key is used for decryption at base station. Each node has private key used for signing the reading. Sensor node generates reading then signed them with the aggregate signature protocol using the nodes private key. Each node homomorphically encrypt the reading with the base station public key. The node sends secured reading, the signature and its public key to its parent. After receiving messages from all its children, the parent combines the messages into one. The parent sums the secured readings, the signatures and the public keys. If the parent also contributes a reading, that reading is treated like any other reading. These are **SUM-ENC**, **SUM-SIG** and **SUM-KEY**. This process is repeated by each parent along the path to the base station. The base station decrypts the received message. The sum of the readings was homomorphically encrypted with the base stations public key. This allows the base station to decrypt the readings. Only the base station which is in possession of the matching private key is able to decrypt the readings. This is **Dec (Enc(x))**. Each node signed its messages, and these signatures were combined along the way. The base station can now verify the sum of the signatures given the sum of the public keys. The aggregate signature protocol ensures that only readings from legitimate sensors are included in the aggregate. It provides confidentiality and integrity by trading off energy consumption.

Suat Ozdemir and YangXiao [19] propose a protocol that allows the aggregation of those packets which are encrypted using different keys. It divides the network in different regions each region has different public keys. Encrypted data is collected in groups and then group's aggregate data is sent towards the base station, which decrypts and gets the data. In figure 17, the concealed data aggregation scheme is shown.

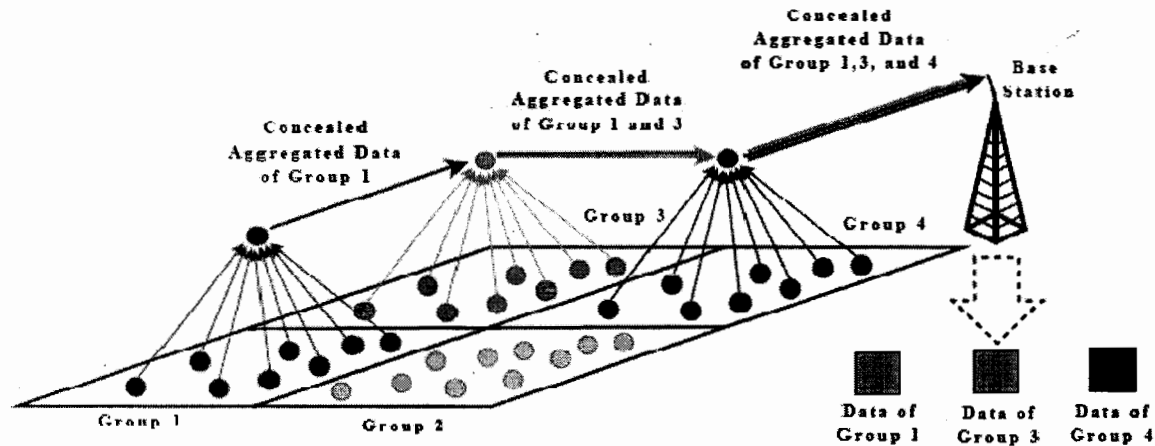


Figure 17 Concealed Data Aggregation in Groups [19]

Thomas Claveirole et al. [20] propose three schemes Secret Multipath Aggregation (SMA), Dispersed Multipath Aggregation (DMA) and Authenticated Dispersed Multipath Aggregation (ADMA). All have common principle, split readings into shares and send over multiple paths. When sink gets enough shares, then it can reconstruct the reading. The number of shares transmitted and received is not necessarily equal. This means that the system tolerates some losses. SMA uses “secret sharing scheme” whereas DMA and ADMA uses “Information Dispersal Algorithm”. The first technique guarantees data confidentiality where as other two data availability. How to obtain multiple paths is the limitation of this scheme.

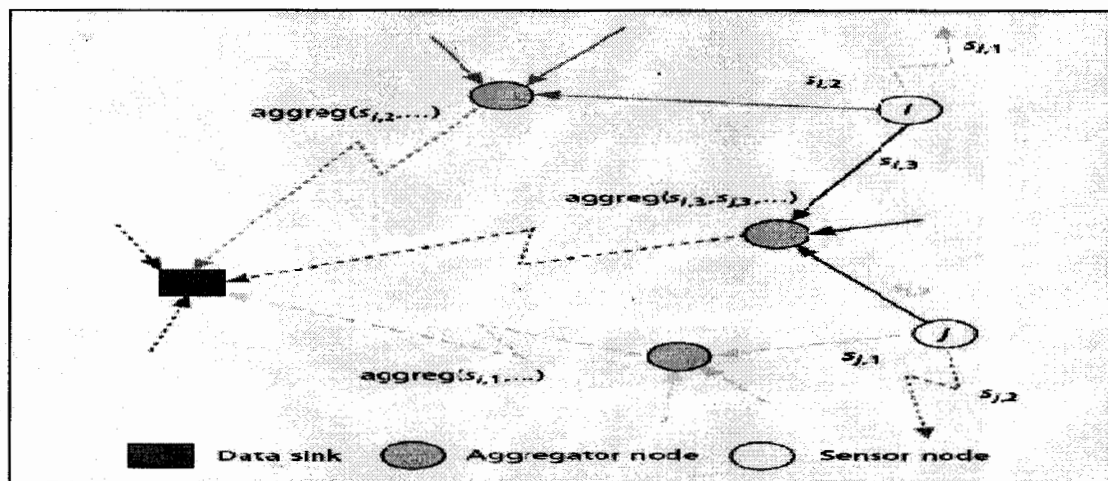


Figure 18 Shares Transmission on Multiple Paths [20]

3.4 Limitations in the Literature Survey

After doing a brief and interesting literature survey, it is found that different researcher proposed a lot of techniques to protect data from adversary form being compromised. Many techniques are also introduced for encryption, different key systems are presented but there are always some

loop-holes in techniques from where adversary can attack. If there is a loop-hole, it is risky to use the technique.

3.4.1 Limitations on Sensor Side

On the sensor side a lot of limitations are there from security point of view. Some researchers proposed techniques by assuming that sensors are secure which is far from reality. There are many algorithms which can provide us best security on sensor side but they are heavy and can't be implemented. If any sensor is compromised then all inner material is fully compromised; so all techniques become useless when sensor is compromised. There is also the need to make such a mechanism so that if a sensor node is under the attack of some adversary even then it works properly.

3.4.2 Limitations on Aggregator Node

When readings are received at aggregator node, it has to perform many mathematical operations over the data. If aggregator is compromised then we can't do anything similar like sensor node, because it will get all inner used techniques, in such a situation we can do only one thing. We send data from sensor node in such a way that it is not understandable by adversary in any case, even though aggregator node is compromised and adversary gets the data. With such a mechanism different types of attack will become useless. In case, we lose data, our information will not be leaked. Existing mechanisms use, heavy techniques which create overhead.

3.4.3 Limitations on Paths

Paths from where sensor node and aggregator node transmit data can be compromised and the transmitted data can be under many attacks, like eavesdropping, data tampering, replay, and denial of service. This is similar to scenario given above.

These limitations show that on sensor side there is a room to make the data in such way that it is not understandable by the adversary even the aggregator node or path is compromised.

3.5 Objectives

We have to develop such a technique which provides the solution to transferring information securely from sensor node to sink, even when the aggregator nodes are compromised. The developed technique provides the following advantages.

1. Provides efficient security
2. Performs less computation
3. Resists against the compromiser
4. Manipulates data and makes it non understandable.
5. Provides authenticity, data availability and protection

Chapter 4
Problem Analysis

Secret Share

A secret is divided into parts, giving each participant its own unique part, where some of the parts or all of them are needed in order to reconstruct the secret [29].

Divide data \mathbf{D} into n pieces in such a way that \mathbf{D} is easily reconstructable from any k pieces, but even complete knowledge of $k - 1$ pieces reveals absolutely no information about \mathbf{D} [30].

Information Dispersal algorithm (IDA)

Information dispersal algorithm is used for splitting data into multiple pieces such that with some (threshold) pieces data can be resembled. In general, the goal of information dispersal is to divide data into f pieces so that a subset of k of those pieces can be used to recover the data [31].

Information dispersal algorithm (IDA) is developed that breaks a file F of length $L = |F|$ into n pieces F_i , $1 \leq i \leq n$, each of length $|F_i| = L/m$, so that every m pieces suffice for reconstructing F . Dispersal and reconstruction are computationally efficient. The sum of the lengths $|F_i|$ is $(n/m) \cdot L$, since n/m can be chosen to be close to 1 [21].

5. Problem Introduction

In the following chapter, we will discuss the problem briefly. Basically, we work on the technique Authenticated Dispersed Multipath Aggregation (ADMA) used in paper [20]. According to this scheme sensor node senses data, places readings in the buffer when the buffer becomes full, it takes secret key " k_i " and the sequence number " s " of the message. A sequence number is generated for every message. Hash function is a one way encrypting function which creates fix size output. Now hash function is applied over the key and sequence number. Finally gets the " $h(k_i, s)$ " which is an authentication key and places it at the end of data buffer. After that information dispersal algorithm is applied over the data buffer. IDA creates several shares then disperses these shares from multiple paths towards sink. These shares are of unitary length, all shares contain information regarding the whole data (readings + authentication key). You need as many shares as required by the threshold to recover all the information. Recovery can be done at once when you have all the requested shares, if you have less shares than the threshold, you cannot recover anything. Sink will try different subset of shares to reconstruct the message. After reconstructing the message the sink has to validate that the last element is equal to $h(k_i, s)$. For that purpose the sink node computes its own authentication key and compare both keys, if both are same then validates it else it assumes that the message is altered.

4.1 Problem Statement

Sensor node creates " n " shares with the sensed data and disperses them on multiple paths. When sink node receives " t " shares out of " n " shares, it can reconstruct the reading where " t " is the threshold value for reconstruction and " n " is the total number of shares created by information dispersal algorithm. If some paths are compromised and adversary becomes successful in capturing " t " shares, it will also come into position to reconstruct the reading with these " t " shares. And if compromiser becomes successful in reconstruction, it will get the message information which can be utilized in variety of ways. With " $t-1$ " shares neither the sink node nor the adversary can perform reconstruction.

Possibilities

If above given problem occurs then:

- Adversary can tamper
- Can replay the message
- Denial of service attack
- Adversary can eavesdrop
- Sink node will lose the messages

4.2 Problem Scenarios

In the below figures, we explain critical attack scenarios and show the multiple compromised paths and on receiving " t " shares, how they reconstruct the reading.

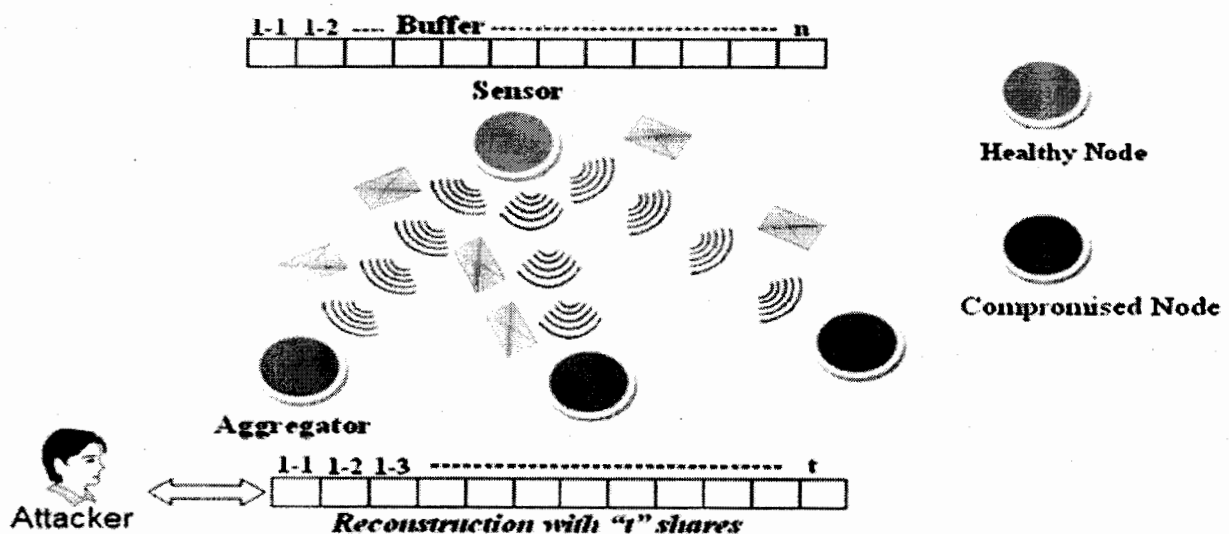


Figure 19 Multiple Compromised Paths

After sensing data, sensor node places readings in the buffer. Each time when the buffer is full, that becomes one message. Each message has a sequence number. In the figure above, the buffer is shown with the sequence number and index (like 1-1, 1-2 - in this, first part is the message number and second part is the index of buffer). When the buffer is full authentication key is computed with the help of sequence number and the secret key. Now information dispersal algorithm (IDA) creates the 'n' shares of data and disperses them on multiple paths which end at the sink node. Sink requires threshold "t" number of shares out of "n" for reconstruction.

