

Denial of Service Attack Mitigation Technique for Wireless Networks using Machine Learning Methodology



By

Jahanzeb Shahid

NUST201362773MSEEC63013F

Supervisor

Dr. Shahzad Saleem

Department of Computing

A thesis submitted in partial fulfillment of the requirements for the degree of Masters in
Information Security (MS-IS)

In

School of Electrical Engineering and Computer Science,
National University of Sciences and Technology (NUST),
Islamabad, Pakistan.

(May, 2017)

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Approval

It is certified that the contents and form of the thesis entitled "Denial of Service Attack Mitigation Technique for Wireless Networks using Machine Learning Methodology" submitted by Jahanzeb Shahid have been found satisfactory for the requirement of the degree.

Advisor: Dr. Shahzad Saleem

Signature: _____

Date: _____

Committee Member 1: Wg Cdr M. Nauman Qureshi

Signature: _____

Date: _____

Committee Member 2: Dr. Rizwan Ahmad

Signature: _____

Date: _____

Committee Member 3: Dr. Saad Qaisar

Signature: _____

Date: _____

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis written by Mr. JAHANZEB SHAHID, (Registration No NUST201362773MSEEC63013F), of ___SEECs_____ (School/College/Institute) has been vetted by undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfillment for award of MS/M Phil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the said thesis.

Signature: _____

Name of Supervisor: **Dr. Shahzad Saleem**

Date: _____

Signature (HOD): _____

Date: _____

Signature (Dean/Principal): _____

Date: _____

*I dedicate my dissertation work to my teachers and parents for
their endless support and love.*

Certificate of Originality

I hereby declare that this submission titled **Denial of Service Attack Mitigation Technique for Wireless Networks using Machine Learning Methodology** is my own work. To the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics, which has been acknowledged. I also verified the originality of contents through plagiarism software.

Author Name: Jahanzeb Shahid

Signature: _____

Acknowledgment

First and foremost, I would like to thank Allah (SWT) for blessing me with the strength, courage, and wisdom to complete my MS thesis. I would like to express my deep gratitude to my parents for their prayers, everlasting love, encouragement and support that led me towards the achievement of this worthwhile goal.

I am highly honored to be mentored and supervised by *Dr. Shahzad Saleem*. His guidance, motivation and mentorship by far had been the most encouraging factors to complete my research work. I am very grateful for his sage advice that helped me get to where I am today and look forward to his guidance in the future as well.

I offer a special thanks to my committee members, who have guided me in many different ways. I am grateful to *Dr. Rizwan Ahmad*, for numerous thought-provoking conversations and suggestions during the implementation phase. Thanks *Wg Cdr Muhammad Nauman Qureshi* for useful guidelines regarding the completion of MS thesis work. The detailed comments, suggestions and observations given by committee member's allowed me to improve my work and I look forward to continuing collaboration with such insightful committee members in future too.

Jahanzeb Shahid

Table of Contents

Abbreviations	10
List of Figures	12
Abstract	14
Chapter 1 Introduction	15
1.1 Background.....	15
1.2 Motivation	16
1.3 Problem Statement.....	16
1.4 Research Objectives and Contribution	17
1.4 Structure of Thesis	18
Chapter 2 Literature Review	19
2.1 Overview of Wireless Sensor Network	19
2.2 Hardware Modules of a Node	21
2.3 Cluster-Based Hierarchical Routing Protocols	22
2.4 Existing Techniques for Energy Drainage Attacks	25
2.5 Cellular Automata based Solutions for Energy Drainage Attacks	26
2.6 Trust based Solutions for Energy Drainage Attacks	26
Chapter 3 Proposed Energy Drainage Attack Model	28
3.1 Network Architectural Model	28
3.2 Energy Drainage Attacks on LEACH protocol.....	30
3.3 Vampire Attack on LEACH protocol.....	31
3.4 Scheduling Attack on LEACH protocol	33
3.5 Gray-Hole Attack on LEACH protocol.....	34
3.6 Network Operational Model	34
3.6.1 Setup Phase.....	34
3.6.2 Steady-State Phase.....	35
3.7 Network Attack Detection Method	35

Chapter 4	Detection of Energy Drainage Attack on LEACH Protocol	38
4.1	Formation of Neighborhood.....	38
4.2	Protocol Stack and Information Sharing between Nodes.....	40
4.3	Self-Monitoring Nodes	41
4.4	Normal and Abnormal Packet Transmission Detection	43
4.5	Detection of Attack and Change of States	44
Chapter 5	Prevention Scheme for Malicious Cluster Head Selection.....	46
5.1	Calculation of Trust	46
5.2	Selection of Cluster Head based on Trust Value	48
Chapter 6	Implementation and Results.....	50
6.1	Simulation Environment	50
6.2	Simulation in MATLAB	50
6.2.1	Downloading of LEACH Protocol	50
6.2.2	Running LEACH Protocol Simulation on MATLAB	51
6.3	Simulating Energy Drainage Attack on LEACH.....	53
6.3.1	Simulating Gray-Hole Attack.....	54
6.3.2	Simulating Scheduling Attack.....	56
6.4	Detection of Energy Drainage Attack	58
6.5	Prevention Scheme for Malicious Cluster Head Selection.....	58
6.6	Simulation in ns-2.....	60
6.7	Simulating Energy Drainage Attack in ns-2	62
6.8	Prevention Scheme Simulation in ns-2.....	67
Chapter 7	Conclusion and Future Work.....	68
7.1	Thesis Contribution.....	68
7.2	Conclusion.....	68
Appendix A		69
References		77

Abbreviations

Abbreviations	Descriptions
ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard
AODV	Ad-Hoc on Demand Distance Vector
BIDRP	Base-Station Initiated Dynamic Routing Protocol
DEEC	Distributed Energy-efficient Clustering protocol
DoS	Denial-of-Service
DTLS	Datagram Transport Layer Security
CA	Cellular Automata
CDMA	Code Division Multiple Access
CH	Cluster Head
GPS	Global Positioning System
IETF	Internet Engineering Task Force
IoT	Internet of Things
IP	Internet Protocol
LEACH	Low-Energy Adaptive Clustering Hierarchy
LEACH-C	Low-Energy Adaptive Clustering Hierarchy-Centralized
MAC	Medium Access Control
MTU	Maximum Transmission Unit
RAM	Random-access memory
RFID	Radio Frequency Identification
ROM	Read-only memory
RRM	Registration Request Message

RSSI	Received Signal Strength Indicator
TCP	Transport Control Protocol
TDMA	Time Division Multiple Access
T-LEACH	Trusted Low-Energy Adaptive Clustering Hierarchy
TLS	Transport Layer Security
UDP	User Datagram Protocol
URL	Uniform Resource Locator
WSN	Wireless Sensor Network
DEEC	Distributed Energy-efficient Clustering
HEED	Hybrid Energy-efficient Distributed

List of Figures

2.1	General WSN Architecture	20
2.2	Usage of WSN in different Environments	21
2.3	Hardware Architecture of a Node.....	21
3.1	First Order Radio Transmission Model	29
3.2	LEACH two phases work flow diagram.....	35
4.1	WSN Field.....	39
4.2	Sensor Node Neighboring Area	40
4.3	Sensor node Neighboring Vector	40
4.4	Protocol Stack.....	41
4.5	Node Energy Levels.....	42
5.1	Node Direct and Indirect Trust.....	46
6.1	MATLAB Run Button	51
6.2	Default LEACH Simulation in MATLAB.....	52
6.3	Default LEACH Energy Drainage plot in MATLAB	52
6.4	Distributed Nodes Simulation in MATLAB	53
6.5	Malicious Cluster Head Simulation in MATLAB	54
6.6	Gray-Hole Attack Energy Drainage plot in MATLAB	55
6.7	Died Nodes Simulation in MATLAB	55
6.8	Scheduling Attack Energy Drainage plot in MATLAB	56
6.9	Comparison of Gray-Hole and Scheduling Attack	57
6.10	Nodes in Blue color depicting Alert State of nodes in MATLAB	58
6.11	CAT-EDP Energy Drainage plot in MATLAB	59
6.12	Comparison of Default LEACH and CAT-EDP in MATLAB	60
6.13	Default LEACH Energy Drainage in ns-2.....	61
6.14	Gray-Hole Attack Energy Drainage plot in ns-2	62
6.15	Comparison of Default LEACH and Gray-Hole in ns-2.....	63
6.16	Scheduling Attack Energy Drainage plot in ns-2.....	64
6.17	Comparison of Default LEACH and Scheduling in ns-2	64

6.18	Comparison of Default LEACH, Gray-Hole and Scheduling Attacks in ns-2.....	65
6.19	Number of Died nodes with Default LEACH in ns-2.....	65
6.20	Comparison of Died nodes with Default LEACH, Gray-Hole and Scheduling Attacks in ns-2.....	66

Abstract

A Wireless Sensor Networks (WSNs) consists of small sensor nodes that collect important information about an area of interest. The ability of these networks to monitor remote and hostile locations has attracted a significant amount of research over the last decades. As a result of this research, WSNs have make their importance in different applications such as transportation, industrial automation, healthcare, habitat monitoring and military surveillance. These networks can be operated in human-unapproachable lands and collect data on an exceptional scale. However, these networks face several technical challenges at the time of deployment as well as operation. Resource limitations such as battery power, inadequate resources for computation arise problems during their operations.

One of the major concern which demands proper attention is energy conservation. To extend the lifetime of these networks an energy-efficient routing protocol is very much required. Moreover, the deployment of these networks in a threatening environment and their operation in the presence of error-prone communication links render these networks to several security loopholes. As a result, there is a great need to design a routing protocol which is more energy efficient and robust against energy drainage attacks.

Low Energy Adaptive Clustering Hierarchy (LEACH) is a hierarchical-based routing protocol which is considered as an energy-efficient routing protocol. This thesis aims to provide an analysis of LEACH protocol under an energy-drainage attacks and then provide effective solution to detect and then minimizing the effects of these attacks. Selection of LEACH protocol is based on usage of this protocol on industrial level and in terms of energy efficiency LEACH protocol is highly adoptable. Due to these reasons LEACH protocol is also prone to different type of attacks. Although it is an energy-efficient protocol but there are some energy drainage attacks that make this protocol less efficient. We also discuss some energy drainage attacks like Grey-Hole, Scheduling and Vampire Attack on LEACH protocol and proposed a scheme to mitigate these attacks. The energy drainage of the nodes is mitigated by using Cellular Automata (CA) scheme for cluster head selection. The main objective of this thesis work is selection of an optimal cluster heads regarding their trust values, reduces the energy consumption and mitigate from energy drainage attacks associated with malicious CH selection. Only suitable nodes are chosen as CHs to improve usage of the resources depending upon their trust values. We also propose some key ideas about the creation of clusters in network and density of the associated nodes with CH in a cluster according to their positions.

Chapter 1

Introduction

This thesis has three main goals. First, the Low Energy Adaptive Clustering Hierarchy (LEACH) cluster-based hierarchical routing protocol of WSN is analyzed to find the loophole from where a malicious cluster head would drain network energy. Second, an energy drainage attack is simulated on LEACH protocol to find out the statistics of network and an individual node life. Finally, we use Cellular Automata (CA) and trust based solution to secure this protocol from energy drainage attack. This chapter is organized as follows. In Section 1.1, we outline the background of the work presented in this thesis. In Section 1.2, we discuss the motivation for our work. The research objectives, contributions and novelty of the work are discussed in Section 1.3 followed by our research objective in Section 1.4. Finally, we conclude the chapter by providing the outline of the structure.

1.1 Background

In modern world human life has been changed by the internet because it gives unified communication with less problems. Technological developments filled the communication gap and also effects other different areas like transportation, healthcare, agriculture and education. Advanced high-tech devices have made it feasible to connect any device anytime and anywhere in the world with the help of internet. This is not possible in the old-fashioned wired networks.

These nodes are tiny, with less computation power and storage capabilities, and they operate on trivial power batteries [1]. In WSN the data is transmitted to a central device called base station for more operations. Two more important features of WSN are scalability and fault tolerance[2]. As a result, they have found their applications in different fields such as agricultural monitoring system [3], remote healthcare monitoring [4], fire monitoring [5] and environmental monitoring system [6].

In WSNs, usage of mobile or static nodes depends on monitored application. Mostly applications are deployed statically, which has several disadvantages [7]. Primary, static deployment of nodes cannot ensure complete coverage of the network. Second, when any node dies or malfunction, it creates “hole” in the network which break communication in network. Gateway or cluster head nodes play important role in static deployment, which are one-hop away from the base

station. They drain more energy because these nodes have responsibility to forward network traffic to base station. By contrast in mobile deployment, nodes change their positions in the field to yield different sets of gateway nodes [8].

In WSNs, nodes are installed to monitor an application with minimal human intervention. Therefore, for efficient utilization of battery usage special attention is needed on the architectures of energy-efficient routing protocols.

1.2 Motivation

The energy-efficient nature of cluster-based hierarchical routing protocols motivated us to use it for our research. Depending on the cluster head selection technique, these protocols are broadly classified into (i) randomly distributed and (ii) centralized protocols. In randomly distributed protocols like LEACH, between the range of 0 and 1, every node selects a random number. If this number is less than a predefined value (explained in Chapter 2), that particular node is get selected as CH. These protocols stress on refining the random number for CH selection. However, refining the random number could not inevitably choose an optimum number of CHs. On these bases, there is a high probability that not a single node or all the nodes in network can become CH. Both situations cause energy drainage, wastage of bandwidth and low quality of aggregated data.

The CH selection decision should not be entirely based on random numbers, but some other parameters like residual energy, location awareness and previous behavior of each node must considered as well. The residual energy, energy consumed by a CH in a round and location information affects CH selection.

Recently, the simple organization of Cellular Automata (CA) has been getting significant attention for the comprehension of energy drainage problems in WSNs [9]. In CA, each cell works as an automaton, it takes outputs of its neighboring cells as an input and change its state based on these inputs [10]. So the states of these cells are based on the states of their neighboring cells. This architecture of CA has drawn attentions of researchers from different fields. Specifically, CA has become a suitable model for operational control and decision making in networks. Energy-conservation for longer network lifetime can also be achieved by using CA models. We will discuss about cellular automata in upcoming chapters.

1.3 Problem Statement

LEACH is an energy-efficient routing algorithm that is used where the longer lifetime of the network is required. But it has some disadvantages like it assumes that all the nodes have same energy value which is not possible for real-time problems [11]. LEACH is a hierarchical cluster based protocol that totally rely

on CHs for routing and data aggregation which make it vulnerable for routing based attacks [11].

Most of the attacks in this protocol depends on the selection of CH phase. The selection of CH totally depends on the energy value of the node and probability of being a CH in a round. A malicious node can easily be introduced by impersonating the node ID or with fake ID in the network to deceive the selection criteria of CH [12]. A malicious node can be introduced in the network which declare itself as a CH and in each round it tends to be a CH to drain the energy value of the associated nodes. The problem here is to make secure the CH selection procedure.

There are many cryptographic authentication [13] based algorithm for the secure selection of CH but they arise the problem of computation and memory storage. In LEACH protocol, number of nodes nominate themselves to become CH in each round so these cryptographic authentication based solution are not feasible [12].

So there is a need to develop an algorithm which poses less overhead on the network nodes and also make sure secure selection of CH. The other problem is the detection of malicious node in the network. In LEACH protocol random selection CHs creates the problem of network monitoring, detection of malicious node is little bit difficult in such covert environment.

Other trust based solutions has also been proposed which uses trust values of the other nodes for selection of CH like T-LEACH [14].

1.4 Research Objectives and Contribution

We have pointed out different research gaps in Section 1.2, which motivated us to work on this topic. We built following objectives and contribution based on those research gaps.

A hierarchical routing protocol LEACH was proposed to improve selection criteria of CH and improve the lifespan of the network. In our proposed solution each node selects its CH based on residual energy of nominated node, its location and energy consumed previously if it had become cluster head in the past.

With the proposed cellular automata based energy efficient solution for LEACH protocol, we could keep track on energy values of neighboring nodes. It will help for the identification of nodes which are under energy drainage attack. We set some threshold value in our proposed scheme which helps to automate the nodes to change their states in case of malicious behavior.

1.5 Structure of Thesis

Prior research work on energy drainage attacks on WSN and their proposed solution based on CA is discussed in chapter 2. Chapter 3 proposes energy efficient cluster based hierarchical routing algorithm like LEACH along with its drawback. Chapter 4 discusses detection mechanism of our proposed algorithm. Prevention scheme for malicious CH selection presented in Chapter 5. Details about simulation environments and their results are discussed in Chapter 6. In the last Chapter 7 conclude our thesis work and give some future work direction.

Chapter 2

Literature Review

Wireless Sensor Networks (WSNs) have been involved in wide range of application since last decade. Working in human unreachable and risky locations have drawn a lot of research attention to solve several problems at different layers. There are several applications which work in such intimidating environments, devices in these environments fail commonly and usually difficult to repair. The problem of regular failure of devices for these applications can be solved by using low-cost replaceable devices. Additionally, these applications demand a cost-effective multi-hop wireless communication for their operations. To lessen the cost of operation, the devices should remain in sleep mode while they are communicating.

In this chapter we discuss a literature review that will also show relation to our proposed solution. We discuss about WSN briefly in Section 2.1, its types and its applications. Sensor node modules are discussed in Section 2.2. In Section 2.3, we present a complete depiction of cluster-based hierarchical routing protocols in terms of their energy-efficient nature. In Section 2.4, we discuss the existing energy drainage attacks detection techniques in WSNs. In Section 2.5 we discuss Cellular Automata (CA) and its components. Energy drainage problems in WSN handled with CA are discussed in section 2.6.

2.1 Overview of Wireless Sensor Network

Small sensor nodes are usually deployed to operate together and sense physical parameters in a dynamic environment [1]. These parameters include humidity, temperature, pressure and movement. For deployment purposes of nodes, WSN have two classes: unstructured and structured. A dense deployment of nodes which operate in an ad-hoc fashion lies in unstructured deployment. In this deployment nodes are left unattended to collect the data. Network maintenance, connectivity management and failure detection are the major problem in this class of deployment [15]. More nodes make these jobs more difficult.

In a structured deployment, number of nodes are small and deployed with some planning. As a result, networks form with structured deployment are easy to maintain. Moreover, nodes connectivity can be managed easily and thus node failure easily be detected as well.

Sensor nodes, a base station and a sensor field are the main components of the network [15]. A general WSN architecture is shown in Fig 2.1. The nodes monitor

the sensor field and collect information about the events. Application in sensor nodes generate data about an interested event like change in temperature, wind speed or an alarm from wildlife monitoring application [16]. The data is then forwarded to the related user devices after examination. User device can be a computer or a smart phone which is registered with the network. Decision makers use these devices to know the condition of the monitored field and then decide in the light of the information. The nodes which sense the environmental parameters are called source or end nodes and which forward the data to the base station are called intermediate nodes.

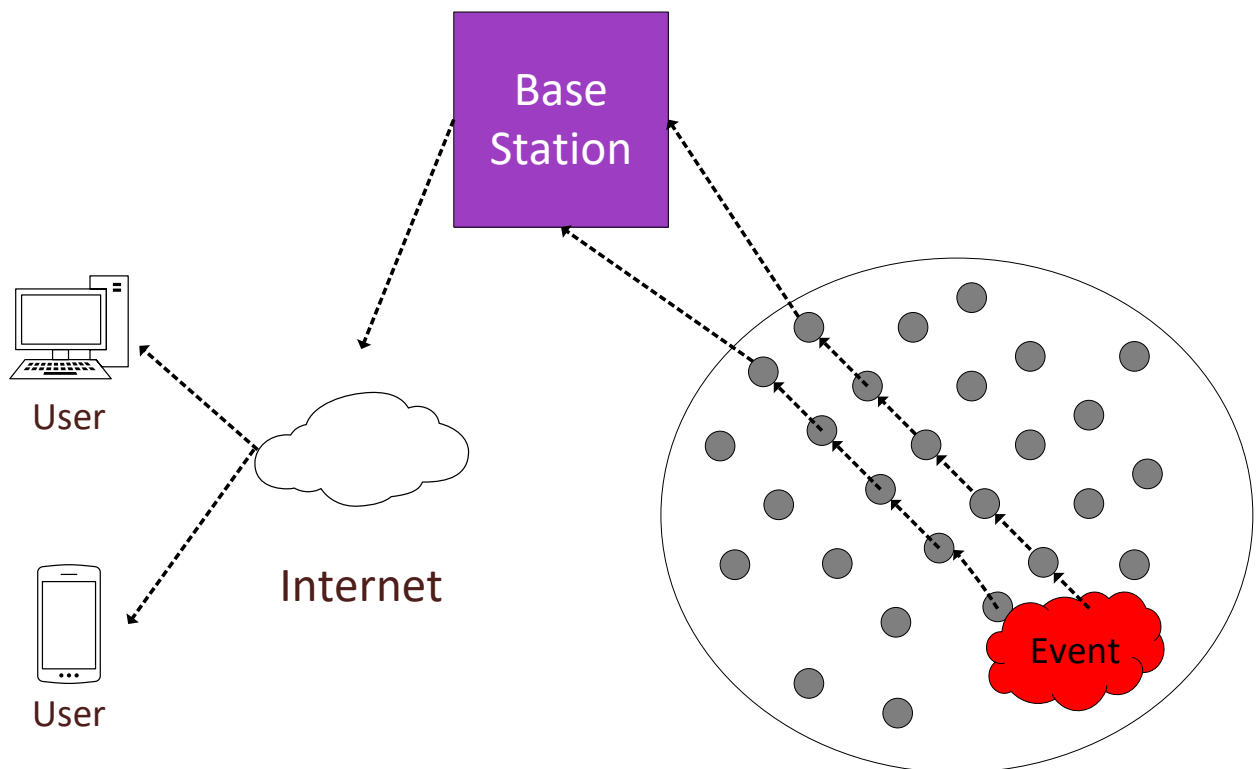


Fig 2.1

In WSNs, multi-hop communication is feasible because the nodes are not capable of long-haul transmission. There are three categories in which we classify these networks [39]: periodic, event-driven and query-based. In periodic network, after a fixed interval of time nodes send their data to base station. In the event-driven network, nodes wake up when some specific event occurs otherwise they remain in sleep mode. In query-based network, nodes only send their sensed data to the base station when an end user makes query about an information.