If adversary becomes successful in getting "t" shares out of "n" it can reconstruct the message. In above figure 19 green color nodes are healthy and they are working properly. Red colored sensor nodes are compromised and adversary is getting there readings and trying to reconstruct the message with "t" shares.

In the below scenario, if adversary and sink node both fail to reconstruct then both can't get the message and with "t-1" shares it is not possible to reconstruct the message for both adversary and the sink node. Figure 20, shows failure of reconstruction for both adversary and sink, because both have less than "t" shares.

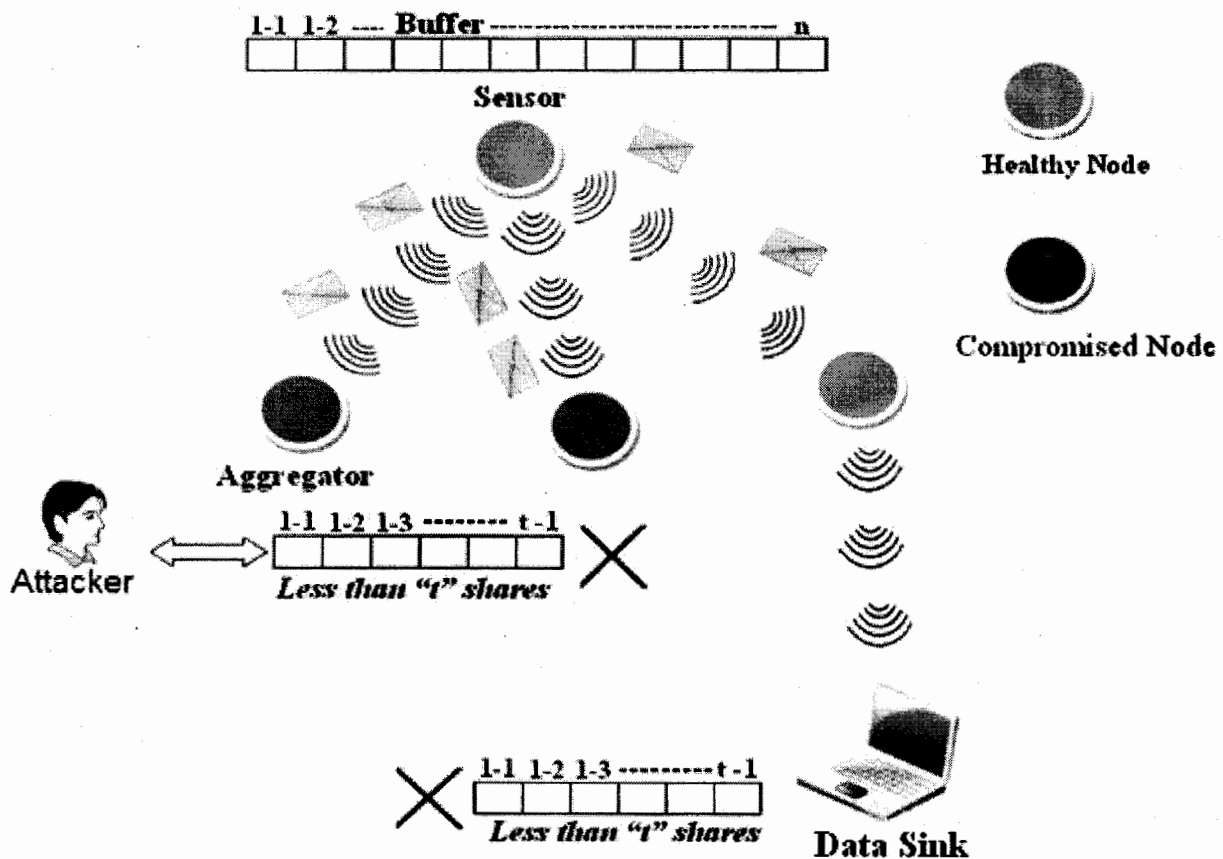


Figure 20 Data Reconstruct with "t-1" Shares

Now from taking overview about the problem scenario, we conclude that sink and adversary cannot reconstruct the reading with less than " t " shares. We can protect our data by using different schemes. These schemes should be of such type, if adversary gets the data reading in spite of that it can't reconstruct the reading.

If adversary reconstructs the reading then it will get all the information in the message and concurrently several messages are traveling through the paths. If it compromises some paths and start getting data readings it will get plenty of messages in the small interval of time. If the compromiser attacks for more time, it will get maximum information and that will be very harmful.

4.3 Focus of Research

If adversary captures messages continuously then the objective of nodes deployment will finish. Our focus is on the situation, in which sensor senses reading and sends towards the base station. During the transmission of data, if adversary becomes successful in capturing threshold number of shares out of ' n ' then it will get the information from the message. During these types of situations there should be some mechanism needed to protect the information from adversary.

Chapter 5

RSS (Re – Sequencing Scheme)

5. Proposed RSS (Re - Sequencing Scheme)

This chapter describes the design goals and the proposed scheme for securely transferring data, from sensor node to the sink. Proposed scheme is a solution of identified problem which is briefly given in the previous chapter.

5.1 Design Goals

We have designed this “**RSS (Re – Sequencing Scheme)**” with certain goals and requirements in mind.

Development of security scheme that:

1. Resists against adversary for more time
2. Provides strong authentication and integrity
3. Will be more efficient than the previous schemes
4. Increases data security with less energy overhead

5.2 Reference Architecture

We have considered [20] as the base idea and chosen the ADMA (Authenticated Dispersed Multipath Aggregation) scheme. According to it a sensor node senses data then authentication key is computed and placed at the end of the buffer. Now buffer is handed over to the IDA [21] which creates its several shares and disperses them on multiple paths. When sink node receives these shares, it reconstructs message with different subsets of shares and authentication key is validated. Finally, sink will get the original data. This scheme contains the problem which is described briefly above.

5.3 Network Assumptions

Like [20] we assume that multiple paths lead to sink node and have the ability of link level encryption. A sensor node breaks the data flow into many sub flows. This dissertation does not address the problem “how to get multiple paths” this is not the scope of this thesis. We also, suppose that sensor node have very limited storage, energy and computation abilities. This designed RSS works within these constraints.

5.4 RSS (Re – Sequencing Scheme)

RSS is a scheme which provides efficient solution to identified problem in the previous chapter. We use this scheme when we want to protect sensor data from adversary, while it is transmitted from sensor node to sink. RSS performs comparatively less computations over the sensed data and re-sequences it with a symmetric key which is only shared between sensor and the sink. As a result data will be protected from adversary during its transmission. We divide this RSS scheme in three phases and we will explain each phase, briefly with the help of figures and flow charts.

1. Initial Preparation
2. Sensor Node
3. Sink Node (On Receiving Shares)

5.4.1 Initial Preparation

Initially, we have to prepare a symmetric key for each sensor node and this symmetric key is randomly generated for each sensor node. We prepare symmetric keys for each node then create a map file with all of these. Each symmetric key is placed against each sensor ID. This map file contains all the sensor nodes ID which are participating in the network deployment and against each ID, its symmetric key is placed which is used when we get the data and arrange it with the help of this symmetric key. Map file is shown in the figure 21 given below.

<u>SINK SIDE</u>												
CONSTRUCTION OF MAP FILE WITH SYMMETRIC KEYS												
Sensor Nodes	Data Reconstruction Sequence Number											
1	7	3	6	0	11	10	9	5	4	1	8	2
2	11	9	8	7	0	6	5	3	10	4	2	1
3	5	8	10	3	7	2	1	11	4	6	9	0
4	3	5	0	1	11	7	4	9	6	10	8	2
5	5	7	8	11	6	3	10	1	4	2	9	0
6	10	3	11	0	5	7	2	9	4	1	8	6
7	4	7	10	5	8	11	1	3	6	2	9	0
8	9	11	5	2	0	10	8	6	1	7	3	4
9	11	0	3	6	8	2	10	1	4	7	5	9
10	7	5	1	8	6	11	9	2	10	3	0	4

Figure 21 Map File Containing Symmetric Keys

The sensor node places 12 readings in the buffer, this number can be increased or decreased as required. The size of the buffer is greater than the placed readings. In this dissertation, we say that the sensor node is placing 12 readings in the buffer. These readings start from 0. That's why the above given symmetric key numbers are 12 (0-11). Below figure 22 shows the flow of generation of symmetric key.

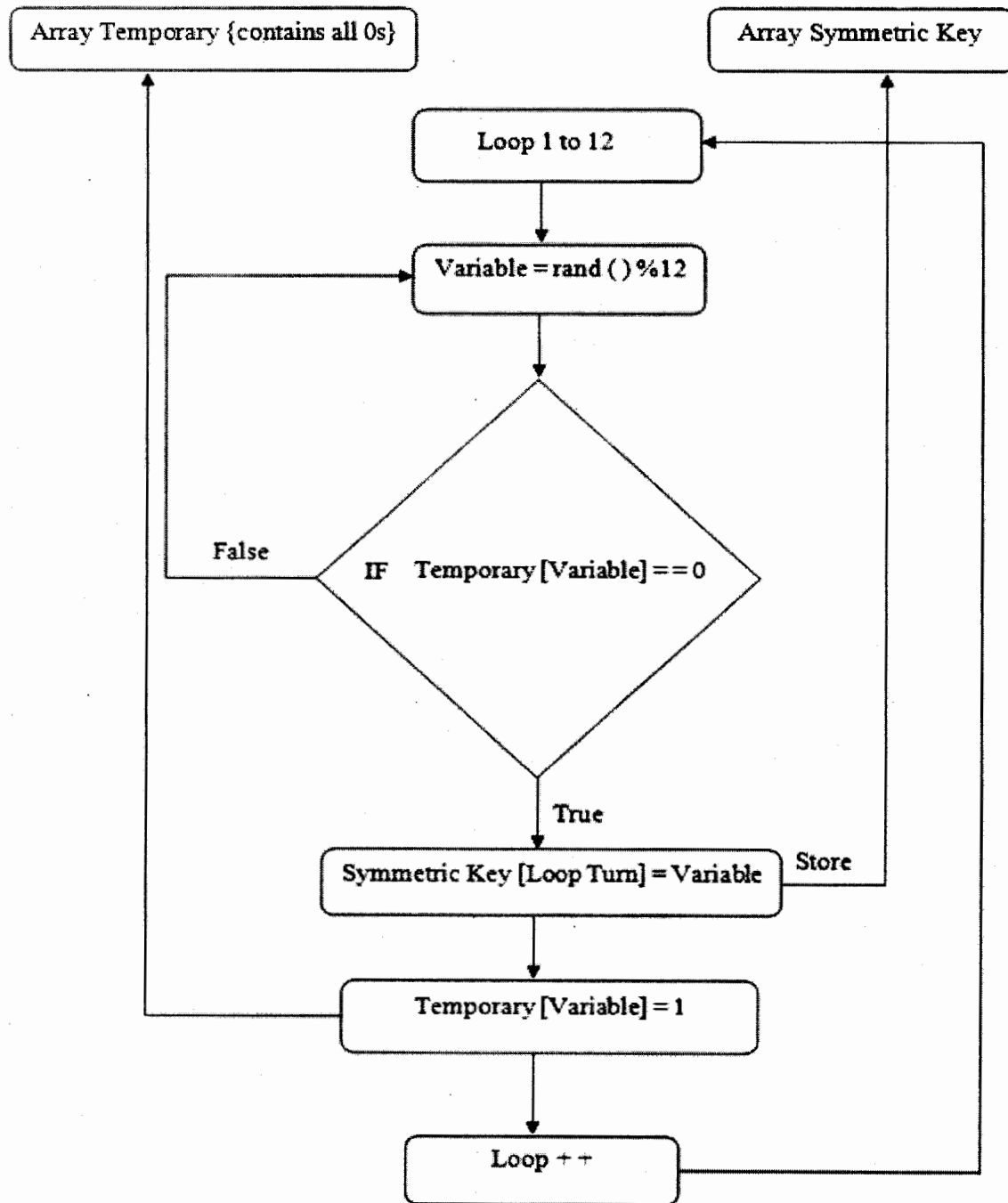


Figure 22 Flow Chart Symmetric key Generation for each Node

With the help of flow chart, it has been explained that the symmetric key generation for each sensor node. The property of this symmetric key is that the sequence numbers in it cannot duplicate because each number address to the index of buffer. The pseudocode for symmetric key generation is given below. By using this pseudocode, we can easily generate nesC code. We have to run this code for generation of symmetric key, for each node separately.

Sink Node (Initial Preparation)

Pseudo Code for Random Sequence Numbers Generation

START

INITIALIZE

Array Symmetric Key of Length 12 {Empty}

Array Temporary of Length 12 {Place all 0's in it}

// End of Initialization

Declare Variable Number

Loop (0 to 12)

Number = rand () % 12

IF (Temporary [Number] is equal to 0)

Symmetric Key [Loop Turn] = Number

Temporary [Number] = 1

Loop ++

End IF

Else repeat Loop

End Loop

END

5.4.2 Sensor Node

At the deployment time the symmetric key is injected in each sensor node. The symmetric key contains the randomly generated sequence numbers between 0 and 11 because sensor stores sensed data from 0 to 11 in the buffer. Injected key is shown in the figure 23.