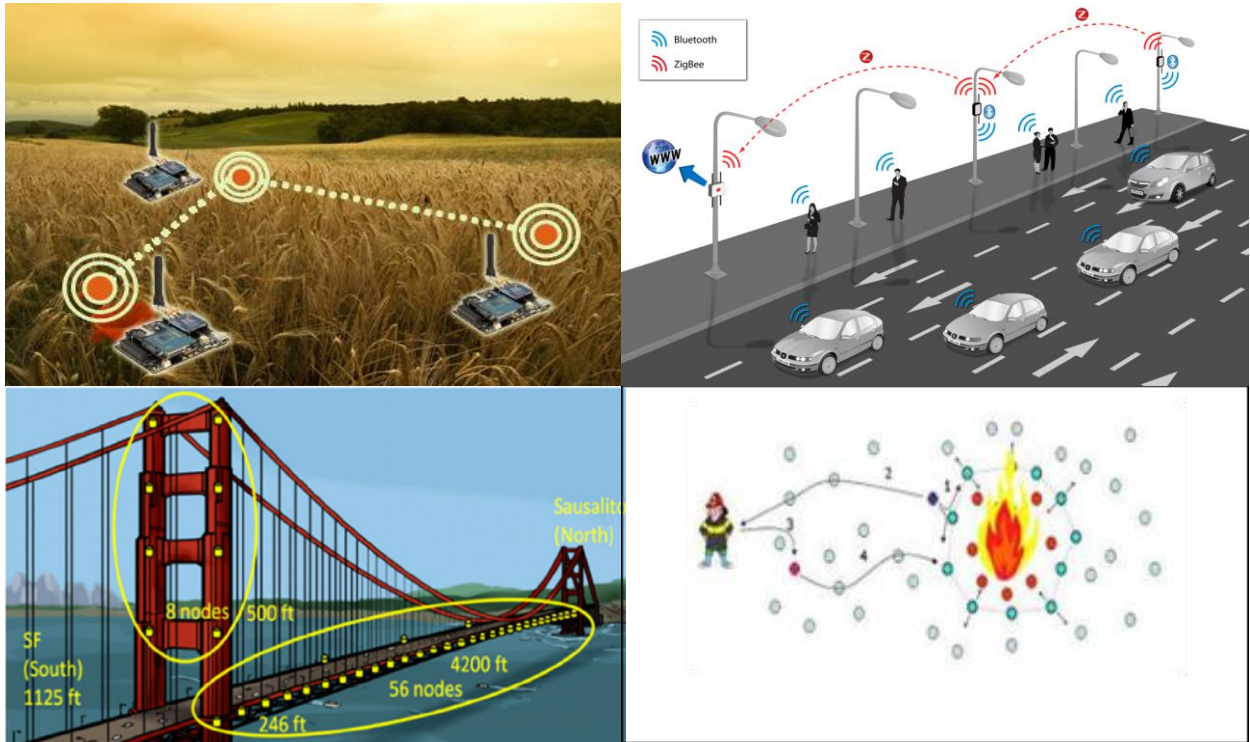


Fig 2.2

2.2 Hardware Modules of a Node

A sensor node consists of four basic modules [17]: a sensor, a radio transceiver, a power unit and a processor. Other than these vital modules, a node can have some other modules such as, a GPS, an accelerometer, a power generator, an analog-to-digital converter (ADC) and a storage module. Fig. 2.3 presents the hardware architecture of a node.

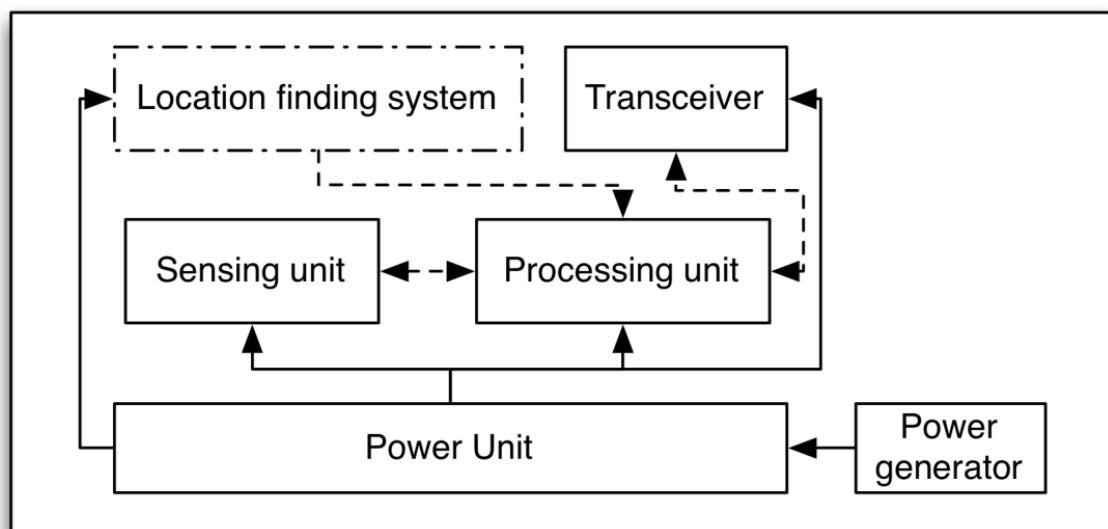


Fig 2.3

Depending on the need of an application, each node may be equipped with multiple sensors. The sensor module senses the environmental physical parameter. This sensed data is in the form of analog physical signals which has to be converted in digital form. The ADC module converts these analog signals to digital form and directs to a processing module. Processing module processes this data and directs to a radio transceiver for eventual transmission to a neighboring node. The radio transceiver can transmit and receive these signals simultaneously. The power or battery module is capable of managing the energy requirements of a node. The location finding module or GPS is a supplementary module. In order to find the right location of the neighboring nodes several localization algorithms are used to program this module. For application specificity, accelerometer module is typically used in applications in which nodes are mobile and its speed of the mobile node is also required. Nodes store their sensed data in storage module. This sensed data is processed by processing unit and then transmitted to neighboring nodes or base station.

2.3 Cluster-Based Hierarchical Routing Protocols

There are two types of cluster-based hierarchical routing protocols: randomly distributed (Self-organized) and centralized (Base Station Assisted). Their cluster features make them distinguishable [16]. In this section, we discuss the randomly distributed cluster-based hierarchical routing protocols followed and then centralized cluster-based hierarchical routing protocols.

2.4 Existing Techniques for Energy Drainage Attacks

Residual energy of the nodes need to be used for the selection of CHs. There are several proposed solutions which use node resources for the selection of CH. For example, protocols such as LEACH-B [21], Distributed Energy-efficient Clustering protocol (DEEC) [22] and Hybrid Energy-efficient Distributed Clustering protocol (HEED) [23] use residual energy for the selection of CH, consumed energy in current round and average node energy. An alternate solution for this problem is to make responsible a central controller for this purpose, i.e. a base station. In LEACH Centralized (LEACH-C) [24], base station is used for the selection of CH. In each round CH is selected based on average residual energy which should be less than remaining energy of the nodes. However, there is a high probability to get selected too many nodes as a CH. For heterogeneous sensor networks (BIDRP) Base-Station Initiated Dynamic Routing Protocol [25] is proposed. In this protocol nodes which have higher energy always selected as CHs by the base station.

2.5 Cellular Automata based Solutions for Energy Drainage Attacks

Solutions for energy drainage problems in WSN with the help of CA have already been proposed. We discuss some of them in this section. CA based energy saving solution for Multi-hop WSN protocol is proposed by Weiqiao Li, et al [26]. They define some transition rules for the nodes by combining CA and routing protocols. These transition rules prolong the network lifetime by switching nodes in active and sleep state. Block Cellular Automata model is proposed by Chayan Banerjee et al [27] to conserve energy in WSN. Group clustering technique is used to create group of nodes. CHs are selected by block of member nodes, which entirely forms a block. The CA rules are applied by considering a whole block as a single unit. Graph-Based Cellular Automata Approach to Maximum Lifetime Coverage Problem in Wireless Sensor Networks is proposed by Antonina Tretyakova et al [28]. This work based on experimental study and proposed an algorithm by comparing some centralized evolutionary algorithm. It includes all the advantages of localized algorithm, i.e. it uses information of neighboring nodes to self-organize themselves to achieve the goal of maximum lifetime of the network. ICLEAR: Energy Aware Routing Protocol for WSN Using Irregular Cellular Learning Automata is proposed by Amir Hosein, et al [29]. This protocols works as energy saver and a manager as well. Energy balancing and increasing network lifetime are the main goals of this work. But there is a tradeoff between these goals because energy balance itself causes energy consumption.

2.6 Trust based Solutions for Energy Drainage Attacks

Solution based on cryptography in WSN cannot be followed because they are costly and incapable of countering behavioral attacks. Now, behavioral attacks are mitigated with trust based solutions. TERP is a trust based energy drainage mitigation solution proposed by Adnan Ahmed, et al [30]. Distributed trust model is used to detect and isolate malicious or faulty nodes. TERP encapsulate residual-energy, trust and hop counts in their functions to make routing decisions. TREE-CR: Trust Based Secure and Energy Efficient Clustering in WSN is proposed by Rashmi Ranjan, et al [9]. They proposed a realistic energy consumption model for the prediction of network lifetime and detection of malicious node in the network. This model keeps track on all the basic activities of a sensor node. They also compare their proposed model with LEACH protocol. TBE-LEACH Trust based energy efficient routing in LEACH for wireless sensor network is proposed by Arzoo Miglani, et al [12]. They proposed an improvement in LEACH protocol by introducing trust factor to avoid secure routing.

An approach for Secure communication in IP-based WSNs via a trusted gateway is proposed by Floris Van den Abeele, et al [31]. They proposed that relies on a trusted gateway lessen the high cost of the Datagram TLS (DTLS) handshake in WSN. As compared to the standard DTLS this approach gives significant energy saving and latency reduction.

Chapter 3

Proposed Energy Drainage Attack Model

Residual energy of nodes play a significant part in the life time of sensor networks. Generally, a sensor node performs the operation of sensing, computation and communication of data, whereas a special node like Cluster Head (CH) performs some advance operations like data aggregation and cluster formation. So the energy consumed by a CH node is much more than other normal nodes. If these CH nodes become malicious than they can execute energy drainage attack in the network. In this way all the nodes associated with this malicious node can become victim to this attack. In this chapter we present our attack model after briefly discussing about network architecture and its communication.

3.1 Network Architectural Model

In order to simplify the network design we assume the following.

- Physical characteristics of the nodes are identical and all nodes have same initial energy.
- All the nodes are evenly distributed.
- All the nodes in the field are static.
- All nodes can communicate with two different frequencies. One is for communication with base station and other is for communicating with the neighboring nodes.
- Location of nodes are known to the system model.
- Transmission power of the nodes can be adjusted.

We use first order radio model for the transmission and receiving of the node [18] as shown in the fig below. Task of electronic component is to sense and then process the data while the amplifier component is liable for the transmission of data over low-power loss links of WSNs.

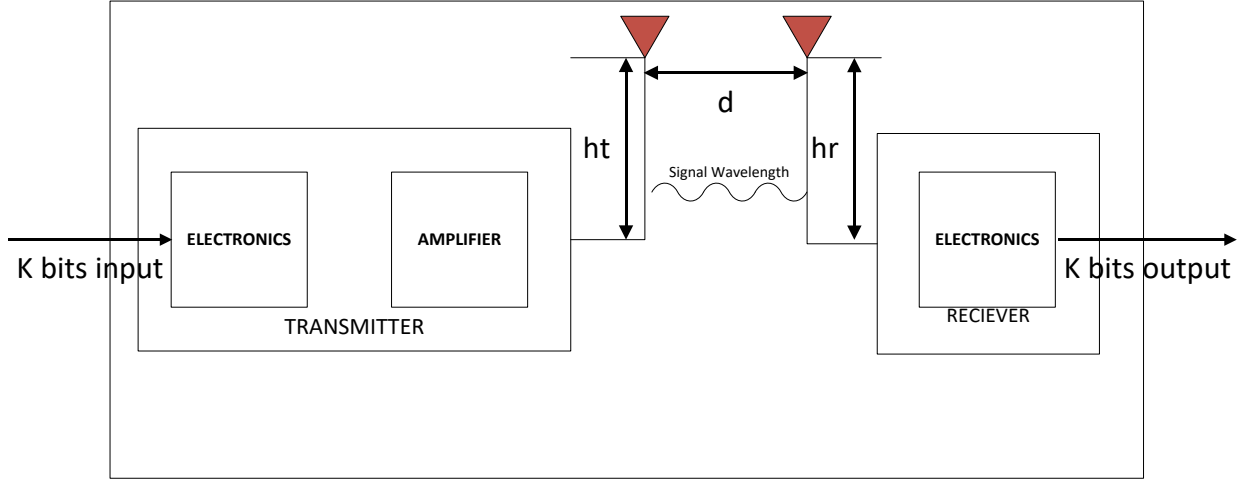


Fig 3.1

In these networks, energy consumed for communication is greater than the energy consumed for data sensing and processing [32]. Communication model can be determined by the distance between the nodes. Free-Space propagation model [33] is used if the transmitter and receiver node distance is less than the crossover distance (dc). Multipath ground propagation model (mp) is used if distance is greater than dc . In free-space model there is a line-of-sight connection between a receiver and a transmitter node. A radio wave travels through several paths because of reflection, refraction and deflection through various obstacles in multipath ground propagation model.

$$dc = 4\pi h_t h_r \sqrt{L} / \lambda \quad (3.1)$$

Here, h_t and h_r are the heights of transmitter and receiver antennas, L is the loss factor during transmission and λ is the wavelength of a radio signal. Generally, $L > 1$, but in ideal case if there is no loss in system hardware, then $L=1$ [34].

First Order Radio Model

Electromagnetic wave signal strength increases as the distance between transmitter and receiver increases. So we can represent the energy on the transmitter side to transmit the k bit data to a distance (d) with these equations.

$$E_{tran}(k, d) = E_{tran - elec}(k) + E_{tran - amp}(k, d) \quad (3.2)$$

Here $E_{tran - elec}(k)$ is the energy taken by the electronic circuit in a node to sense and process data with addition of $E_{tran - amp}(k, d)$ total energy consumed by amplifier unit of a node.

$$E_{tran - amp}(k, d) = E_{elec} * k + \epsilon_{amp} * k * d^2 \quad (3.3)$$

In this equation $E_{elec} * k$ represents energy consumed by the electronics circuit in an amplifier and $\epsilon_{amp} * k * d^2$ represents the energy consumed by amplifier for transmitting 1 bit data to unit area where d is the distance of radio transmission between two nodes.

And to receive this message, the radio expends:

$$E_{rec}(k, d) = E_{rec - elec}(k) \quad (3.4)$$

$$E_{rec - elec}(k) = E_{rec - elec} * k \quad (3.5)$$

Here $E_{rec - elec} * k$ represents the energy consumed by the receiver to receive k bit data. Receiver energy does not depend on the distance between nodes. So the total energy on the receiving side is almost equal to energy consumed by electronics of receiving module of the node.

So $E_{rec - elec}(k) \ll E_{tran - amp}(k, d)$, so we can say that the smaller the distance between the nodes, lesser energy will be consumed.

When the transmission distance is less than the crossover distance d_c , there is a linear relationship between attenuation of radio signal strength and the transmission distance d^2 . In other case it has linear relationship with d^4 .

$$E_{tran}(k, d) = \begin{cases} E_{elec} * k + \epsilon_{amp - freeSpace} * k * d^2 \\ E_{elec} * k + \epsilon_{amp - multiPath} * k * d^4 \end{cases} \quad (3.6)$$

$\epsilon_{amp - freeSpace}$ is the loss coefficient in free space model and $\epsilon_{amp - multiPath}$ is the loss coefficient in multi path model, cross over distance d_c is related to the transceiver.

3.2 Energy Drainage Attacks on LEACH

We discuss here three types of energy drainage attacks we launched on LEACH protocol.

1. Gray Hole Attack
2. Scheduling Attack
3. Vampire Attack

We try to explain these attacks on LEACH protocol for more understanding of energy drainage attack model.

Grey Hole Attack: This is the most known attack performed by manipulating the routing. An attacker simply forward all the traffic to an attractive node which other nodes consider as shortest hop count path and then drop all data

packets to a malicious node [5]. If a CH pulls all the traffic of a cluster and then drop this traffic then it becomes denial of service attack as well as energy drainage attack because cluster nodes drain their energies to transmit data to cluster head.

Scheduling attack: It is actually a DoS attack which can be launched in setup phase of LEACH protocol, during the scheduling of TDMA for the associated nodes to transmit their data. The malicious node which acts as a CH will assign same time slot to all nodes to send their data [35]. This change results in packet loss and leads to DoS.

We launched these attacks on LEACH protocol and analyze the effects of these attacks. Firstly we will explain, how these attacks can be launched on LEACH protocol and where is the vulnerability in the LEACH protocol that can be exploited to launch these attacks.

Vampire Attack: It is a new kind of DoS attack befalling on network layer and resource depletion (energy or battery drainage) attack [54]. This attack is difficult to detect because it does target multiple network layer protocols like SEAD, Ariadne and SAODV [55]. We launch this attack on LEACH protocol and tried to analyze the effects of Vampire attack. This attack further classified into Carousel attack and Stretch attack [56]. In Carousal attack a packet route is set in such a way that it keeps moving in series of loops. In Stretch attack victim node builds a long source path that packets pass through more number of nodes then present.

3.3 Vampire Attack on LEACH protocol

In this chapter we will discuss that how we can launch Vampire Attack on LEACH protocol. But before this we will see how Vampire Attack drain energy for a particular protocol. This attack is not specific to any protocol [36] and does not rely on routing properties or implementation fault of any protocol. But it exploit other properties of a protocol like source routing, distance-vector, and geographic and beacon routing [36]. There are two types of this attack Stretched Vampire Attack and Carousal Vampire Attack. Let us discuss these two types of attack.

Carousal Vampire Attack: In this attack a malicious node sends packet to a route which is consisted on series of loops [37], such that a packet passes through same nodes in a loop many times. This attack basically extend the route length in the network. As you can see in the Fig 3.x. This attack drain energy of the victim nodes if malicious node set higher value of loop count.

Stretch Vampire Attack: Another type of Vampire Attack is Stretch attack. In this attack a malicious node originate long path to the destination node. It causes packets to pass through more number of nodes than optimal number [37]. As in the Fig 3.x shows that packets on malicious path taking 4 hop counts from source node to sink node, but original path take 1 hop count. This attack drain energy more rapidly if large number of packets passes through malicious path.

We choose Stretch attack to launch on LEACH protocol. In which a malicious CH deceives its associated nodes to forward their packets to a longer distance CH. As we have discussed in section 3.2 how LEACH protocol works. In Set up phase when a CH broadcast (HEAD_Adv_Msg), this message includes CH node ID and its coordinates in the network. In LEACH protocol all the nodes join their CH based on Received Signal Strength Index (RSSI) value. In setup phase all nodes in the network received (HEAD_Adv_Msg) from CHs elected in a round. A node compares RSSI value of all received (HEAD_Adv_Msg) messages and choose a CH for itself which message has larger RSSI value.

To launch a vampire stretch attack, firstly a malicious node should be elected as CH and then deceive its associated nodes to forward their data to a longer distance CH. But how a node could be elected as CH in every round. For this we should see the formula in LEACH, which a node use to become a CH.

We have already define parameters of this formula. Here G is the value which describes a set of nodes which have not become a CHs in past $1/popt$ rounds. Suppose we set the value for $popt = 0.1$, then $1/0.1 = 10$, so if a node become a CH in a 1st round, then it will get a chance to become a CH after 10th round, may be in 11th, 12th or 13th round. After becoming a CH this value decremented by 1 in each round and when it will drop down to 0 after 10 rounds then it added to the group of nodes which have not become CH in last 10 rounds. So to make a malicious node CH an adversary needs to keep this value 0. In this way a malicious node always remain in the group of nodes which has the chance to become a CH in a current round.

Second thing is the selection of random number which is compared with calculated value of $T(n)$ in equation 2.1. So an adversary choose the value closer to the 0 but not zero and make it static i.e. random number generation function not be executed in malicious node. LEACH compare this value with its threshold value $T(n)$, it will select this node as CH.

Next thing is to launch the Vampire attack with the malicious node. Stretch Vampire attack can be launched on LEACH protocol. Setup phase is really a critical phase in LEACH protocol. Most of the attacks occur on this phase. CH selection and cluster formation involves lots of steps and functions. So an adversary mostly consider a CH node to launch attack because it alters the function of CH and make whole cluster malfunction. To launch Stretch vampire attack a malicious node receive (HEAD_Adv_Msg) messages from other CHs, which include their node ID and network coordinates. It sends its (HEAD_Adv_Msg) with node ID and coordinates of any other CH which is away from cluster nodes. But it broadcast this message with strong signal strength and the nodes receive this message with good RSSI value. Victim nodes associate with wrong CH and send it (JOIN_Clu_Msg) so that it allocate time slots for data transmission. Now the route of data packet has been extended to other CH. The purpose of this activity is cluster energy drainage. When the setup phase completed, nodes are ready to send their data to their associated CH. Each node calculate distance between associated CH and itself and based on this calculated distance it transmit data with the energy required to cover the distance. More the distance more the energy required.

3.4 Scheduling Attack on LEACH protocol

Basically scheduling attack is a DoS attack but it works also as an energy drainage attack [35] when CH make irregular schedule for its cluster nodes. This attack also occurs in setup phase when CH makes TDMA schedule for its nodes. Generally in scheduling a malicious node assign same time to every node within its cluster. So all the nodes send their sensed data at the same time and it leads to packet collision. But the other form of this attack is to alter the TDMA scheme so that it would not let the nodes in sleep mode. It can be done by increasing the time limit from its normal limit. This extension of time slots causes slot overlapping. Due to the over extension of time limit the nodes remain in active mode for longer time whether they have transmitted their data to the CH. This attack may be would not effect in a single round but when the nodes get associated with malicious node every time then it drain network node energy abnormally. TDMA scheme basically designed to avoid packet collision and save energy [38]. But if an adversary assign longer duration of time for the slots allocated to the nodes then it remain active without any purpose.

3.5 Gray-Hole Attack on LEACH protocol

Gray-Hole attack is a sort of DoS attack that has similar effect on the LEACH protocol by declaring on a node acting as CH for the other nodes. When the CH receives the data packets, it discards to forward selected data packets to the BS by dropping them. In Gray- Hole attack, malicious nodes try to stop the packets in the network by refusing to forward or drop the packets passing through them for considerable periods [39]. There is a difference between Gray-Hole and Black-Hole attack, in Gray-Hole during the packet transmission phase, it drops the packets for some specified interval of time.

3.6 Network Operational Model

Here, we provide a brief overview of the operational model of LEACH protocol.

LEACH algorithm operates in two phases, first is Setup phase and second is Steady-State phase.

3.6.1 Setup Phase

Setup phase is further divided in three sub-phases.

Cluster Head Selection

In first sub-phase, a random number is chosen between 0 and 1 by every applicant node. For example a number 0.05 has selected by a node. Then this number is compared by already calculated value we call it threshold value $Pt(n)$. If this threshold value $Pt(n)$ is greater than selected number, then that nominated node will be elected as a CH for current round.

$$Pt(n) = \begin{cases} \frac{k}{1-k(r \bmod \frac{1}{k})} & \text{If } n \in Gt(n) \\ 0 & \text{otherwise} \end{cases}$$

here k represents the probability factor of a node for being selected as a CH; r represents the current round of LEACH protocol; $Gt(n)$ are nodes in the network that has not been selected as a CH in last $1/k$ rounds, and $Pt(n)$ is a probability factor for a node to get selected as CH. The above equation makes sure that nodes that have become CHs in the last $1/k$ rounds will not become CH in the running round. The node immediately declares its present CH status to the other nodes in the network. (HEAD_Adv_Msg) is an advertisement message broadcasted by the elected CH. In this message CH node send its ID and its geographic location.

Cluster Formation

Other nodes which are not CH receive the HEAD_Adv_Msg and then send a (JOIN_Clu_Msg) to the CH for which it has the maximum received signal strength. This message include the CH's ID and the node's ID.

Schedule CDMA and TDMA

After the organization of clusters in the network, TDMA time slots are created by each CH and allocates them to each associated node in the cluster. Each CH also selects a CDMA code that it will use to forward sense data to the BS.

3.6.2 Steady-state Phase

CH nodes allocate TDMA time slots to their associated cluster nodes and these sensor nodes start sending their sensed data within their allocated time slots. This sensed data is received by respective CHs nodes within their clusters. It is the responsibility of CH nodes to forward this data to BS by aggregating it using data fusion. After that the protocol will again start the setup phase for a new round. Figure 3.2 depicts these two phases workflow [10, 11].

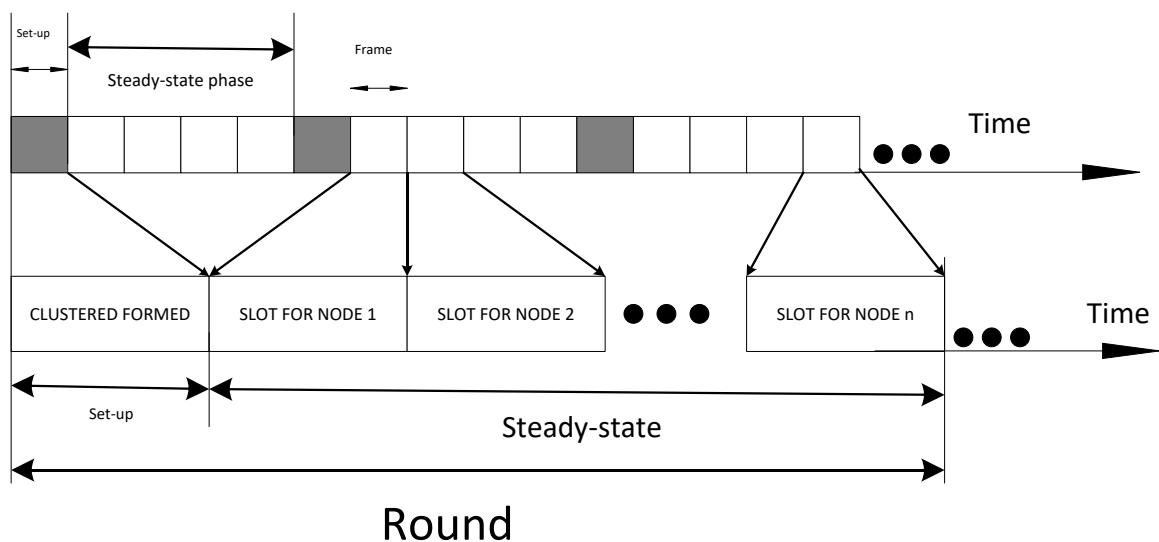


Fig 3.2

3.7 Network Attack Detection Methods

The main objective of proposed attack detection mechanism in the sensor network is to detect abnormal energy drainage in the network. The next task then is to isolate the malicious cluster head node.

In conventional LEACH protocol at the start of the round, each node selects a random number ranges 0 and 1. This number is compared with threshold value $T(n)$ in equation 2.1, if the selected number is lower than this threshold value

the node is elected as a cluster head for the current round. Unlike conventional LEACH protocol our proposed algorithm elects CH based on the trust value, which depends upon some other parameters like energy consumed by the CH in the past, distance between a node and CH and residual energy of the elected CH.

Trust on any entity is always gauged by observing its behavior and performance on some assigned task. In our proposed model we calculate trust value by observing the node behavior and performance. For this purpose we use cellular automata to keep track on behavior and performance of the nodes. Cellular automata is primarily used for originating neighborhood around a particular node and exchange information between nodes. We use this information to calculate trust on the neighboring nodes. In this proposed solution we did not alter anything in the conventional LEACH protocol. We just integrate cellular automata function in this protocol.

At the start of the first round each node make their own neighborhood within a geographical area. This area is calculated with the following equation.

$$T(n) = \begin{cases} \frac{popt}{1-popt(rmod\frac{1}{popt})} & \text{if } n \in G \\ 0, & \text{otherwise} \end{cases} \quad (2.1)$$

Each node have a number of nodes in its neighborhood. We call it neighborhood vector. At the start trust value is 1 for all the nodes, i.e. all the nodes have equal trust on each other. Energy value of all the nodes is also equal as we have supposed in our architectural model. With the help of cellular automata nodes exchange information like its energy value, location coordinates and some queries in case of malicious behavior. We discuss about these queries later.

At the start of a round, every node monitors its own energy value and shares this value with its neighboring nodes. In the case of energy drainage attack a node will notice that its energy value is draining faster than normal. We will discuss normal pattern of nodes in Section 4. It will then check energy consumption trend of its neighbor nodes. If it observes that there is an abnormal rise in energy consumption within its neighborhood, it will stop communicating with their associated CH and wait for the next round. In the next round new CHs will be selected, nodes exchange information about their status and keep monitoring their energy values. In the new round, nodes will not select that particular node as a CH which was causing energy drainage for them. If the attack continues to drain energies of the network and nodes energies fell below some threshold value say ϵ_2 then the nodes stop

communicating with the network. The purpose of ending the communication of the victim's nodes is to save them from more energy drainage. When the nodes stop communication it will send a mayday message to the base station and they share node ID of the CH with which they were associated. This model is associated with detection of energy drainage attack in LEACH protocol. Detail explanation about detection of energy drainage attack in network will be discussed in Chapter 4. To make secure LEACH protocol from energy drainage attack we also propose prevention technique for energy drainage attacks. Chapter 5 includes all the detail regarding prevention scheme.

Chapter 4

Detection of Energy Drainage Attack on LEACH protocol

A biological cell is simplest form of living organism but multicellular organisms are the complex form of life that is formed with association of cells. In this way a multicellular system is formed with several replicas of cell and these cells interact with each other to make a comprehensive behavior [10]. Cellular Automata (CA) is a simplest form of cellular system [40]-[41]. It is firstly introduced by Neumann, V and Ulam [42]. A CA is a frame of cells, each cell has number of discrete states and changes its states by following some transition rules. In each time step each cell takes some input, applies transition rule on it and update its state according to the rules applied. A CA architecture consists of four sets (L, S, N, f) , L is the working space of the cells, $S = \{0, 1, \dots, s - 1\}$ represents the finite states of each cell, number of neighboring cells for each cell is denoted as N , transition functions [42] for each cell is denoted as f . These functions are deterministic in nature and give the state $S_i(t + 1)$ for i th cell as a function of the states of the cells in neighborhood N_i at time t , which can be represented as follows,

$$S_i(t + 1) = f(S_j(t) : j \in NS_i) \quad (4.1)$$

In this model we consider only three states. Active State, Alarming State and Isolation State. So $S = \{11, 01, 00\}$, the number of states on each cell is 3. 11 represents active state, 01 represents the alarming state and 00 represents isolation state [26].

4.1 Formation of Neighborhood

Initially a node has its energy information, location information and sensed data. When we program the node to work in CA environment, in the first round each node determines its neighborhood. It applies basic distance formula to calculate distance between nodes. Each node determines its neighborhood in a radial form. It applies formula to calculate circle radius R .

In our simulation we have set the dimensions of the field in $X = 100 \times Y = 100$ points. As shown in Fig 4.1

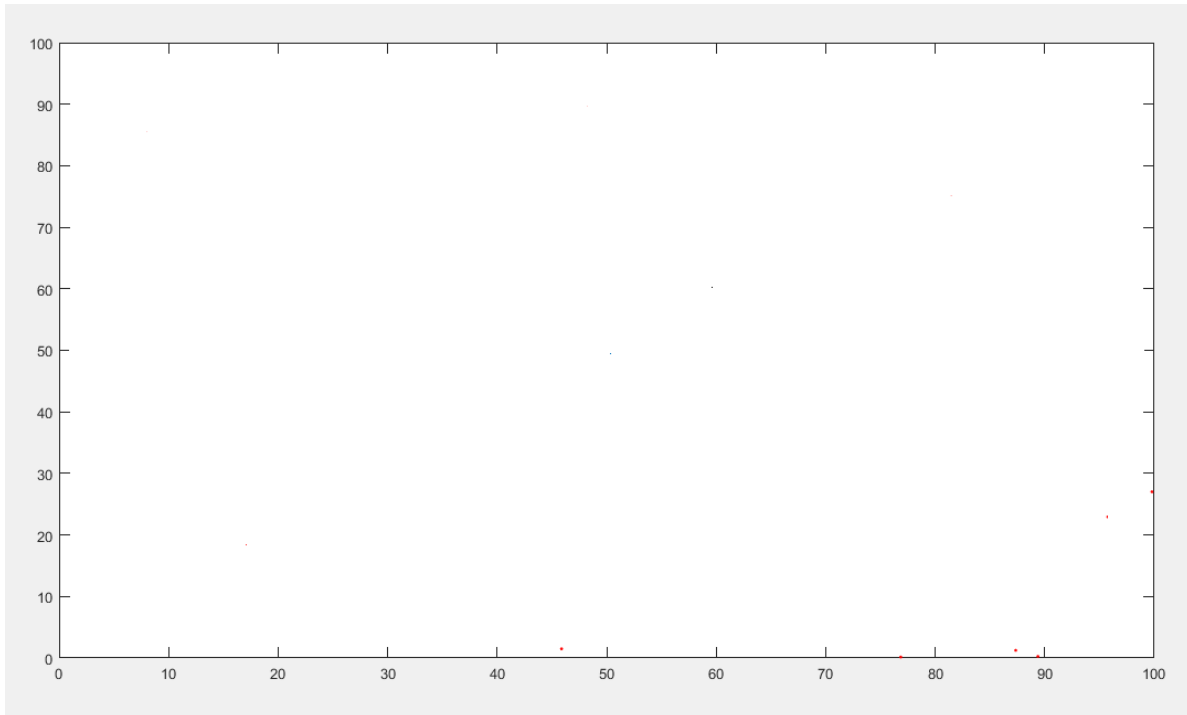


Fig 4.1

We select the value for $R=3$. It is the optimum value for the above field. Value greater than 3 covers extravagant area for a single node. Area of circle or we can say area of neighborhood is $A = \pi R^2 = 28$, if $R=3$. Fig 4.2 below shows a node calculates its area and which nodes lie in this area become its neighborhood nodes.

Each node calculate its neighborhood area in this way. The number of nodes in neighborhood for each node is called its neighborhood vector. Density of network and covered area for an individual node also effects neighborhood vector value. As you can see in the Fig 4.3 the nodes in green color become neighborhood nodes of a node in red color.

At the start of the 1st round each node performs area building process and share its energy value, ID, coordinates and trust values. Each node keeps this information in its local CA table. If table has no value then node started area building process [26]. We will show CA table formation process in Chapter 6, where we will explain simulation and results.

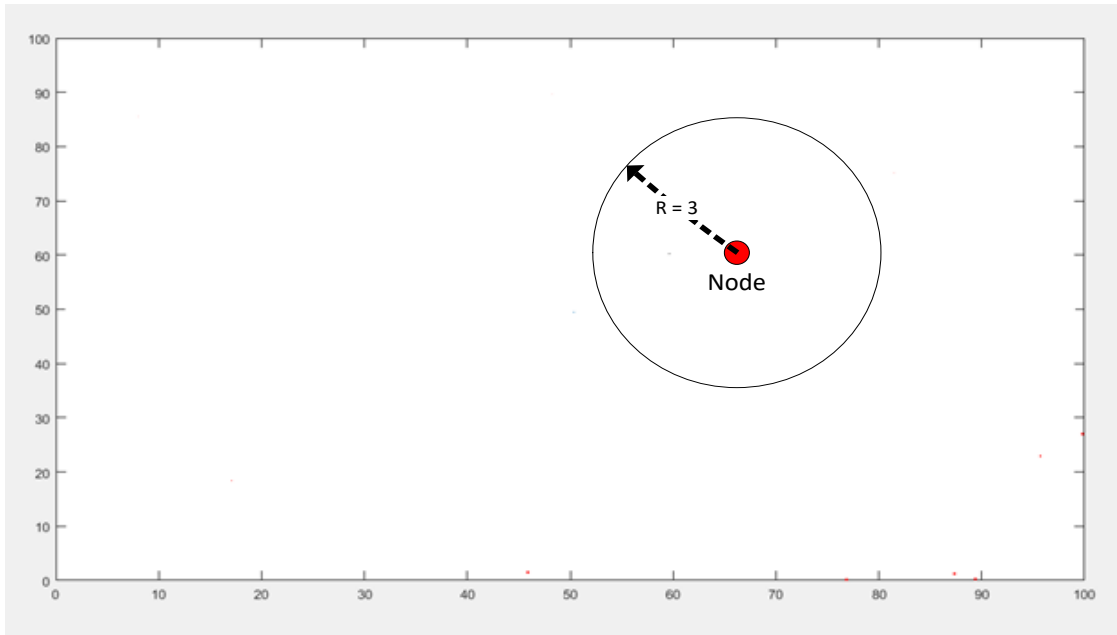
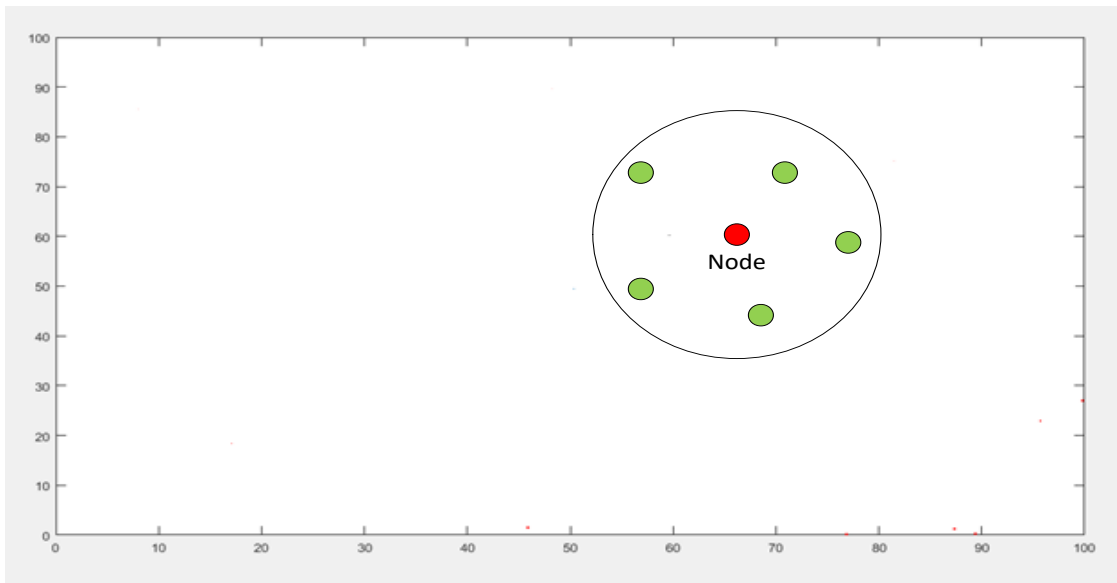


Fig 4.2



Node

Neighboring
Nodes

Fig 4.3

4.2 Protocol Stack and Information Sharing between Nodes

Little modification required on protocol stack by adding CA processing unit in protocol. CA processing unit [10] updates neighboring nodes states, control event-driven mechanism, and understand state transition rule [26]. Modification

in protocol stack is shown in Fig 4.4. When nodes start updating their CA information table and exchange information, routing packets contains this information. CA information contains node ID number, type of information, residual energy, and some trust information of neighboring nodes. After updating their table, nodes apply rules of the state transitions for switching their state.

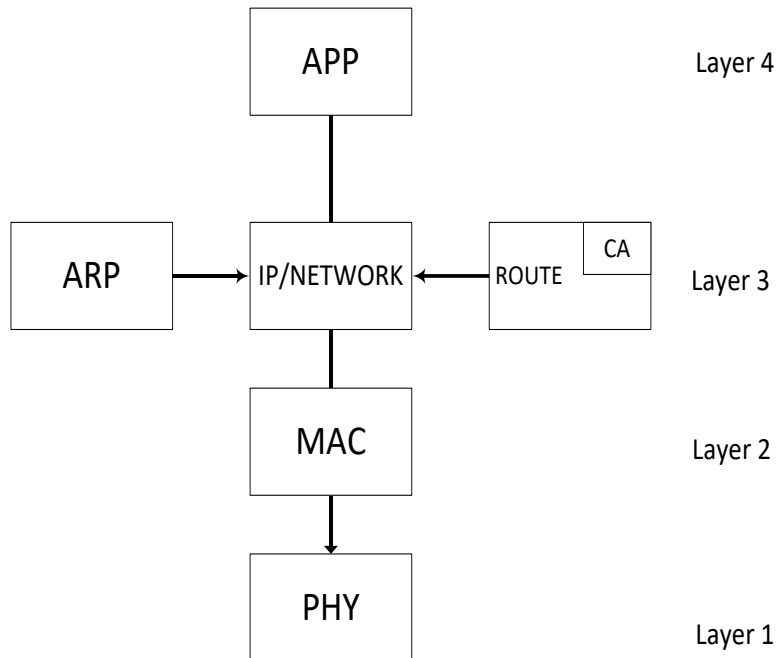


Fig 4.4

In above figure CA packets are exchanged at network layer. Nodes exchange information with its neighboring nodes by receiving CA_start type packet. Then a node sends its updated information to all its neighboring nodes with CA_response packet.

4.3 Self-Monitoring Nodes

In this energy drainage detection model each node keeps monitoring its energy status. For this we could maintain a small database in the memory of a node. In which we could say that in a regular LEACH round a node usually send maximum x number of packets if it is behaving normally. But before that we have made some assumptions, that sensor nodes are not mobile and randomly distributed [43]. CH and base station node coordinates are known, dimensions of sensor fields are known and all nodes have equal amount of energy.