<u>Sensor 1</u>	7	3	6	0	11	10	9	5	4	1	8	2
<u>Sensor 2</u>	11	9	8	7	0	6	5	3	10	4	2	1
<u>Sensor 3</u>	5	8	10	3	7	2	1	11	4	6	9	0

Figure 23 Injected Symmetric Key

When sensor node is deployed it starts sensing and storing data readings in the buffer, until the buffer becomes full then re-sequencing process will start.

Symmetric key is used at this place when the buffer becomes full. It re-sequences the sensed data buffer according to this symmetric key. With this process, the data positions are changed and there will be no original order of the data. Before it, data was in some sequence but after re-sequencing the positions of the data changes and there will be no sequence in the data. As an example, we show that sensed buffer contains data in numerical form in the figure 24. Re-sequencing with symmetric key is shown in the given below figure 25. In this process, we are changing the positions of the data but their sequence is also changing. When this process completes, then almost each part of data gets new location in the buffer that also has a sequence number. That is why we call this process as “**Re-sequencing**”.

<u>Sensed Buffer</u>																
<u>Index</u> →	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0	1	2	3	4	5	6	7	8	9	10	11				

Figure 24 Sensed Data Buffer

Now we will show briefly, all the re-sequencing process of sensed buffer. This re-sequencing process consists of three elements.

- Empty Buffer
- Sensed Buffer
- Symmetric Key

Deployed Symmetric Key

<u>Index</u> →	0	1	2	3	4	5	6	7	8	9	10	11
	7	3	6	0	11	10	9	5	4	1	8	2

Re - Sequence Buffer

<u>Index</u> →	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
----------------	---	---	---	---	---	---	---	---	---	---	----	----	----	----	----	----

Sensed Buffer

<u>Index</u> →	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0	1	2	3	4	5	6	7	8	9	10	11				



Figure 25 RSS (Re-Sequencing Scheme) working

The flow chart shows the process of the sensor side process. How sensed buffer is re-sequenced according to the symmetric key. Figure 26 shows the detail process of re-sequencing on sensor side according to our scheme. This is not the complete flow of sensor side, it shows only the portion which our scheme integrate with the existing process. Flow chart explains how sensor node uses the symmetric key and re-sequences the sensed buffer.

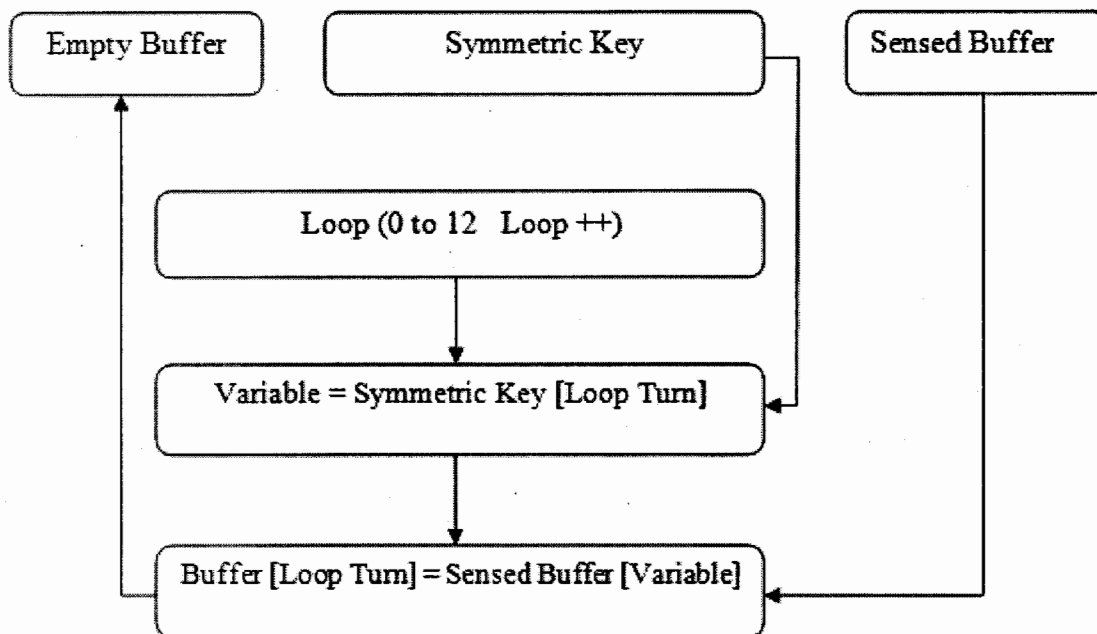


Figure 26 Flow Chart of Sensor Side process

We have prepared the pseudocode of the sensor side which is given below. The above flow chart and below given pseudocode show that very less computation is required to complete this re-sequencing process. There are some more processes which will be performed over the re-sequenced data, like SHA-1, IDA. That is why we have designed very light process which integrates with the existing process.

Sensor Node

Pseudo Code for Re-Sequencing Process

START

INITIALIZE

Symmetric Key of Length 12

Sensed Buffer of Length 16 {Contains Sensed Data}

Buffer of length 16 {Empty}

// End of Initialization

Declare Variable Number

Loop (0 to 12 Loop ++)

Number = Symmetric Key [Loop Turn]

Buffer [Loop Turn] = Sensed Buffer [Number]

End Loop

END

When the above process completes, the authentication key $h(k_i, s)$ is computed with the secret key and sequence number of the message. The secret key used to create authentication key is a separate and it is only used to create the authentication key. It is also shared between sensor node and sink and placed in node at deployment time. After creating authentication key, it hands over the buffer to the IDA (Information Dispersal Algorithm) [21] which create its several shares then disperses them on multiple paths. These multiple paths lead towards the sink node. All transmitted shares are not necessarily required for reconstruction. This shows that IDA is loss tolerant.

5.4.3 Sink Node (On Receiving Shares)

Sink node start receiving shares from multiple paths, and arrange them. It reconstructs the reading with different subset of shares. After reconstructing a $\sum R_i$ the sink node has to validate its last reading which should be equal to $\sum h(k_i, s)$. If this is equal to that then the message is validated else sink will try another subset of shares. To validate authentication key, sink node computes the authentication key by itself and compares both. If it found both are same then validates it and if both are not the same then there is cheating. When sink successful receives the data it applies the process of our proposed scheme over it. According to that sink re-sequences data by using map file, then it is in arranged form. This process is shown in the figure 27.

Deployed Symmetric Key

Index →	0	1	2	3	4	5	6	7	8	9	10	11
	7	3	6	0	11	10	9	5	4	1	8	2

Received Buffer

Index →	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
---------	---	---	---	---	---	---	---	---	---	---	----	----	----	----	----	----

Sensed Buffer

Index →	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0	1	2	3	4	5	6	7	8	9	10	11				

Figure 27 Re-sequencing of Received Buffer

When the sink node successfully receives data and finds that it is correct, then it uses the map file which is created initially. Message contains the ID of each sensor node that transmits it. With the sensor identification sink node searches the symmetric key against that sensor node from the map file and on getting that symmetric key it rearranges the received message. If adversary is very cleaver, it can send tampered message with complete shares and authentication key. Then our scheme can also verify it because its identification and key will not exist in the map file which means that this message is not accurate. This process is also very light and sink has to perform small computation. After the reconstruction of the shares, the re-sequencing process starts and we show this process with flow chart which is given below in figure 28.

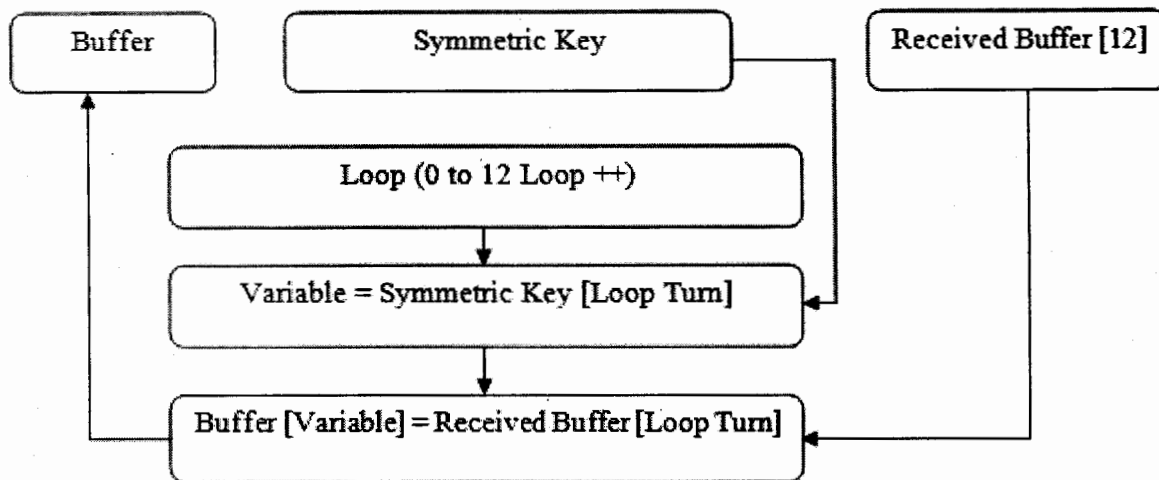


Figure 28 Flow Chart of Re-sequencing Process on Sink Side

The flow chart explains the re-sequencing process with the help of symmetric key stored in map file. We prepare the pseudocode of this process which helps to implement this process in the nesC programming code. Pseudocode of the process is given below.

Sink Node (On Receiving Shares)

Pseudo Code for Rearrangement

START

INITIALIZE

Symmetric Key of Length 12

Received Buffer {Contains Reconstructed Message}

Buffer of Length 12 {Empty}

// End of Initialization

Declare Variable Number

Loop (0 to 12 Loop ++)

Number = Symmetric Key [Loop Turn]

Buffer [Number] = Received Buffer [Loop Turn]

End Loop

END

The RSS (Re-Sequencing Scheme) ends here and in the above pages, we show the whole process of the RSS in detail, how RSS works from initial stage to the final stage. Each step is briefly defined.

5.5 Scheme Analysis

Now we compare and analyze, both schemes ADMA and our proposed RSS in different scenarios. These scenarios are given below

ADMA Scenario -1

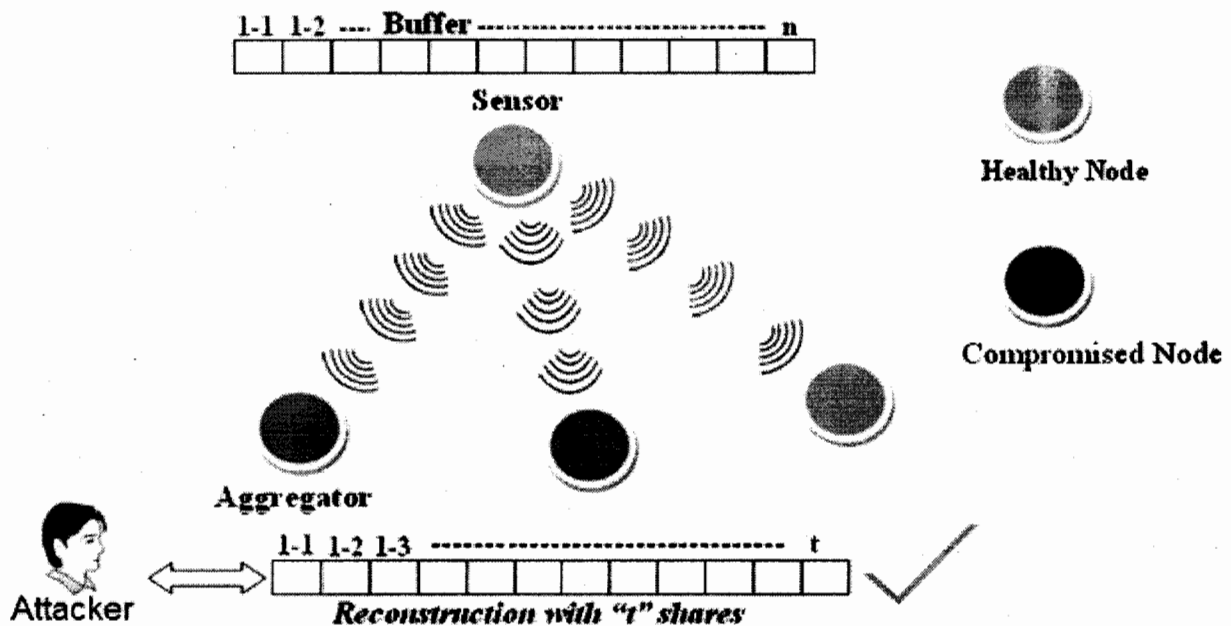


Figure 29 ADMA Scenario – 1

In the above scenario shown in figure 29, when some nodes are compromised in the network and attacker gain full control over them, if it becomes successful in getting “t” shares (threshold) so it will be in position to reconstruct the message. Now information transmitted by the sensor node is reconstructed by the adversary.

RSS Scenario – 1

RSS scenario is shown in the given below figure 30, if attacker captures “t” shares out of “n”, it can easily reconstruct the message. In spite of this it cannot get the data in the message until it re-sequences it. Attackers don’t have the symmetric key for re-sequencing because it is only shared between the sensor and the sink. To arrange data without symmetric it needs “s!” permutations.