Initially, each node shares its residual energy E_r with its neighboring nodes. Normally in each round a regular node sends 20 data packets. We also set 5% to

10% threshold value for normal packet loss depends on distance between a node and CH. We have calculated energy dissipation per round for a single node in Table 4.1. Nodes keep monitor their energy values after transmitting their data in their TDMA time schedule. If a node detects it has drain more energy than usual it checks how many packets it has transmitted in current round. If this value show abnormal packet transmission then it will make query to its neighboring nodes by sending CA_Start message at the end of round. Basically self-monitoring of nodes depends energy dissipated in each round by packet transmission. We have set two threshold level for energy, ϵ_1 and ϵ_2 . These two values are used to change the state of the node. As you can see in the Fig 4.5 if the energy of a node falls below ϵ_1 , then it will go into the alarming state and if this level falls below ϵ_2 then it will go into the isolation state.

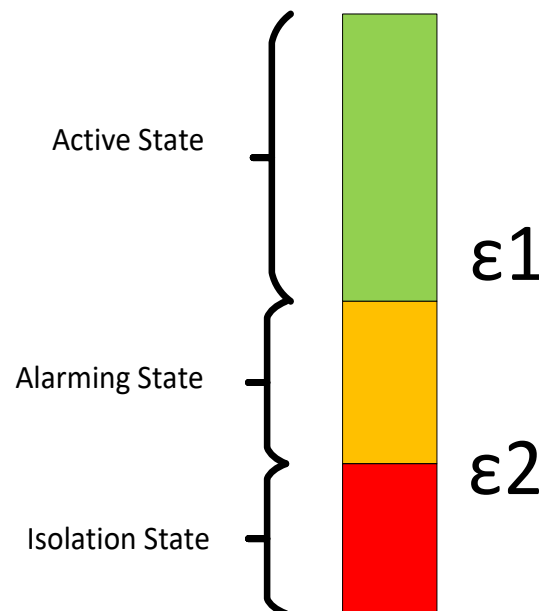


Fig 4.5

But this transition of states also depends on the energy values of the neighboring nodes. Node checks its energy value after an interval. In next section we will discuss on what parameters a node transit its state and how CA helps a node to detect an energy drainage attack.

4.4 Normal and Abnormal Packet Transmission Detection

We perform abnormal packet transmission detection test on MATLAB. It is a numerical computing environment developed by MathWorks [44]. We assumed some parameters to simulate this test which are given in the table below.

We simulate this environment in 100 m² field with randomly deployed 100 sensor nodes. Base station is situated in the center of the field.

Nomenclature	Symbol	Value
Number of nodes	n	50
Initial Energy	E_0	0.5J
Energy used by the electronic circuit to transmit or receive the signal	E_{tx}/E_{rx}	50pJ/bit
Energy used by the amplifier to transmit at short distance	E_{sd}	10pJ/bit/m ²
Energy used by the amplifier to transmit at long distance	E_{ld}	15pJ/bit/m ²
Data packet	K	20 /round
Size of each data packet	P	200 bits
Bits transmitted to CH by a node in each round	B	4000 approx.
Energy used by a node to transmit a bit to CH at short distance	$E_{tx} + E_{sd}$	50 + 10 = 60pJ/bit/m ²
Energy used by a node to transmit 20 data packets in each round at short distance	$B * (E_{tx} + E_{sd})$	4000 * 60 = 240000 pJ/bit/m ² or 240 nJ/bit/m ²
Energy used by the node to transmit a bit to CH at long distance	$E_{tx} + E_{ld}$	50+15=65pJ/bit/m ²
Energy used by a node to transmit 20 data packets in each round at long distance	$B * (E_{tx} + E_{ld})$	4000 * 65 = 260,000pJ/bit/m ² or 260nJ/bit/m ²
Data Aggregation Energy	E_{da}	5pJ/bit/report
Area	$X_m \times Y_m$	100 x100m

Table 4.1

We have considered that during each round a node sends 20 packets to CH. If each packet is 200 bits long then there are approximately 4000 bits sent per round. To deliver more accurate simulation results, natural loss of packets between nodes has also considered [45].

In the table 4.2 we can see that if a node is not a CH then it will consume 260 to 300 nJ/bit/m² in each round to transmit 20 packets. If the CH drops these packets 10 times for each node and sends drop packet message to the nodes then it will take more energy to retransmit data packets.

In LEACH simulation we also assume that all the nodes in the network experience same packet loss. There are some measurements in table 4.2 for the packet loss depending on the distance between CH and nodes [45].

Distance(m)	Min (%)	Max (%)	Average (%)
10	0	3	0.57
20	0	5.59	1.60
30	0	4	0.80
40	1.50	7.50	4.10
50	2.50	21.10	9.40
60	2.80	11.10	5.50

Table 4.2

In our detection model nodes keep track on their energy level so they need to check the value of number of packet transmitted and energy drained during this transmission. Suppose a node has transmitted 10 packets in continuing round then it let consume 120nJ energy for short distance and 130nJ energy for long distance. We also assume that packet loss ratio is more for longer distance. We added 10% packet loss ratio for long distance nodes from CH. So in that case a node consume 270 to 300nJ energy for normal packet transmission in one LEACH round. The payload size may also be vary but for dynamically changing environment energy values will also vary. Residual energy value also depends on completed number of rounds and how many time a particular node have become a CH. When nodes checks its energy and found that its energy is getting drain rapidly due to over transmission of data packets. It will make query from its neighboring nodes whether their energies are also getting drain with same speed. It will become more serious when a malicious node is becoming CH in every round. We set the energy values for the nodes, how much energy a normal behaving node should consume after particular number of rounds.

4.5 Detection of Attack and Change of States

Our primary objective is to detect energy drainage attack with the help of CA. In previous section we try to illustrate how a malicious node can drain energy of the cluster. Every node will compare its energy value with number of packets it has

transmitted in certain number of rounds. For example if a node consume 260 to 300nJ energy in a single round then in coming two to three rounds it will consume 600 to 900nJ energy. We configure a node to check its energy states after two rounds. It will check its energy consumption in last two rounds and number of packets it has forwarded to base station. In normal case it will consume 500 to 700nJ energy with maximum threshold value. But if it has consumed more energy and not become a CH in last two rounds then it will make query to the neighboring nodes. If the neighboring nodes experiencing with same situation then it will check how many neighboring nodes are under attack.

Nodes send these query messages on IP layer as we have discussed in section 4.2. After updating their CA information table nodes take decisions about their states. But a single node cannot take their decision at its own. It should follow the states of its neighboring nodes. In a cluster if more than 70% of nodes experience that they are draining their energies swiftly then they will go into the alarming state. In this state they will not communicate with CH till next round. In the next round if they will experience same problem and energy values drop below ϵ_2 , nodes will go into the isolation state and will not communicate with the network anymore. They will send mayday message to the base station with their node IDs to tell the base station how many nodes stop communication in network.

Change of state depends on the state of the neighboring nodes. If a node i checks its neighboring node states that has been changed to alarming or isolation state in $S_j(t)$ where $j \in N$, then it change its state in $S_i(t + 1)$ according to its neighboring nodes. A node checks its neighborhood states and if more than 70% of the nodes have changed their states at time $S_j(t)$ then it will change its state accordingly at time $S_i(t + 1)$. Nodes also share their change state information when CA table is updated. Detection of attack involve isolation of nodes from the network, when nodes detects that they are associated with malicious CH, they stop communication with it and if they will experience same behavior in upcoming rounds, they isolate themselves from the network. A database with state information is used to define a normal behavior and then we used it to detect an abnormal behavior. Detection of abnormal behavior triggers corrective actions which involve change in state and a Mayday signal as a last resort.

One of the problems is the malicious cluster head, and in the next chapter we will suggest a solution to avoid selection of a malicious cluster head.

Chapter 5

Prevention Scheme for Malicious Cluster Head Selection

Trust is the factor which we used to prevent nodes to select malicious CH. Nodes calculate trust on its neighboring nodes based on energy consumption. To ensure that sensor nodes select benign node as a CH, trust value is considered during selection process. Trust determines level of assurance for a node N_i on neighbor vector N_j depending on the performance of node N_i as a CH in the past. The trust can be assessed from energy consumed by a node and trust values suggested from the neighboring nodes [46].

5.1 Calculation of Trust

Trust on any other node is calculated as direct and indirect trust, and total trust will be the aggregation of both the values. Calculation of direct trust is based on previous performance of a node when it had become a CH. Indirect trust is an aggregated value based on direct trust values of the neighboring nodes for a particular node. As you can see in Fig 5.1 node N_1 wants to calculate trust for node N_2 . So it gets direct trust values from its neighborhood, aggregate them and add with its own calculated direct trust value.

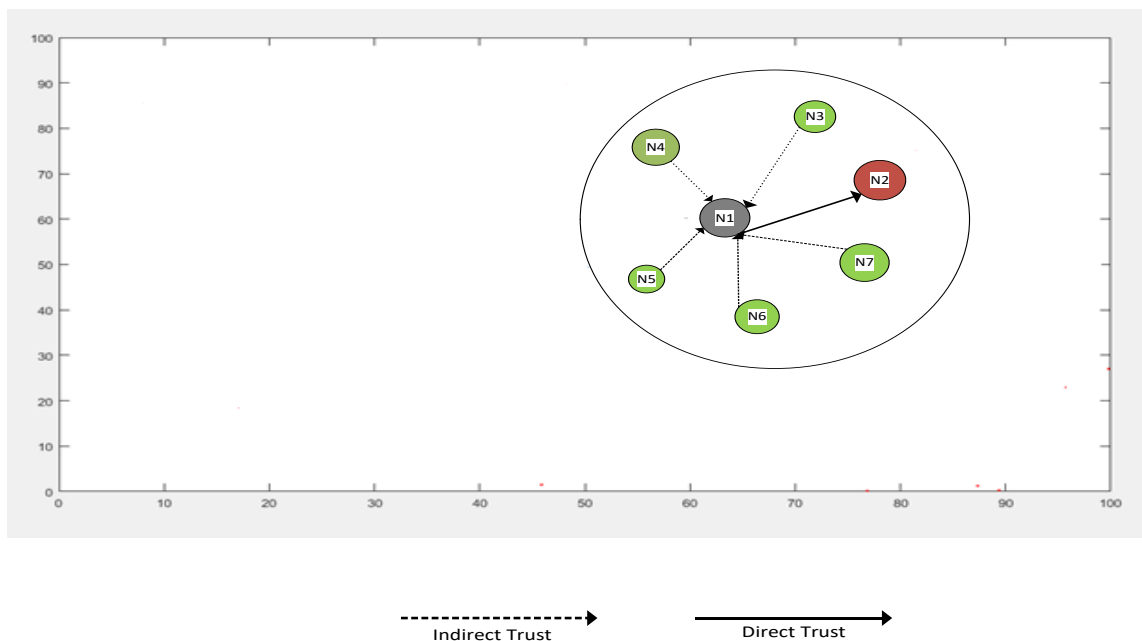


Fig 5.1

Following equations describe the calculation of direct and indirect trust values.

Suppose node $N2$ nominates itself to become a CH. So node $N1$ calculates its trust value for $N2$ as

$$Trust_{N1 \rightarrow N2} = DT_{N1 \rightarrow N2} + IT_{N1 \rightarrow N2} \quad (5.1.1)$$

Where $DT_{N1 \rightarrow N2}$ is direct trust and $IT_{N1 \rightarrow N2}$ is the indirect trust of node $N1$ to node $N2$ based on the recommendations of neighboring nodes [12].

Where:

$$IT_{N1 \rightarrow N2} = \sum_i^n DT_i \rightarrow N2 \quad (5.1.2)$$

Here n is the neighborhood vector of node $N1$ [14].

Now we see how a node calculates direct trust for any other node.

$$DT_{N1 \rightarrow N2} = 1 - T(cal) \quad (5.1.3)$$

$$and \quad T(cal) = Econ(t - 1) - Econ(t) \quad (5.1.4)$$

In above two equations direct trust is calculated based on energy consumed by node $N2$ when it had become CH at $(t - 1)$ and energy consumed by it in current round at time t . Neighborhood nodes also share their calculated direct trust value to each other, and all nodes in the cluster receive these values and aggregate them to calculate indirect trust value. At the end of round, each node calculates energy consumed by the CH and subtract it from the value when that node previously became CH. We subtract it from 1 and it gives direct trust value for node $N2$ calculated by node $N1$. All the nodes in the cluster perform this operation to calculate direct trust for current CH node and save it in its CA information table after incorporating indirect trust values [30]. When this particular node again nominates itself to become CH, then other nodes use this trust value to select this node as CH.

We have also considered some assumptions to calculate this trust value [30].

- Every node has equal resources for power and processing.
- Malicious node can drop received packets randomly.
- Malicious node does not have the ability to collude themselves.
- No node added or removed from network after deployment.
- Each node takes indirect trust value from each of its 1-hop neighbor node.

5.2 Selection of Cluster Head based on Trust value

In this model selection of CH based on three different parameters

- Residual Energy of node nominated as CH
- Distance between simple node and nominated node
- Trust value calculated for nominated node

Initially all the nodes have same trust value equal to 1. At the start of the round nodes after being elected as CH broadcast (HEAD_Adv_Msg) message to other nodes. Other nodes received this message with associated node ID. Nodes check its CA information table and find trust value regarding this node ID. If any node elected first time as a CH then its trust value should be equal to 1 for all the nodes in the cluster. Following formula is used to select CH rather Received Signal Strength Index (RSSI) value.

$$WT_{j \rightarrow i} = \alpha \left[\frac{E_{rem_i}}{D(node_j, node_i)} \right] + \beta (T_{node_{j \rightarrow i}}) \quad (5.2.1)$$

In this equation E_{rem_i} denotes remaining energy of node i in the network nominated as CH. $D(node_j, node_i)$ denotes the distance between any node j in the cluster and nominated CH node i . $WT_{j \rightarrow i}$ denotes weighted trust of node j in the cluster over nominated CH node i . It is the final trust for any node to the other node in its neighborhood. α and β are the weight value in this formula. These value decide how much weight should be assigned to the trust value.

All the nodes keep record of the trust values present in their neighborhood. In this formula node ensure that nominated CH node have maximum remaining energy, minimum distance for a particular node and maximum trust for being as a CH. It gives the optimum value for selection of benign node as a CH.

Suppose a malicious node get selected as a CH and when it drains cluster nodes energy in one round then nodes lose their trust on that malicious node. They will update lower trust value in their CA information table and share it with other nodes. When this node again nominate itself as a CH then nodes in the cluster negate it to select it as a CH due to lower trust value. We have assigned two weight values in this formula α and β , where $0 << \alpha + \beta << 1$. By controlling these values we can decrease or increase trust factor weight. For the current setting we have set α value 40% and β value at 60%, to make the trust factor weight 60% to the total value of $WT_{j \rightarrow i}$. Second important thing in this formula is calculation of distance between two nodes. Every node share its network coordinates with other neighboring nodes. So we use simple distance formula to calculate distance

between two nodes based on their coordinates. Here the distance formula for calculating distance between node N1 and N2 is [9]:

$$D(N1, N2) = \sqrt{(X_{N1} - X_{N2})^2 + (Y_{N1} - Y_{N2})^2} \quad (5.2.2)$$

In this formula X_{N1} , Y_{N1} are X and Y coordinates of node N1 and X_{N2} , Y_{N2} are X and Y coordinates of node N2. Every node calculate its distance from any other node using this formula.

Chapter 6

Implementation and Results

This chapter covers the facts about the implementation of our proposed solution against energy drainage attack on WSN and its results. We present the proof-of-concept for LEACH protocol, which fulfills the architectural and operational requirements discussed in detail in Chapter 3.

6.1 Simulation Environment

We have implemented our proposed algorithm in a simulated environment using MATLAB and Network Simulator-2 (ns-2). We also compared our proposed algorithm CAT-EDP with default LEACH algorithm and other trust based proposed algorithms namely TERP [30] and TREE-CR [9]. These algorithms [30] [9] also used trust based scheme for optimum selection of CH. Experimental results showed that our algorithm performs better in terms of energy drainage attack detection and selection of benign CH. We will explain these results in Section 6.3 and 6.4.

Firstly we have setup our environment in MATLAB and simulated its results, then we will talk about ns-2 and its results. We simulate our proposed algorithm to validate the significance of it and how it will react at different environment. MATLAB simulation environment works on application level, but it does not give packet level simulation results. On the other hand ns-2 is purely designed to simulate networks. It gives packet level details on the run time of the simulation.

6.2 Simulation in MATLAB

MATLAB is a product of MathWorks, Inc. is a numerical computing environment and a high level programming language. After downloading and installing MATLAB you need to configure it to simulate WSN. For this you need to update it with SIMULINK libraries. For model based design it is an environmental setup for block diagram and multi domain simulation [47]. It also provides graphical editor to simulate dynamic systems. In newer versions of MATLAB it is already integrated.

6.2.1 Downloading of LEACH protocol

We downloaded the source code of LEACH-V-2014 from <http://wsnlab.org/downloads/>. It is supported by wsnlab.org. It will be downloaded in your computer. Unzip it and place this unzipped file in the default

directory of the MATLAB setup like C:\Program Files\matlab\leach. Now open the file C:\Program Files\matlab\leach\start.m, it will open in MATLAB editor windows. We also downloaded the source code of LEACH protocol for ns2 from the link given above.

6.2.2 Running LEACH protocol simulation on MATLAB

As discussed in previous part to run leach simulation open start.m file as specified above and click “Run” button to start the simulation as shown Fig 6.1



Fig 6.1

In the table below Simulation parameters are listed.

Parameters	Values
Network Dimensions	100 x 100 sq-m
Initial Node Deployment	Random
Number of Nodes	100
Base Station Location	50x,50y
Initial Node Energy	0.5J
Data Packet Size	200 bits
Transmitter/Receiver Energy Dissipation	50nJ
Data Aggregation Energy	5nJ
Cluster Head Energy Consumption	70nJ
Free Space Energy Consumption	10pJ
Multi Path Energy Consumption	13pJ
Data Packets per round	20

Table 6.1

If we run the simulation for 950 rounds then we get following results.

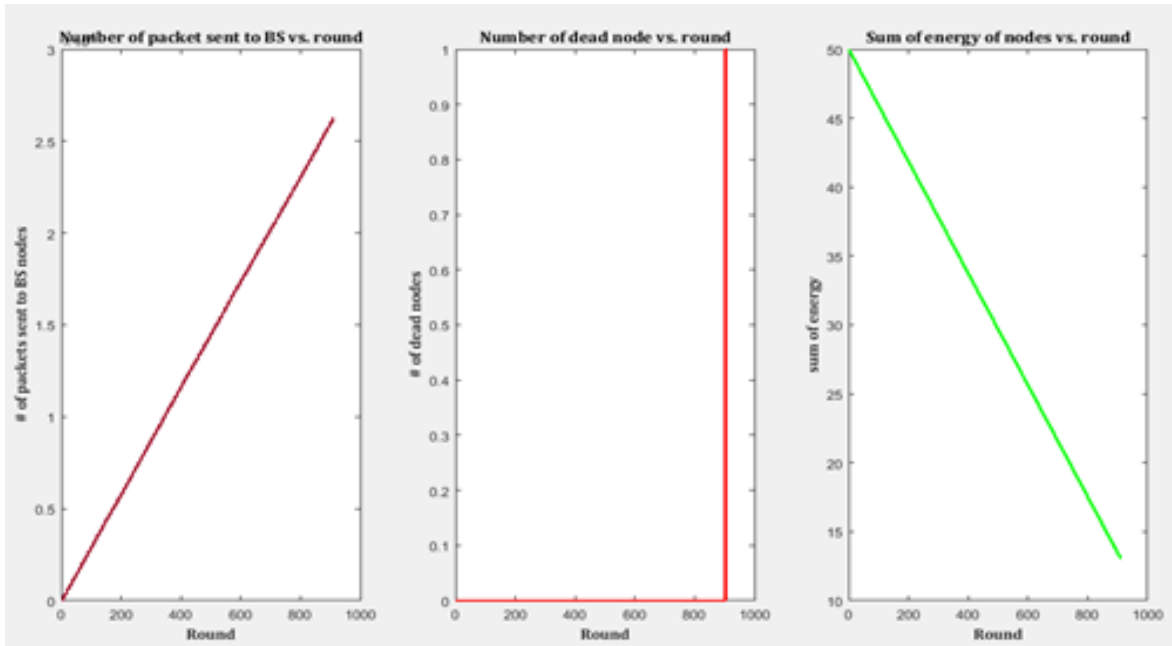


Fig 6.2

As you can see in Fig 6.2, we ran the simulation for 1000 rounds and more than 25000 packets transferred to the base station. One node died out in this duration so in this simulation network lifetime is about 920 rounds. Initially total energy of the network was 50J and it dropped down to 13J after 920 rounds.