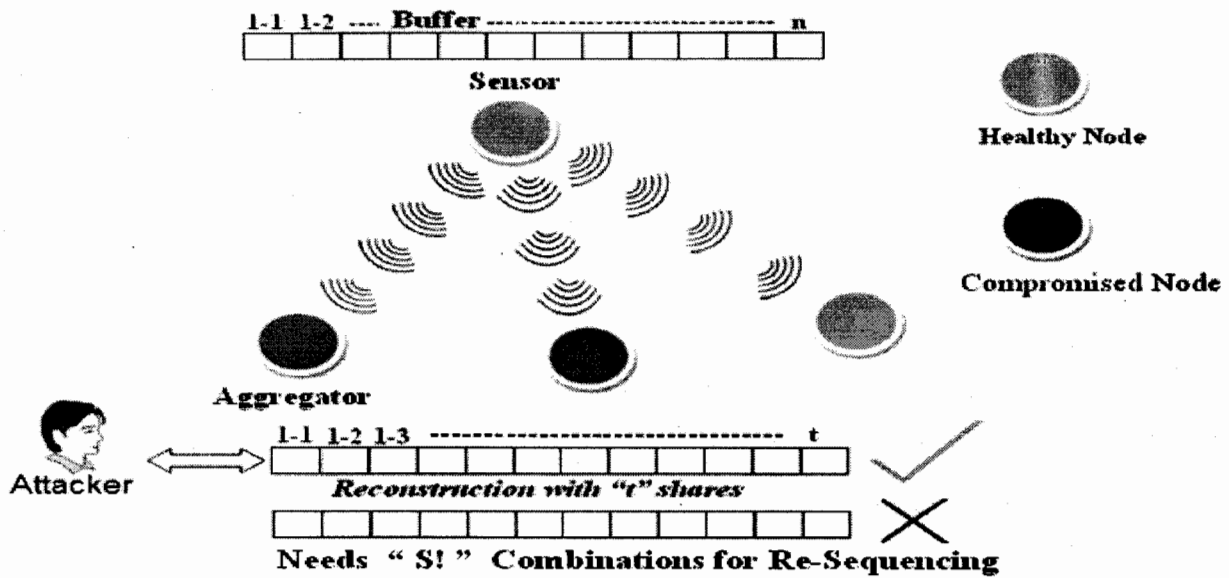


Figure 30 RSS Scenario – 1

ADMA Scenario – 2

ADMA scenario 2 is given in below figure 31, if attacker is successful in capturing “t” shares, it will reconstruct the message and get the information in the message. Similarly the sink will also get “t” shares and reconstruct the message. Both adversary and the sink will get the information.

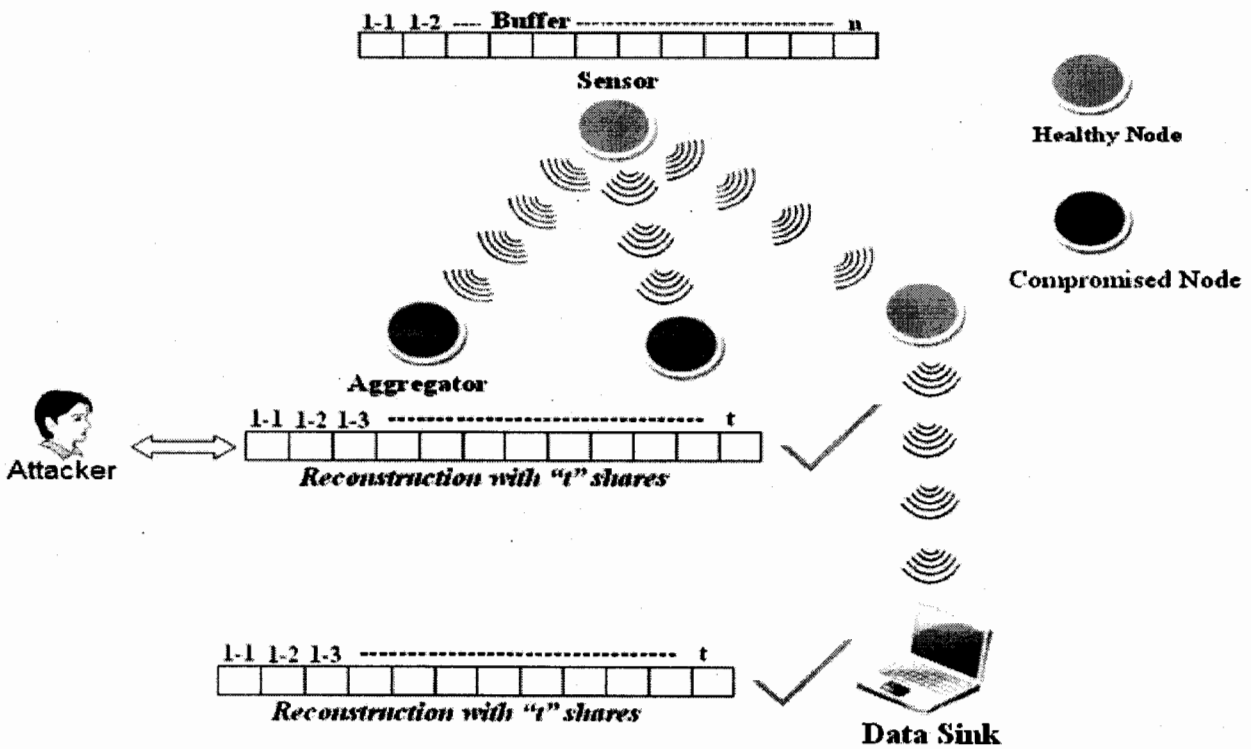


Figure 31 ADMA Scenario – 2

RSS Scenario – 2

We observe the scenario, in the below figure 32, if both attacker and the sink become successful in getting “t” shares after that both can reconstruct the message. Now the sink has symmetric key. It can easily re-sequence data in the message but on the other hand attacker cannot rearrange the data because it doesn't have symmetric key. To re-sequence data without symmetric key requires “s!” permutations.

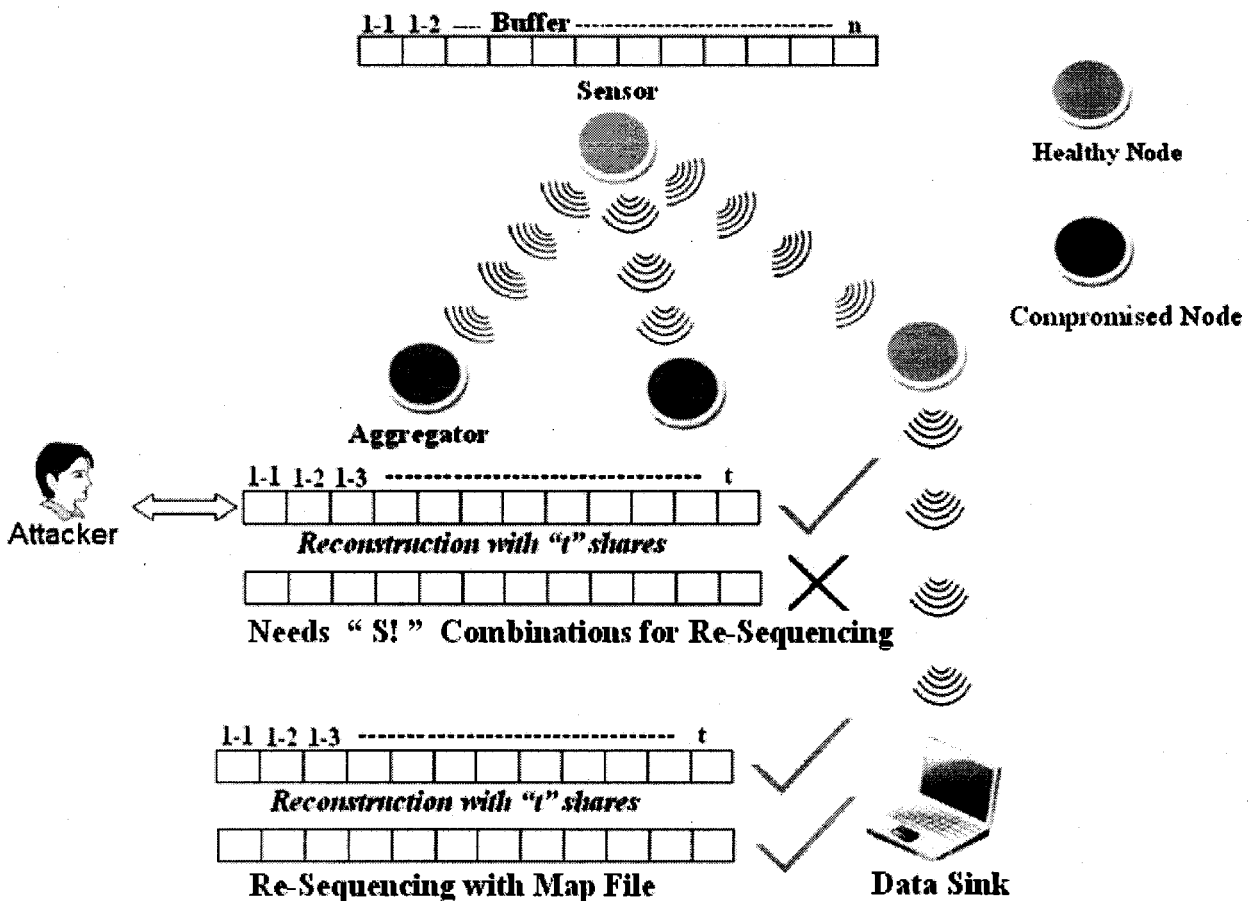


Figure 32 RSS Scenario – 2

ADMA Scenario – 3

Figure 33 shows the situation in which whole network is compromised and all nodes are in the control of the adversary. In such situation attacker can perform attacks over the nodes and data.

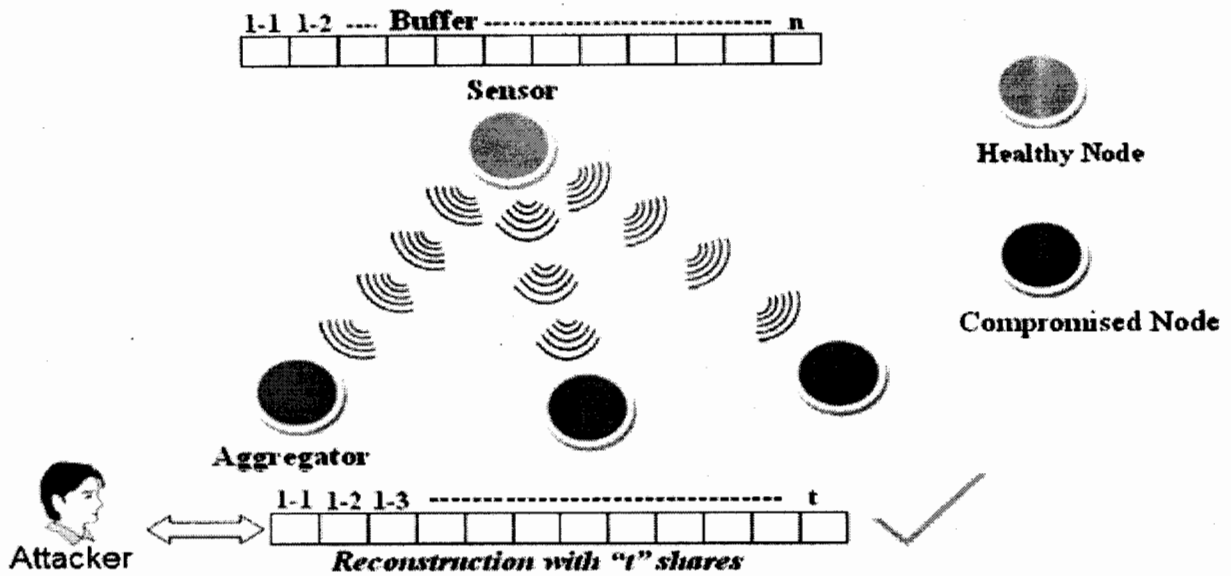


Figure 33 ADMA Scenario – 3

Attacker can get all shares and can reconstruct the message in such scenario. Now it is up to the adversary which type of attack it performs over the node. If attacker is only eavesdropping then the sink will get the messages. In case of tampering, the sink will get tampered messages and if it performs denial of service attack then sink will get no message.

RSS Scenario – 3

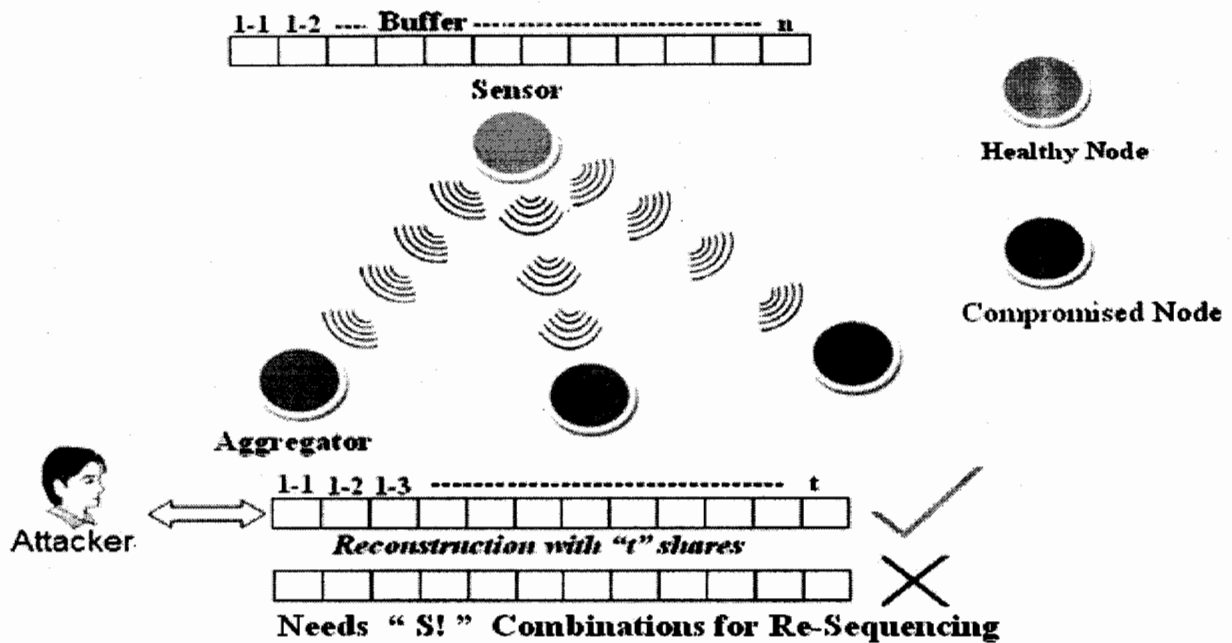


Figure 34 RSS Scenario – 3

In the above figure 34, RSS shows the scenario in which all nodes are compromised. If all nodes are compromised and attacker gets all shares and reconstructs the message. In spite of reconstruction it cannot get the information from the message because the data in the message is re-sequenced with symmetric key. The attacker needs “s!” permutations to arrange the tamper data.

The eavesdropping attack becomes difficult to perform and in case of tampering it cannot find the specific portion to which it specially wants to tamper.

Sensor stores data in buffer up to 12 readings and all these are re-sequenced with symmetric key so 12! (479001600) are possible permutations for data arrangement of single message. Sensor is sending number of messages in small time frame.

Now we will analyze our proposed RSS scheme for how much time it will resist against the adversary because there is no scheme which is 100% perfect and unbreakable. All security schemes are breakable and there quality lies in the time for which they resist against the attacker. So to rearrange the message, we are going to calculate, for how much time our proposed scheme, one message resists against the adversary. For analyzing this thing we suppose smaller unit of time to start. The calculation for testing is given below.

1 Buffer = 12 readings

$$\begin{aligned}
 T_{hours}(s) &= \frac{\text{Size of Buffer}}{\text{Milliseconds} * \text{Seconds} * \text{Minutes}} \\
 &= \frac{S!}{1000 * 60 * 60} \\
 &= \frac{12!}{1000 * 60 * 60} \\
 &= \frac{479001600}{3600000} \\
 T_{hours}(s) &= 133.056
 \end{aligned}$$

$$\begin{aligned} T_{Days}(s) &= \frac{Hours}{24} \\ &= \frac{133.056}{24} \\ T_{Days}(s) &= 5.544 \end{aligned}$$

5 Days 13 Hours and more than 30 minutes

This above analysis shows that 5 days 13 hours and more than 30 minutes are required to rearrange only one message, but in 10 minutes some sensors send more than 20 messages.

It is providing strong security to the sensor data. Next chapter will explain the simulations performed. It also discusses the tools which are used for simulations.

Chapter 6
Simulation Detail

6. TinyOS implementation

In this chapter, we are going to discuss the implementation details of scheme and different simulations performed using the technique. We use TinyOS as working environment for wireless sensor networks and in it, we use TOSSIM simulator, for sensors simulations.

6.1 TOSSIM Simulator

We briefly discuss the TOSSIM simulator in this portion. Simulator is used before using some hardware because the hardware is so costly that we can't buy it. May be the hardware which is required for our application is not accessible. Sometimes our application requires such hardware which is used by military only that type of hardware is not accessible to civilians. With these limitations, we use simulator. TinyOS is an environment, working for wireless sensor applications and it provides best simulator for sensors that is TOSSIM. As TinyOS is specially designed to work for sensor applications, the program and applications are directly burnt into the hardware, then if the hardware is not available, it also provides the facility to simulate the application in the TOSSIM. It helps us to debug applications very easily.

Single sensor node is much costly. It is not affordable if we have to buy several nodes for experiment that will not be possible for us. Many other simulators are available but we prefer to use TOSSIM because it has many special features. Application programmed to work on TOSSIM can easily be burned on the hardware and it is a big advantage. Some characteristics are given below.

Fidelity

TOSSIM emulates the hardware at component level. It provides us the simulation at bit level and it captures the network with high level of fidelity.

Time

The mica platform mote instruction frequency cycle is 4MHz. TOSSIM has mica platform built in which it uses for simulations.

Application Building

TOSSIM directly builds the application from the TinyOS source code and generate its simulation.

Models

TOSSIM lacks some models which are required for some applications, however these models can be attached externally and they work perfectly with it.

Radio Models

Two radio models are present in it. First is “Simple” and the second one is “LossyBuilder”. We can get the bit error rate and propagation delay with it. In simple model each transmitted bit is received properly and while in the LossyBuilder the probability of the corruption per bit is known.

Scalability

It can predict the behavior of large applications. We can simulate thousands of nodes with TOSSIM and in the real world single node is so much costly. It is not feasible to perform experiment on a large number of nodes.

Power TOSSIM

To calculate the energy it uses a specially designed component called power TOSSIM. This component will tell you the total energy used, number of total CPU cycles used and total usage of radio, ADC, led's etc.

Networks

It simulates the network at bit level. TOSSIM simulates 40Kbit RFM networking stack. It consists of encoding, MAC, timing and acknowledgements.

Documentation

By giving some commands it generates the brief descriptive documents and the graphical figures which completely show the working of the application. These documents contain brief information about the components, modules up to command level.

TOSSIM is a simulator which shows the perfect results, but in the real world this is not true. There is some preemption which involves in the real world applications. That is the difference in simulation and the real world application which work on the notes. These preemptions may be some interrupts etc.

6.2 How to Compile and Run Application in TOSSIM

TOSSIM has a compiler support; it directly compiles the application from the source code. Figure 35 shows the compilation of application.

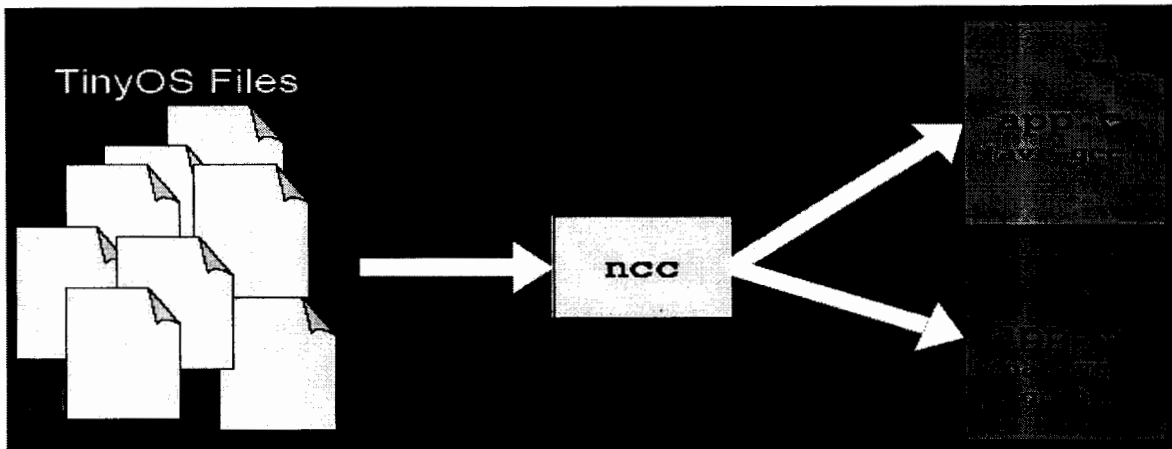


Figure 35 Compiling TinyOS Applications [22]

Basically a program has two portions the module and its configurations. With “make” command we can compile the program. Compiling for any platform we use command “make” and platform for which we are compiling like for mica we give command “make mica”. If we want to compile application for simulation we use “make pc”.

Before running application we enter the DBG modes which are used to debug at run time. These are am, clock, task, led, sensor, all, crypto, route, logger, radio, packet, encode, crc, power etc. Three user modes are reserved for output usr1, usr2, and usr3. Before running the application we enter the command and the debug mode “export DBG= usr1, led, radio, power”.

After compiling the application it creates the “main.exe” file in the directory “build/pc/” then we have to run the applications with the command “build/pc/ main.exe 100” here 100 is the number of nodes for which we are going to run the application.

6.3 TinyViz

TinyViz provides the GUI for debugging the application. It provides the visualization of application. It has many plug-in by using those we can check many things from the simulation. Before running our simulation, we have to enable or check our required plug-in then it will show its working. Plug-in includes debug messages, sent radio packets, radio model, radio links etc. It has a button on the top which is used to run and stop the simulation when needed. You can add delay in the simulation by using the sliding tab. Graphs can also be generated with TinyViz. To run the TinyViz we use the commands according to the given below steps.

Open two terminals windows and follow these steps in them.

1. First Terminal Window

(Directory: opt\tinyos-1.x\apps\desired application)

- a. make filename
- b. make pc

- c. `build/pc/main.exe -gui number of nodes`
2. Second Terminal Window
(Directory: `opt\tinyos-1.x\tools\java\net\tinyos\sim`)
 - a. `java net.tinyos.sim.TinyViz`

Then you will see the TinyViz on the screen of computer which looks like figure 36.

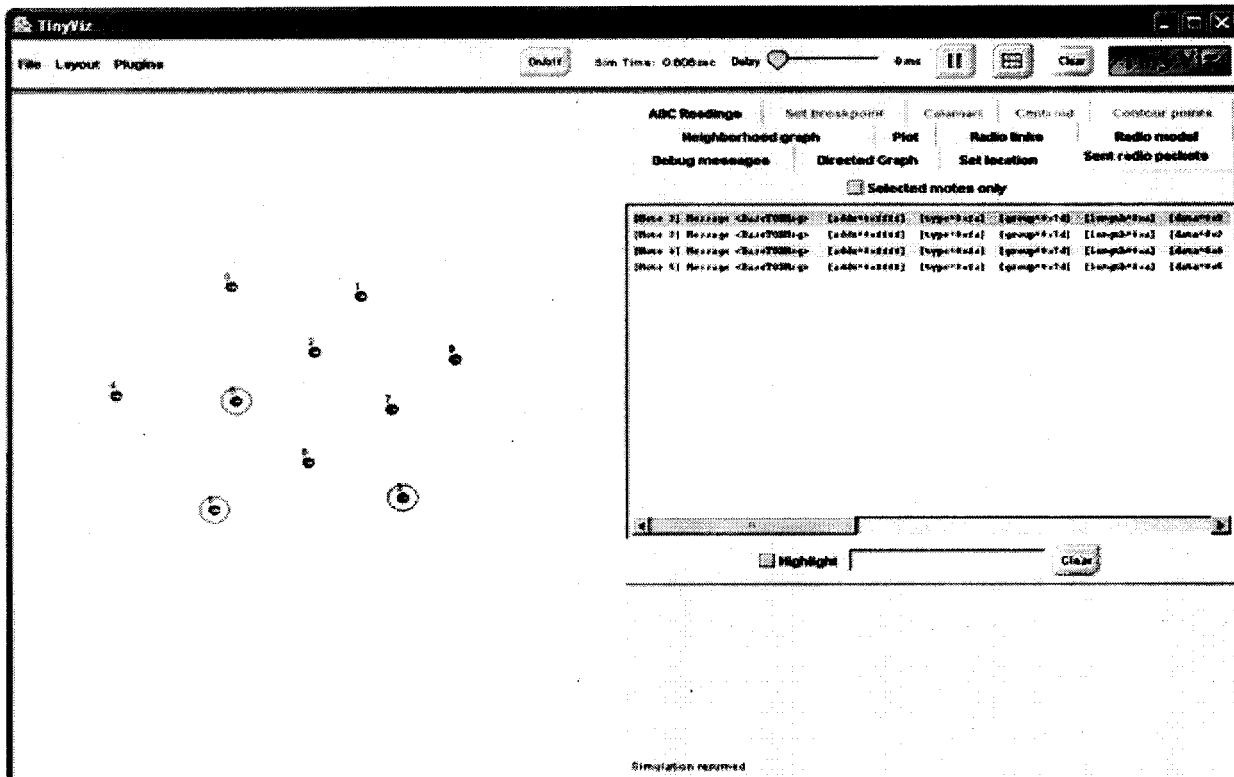


Figure 36 Screen Shot of TinyViz

6.4 NesC Language

NesC is a programming language used in TinyOS for application building, NesC stands for “Network embedded systems C”. NesC has syntax like C language. Applications consist of components and these components are made up of modules and configuration files. Modules contain interfaces and commands, whereas the configuration file contains the binding of the interfaces and the components with each other. Some interfaces are built-in the TinyOS while other required has been written by us, according to our application requirements. In the given below figure 37, we will show the formation of the component.