Now we ran the simulation and plot energy consumed for whole network against number of rounds. We ran the simulation here for 1500 rounds as shown in the Fig 6.3 below:

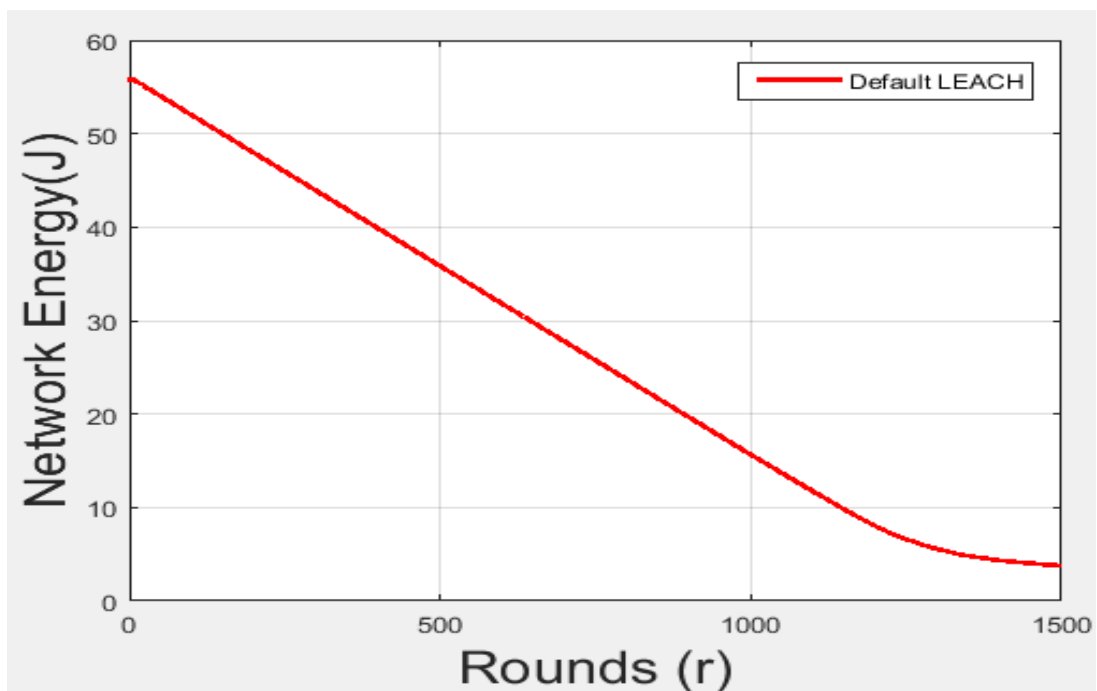


Fig 6.3

In this plot we ran the simulation again for 1500 rounds and we network energy drop down to 4J. We included this plot because our main concern with network life time and network energy consumed by the nodes in normal and abnormal behavior.

We can see in Fig 6.4 below that there is only one node that died after 1000 rounds. Base station, normal and cluster head nodes are also highlighted in this figure to show how they are behaving in this simulation. Nodes are randomly distributed over the network as discussed previously so cluster head node changes after every round.

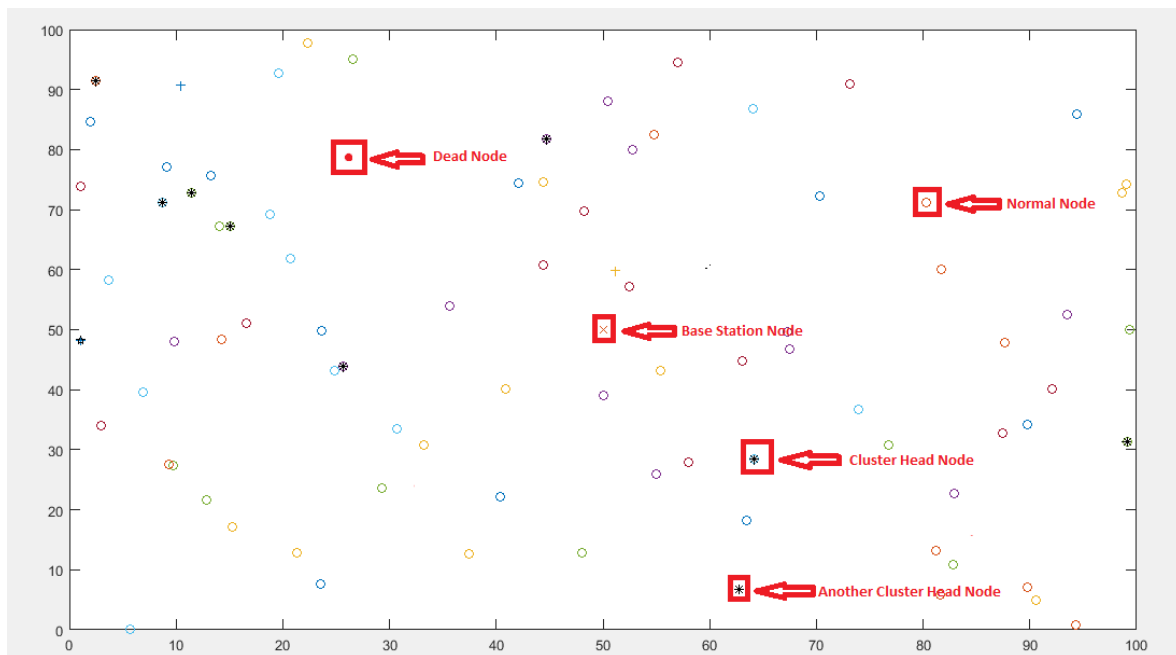


Fig 6.4

6.3 Simulating Energy Drainage Attack on LEACH

Now we simulate some energy drainage attack on LEACH protocol like gray-hole attack and scheduling attack. We have discussed about these attacks in Chapter 3.

We introduced a malicious node in deployment process of the nodes. This node nominate itself as CH in every round. The process of getting elected as a CH is explained in Chapter 3. This node consumed nodes energy associated with it in a cluster. We can see in Fig 6.4 a malicious node is introduced in the network field which is acting like a CH. We have set the same parameter for it like any other node in the network such as initial, transmitting, receiving and aggregation energy. But this node broadcast false information to the other nodes in the network like remaining energy, how many time it has become CH and number of

packets transferred to the base station. We are simulating these attacks on default version of LEACH protocol.

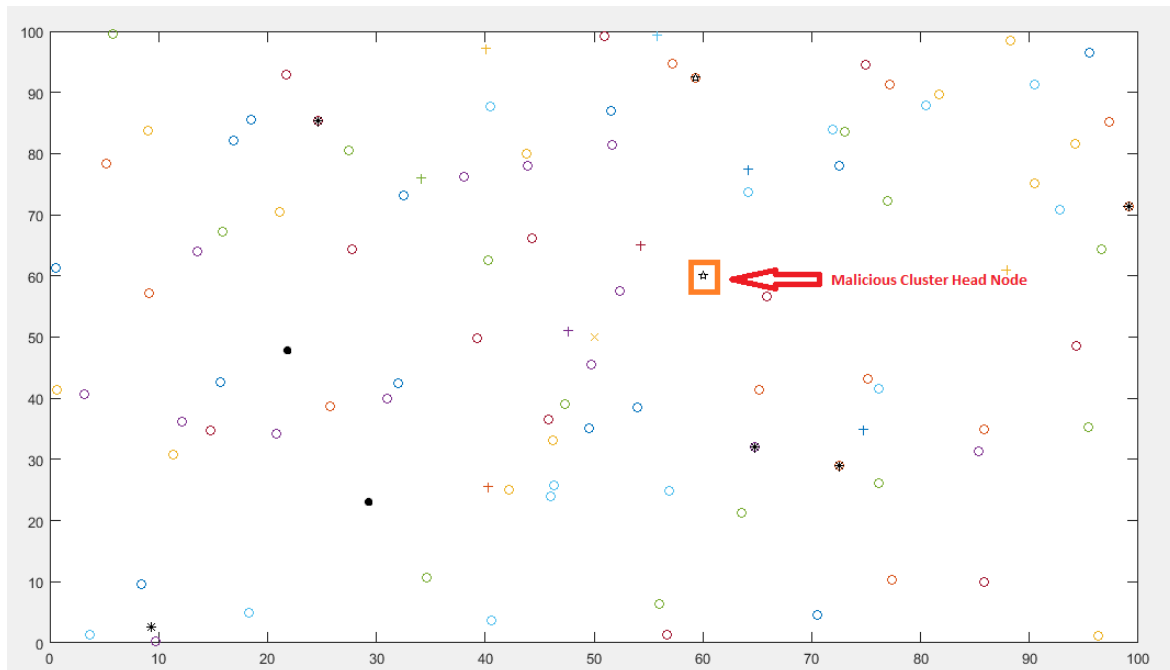


Fig 6.5

6.3.1 Simulating Gray-Hole Attack

We launch gray-hole attack in which a malicious CH node drops all the packet coming from its associated nodes and sends resend request to the associated nodes. It may not effect cluster nodes in one round but when a malicious CH gets selected in every round then it leads to denial of service attack and energy drainage attack also. We set the packet drop ratio to 60% of the packet received at the CH node. A malicious node saves its aggregation and transmitting energy because it does not aggregate and forward received packets to the base station.

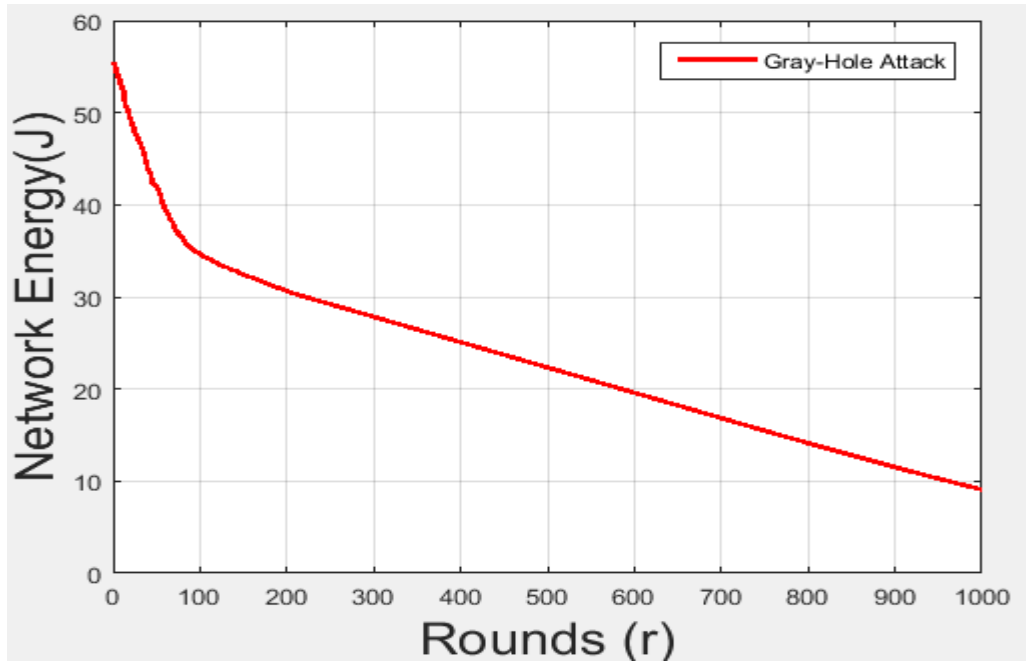


Fig 6.6

In the Fig 6.6 we plot number of rounds and network energy graph for gray-hole attack. We can see that a malicious CH node abruptly drain its cluster energy in initial rounds. When these node died in first 110 rounds then other network energy drain as usual. Whole network nodes died after around 800 rounds. There is only one malicious CH node present in the network. If there are more malicious nodes present in the network then results are more dangerous. Red dots in this figure 6.7 shows died nodes around malicious CH.

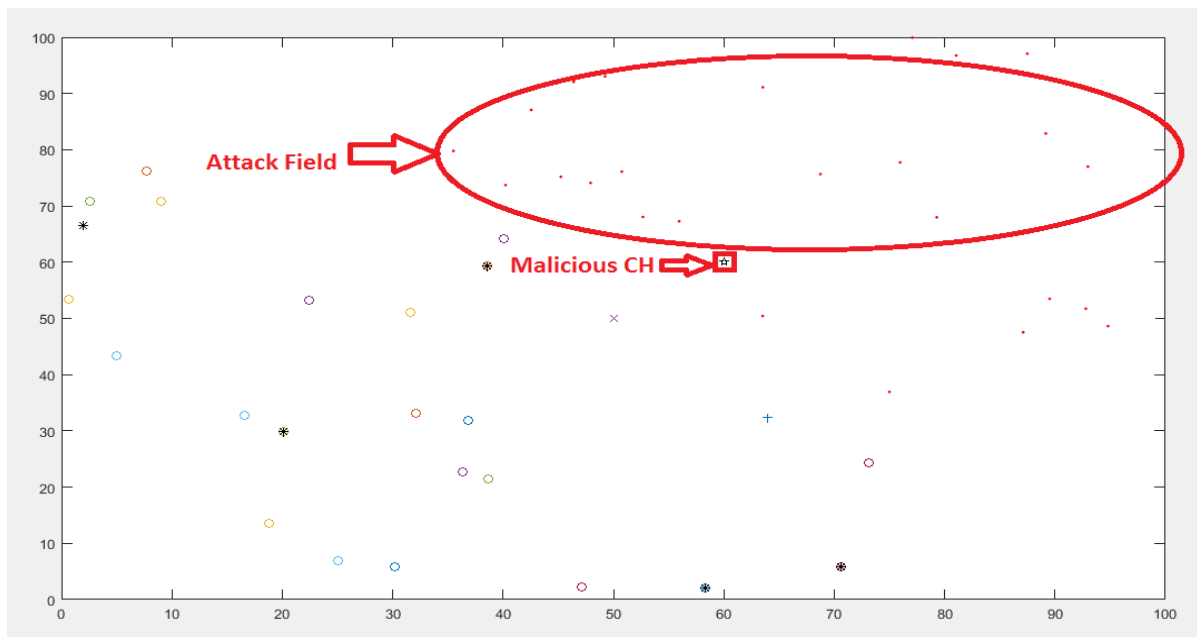


Fig 6.7

In figure above malicious node nominate itself as a CH node in every loop and drain cluster nodes energy. Other nodes in the network drain their energy normally but 30% of network nodes died in initial rounds.

6.3.2 Simulating Scheduling Attack

We simulated scheduling attack on LEACH protocol. This attack alters time schedule of cluster nodes and make collision on CH node. This attack may not effect for fewer rounds, but if a malicious CH node disturbs time schedule of nodes to transmit their data in every round then it can drain significant amount of nodes' energy.

In this attack, packet drop ratio is greater than gray-hole attack. Due to collision, more packets drop on CH node. Here packet drop ratio is 80% of the packets received. So more energy will be consume in this attack.

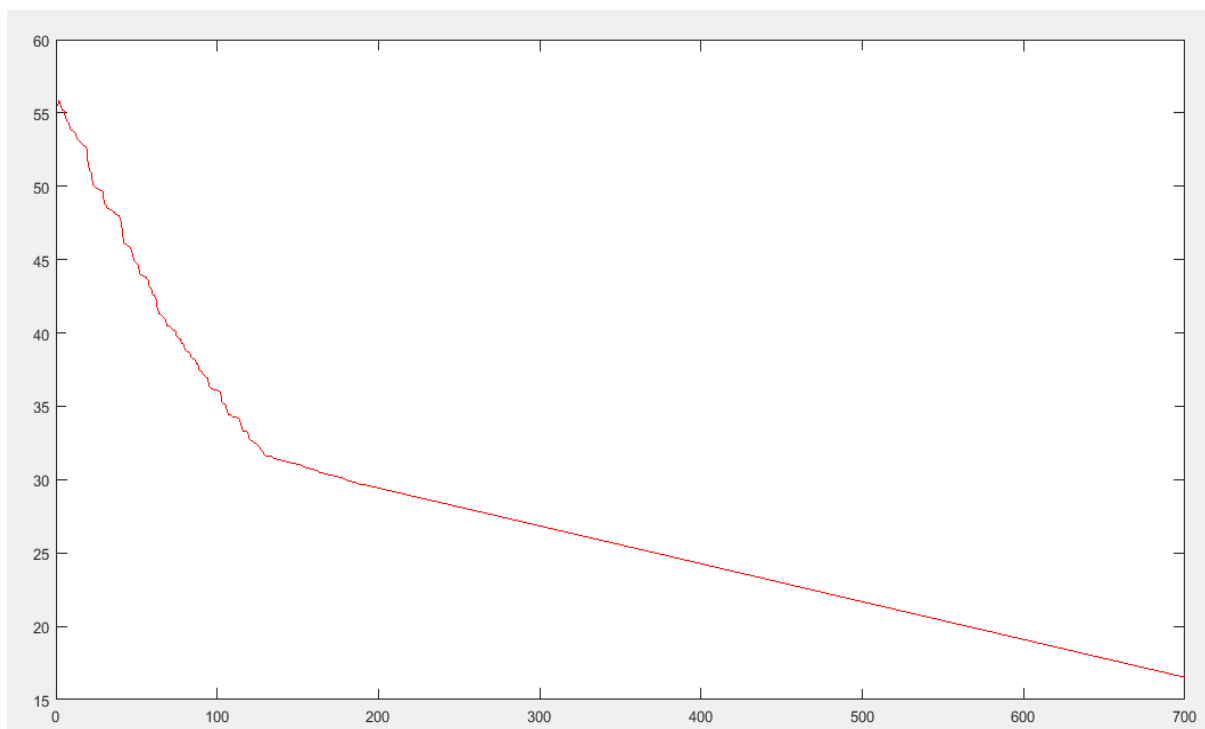


Fig 6.8

We can see in this figure that in first 100 rounds network energy drops down to 35.5J. So it has drain 40% of network energy in first 100 rounds and rest of energy drain early than usual. In this attack nodes get less time for sleep mode and most of the time they keep transmitting data packets.

If we plot the comparison between gray-hole and scheduling attack then it can be seen that scheduling attack drain more energy of the cluster nodes because packet drop ratio is higher in this attack.

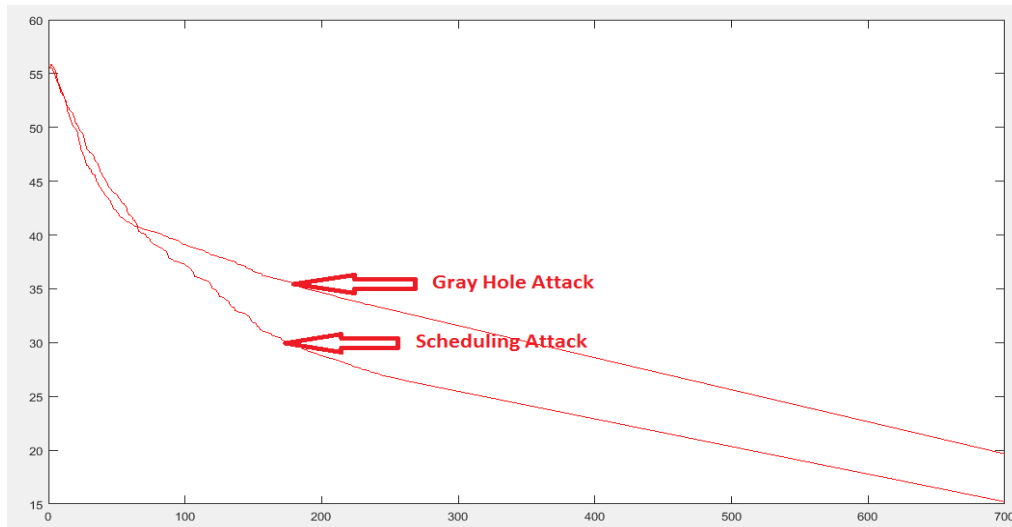


Fig 6.9

In figure 6.9 we plotted comparison between two attacks and after 150 rounds scheduling attack has drain 10% more energy than gray-hole attack.

6.4 Detection of Energy Drainage Attack

We have discussed the attack detection method in chapter 4. Detection process involve real time monitoring of nodes. Each node keeps monitoring its own energy level with number of packets transmitted to the base station. If any node experiences abnormal drainage of energy, it checks its neighboring nodes energy and if their energy levels are also draining abnormally then they switch to alarming state and stop the communication with the CH.

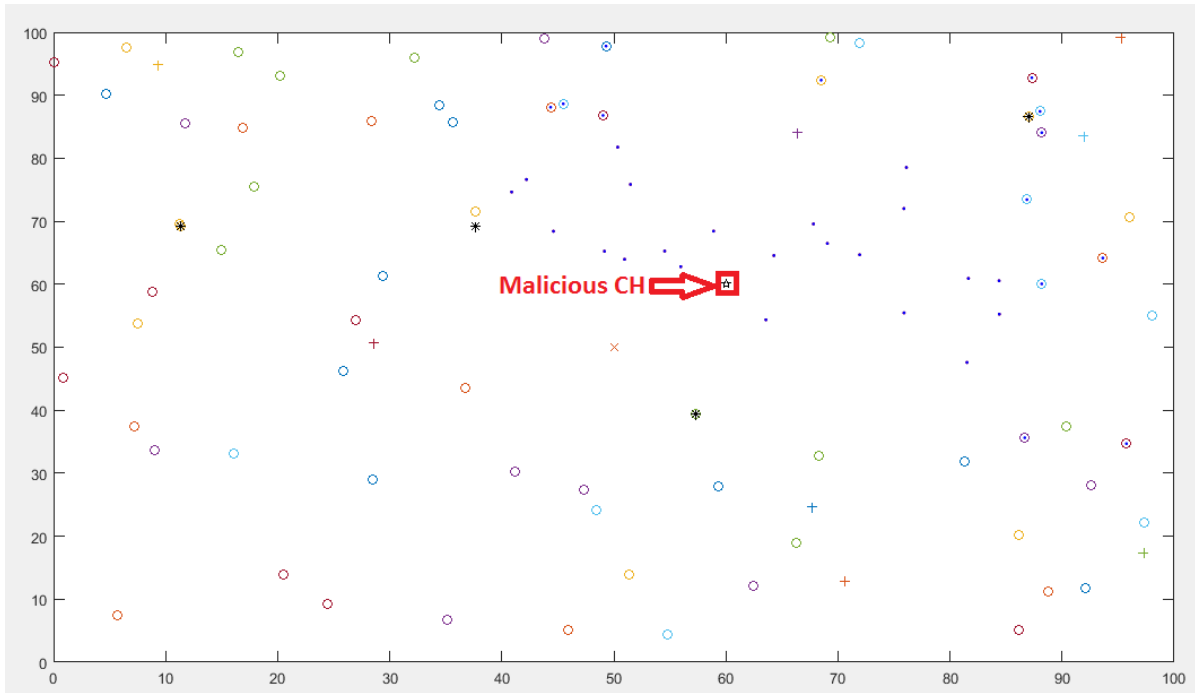


Fig 6.10

In figure 6.10 we can see that nodes around malicious CH are in blue colors. When nodes detect energy-drainage attacks in their neighborhood they change their states to 01 as discussed in chapter 4. They stop communication with CH and wait for the next round to work with the new CH node.