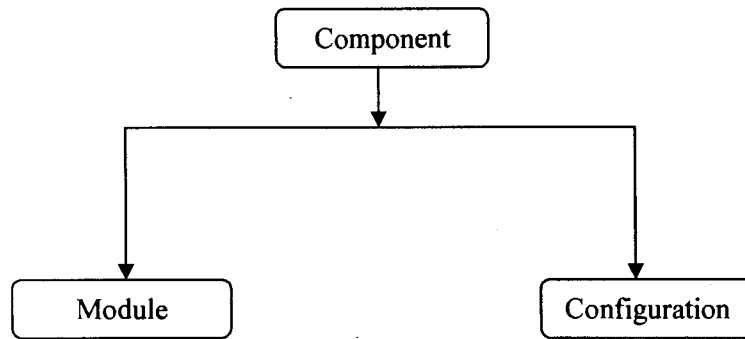


Figure 37 Formation of Component

TinyOS is configured in layered structure and compiler feel ease to handle layered structure. During the formation of each application all the layers collaborate. Figure 38 shows the layered structure.

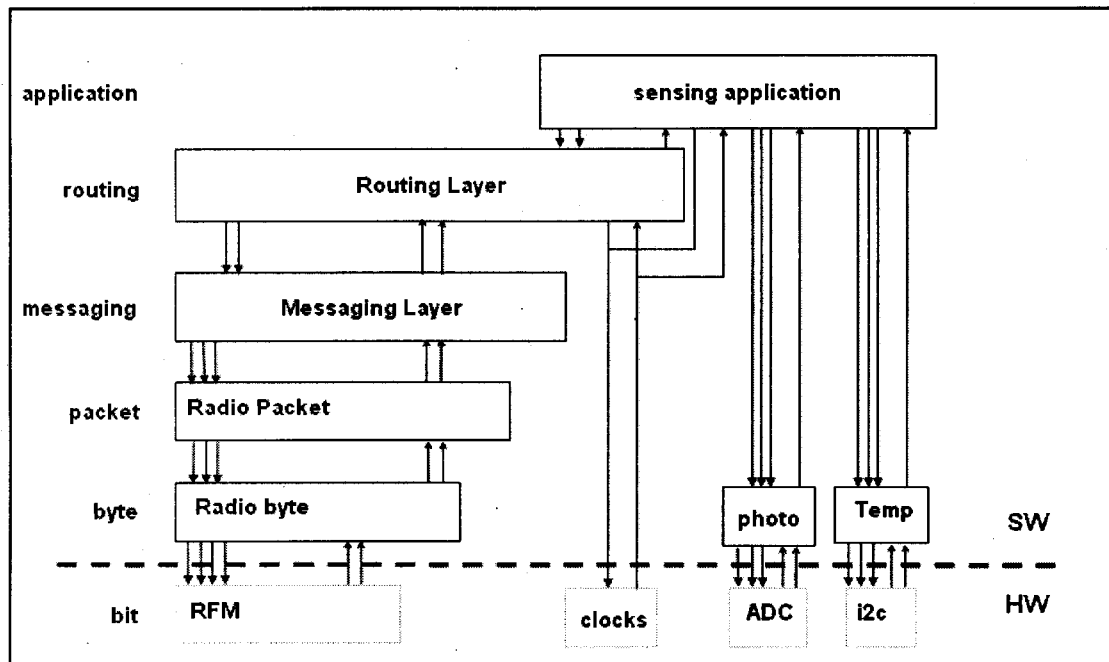


Figure 38 TinyOS Layered Structure [23]

6.5 Simulation Setup

In this section, we will explain some key things before discussing the simulations. These are; we use XUBUNTOS as a working platform. In this platform we have set its environment for TinyOS 1.x. For displaying simulation of the application, we use TinyViz form TinyOS. Above figure shows the TinyViz and its information. For the simulations of the energy consumption we use PowerTOSSIM. Figure 39 shows the network of 29 sensor nodes with one sink. From this figure you can get the idea about the simulation.

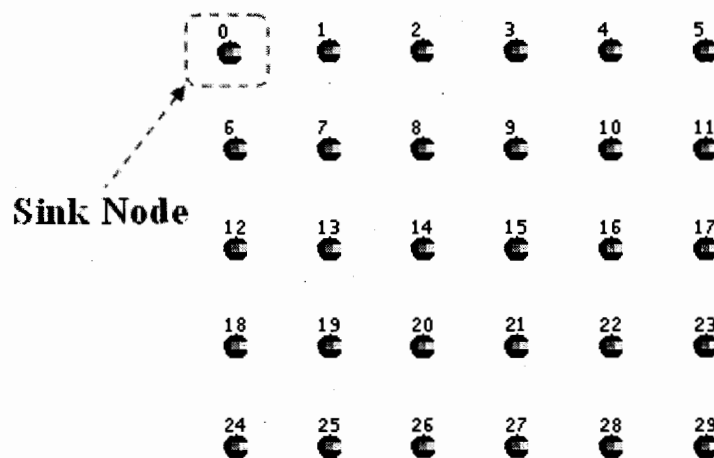


Figure 39 Screen Shot of Network Topology

In our application “0” node is always the sink node and others are sensor nodes. They sense data reading and transmit the data readings towards the sink node from multiple paths. These paths are multiple in number, so it is difficult to tell which one is the exactly path followed by sensor for next communication. Some nodes behaves as aggregator, they collect readings from other sensor nodes and transmit towards the sink node. We can observe them, in the simulation when some nodes send their data readings to other nodes and finally they send towards the sink node. Pink color arrows show the communication between the nodes and the sink.

6.6 Simulations

Some simulations have been done to test the scheme “RSS” with different perspectives. We use the “PowerTOSSIM” to check the total energy consumed by our proposed scheme. TinyViz is used to graphically view the communication of the sensor nodes with the sink. With the help of figures, we show the simulations in detail and in the next chapter, we will analyze those results generated by PowerTOSSIM and TinyViz.

6.6.1 Simulation with TinyViz

In this section, we will use TinyViz to check different things like communication between nodes to sink and number of packets sent individually. We have used some parameters which are given below in the table 2. This table shows that number of simulations done for it and how much time we run each simulation. It also shows for how many nodes we have taken for our simulations and some more details.

Table 1 Simulation Parameters for TinyViz

Parameters	Values
Topology	Grid
Number of Nodes	30
Sink Node (number)	0
Broadcast ID	65535
Number of Simulations	5
Each Simulation Time	(10 min) (15 min) (20 min) (30 min) (45 min)
Grid Placement	5x6
Unit of Distance	Feet
Deployed Area	1000 Sq. Feet

According to parameters given above, we did five simulations with both applications for above mentioned time periods. We use 30 nodes for simulations, of which 0 node is a sink node. As defined above the remaining are 29 nodes. They sense and communicate with each other; figure 40 below will show the simulation result, of the application.

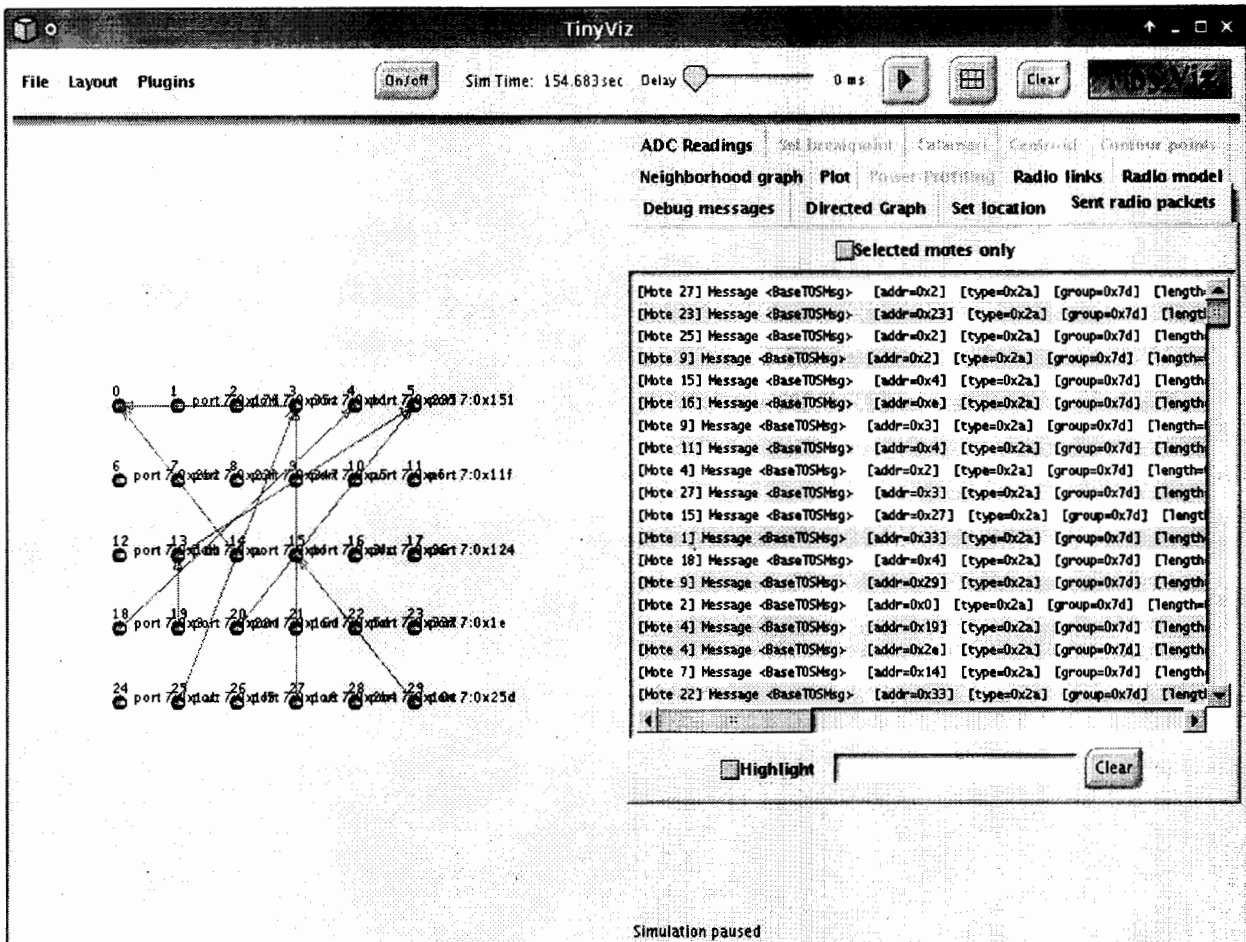


Figure 40 Screen Shot of RSS (Re – Sequencing Scheme) working

Pink color arrows show, the communication, 0 is the sink node. The arrows which connect other nodes, except the 0 node, are aggregator nodes. The nodes which are aggregating data also finally send data towards the sink node. As in the above figure, node 25 and 27 send data to node 3 which aggregate the data finally send towards the 0 sink node. Node 14 directly communicates with the sink. Node 19 sends data to node 13 which further sends towards the node 5, similarly node 20 also sends its data to node 5 which collects the data from all these nodes and sends towards the sink. TinyViz provides the facility of plug-ins, on right side of the simulation there are many plug-ins which you can enable to check your simulation. In the figure 40, it shows radio packets sent by each sensor node. Many other plug-in are there which you can use for your simulation.

6.6.2 Simulation with PowerTOSSIM

Another TinyOS tool used to simulate our “RSS” scheme is PowerTOSSIM. It will show the details of how much exactly the energy budget used, total CPU usage, total radio usage, total ADC, led’s, CPU cycles, total eeprom and total sensor. It takes some parameters which are given below in the table 2.