6.5 Prevention scheme for malicious CH selection

In this section we simulated our prevention scheme for malicious CH selection and analyze results. Prevention mechanism for selecting malicious CH has been discussed in Chapter 5 with detail. This section includes simulation results we get from our proposed mechanism. We simulate it in MATLAB [47] and set the same parameters like field area, number of nodes and their energies as we have set in Section 6.4. We again introduced a malicious node in the network and it tried to drain network energy but network nodes built trust based on the performance and energy consumption. This trust value is used for selection of optimum node as a CH. There are some simulation results we plot with malicious node in the network. There is a significant change in the network performance regarding CH selection and energy consumption. We compare our results with TREE-CR [9] and *TRUST based energy efficient routing in LEACH for wireless sensor network* [12]. They both simulate their proposed solutions in MATLAB and compare their results with benchmark LEACH protocol. If we compare our proposed solutions with TREE-CR then there is little improvement in terms of energy consumption.

But if we compare our simulation results with *TRUST based energy efficient routing in LEACH for wireless sensor network* [12] then there is a significant improvement of performance and energy consumption.

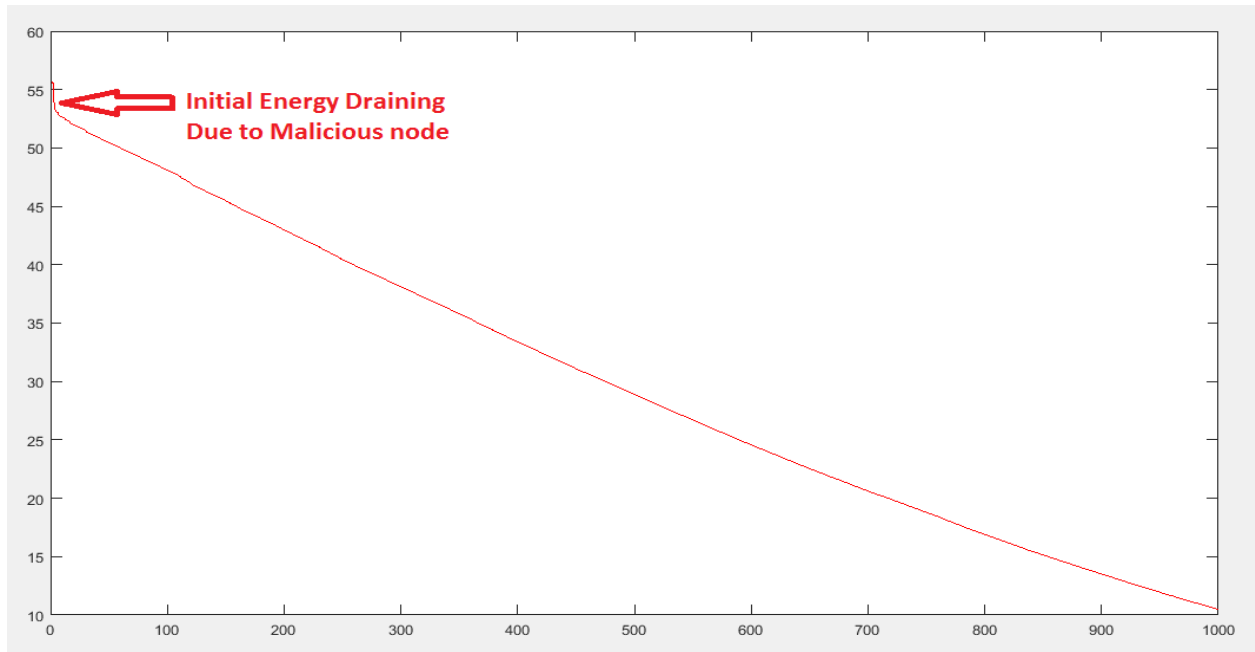


Fig 6.11

In fig 6.11 you can see constant drainage of energy of network. Initially malicious CH node try to drain network energy and some nodes energies drain abruptly. But when nodes start to maintain their neighboring nodes trust values and select CH based on these trust values, malicious node has no chance to get elected as a CH. A malicious node get lower level of trust value due to maximum drainage of energy in early rounds. This node never elected as CH whether it is nominating itself as CH in every round.

Network lifetime has also been improved. In conventional LEACH protocol first node died around 900 rounds but now it has extended its limit around 1200 rounds. Some energy consumed in CA information table update process. Nodes exchange information with their neighboring nodes and results in significant increase of network overhead. Each node donate its 15% of energy in CA information update process.

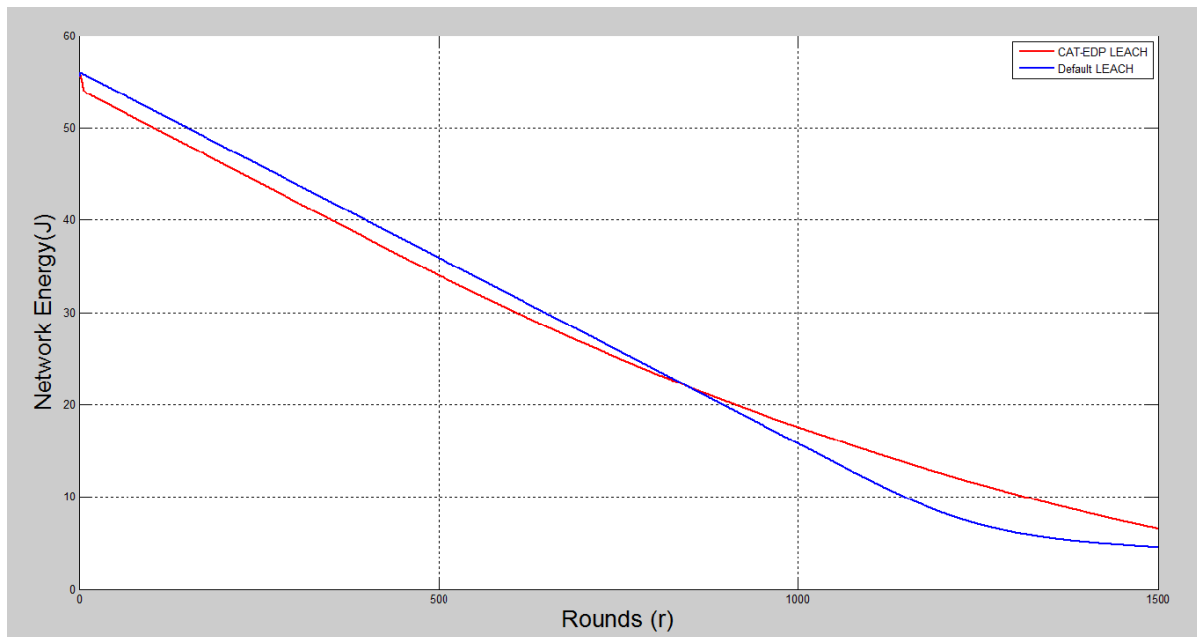


Fig 6.12

In figure 6.12 we plot comparison between conventional LEACH with our proposed LEACH protocol. You can see our proposed solution is more stable than conventional LEACH protocol. In proposed LEACH graph we plot it with malicious CH in network. That is why it drain little energy initially but after that it become stable by selecting CH based on trust values. Some energy also drain due to CA information table update process but still it gives significant improvement in CH selection process and network lifetime. Network life time has also been increased from 20-25% based on number of nodes and CH's in the network.

6.6 Simulation in ns-2

Network Simulator-2 (ns-2) is a discrete event network simulator which is basically used in research to simulate network environments. We have used **ns-allinone-2.35** version and build it in Ubuntu 14.04 LTS 32-bit machine. You can download ns-2 from the following link:

<https://sourceforge.net/projects/nsnam/>

6.6.1 Installing and Configuring ns-2 in Ubuntu 14.04 LTS

Complete installation and configuration steps of ns-2 are given in Appendix A.

6.6.2 Simulation of LEACH in ns-2

We have set the parameter to simulate LEACH protocol as given in table 6.2.

Simulation Environment Parameters

Parameters	Value
Network Field	1000 x 1000
Number of Nodes	100
Simulation Time	3600 Seconds
MAC protocol	IEEE 802.15.4
Protocol	LEACH
Node Initial Energy	2 J
Base Station Location	X =50, Y =175
Transmitter/Receiver Dissipation Energy	50 nJ

Table 6.2

Energy drainage graph for default LEACH protocol is shown below. We ran simulation for 450 rounds as shown in figure 6.20.

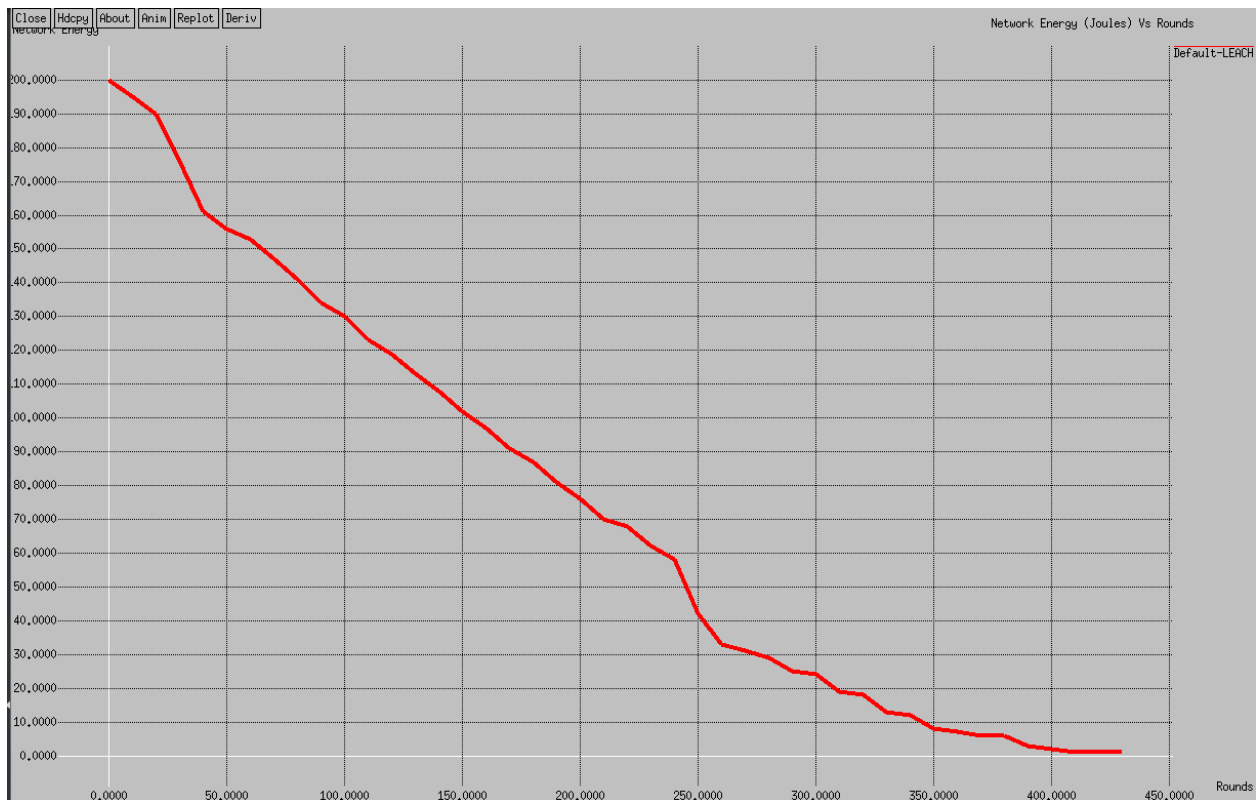


Fig 6.13

Total energy of the network is 200 j. Initial energy of each node is 2 j. After 440 rounds whole network died out. Only 4 nodes remain alive and at the end of simulation 198.90 j energy has consumed. First node died at 60 rounds.

6.7 Simulating Energy Drainage Attack in ns-2

We simulated grey-hole attack on leach protocol. First thing is to introduce malicious CH in network and it should advertise itself as a CH in every round. We set the node 101 as a malicious node in network. For this we have to set number of nodes in network equal to 101 to simulate one more node as a CH. Detail coding is implemented in source code to declare node as CH and is placed in Appendix.

After introducing malicious node in network and setting the CH to drop all the packets coming towards itself from its associated nodes, it will drain energy of the network. Energy graph of this simulated attack is shown figure 6.21

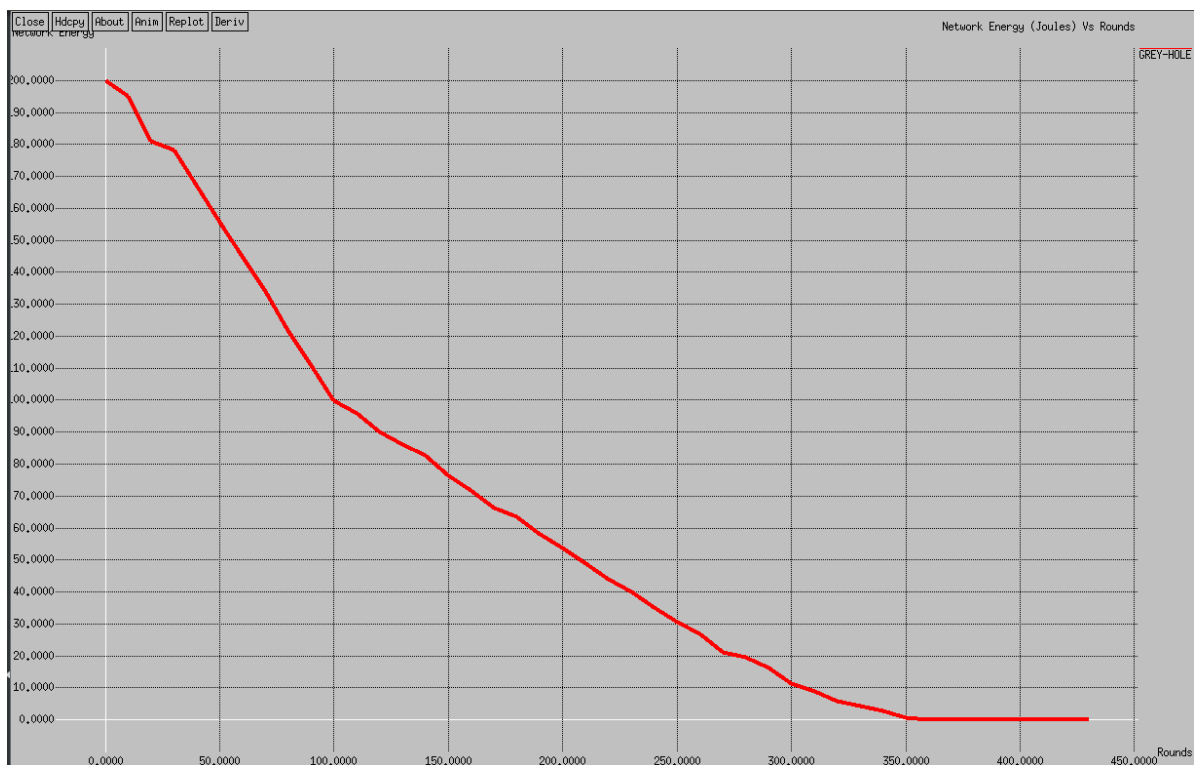


Fig 6.14

In figure 6.21 energy dropped down to 100 j in first 100 rounds. Other nodes energy fell as usual but network lifetime reduced 400 rounds to 350 rounds.

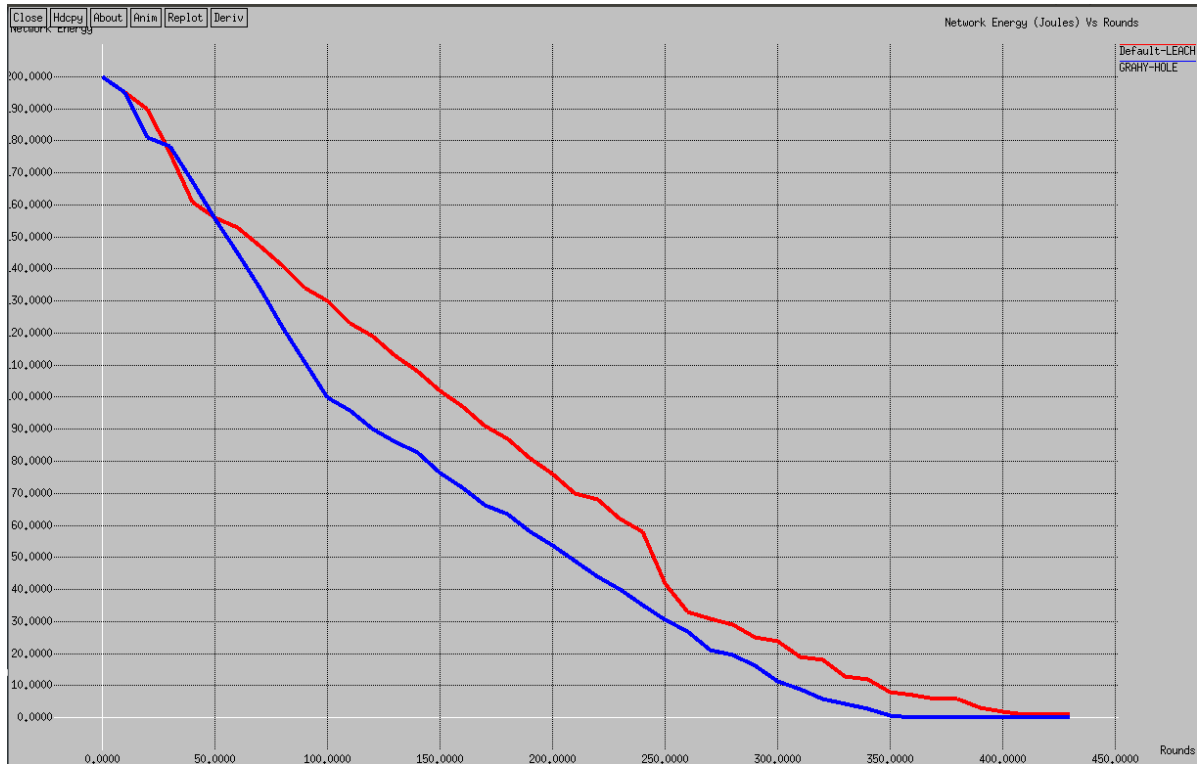


Fig 6.15

In figure 6.22 we plotted comparison between default LEACH protocol and Grey-Hole attack on LEACH. Due to one malicious CH node network lifetime decreased significantly. All nodes died around 350 rounds because 12% of network nodes died early.

In figure 6.23 we plotted Scheduling attack on LEACH protocol. In this attack packet drop ratio is higher than Grey-Hole attack as we discussed in Chapter 4. Half of the network energy drained in first 92 rounds and whole network died in 320 rounds. So network lifetime reduced from 400 to 320 rounds.