Table 2 Input Parameters for PowerTOSSIM

Parameters	Values
t	Number of Seconds the Application Run for Calculating the .trace file
p	Number of the Nodes for which this .trace file is Calculated

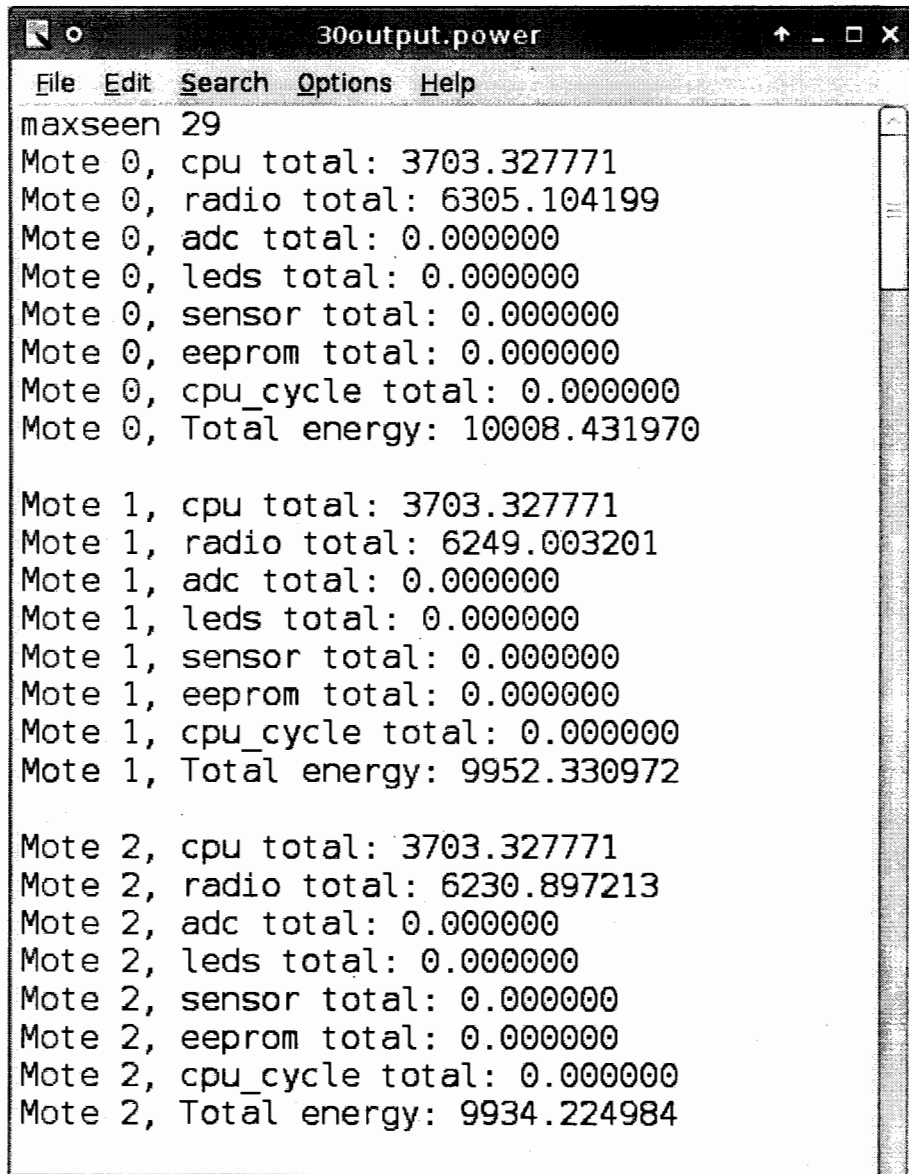
First of all, PowerTOSSIM creates “.trace” file which takes the above values as parameters. When this file is created, it takes file as input to “.power” file. Finally it shows the results.

We have done three simulations with PowerTOSSIM to calculate different results for analyzing both schemes from energy perspective. We have performed simulations by changing the values of input “t”. Different simulation parameters are given in the below table 3.

Table 3 Simulation Parameters for PowerTOSSIM

Simulation	Parameters
1	t = 300 p 30
2	t = 600 p 30
3	t = 900 p 30

PowerTOSSIM shows the details of an application for which it is used. Final report which it generates of an application contains all the statistics for each mote turn by turn. Figure 41 shows briefly the output file generated by the PowerTOSSIM. We will discuss in detail only the energy consumption comparison for both the schemes.



```

30output.power
File Edit Search Options Help
maxseen 29
Mote 0, cpu total: 3703.327771
Mote 0, radio total: 6305.104199
Mote 0, adc total: 0.000000
Mote 0, leds total: 0.000000
Mote 0, sensor total: 0.000000
Mote 0, eeprom total: 0.000000
Mote 0, cpu_cycle total: 0.000000
Mote 0, Total energy: 10008.431970

Mote 1, cpu total: 3703.327771
Mote 1, radio total: 6249.003201
Mote 1, adc total: 0.000000
Mote 1, leds total: 0.000000
Mote 1, sensor total: 0.000000
Mote 1, eeprom total: 0.000000
Mote 1, cpu_cycle total: 0.000000
Mote 1, Total energy: 9952.330972

Mote 2, cpu total: 3703.327771
Mote 2, radio total: 6230.897213
Mote 2, adc total: 0.000000
Mote 2, leds total: 0.000000
Mote 2, sensor total: 0.000000
Mote 2, eeprom total: 0.000000
Mote 2, cpu_cycle total: 0.000000
Mote 2, Total energy: 9934.224984

```

Figure 41 Screen Shot of PowerTOSSIM Output File

All the results for ADMA and proposed “RSS” are generated by TinyViz and PowerTOSSIM. The results are briefly discussed in the next chapter.

Chapter 7

Results

7. Results

In this chapter, we are going to discuss the results of the simulations which will show the significance of this scheme. When results are compiled, they show that the targeted results have been achieved about which we have stated in the previous chapters. On the basis of results, we can say that this scheme provides better security as we required.

Number of simulations has been carried out to compare our RSS scheme with A-DMA. These simulations are shown in the previous chapter. Two different types of tools are used for simulations. We have done five simulations with TinyViz and three with PowerTOSSIM for different sets of time frames. There are two types of comparisons between both the schemes; first is number of packets and second is the most important issue energy. One thing is that “**security always comes with some cost**”. Our RSS scheme is providing better security of the data than ADMA so it will create some overhead as the cost for providing security.

7.1 Performance Metrics

There are some performance metrics:

1. RSS energy overhead
2. Energy consumed individually
3. Total energy consumed by the network
4. Packets sent individually to sink
5. Total time required to rearrange one message

7.2 Simulation Results

We start with the number of packets sent by individual node then the comparison of RSS and the ADMA techniques. We have performed five simulations for different time intervals and in all of them there is a comparison between both the schemes proposed RSS and ADMA.

In this simulation we calculate the number of packets sent by individual node towards the sink for both schemes. We run the first simulation for 10 minutes then increase time frame for next simulations. After completing all simulations we calculate the results and found maximum number of nodes sent equal number of packets, as compared to the ADMA. Some nodes from both schemes show variations in packet sending. Below figure 42 shows that RSS performs very light computation which has less effect on packet sending.

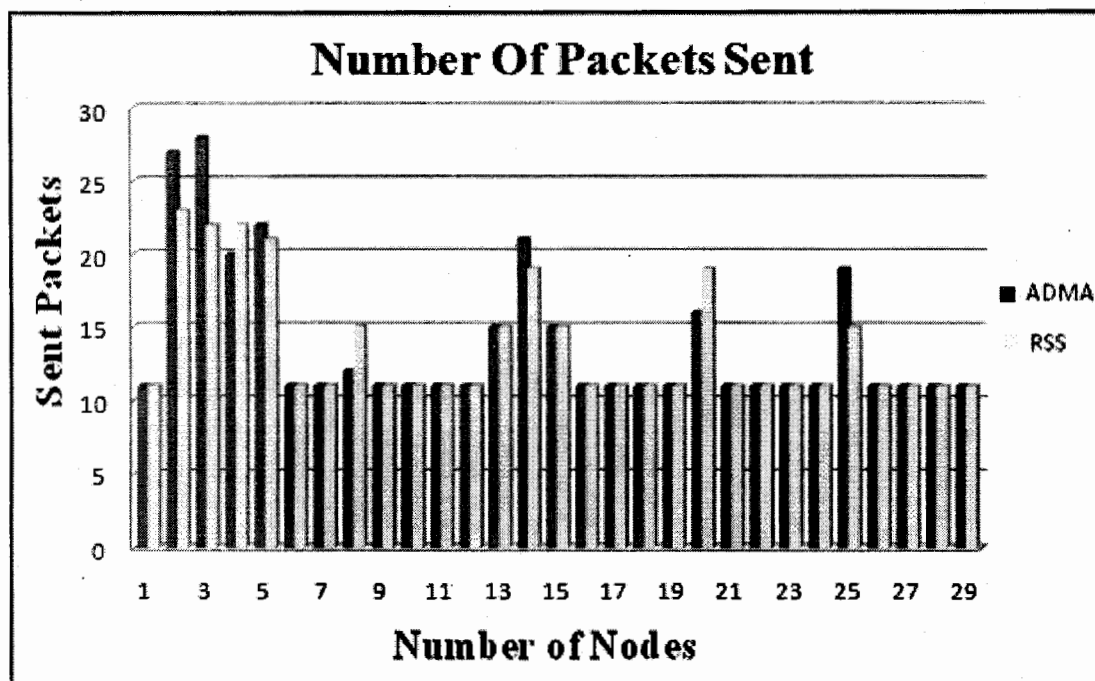


Figure 42 Number of Packets Sent by Individual Node

All next simulations for calculating the packet sending by individual node run according to the time frames which are given in the table 1. After completing all these simulations we found that our proposed scheme is working properly.

We know that energy resource is the life of sensor node. Sensor nodes are deployed for information about the interested area and if this information is not protected then all the energy resources consumption for getting this information becomes useless. We have made our data secure from adversary. As we know that “security always comes with some cost”, so some energy overhead is created in RSS for providing better security than ADMA. In order to calculate energy resource, used by both schemes, we use PowerTOSSIM, which shows the comparison of energy consumed by both schemes.

Three simulations have been performed with PowerTOSSIM for both the schemes RSS and ADMA. Simulation parameters are shown in the table 3 in previous chapter. According to these parameters, simulations have been performed to check the variation in the results.

After completing the PowerTOSSIM simulations, we calculate results from them. As we have stated before our RSS creates some energy overhead for providing better security so from the simulation the calculated results are given below. Individual node energy consumption is shown in the figure 43.

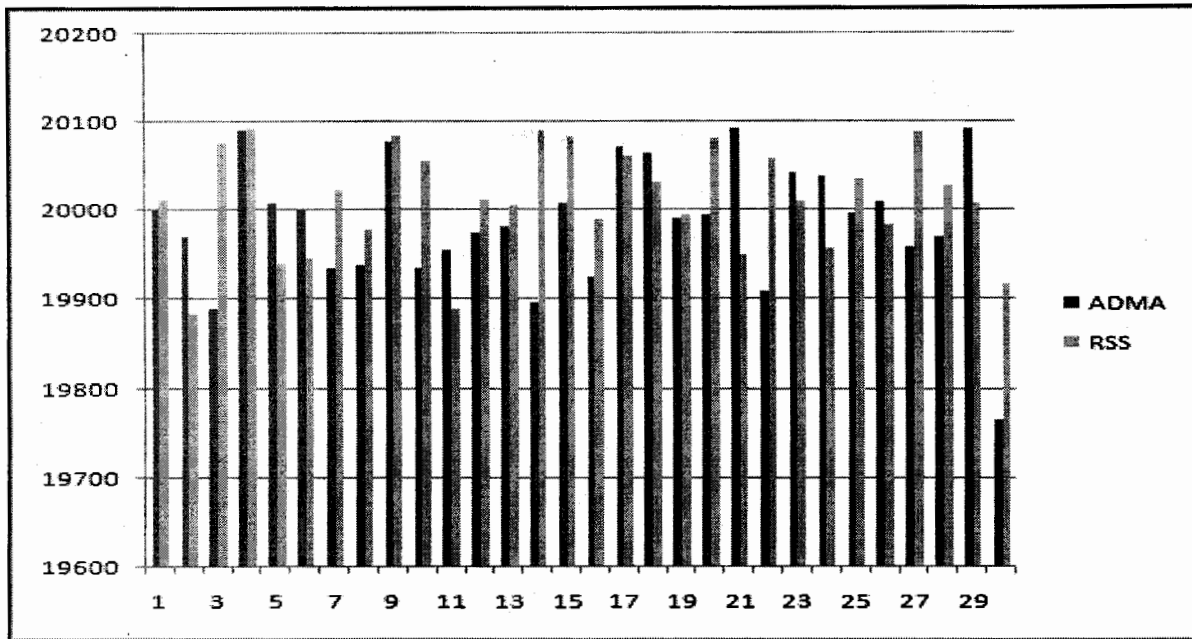


Figure 43 Energy Consumption by Individual Node

The above energy graph represents the optimum results for individual node energy consumption. This graph represents the energy consumed by nodes in only one simulation. Subsequent graph, will present the total energy consumption in each simulation. In figure 44, the statistics shows that RSS scheme consume a little more energy than the ADMA. This more energy consumption is due to more calculation performed by RSS to provide better security for the data. This difference is minor and we will explain it in the next figure.