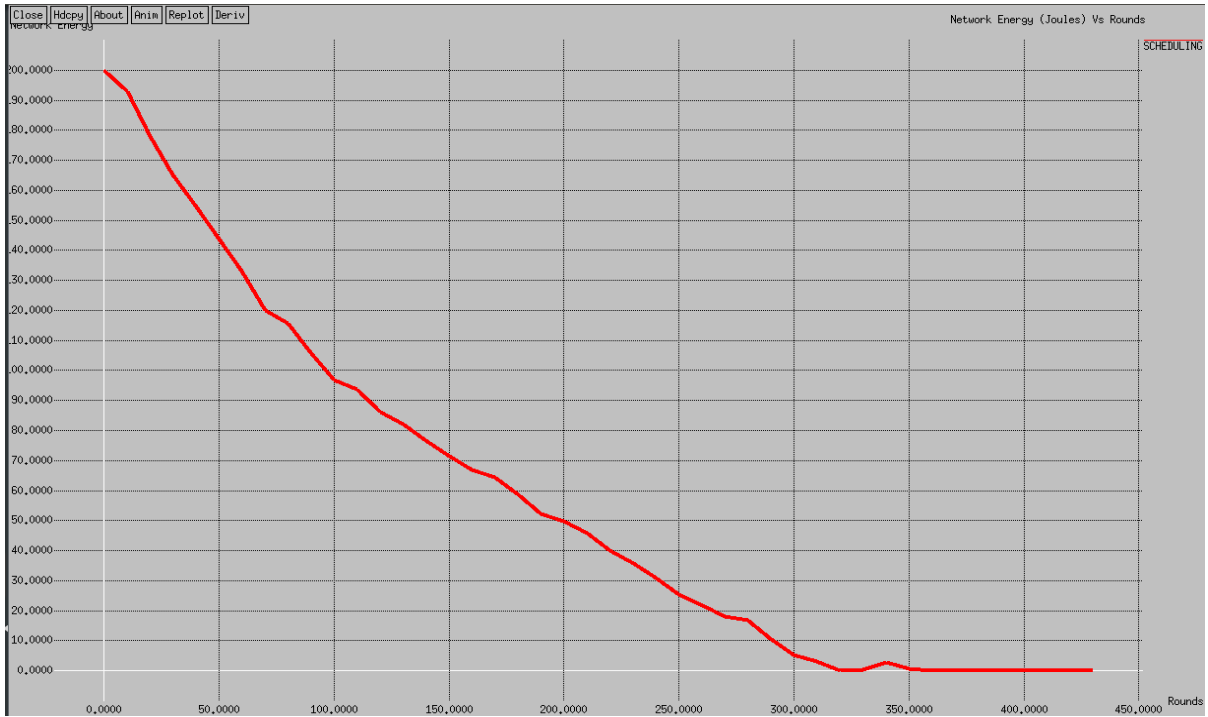


Fig 6.16

Figure 6.24 shows energy drainage comparison between default LEACH and Scheduling attack on it. This figure is the evident of prominent effect of Scheduling attack on LEACH protocol.

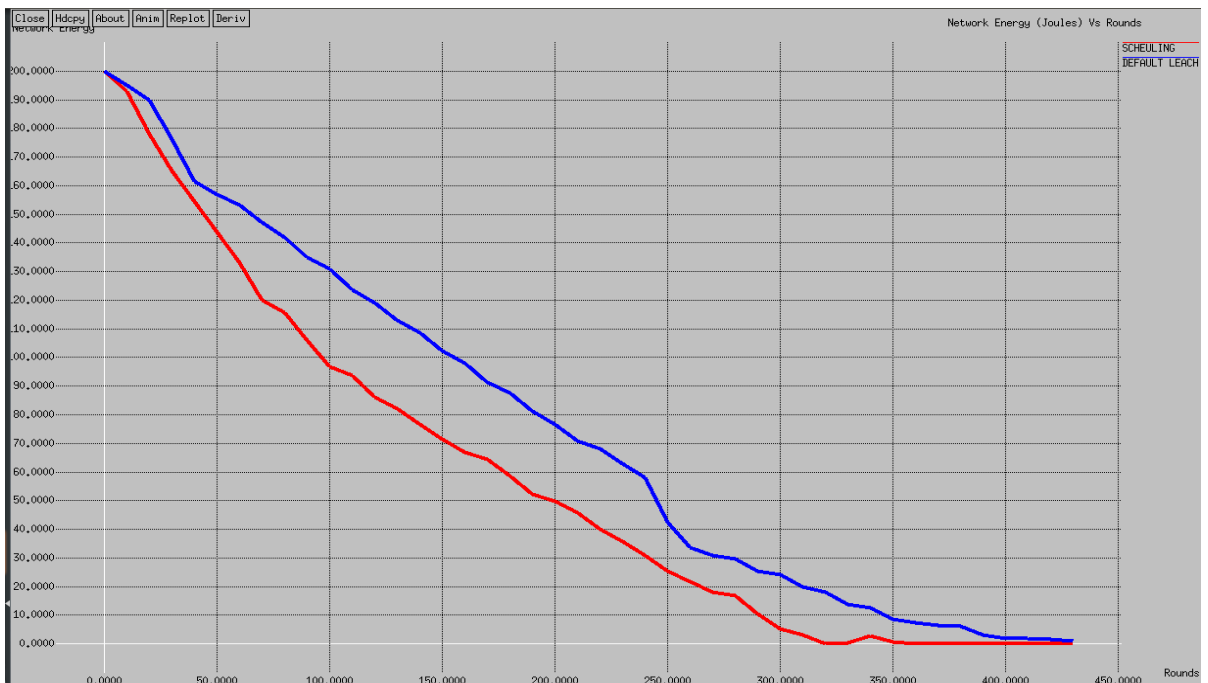


Fig 6.17

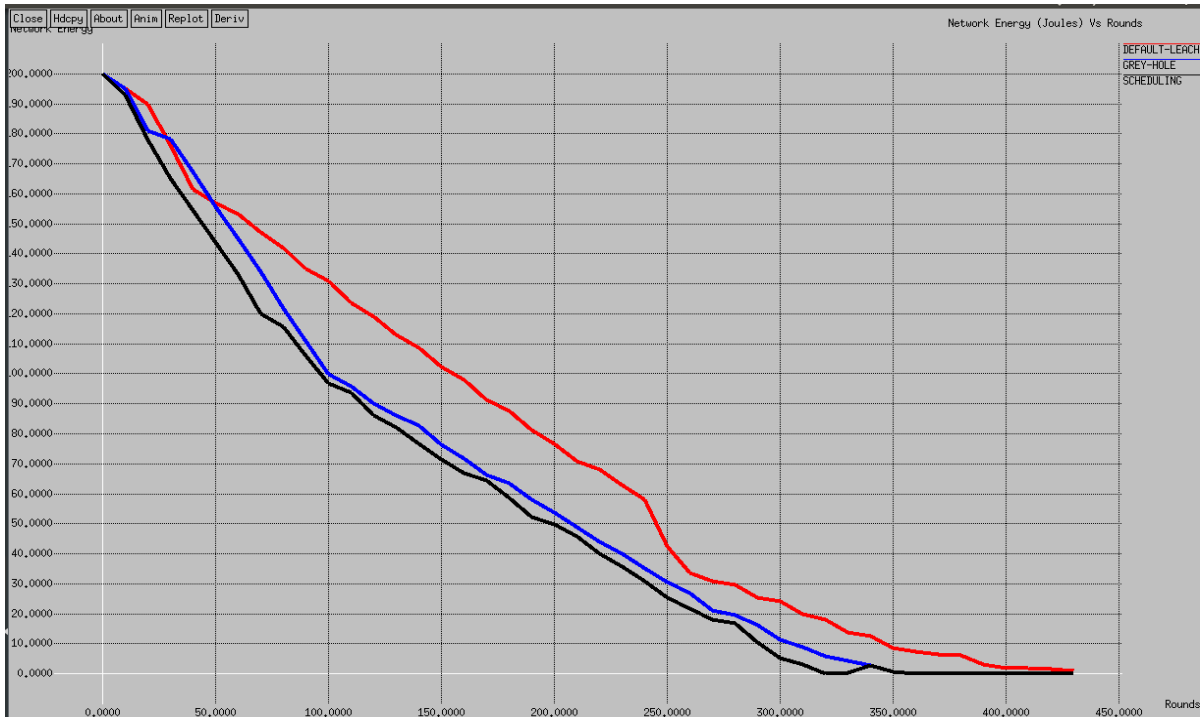


Fig 6.18

In figure 6.25 we plotted simulation results of default LEACH protocol with Grey-Hole and Scheduling attacks on it.

In figure 6.26 we plotted number of nodes died in 450 rounds for default LEACH protocol. First node died in 61st round.

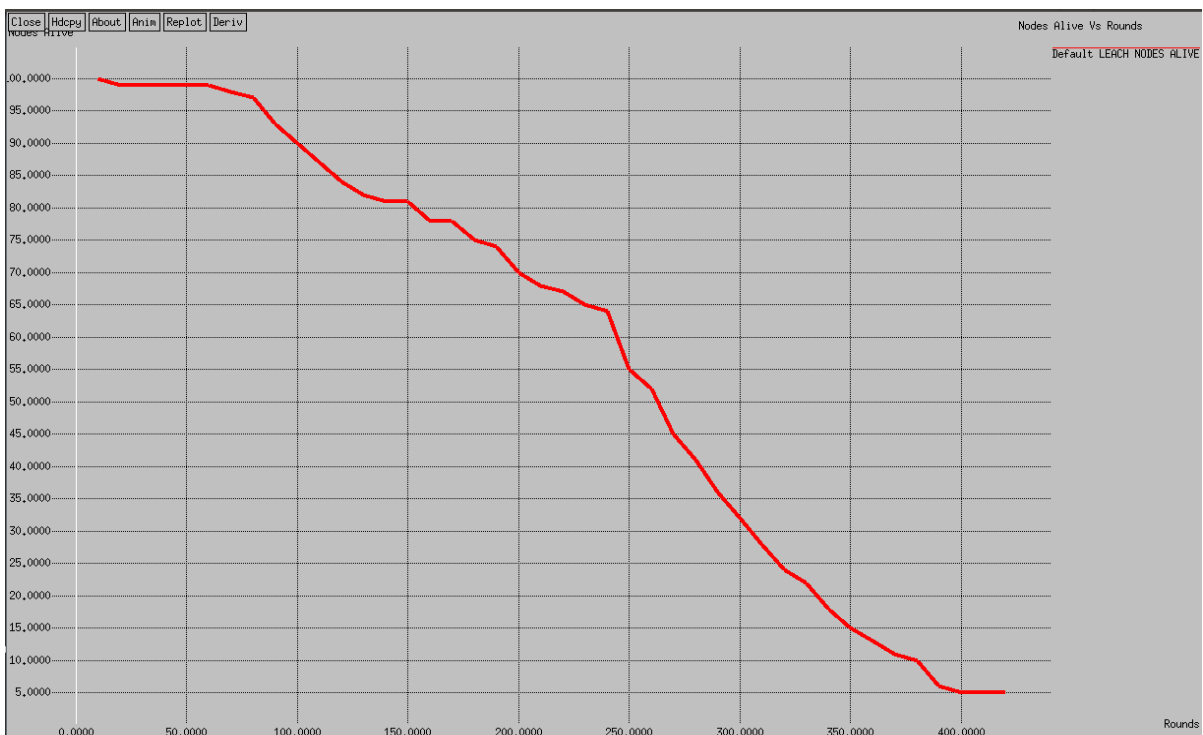


Fig 6.19

Figure 6.27 shows rate of nodes died in default LEACH protocol with grey-hole and scheduling attack on LEACH protocol. Red graph shows rate of nodes died in default LEACH protocol, green graph depicts rate of nodes died due to grey-hole attack and blue graph depicts the rate of nodes died due to scheduling attack.

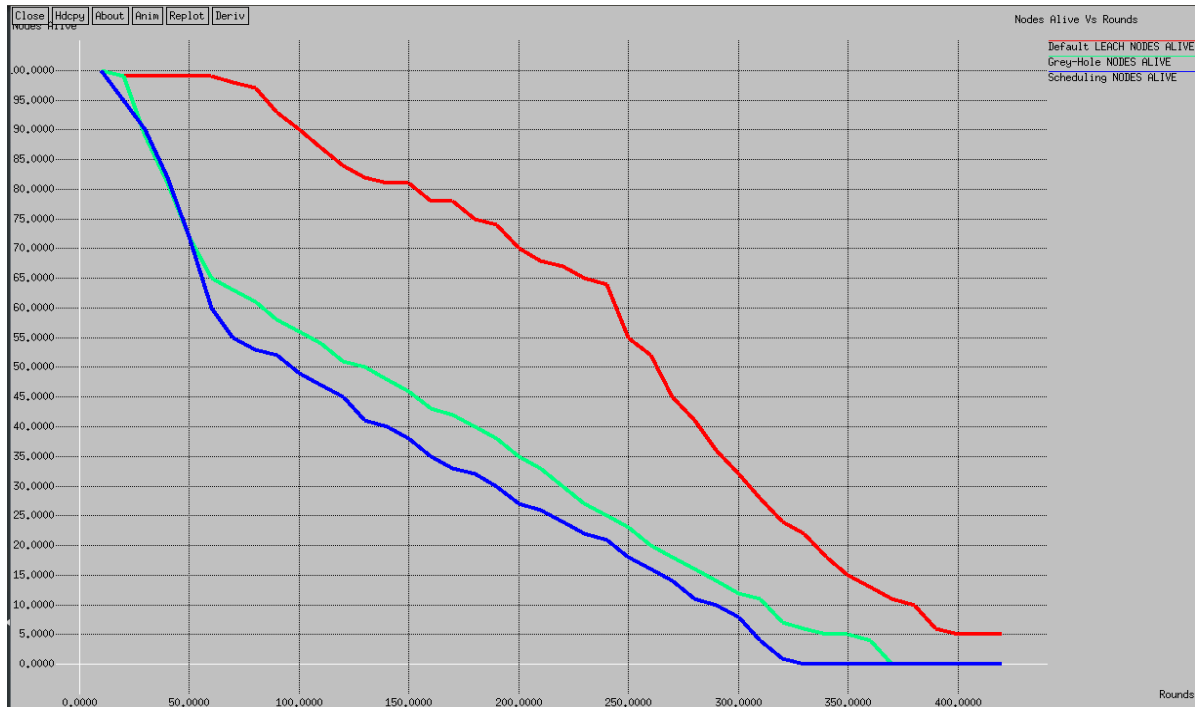


Fig 6.20

6.8 Prevention Scheme Simulation in ns-2

In this section we have simulated prevention scheme for malicious CH selection in ns-2. Details about prevention scheme mechanism already presented in Chapter 5. We also presented simulation results we got from our proposed solution. We used same environmental parameters as listed in Table 6.2. Malicious CH tried to drain network energy initially, but after some rounds when cluster nodes build their trust for neighboring nodes they will not select malicious node as CH. A malicious node gets lower level of trust value due to drainage of energy in early rounds. This node is never elected as CH whether it is nominating itself as CH in every round.

Chapter 7

Conclusion and Future Work

In this chapter we will describe our thesis contribution, conclude our research work and draw some light on the future work.

7.1 Thesis Contribution

We have conducted our research in three segments.

1. We have performed research work for energy drainage attack on LEACH protocol which is an energy efficient WSN routing protocol but still has vulnerabilities in it.
2. We have proposed a solution to cater energy drainage attack on LEACH protocol based on Cellular Automata and Trust.
3. We have simulated attack scenario in different environment and implemented our proposed solution in these environments.

7.2 Conclusion

In this research, we have proposed a Cellular Automata and trust based solution to detect and mitigate energy drainage attack on LEACH protocol. We simulate energy drainage attack like gray-hole attack, vampire attack and scheduling attack on LEACH protocol using MATLAB and ns-2. Analyzed their effects on LEACH protocol and implement our solution purely based on cellular automata. Nodes make neighborhood with its nearby nodes and exchange information to check they are draining their energies normally or abnormally. Based on this information they calculate trust on its neighboring nodes, change their states in case of an attack and use this trust value for selection of cluster head. This solution not only detect any energy drainage attack but also prevent nodes to select malicious node as a cluster head.

7.3 Future Research Work

There are certain things that need improvement as a future work. Proposed technique can be implemented on other WSN protocols like AODV, PEGASIS and DSDV. This proposed solution can be implemented to mitigate other DoS attacks like Sink-Hole, Flooding and Wormhole attacks. Cellular automata based detection mechanism needs further improvement to make this solution more

robust and lightweight. More work required to reduce the overhead due to extra communication between nodes.

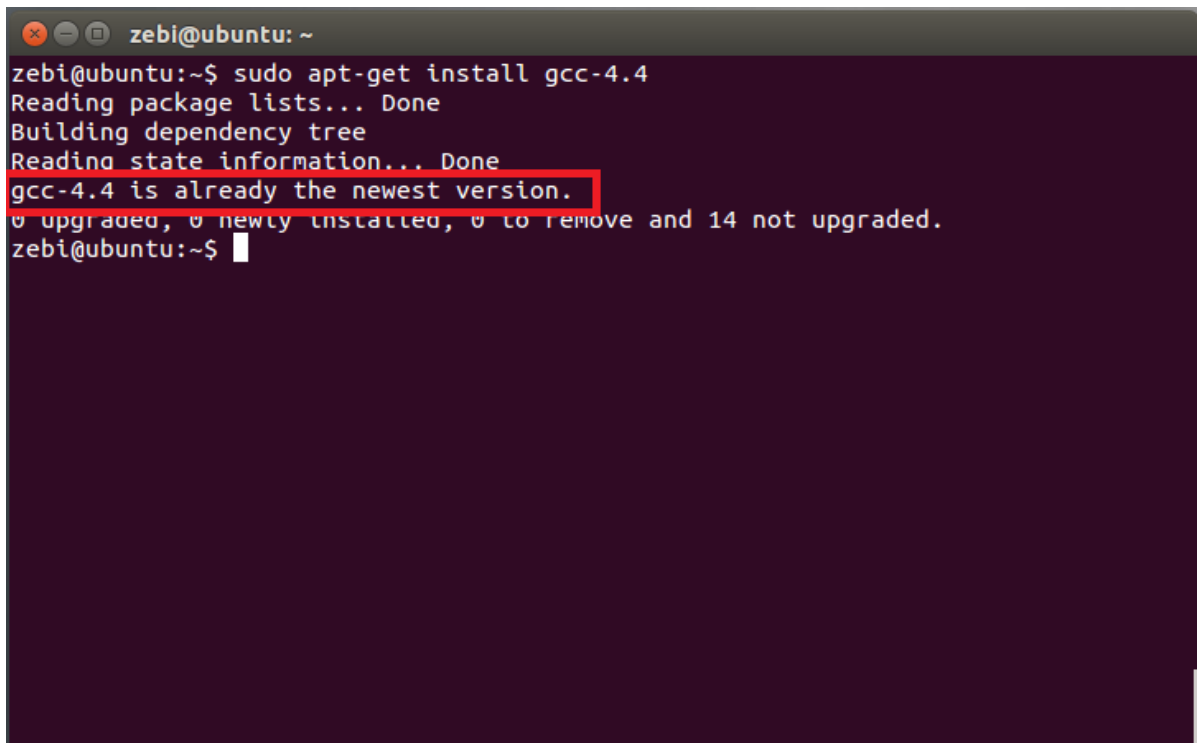
Moreover, each node forms neighborhood at pre-defined distance. But in the cluster a node may have neighboring nodes that belong to another cluster in the network. If nodes experiencing energy drainage in a cluster, some nodes that located at the border of the cluster have neighboring nodes belong to other cluster and not experiencing same energy drainage. So if these nodes are not associated with same CH or not a subset of malicious cluster they could have higher trust value for malicious CH.

These trust values increase indirect trust values received from neighboring nodes and leads to misjudgment of malicious CH. In future we work on this problem and give solution how a node in malicious cluster receive indirect trust value from its neighboring nodes that are not the part of cluster in current round.

Appendix A

You can install Ubuntu OS both in physical and virtual machine. We build this environment in virtual machine using VMware Workstation 12. After installing and configuring Ubuntu 14.04 we use following command to install ns-2 in Ubuntu.

1. To update the Ubuntu machine enter
`$ sudo apt-get update`
This command configure Ubuntu to download latest dependencies from Ubuntu repositories.
2. To upgrade the Ubuntu machine enter
`$ sudo apt-get upgrade`
This command download dependencies from repositories and install them in Ubuntu system.
3. ns-2 requires some more packages to be installed. So type following commands to download and install them.
`$ sudo apt-get install build-essential autoconf automake libxmu-dev`
This command install and configure essential packages require for building ns-2 setup.
4. Then you need to install GCC version 4.4 compiler and other programs to compile ns-2 files. You can download and install it by typing this commands.
`$ sudo apt-get install gcc-4.4`
`$ sudo apt-get install tcl8.5-dev`
`$ sudo apt-get install perl xgraph libxt-dev libx11-dev libxmu-dev`
If you have already installed it then it will show you the message like highlighted in fig 6.13 below.



```
zabi@ubuntu: ~  
zabi@ubuntu:~$ sudo apt-get install gcc-4.4  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
gcc-4.4 is already the newest version.  
0 upgraded, 0 newly installed, 0 to remove and 14 not upgraded.  
zabi@ubuntu:~$
```

Fig A.1

5. Now download the package from the link given above, copy it to your /home folder and untar it by typing following command
`$ tar zvwf ns-allinone-2.35.tar.gz`
This command untar/extract ns-2 program and place it in your /home directory.
6. Before installing ns-2 we have to edit some files to complete installation and without errors. If you are in your /home directory type
`$ cd ns-allinone-2.35` and press enter
Now open the file **ls.h** located in the directory /ns-allinone-2.35/ns-2.35/linkstate/ls.h. Open this file by typing this command.
`$ gedit ns-2.35/linkstate/ls.h`
Search this line
`void eraseAll() { erase(baseMap::begin(), baseMap::end()); }`
Replace with
`void eraseAll() { this->erase(baseMap::begin(), baseMap::end()); }`
7. Now pre-installation requirements are done you can install ns-2 by changing directory to /home/ns-allinone-2.35 and type command `./install` to start installation process.
`$./install` and press enter
It will start installation of ns-2 and take some time for complete installation.
8. After the successful installation you need to add some environmental path in .bashrc file. To do this type command
`$ sudo gedit .bashrc`

Add the following lines at the end of this file.

```
# LD_LIBRARY_PATH
OTCL_LIB=/home/ns-allinone-2.35/otcl-1.14/
NS2_LIB=/home/ns-allinone-2.35/lib/
USR_Local_LIB=/usr/local/lib/
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OTCL_LIB:$NS2_LIB:$USR_Local_LIB

# TCL_LIBRARY
TCL_LIB=/home/ns-allinone-2.35/tcl8.5.10/library/
USR_LIB=/usr/lib/
export TCL_LIBRARY=$TCL_LIBRARY:$TCL_LIB:$USR_LIB

# PATH
XGRAPH=/home/ns-allinone-2.35/xgraph-12.2/:/home/ns-allinone-2.35/bin/:/home/ns-allinone-
2.35/tcl8.5.10/unix/:/home/ns-allinone-2.35/tk8.5.10/unix/
NS=/home/ns-allinone-2.35/ns-2.35/
NAM=/home/ns-allinone-2.35/nam-1.15/
export PATH=$PATH:$XGRAPH:$NS:$NAM
```

Save and close this file and restart your system.

9. After rebooting the system go to the directory `/home/ns-allinone-2.35/ns-2.35` and hit `./validate` command. It will perform some tests on the installed modules of ns-2. It will take more than about half an hour for complete validation.
10. When all the steps has completed type `ns` command in your `/home` directory to start ns-2 like this:
\$ ns
It will prompt on your command line like this:
%

Now you have successfully installed ns-2 in your Ubuntu 14.04 machine.

6.6.2 Installing LEACH protocol in ns-2

You can download the source code of the LEACH protocol from the following link:

<https://drive.google.com/drive/folders/0B4nUSbTYSK4TfkVrVEhzaHR1aXYwQ296bXZKZE14a21wSXFtRmFqNW4tQ2VKbVlGWnJUeXM>

Place the downloaded file in your `/home` directory. Untar it with the following command.

```
$ untar zvwf leach-ns-2.35.tar.gz
```

After extracting the leach code you have to copy some files into ns-2 installation directory.

Or simply you can run the leach-setup.sh script to copy all necessary files from LEACH source code to ns-2.35 source installation directory.

```
#!/bin/bash
tar -xvzf ns-leach-2.35.tar.gz
cd ns-leach-2.35/
cp -r mit /home/jahanzeb/ns-allinone-2.35/ns-2.35
cp apps/app.* /home/jahanzeb/ns-allinone-2.35/ns-2.35/apps
cp mac/channel.cc /home/jahanzeb/ns-allinone-2.35/ns-2.35/mac
cp mac/ll.h /home/jahanzeb/ns-allinone-2.35/ns-2.35/mac
cp mac/wireless-phy.* /home/jahanzeb/ns-allinone-2.35/ns-2.35/mac
cp mac/phy.* /home/jahanzeb/ns-allinone-2.35/ns-2.35/mac
cp mac/mac.cc /home/jahanzeb/ns-allinone-2.35/ns-2.35/mac
cp mac/mac-sensor* /home/jahanzeb/ns-allinone-2.35/ns-2.35/mac
cp trace/cmu-trace.* /home/jahanzeb/ns-allinone-2.35/ns-2.35/trace
cp common/packet.* /home/jahanzeb/ns-allinone-2.35/ns-2.35/common
cp common/mobilenode.cc /home/jahanzeb/ns-allinone-2.35/ns-2.35/common
cp tcl/mobility/leach-c.tcl /home/jahanzeb/ns-allinone-2.35/ns-2.35/tcl/mobility
cp tcl/mobility/leach.tcl /home/jahanzeb/ns-allinone-2.35/ns-2.35/tcl/mobility
cp tcl/mobility/mte.tcl /home/jahanzeb/ns-allinone-2.35/ns-2.35/tcl/mobility
cp tcl/mobility/stat-clus.tcl /home/jahanzeb/ns-allinone-2.35/ns-2.35/tcl/mobility
cp tcl/ex/wireless.tcl /home/jahanzeb/ns-allinone-2.35/ns-2.35/tcl/ex
cp leach_test /home/jahanzeb/ns-allinone-2.35/ns-2.35
cp Makefile.in /home/jahanzeb/ns-allinone-2.35/ns-2.35
```

Fig A.2

After copying all necessary files you need to reconfigure the ns-2 code. You will change your directory to /ns-allinone-2.35/ns-2.35 by typing following commands.

```
$ cd ns-allinone-2.35          press enter
```

```
~/ns-allinone-2.35 $ cd ns-2.35          press enter
```

```
~/ns-allinone-2.35/ns-2.35 $ ./configure          press enter
```

This command reconfigure ns-2 code with updated source code file of LEACH protocol.

By following above steps we installed LEACH code in ns-2. It copy all its files in ~/ns-allinone-2.35/ns-2.35/mit folder. Simulation results will be generated in ~/ns-allinone-2.35/ns-2.35/mit/leach_sims/ folder.

Some files are needed to be modified, placed in the directory /home/ns-allinone-2.35/ns-2.35/mit/uAMPS/sims folder. These files are leach.tcl, leach-c.tcl, mte.tcl, uamps.tcl and stat-clus.tcl. In newly installed LEACH source code these files have default for other source files. So you have to update them with current locations of source files. For example in file uamps.tcl it has default paths for

source	file	as	shown:
--------	------	----	--------

```

uamps.tcl x
#####
#
# This code was developed as part of the MIT uAMPS project. (June, 2000)
#
#####

global opt bs

#source $env(RCA_LIBRARY)/ns-ranode.tcl
#source $env(uAMPS_LIBRARY)/ns-bsapp.tcl
#source $env(uAMPS_LIBRARY)/extras.tcl
#source $env(uAMPS_LIBRARY)/stats.tcl

```

Fig A.3

All these commented source paths are needed to modify according to their current location in code. In our case these paths are:

```

uamps.tcl x
#####
#
# This code was developed as part of the MIT uAMPS project. (June, 2000)
#
#####

global opt bs

#source $env(RCA_LIBRARY)/ns-ranode.tcl
#source $env(uAMPS_LIBRARY)/ns-bsapp.tcl
#source $env(uAMPS_LIBRARY)/extras.tcl
#source $env(uAMPS_LIBRARY)/stats.tcl

source /home/zebi/ns-allinone-2.35/ns-2.35/mit/rca/ns-ranode.tcl
source /home/zebi/ns-allinone-2.35/ns-2.35/mit/uAMPS/ns-bsapp.tcl
source /home/zebi/ns-allinone-2.35/ns-2.35/mit/uAMPS/extras.tcl
source /home/zebi/ns-allinone-2.35/ns-2.35/mit/uAMPS/stats.tcl

```

Fig A.4

We perform same procedure with other four files and update their source file path according to their locations.

6.6.3 Running LEACH protocol in ns-2

There should be a file name leach_test in ~/home/ns-allinone-2.35/ns-2.35/ folder. It is a script file to run leach protocol. If you open this file it will look like:

```

#!/bin/bash
# This file runs a generic LEACH protocol simulation.

#This is the algorithm that we are going to run. Type leach, leach-c, stat-clus or pegasis
alg=leach
#dirname, filename =
# The directory and filename that we want our output to be written.
dirname="mit/leach_sims"
filename=$alg

#Topology
# This file is the scenario that we are going to
# This file can be edited manually if you are very careful to create
# a predefined topology. To generate a random topology go to the
# ./mit/uAMPS/sims directory and run 'ns genscen'.
topology_file="mit/uAMPS/sims/100nodes_random.txt"

# number of clusters we want. It is recommended to use 5% of the total for LEACH and LEACH-C
# WARNING! - It is important to use 1 cluster for Pegasis
# number of nodes in the scenario.
num_clusters=5
# energy values. How much energy does each node have initially
eq_energy=1
init_energy=2
# stop is the time to stop the simulation if it is still running
stop=50
# x,y is the size of the field
x=1000
y=1000
# bs_x, bs_y is the location of the base station in the field.
bs_x=50
bs_y=175
# Number of nodes. WARNING! This should be 1 higher then the number
# of nodes generated in the scenario.
nn=101

ns wireless.tcl \
-sc mit/uAMPS/sims/nodescen.tcl \
-rp $alg \
-x $x \
-y $y \
-nn $nn \
-stop $stop \
-eq_energy $eq_energy \

```

Fig A.5

This file initiates LEACH protocol and some parameters like number of clusters, initial node energies, base station location, simulation time and total number of nodes in network. It also defines the directory where it puts output results. You can run simulation by going into the directory ~/home/ns-allinone-2.35/ns-2.35. Like if you are in your /home directory then change your directory to the above mentioned path with these commands.

\$ cd ns-allinone-2.35 press enter

~/ns-allinone-2.35 \$ cd ns-2.35 press enter

~/ns-allinone-2.35/ns-2.35 \$ ls press enter it will appear in you in this directory as shown below:

```

zebi@ubuntu: ~/ns-allinone-2.35/ns-2.35
aomdv          diffusion3     makefile.vc   satellite
apps           doc           mcast         sctp
asim           dsdv         mdart         sensor-nets
autoconf.h     dsr          mit           src_rtg
autoconf.h.in empweb       mit-leach-protocol tcl
autoconf-win32.h emulate     mobile        tcp
BASE-VERSION  FILES        mpls         test
baytcp        gaf          nix          test-all
bin           gen          ns           tmix
bitmap        HOWTO-CONTRIBUTE ns.1         TODO.html
CHANGES.html imep         ns_tclsh.cc  tools
classifier     indep-utils  nstk         tora
common        install-sh   packmime     trace
conf          INSTALL.WIN32 pgm          validate
config.guess  leach.nam   plm          validate.out
config.h       leach_test  puma        VERSION
config.log    leach_test~ pushback     webcache
config.status leach.tr    qs          wireless.tcl
config.sub    lib         queue       wireless.tcl~
configure     LICENSES    rap         wpan
configure.in  link        README      xcp
COPYRIGHTS   linkstate   realaudio
dccp         mac         release_steps.txt
zebi@ubuntu:~/ns-allinone-2.35/ns-2.35$

```

Fig A.6

To run this script type

~/ns-allinone-2.3/ns-2.35\$./leach_test press enter

```

zebi@ubuntu: ~/ns-allinone-2.35/ns-2.35
apps           doc           mcast         sctp
asim           dsdv         mdart         sensor-nets
autoconf.h     dsr          mit           src_rtg
autoconf.h.in empweb       mit-leach-protocol tcl
autoconf-win32.h emulate     mobile        tcp
BASE-VERSION  FILES        mpls         test
baytcp        gaf          nix          test-all
bin           gen          ns           tmix
bitmap        HOWTO-CONTRIBUTE ns.1         TODO.html
CHANGES.html imep         ns_tclsh.cc  tools
classifier     indep-utils  nstk         tora
common        install-sh   packmime     trace
conf          INSTALL.WIN32 pgm          validate
config.guess  leach.nam   plm          validate.out
config.h       leach_test  puma        VERSION
config.log    leach_test~ pushback     webcache
config.status leach.tr    qs          wireless.tcl
config.sub    lib         queue       wireless.tcl~
configure     LICENSES    rap         wpan
configure.in  link        README      xcp
COPYRIGHTS   linkstate   realaudio
dccp         mac         release_steps.txt
zebi@ubuntu:~/ns-allinone-2.35/ns-2.35$ ./leach_test
zebi@ubuntu:~/ns-allinone-2.35/ns-2.35$

```

Fig A.7

This script runs the LEACH protocol in background and stores all the results in folder ~/ns-allinone-2.35/ns-2.35/mit/leach_sims. It will generate files leach.out, leach.err, leach.alive, leach.energy and TDMASchedule.txt files. All the results

of the simulation store in these files. leach.out is the file which store the results in detail.

References:

- [1] J. Yick, B. Mukherjee, and D. Ghosal, “Wireless sensor network survey,” *Comput. Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.
- [2] W. Dargie and C. Poellabauer, “FUNDAMENTALS OF WIRELESS SENSOR NETWORKS THEORY AND PRACTICE.”
- [3] J. Gutierrez, J. F. Villa-Medina, A. Nieto-Garibay, and M. A. Porta-Gandara, “Automated Irrigation System Using a Wireless Sensor Network and GPRS Module,” *IEEE Trans. Instrum. Meas.*, vol. 63, no. 1, pp. 166–176, Jan. 2014.
- [4] J. M. Corchado, J. Bajo, D. I. Tapia, and A. Abraham, “Using Heterogeneous Wireless Sensor Networks in a Telemonitoring System for Healthcare,” *IEEE Trans. Inf. Technol. Biomed.*, vol. 14, no. 2, pp. 234–240, Mar. 2010.
- [5] Y. E. Aslan, I. Korpeoglu, and Ö. Ulusoy, “A framework for use of wireless sensor networks in forest fire detection and monitoring,” *Comput. Environ. Urban Syst.*, vol. 36, no. 6, pp. 614–625, 2012.
- [6] K. K. Khedo, R. Perseedoss, A. Mungur, U. of Mauritius, and Mauritius, “A Wireless Sensor Network Air Pollution Monitoring System,” May 2010.
- [7] J. Rezazadeh and J. Rezazadeh, “Mobile Wireles Sensor Networks Overview,” *Int. J. Comput. Commun. Networks*, vol. 2, no. 1, 2012.
- [8] S. A. Munir, B. Ren, W. Jiao, B. Wang, D. Xie, and J. Ma, “Mobile Wireless Sensor Network: Architecture and Enabling Technologies for Ubiquitous Computing,” in *21st International Conference on Advanced Information Networking and Applications Workshops (AINAW’07)*, 2007, pp. 113–120.
- [9] R. R. Sahoo, M. Singh, A. R. Sardar, S. Mohapatra, and S. K. Sarkar, “TREE-CR: Trust based secure and energy efficient clustering in WSN,” *2013 IEEE Int. Conf. Emerg. Trends Comput. Commun. Nanotechnol.*, no. Iceccn, pp. 532–538, 2013.
- [10] H. Byun and J. Yu, “Cellular-Automaton-Based Node Scheduling Control for Wireless Sensor Networks,” *IEEE Trans. Veh. Technol.*, vol. 63, no. 8, pp. 3892–3899, 2014.
- [11] M. Shankar, M. Sridar, and M. Rajani, “Performance Evaluation of LEACH Protocol in Wireless Network,” *Int. J. Sci. Eng. Res.*, vol. 3, no. 1, pp. 1–7, 2012.

- [12] A. Miglani, T. Bhatia, and S. Goel, “TRUST based energy efficient routing in LEACH for wireless sensor network,” no. Gcct, pp. 361–365, 2015.
- [13] C. Karlof, N. Sastry, and D. Wagner, “TinySec,” in *Proceedings of the 2nd international conference on Embedded networked sensor systems - SenSys '04*, 2004, p. 162.
- [14] F. Song and B. Zhao, “Trust-Based LEACH Protocol for Wireless Sensor Networks,” in *2008 Second International Conference on Future Generation Communication and Networking*, 2008, pp. 202–207.
- [15] M. A. Jan, “Energy-efficient Routing and Secure Communication in Wireless Sensor Networks,” no. February, p. 209, 2016.
- [16] J. Kumari and Prachi, “A comprehensive survey of routing protocols in wireless sensor networks,” pp. 325–330, 2015.
- [17] S. Kaplantzis, “Security Models for Wireless Sensor Networks,” *PhD Convers. Report, Cent. Telecommun. Inf. Eng. Monash Univ. Aust.*, 2006.
- [18] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energy-efficient communication protocol for wireless microsensor networks,” in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, vol. vol.1, p. 10.
- [19] P. Reindl, K. Nygard, and X. Du, “Defending Malicious Collision Attacks in Wireless Sensor Networks,” *2010 IEEE/IFIP Int. Conf. Embed. Ubiquitous Comput.*, pp. 771–776, 2010.
- [20] S. Magotra and K. Kumar, “Detection of HELLO flood attack on LEACH protocol,” *Souvenir 2014 IEEE Int. Adv. Comput. Conf. IACC 2014*, pp. 193–198, 2014.
- [21] A. Depedri, A. Zanella, and R. Verdone, “An energy efficient protocol for wireless sensor networks,” *Auton. Intell. Networks Syst. (AINS 2003)*, Menlo Park. CA, pp. 1–6, 2003.
- [22] L. Qing, Q. Zhu, and M. Wang, “Design of a distributed energy-efficient clustering algorithm for heterogeneous wireless sensor networks,” *Comput. Commun.*, vol. 29, no. 12, pp. 2230–2237, 2006.
- [23] C. H. Lin and M. J. Tsai, “A comment on ‘HEED: A Hybrid, Energy-Efficient, Distributed clustering approach for ad hoc sensor networks,’” *IEEE Transactions on Mobile Computing*, vol. 5, no. 10, pp. 1471–1472, 2006.
- [24] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, “An application-specific protocol architecture for wireless microsensor

- networks,” *IEEE Trans. Wirel. Commun.*, vol. 1, no. 4, pp. 660–670, 2002.
- [25] S. Varma, N. Nigam, and U. S. Tiwary, “Base station initiated dynamic routing protocol for Heterogeneous Wireless Sensor Network using clustering,” *Wireless Communication and Sensor Networks, 2008. WCSN 2008. Fourth International Conference on*. pp. 1–6, 2008.
- [26] W. Li, L. Ma, and Q. Yu, “Cellular automata-based multi-hop WSN routing protocol energy saving technology,” *2013 Int. Conf. Commun. Circuits Syst. ICCAS 2013*, vol. 1, pp. 113–117, 2013.
- [27] C. Banerjee and S. Saxena, “Energy Conservation in Wireless Sensor Network using Block Cellular Automata,” vol. 4, no. 5, pp. 2216–2220, 2013.
- [28] A. Tretyakova, F. Seredynski, and P. Bouvry, “Graph-based cellular automata approach to maximum lifetime coverage problem in wireless sensor networks,” *Proc. Int. Parallel Distrib. Process. Symp. IPDPS*, pp. 439–447, 2014.
- [29] A. H. F. Navid and H. H. S. Javadi, “ICLEAR: Energy aware routing protocol for WSN using irregular cellular learning automata,” *2009 IEEE Symp. Ind. Electron. Appl. ISIEA 2009 - Proc.*, vol. 1, no. Isiea, pp. 463–468, 2009.
- [30] A. Ahmed, K. A. Bakar, M. I. Channa, K. Haseeb, and A. W. Khan, “TERP: A Trust and Energy Aware Routing Protocol for Wireless Sensor Network,” *IEEE Sens. J.*, vol. 15, no. 12, pp. 6962–6972, 2015.
- [31] F. Van Den Abeele, T. Vandewinckele, J. Hoebeke, I. Moerman, and P. Demeester, “Secure communication in IP-based wireless sensor networks via a trusted gateway,” *2015 IEEE 10th Int. Conf. Intell. Sensors, Sens. Networks Inf. Process. ISSNIP 2015*, no. April, pp. 7–9, 2015.
- [32] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: a survey,” *Comput. Networks*, vol. 38, no. 4, pp. 393–422, 2002.
- [33] W. B. Heinzelman, “Application-specific protocol architectures for wireless networks,” vol. 689, p. 154, 2000.
- [34] Yong Soo Cho and Kim Jaekwon, *MIMO-OFDM Wireless Communications with MATLAB - Yong Soo Cho, Jaekwon Kim, Won Young Yang, Chung G. Kang - Google Books*. 2010.
- [35] I. Almomani and B. Al-Kasasbeh, “Performance analysis of LEACH protocol under Denial of Service attacks,” *2015 6th Int. Conf. Inf.*

- Commun. Syst.*, pp. 292–297, 2015.
- [36] E. Y. Vasserman and N. Hopper, “Vampire attacks: Draining life from wireless ad Hoc sensor networks,” *IEEE Trans. Mob. Comput.*, vol. 12, no. 2, pp. 318–332, 2013.
- [37] A. A. Patel and S. J. Soni, “A novel proposal for defending against vampire attack in WSN,” *Proc. - 2015 5th Int. Conf. Commun. Syst. Netw. Technol. CSNT 2015*, pp. 624–627, 2015.
- [38] M. Vidya and S. Reshmi, “Preventing Denial of Service Attacks in Wireless Sensor Networks,” pp. 16–21, 2014.
- [39] S. P. Dongare and R. S. Mangrulkar, “Implementing Energy Efficient Technique for Defense against Gray-Hole and Black-Hole Attacks in Wireless Sensor Networks,” pp. 167–173, 2015.
- [40] “universality-complexity-cellular-automata.pdf.” .
- [41] A. Ilachinski, *Cellular Automata*. WORLD SCIENTIFIC, 2001.
- [42] M. Peng, K. Xu, Q. Yu, W. Jiang, S. Leng, and Y. Mao, “Cellular Automata Self-organization Algorithm for Wireless Sensor Network.”
- [43] D. Kumar, T. C. Aseri, and R. B. Patel, “EEHC: Energy efficient heterogeneous clustered scheme for wireless sensor networks,” *Comput. Commun.*, vol. 32, no. 4, pp. 662–667, 2009.
- [44] K. Manohar and A. I. Darvadiya, “Clustering Based Routing Protocol (LEACH Protocol),” *Int. J. Innov. Res. Comput. Commun. Eng. (An ISO Certif. Organ.)*, vol. 3297, no. 4, 2007.
- [45] C. Cirstea, M. Cernaianu, and A. Gontean, “Packet loss analysis in wireless sensor networks routing protocols,” *2012 35th Int. Conf. Telecommun. Signal Process.*, no. July, pp. 37–41, 2012.
- [46] R. R. Sahoo, R. Panda, D. K. Behera, and M. K. Naskar, “A trust based clustering with Ant Colony Routing in VANET,” in *2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT’12)*, 2012, pp. 1–8.
- [47] “Simulink - Simulation and Model-Based Design.” [Online]. Available: <https://www.mathworks.com/products/simulink.html>. [Accessed: 13-Feb-2017].