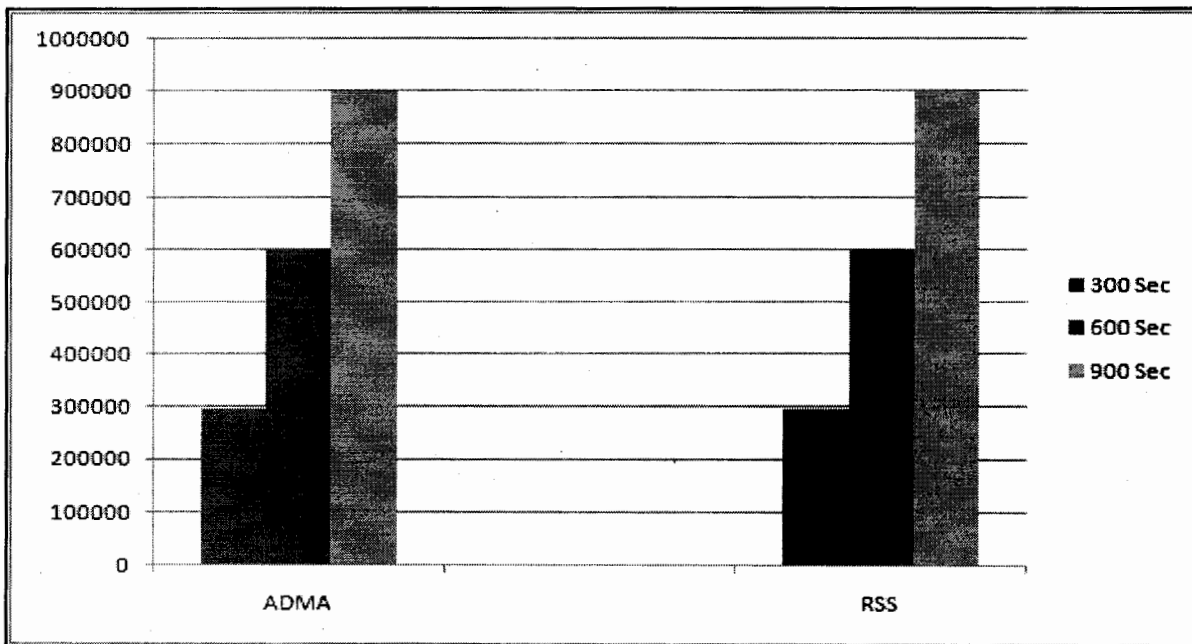


Figure 44 Total Energy Consumption by Each Simulation

Total energy consumed by the ADMA scheme is “1798077” joules and proposed RSS scheme consumed “1799444” joules. We analyze both the readings, and then we get the difference of 1366joules. RSS consumes 1366 joules more energy than the ADMA because it performs more computation than the ADMA and with this computation it makes the data more secure. This comparison is shown in the below figure 45.

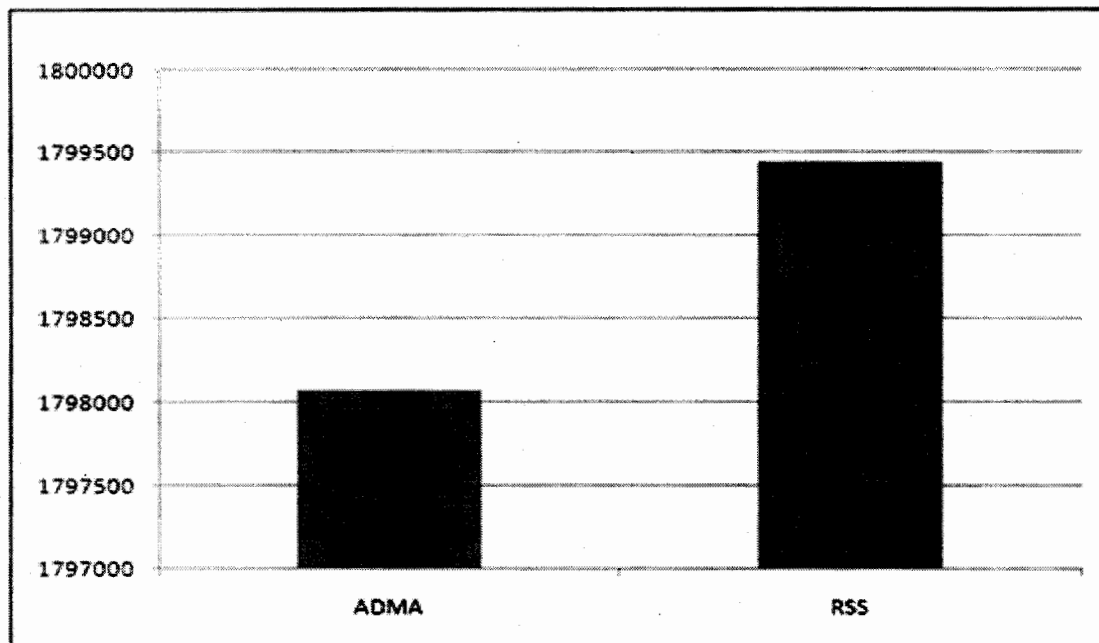


Figure 45 Total Energy Overhead

According to the above graph RSS utilized “1366” joules more energy than ADMA. As we know that we have to pay the cost for making the security of data more effective. We have to bear this minor overhead, and if divide this energy overhead between 30 nodes then it is “45” joules for single node, for 30 minutes simulation. In energy consumption, few hundred are small overhead for making data more secure. If we check it according to percentage, the overhead is just 0.075%.

7.3 Comparison with other Schemes

In the analysis first, we compare our RSS with some secure aggregation techniques then we discuss the analysis of our scheme from different perspectives. These schemes are given below.

1. Simple Aggregation
2. Hu. and Evas [10]
3. Secret Multipath Aggregation[20]
4. Dispersed Multipath Aggregation[20]
5. Authenticated Dispersed Multipath Aggregation[20]

Table 4 Comparison of RSS with other Secure Aggregation Schemes

Schemes	Provides Protection From			Security
	Eavesdropping	Data Tampering	Denial of Service DoS	
Simple Aggregation	No	No	No	Low
Hu and Evas[10]	No	Yes	No	Low
SMA[20]	Yes	Weak	Weak	Low
DMA[20]	Weak	Weak	Weak	Low
ADMA[20]	Weak	Weak	Yes	Low
RSS	Yes	Yes	Yes	High

We have compared our RSS scheme with some well known secure aggregation schemes and the results are displayed in the above table 4. Proposed scheme provides strong protection against eavesdropping, data tampering and denial of service attacks.

RSS provides more protection to the data from attacks than all the schemes of data aggregation, given in the table 4. In the subsequent chapter, we will finally conclude about the RSS scheme.

Chapter 8

Conclusion and Future Work

8. Conclusion

We have proposed RSS (Re-Sequencing Scheme) for secure data aggregation. This scheme provides better data security from adversary during the transmission. It resists against the adversary to protect the data. This scheme depends on symmetric key, information dispersal algorithm and multiple paths routing. RSS provides protection against eavesdropping, data tampering and denial of service attacks even in the presence of compromised nodes. It provides data authenticity, protection and data availability. Some energy overhead is the cost for making data more secure and this cost is 0.075 %.

8.1 Future Work

We want to remove the loop-holes in this scheme and make it more efficient. Still we have done the simulation of the RSS scheme but in the future we plan to implement this scheme with real time nodes to get real results.

We also plan to simulate different types of attacks on the scheme to analyze its resistance against the attacks.

References

- [1] J.K. Hart, K. Martinez, "Environmental Sensor Networks: A revolution in the earth system science", *Earth-Science Reviews*, 78. pp. 177-191.2006
- [2] Tiwari, Ankit et al. "Energy-efficient wireless sensor network design and implementation for condition-based maintenance" *ACM Transactions on Sensor Networks (TOSN)*, <http://portal.acm.org/citation.cfm?id=1210670>
- [3] Banner Engineering (March 2009), Application Notes, http://www.bannerengineering.com/en-US/wireless/surecross_web_appnotes
- [4] <http://www.sensor-networks.org/index.php?page=1011111101>
- [5] http://apsellab.ece.cornell.edu/imageszhongtao_sensor_network2.jpg
- [6] <http://blogs.dolcera.com/files200908wireless1.JPG>
- [7] http://www2.imse-cnm.csic.es/vmote/english_version/imagesWSN_sketch.png
- [8] Sarah Rutledge, "The PIC Farm: An Multiprocessor Sensor Board Msc. Advanced Computer Science" Chapter 2, Node Architecture September 2008. www.lancs.ac.uk/postgrad/rutledge/pdf/ch2.pdf
- [9] [https://ws.edu.isoc.org/trac/wirelessu/attachment/wiki/Material%27/17 Introduction_WSN-v1.0.pdf](https://ws.edu.isoc.org/trac/wirelessu/attachment/wiki/Material%27/17%20Introduction_WSN-v1.0.pdf)
- [10] Hu, L., Evans, D. "Secure aggregation for wireless network" In: *Proc. IEEE Symposium on Applications and the Internet Workshops (SAINT'03)*, pp 384–394, 2003.
- [11] David Wagner, "Resilient Aggregation in Sensor Networks" ,*Workshop on Security of Ad Hoc and Sensor Networks*, ISBN1-58113-972-1, pp 78-87, ACM/NY-USA, 2004.
- [12] Pawan Jadia and Anish Mathuria, "Efficient Secure Aggregation in Sensor Networks" *High Performance Computing - HiPC* , pp 40-49, Bangalore,-India, 2004.
- [13] Claude Castelluccia et al. "Efficient aggregation of encrypted data in wireless sensor networks" In *MobiQuitous*, ISBN: 0-7695-2375-7, pp 109-117, IEEE Computer Society, 2005.

- [14] Haowen Chan et al. "Secure Hierarchical In-network Aggregation in Sensor Networks" Conference on Computer and Communications Security, ISBN: 1-59593-518-5, pp 278-287, ACM, NY-USA, 2006.
- [15] Wembo He et al. "PDA: Privacy Preserving Data Aggregation in WSN" International Conference on Computer Communications, ISSN: 0743-166X, pp 2045 – 2053, IEEE, Anchorage, AK, 2007.
- [16] Robert Di Pietro et al. "Confidentiality and Integrity for Aggregation in WSN Using Peer monitoring" journal Security and Communication Networks, Volume 2, Issue 2, pp 181-194, 2007.
- [17] C. Castelluccia, C. Soriente, "ABBA: A Balls and bins approach to secure aggregation in WSNs" in: Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks and Workshops, WiOPT 2008.6th International Symposium on, April 2008, p. 185-191, 2008.
- [18] Julia Albath and sanjay Madria, "Secure Hierarchical Data Aggregation in Wireless Sensor Networks" Wireless Communications and Networking Conference, 5-8 April, ISBN 1525-3511, pp 1-6, IEEE, Budapest, 2009.
- [19] Suat Ozdemir and Yang Xiao, "Hierarchical Concealed Data Aggregation for Wireless Sensor Networks", in Proc. of Embedded Systems and Communications Security Workshop in conjunction with IEEE SRDS , September 27-29, Niagara Falls, NY 2009.
- [20] Thomas Claveirole et al. "Securing Wireless Sensor Networks Against Aggregator Compromises" in: Communications Magazine, IEEE, Security in Mobile ad hoc and Sensor Networks, Volume 46, Issue 4, ISSN: 0163-6804, p 134 – 141, 2008.
- [21] M. O. Rabin, "Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance," J. ACM, vol. 36, no. 2, pp. 335–48, 1989.
- [22] Philip Levis et al. "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications" SenSys'03: Proceedings of the 1st international conference on Embedded networked sensor systems, November 5–7, pp 126-137, ACM press, Los Angeles, California, USA, 2003. <http://www.cs.berkeley.edu/~pal/pubs/tossim-talk.pdf>
- [23] <http://www.ee.duke.edu/~romit/courses/s07/material/lecture-sensor-ddiffusion-leach.ppt>
- [24] Steffen Peter et al. "On Concealed data aggregation for wireless sensor networks" In Proceedings of the IEEE Consumer Communications and Networking Conference, Jan. 2007.

-
- [25] Hasan Cam et al. "Energy-Efficient and secure pattern based data aggregation for wireless sensor networks", Special Issue of Computer Communications on Sensor Networks, pp. 446-455, Feb. 2006.
- [26] http://networks.cs.ucdavis.edu/~yickresearchSensorCost_clip_image001.jpg
- [27] http://wsn.oversigma.com/wiki/index.php?title=WSN_Platforms
- [28] Kewei Sha et al. "Using Wireless Sensor Networks for Fire Rescue Applications: Requirements and Challenges", Electro/information Technology, 2006 IEEE International Conference, pp 239 – 244, East Lansing, MI, 04 December 2006.
- [29] http://en.wikipedia.org/wiki/Shamir%27s_Secret_Sharing
- [30] A. Shamir, "How to Share A Secret, *Commun. ACM*, vol. 22, no. 11, 1979, pp. 612–13.
- [31] <http://bryanismills.net/archives/2007/09/information-dispersal-algorithms/>

Acronyms

RF:	Radio Frequency
PH:	Privacy Homomorphism
ID:	Identification
I/O:	Input / Output
WSN:	Wireless Sensor network
MAC:	Message Authentication Code
RAM:	Random Access Memory
MHz:	Mega Hertz
GHz:	Giga Hertz
CPDA:	Cluster-Based Private data Aggregation
SMART:	Slice Mix Aggregation
ABBA:	A Balls and Bins Approach
SMA:	Secret Multipath Aggregation
DMA:	Dispersed Multipath Aggregation
ADMA:	Authenticated Dispersed Multipath Aggregation
DBG:	Debug
GUI:	Graphical User Interface
Sq. ft:	Square Feet
TOSSIM:	TinyOS Simulator
NesC:	Network Embedded System C
SOS	Sensor Network Operating System