AOMDV-E: Energy Aware Event-Based Routing Protocol for Environmental Monitoring

A Thesis Submitted in

Partial Fulfilment of the Requirements for the Degree of

Doctor of Philosophy in Information Science,

School of Information and Sciences

Middlesex University

By

Maher Rantisi

May 2017

Director of Studies: Dr. Orhan Gemikonakli

Second Supervisor: Dr. Glenford Mapp

Abstract

Due to rapid changes in climatic conditions worldwide, environmental monitoring has become one of the greatest concerns in the last few years. With the advancement in the wireless sensing technology, it is now possible to monitor and track fine-grained changes in different environments. Wireless sensor networks (WSN) provide very high quality and accurate analysis for monitoring of both spatial and temporal data thus providing the opportunity to monitor harsh outdoor environments. But to deploy and maintain a WSN in such harsh environments are great challenges for researchers and scientists. Several routing protocols exist for data dissemination and power management but they suffer from various disadvantages including lack of energy-aware operation, lack of ability to react to sudden environmental changes, etc. There are a lot of challenges which need to be addressed for latest WSNs applications which include timely delivery of data, particularly for real-time applications. Moreover, reactive event-based routing functions are required to deal with changing outdoor environments. For example, with limited water resources in the Middle East, soil moisture measurements must be taken into account to manage irrigation and agricultural projects. The main factors for changing the soil moisture are the seasonal rains. So an event-based routing protocol to determine the correct routing path for sending the data is needed for optimizing irrigation operations.

This study outlines the challenges in supporting such an environment and demonstrates a solution approach based on the modification of a popular reactive routing protocol known as AdHoc On-Demand Multiple Path Distance Vector (AOMDV). Moreover, additional enhancements have been proposed for AOMDV to make it an energy-aware routing protocol with the additional characteristic of being capable of acquiring energy from the solar system. The proposed modifications, AOMDV-E (event-based, energy-aware routing functions) in AOMDV are desirable to facilitate better wireless connectivity for current and future needs. In our case study, there are very limited water resources in the Middle East, hence soil moisture measurements must be taken into account to manage irrigation and agricultural projects. In order to meet these challenges, a testbed that supports an energy aware, reactive, event-based routing protocol is developed using AOMDV. A prototype WSN network of 5 nodes was built. Three simulations

have been done to test the proposed algorithms and their scalability: the first consisted of 5 nodes with one of them affected by rain. The second simulation considered 7 nodes in which 2 of them are affected by rain while the final simulation is based on 30 nodes with 5 of them affected by rain.

AOMDV-E event-driven enhancements not only increase the performance of the proposed protocol but also make it energy efficient as the energy consumption is considerably reduced for the nodes experiencing the rainfall and also in general because we use the sleep mode when it is not raining. Simulation results also show that when the rainfall is heavier, then also the amount of energy consumed is reduced which shows that the proposed AOMDV-E protocol is robust in terms of the amount of rainfall.

The simulation results also show that the enhanced AOMDV-E protocol is scalable. Its performance is compared with prior AOMDV protocol. The results clearly show that AOMDV-E reduces average delay while at the same time increases the throughput of the nodes being affected by rain.

Acknowledgments

I would like to express my gratitude to all those who gave me the opportunity to complete this thesis. I want to thank the Facility of Science and Technology, Middlesex University for giving me the opportunity to commence this thesis in the first instance, to do the necessary research work and to use departmental resources.

I would like to express my deep and sincere gratitude to my supervisors, Dr. Glenford Mapp and Dr. Orhan Gemikonakli from the Middlesex University whose help, stimulating suggestions and encouragement motivated me to complete this thesis. My colleagues in the Faculty of Science and Technology supported me in my research work. I want to thank them for all their help, support, interest and valuable feedback.

I am deeply indebted to my wife Raniah Hussein, for much of what I have become. I thank her for her patience, love, and support. She kept me away from family responsibilities and encouraged me to concentrate on my studies. I thank her and my children for their all support and love which enabled me to complete this work.

Last but not the least, I would like to thank my parents who supported me at the start of BSC and MSC which is the first step of my study also I pray to god bless their soul.

List of Publications

The results presented in this thesis have also appeared in the following publications:

- Rantisi, M.A.A.; Mapp, G.; Gemikonakli, O. ***A Smart Sensor Grid to Enhance Irrigation Techniques in Jordan Using a Novel Event-Based Routing Protocol***. *Multimodal Technologies Interact.* 2017, *1*, 9.
- Rantisi, M.A.A, Mapp G and Gemikonakli O: ***A WSN Testbed to Enhance Irrigation Techniques Using a Novel Event-Based Routing Protocol***: 13th International Conference on Wireless and Mobile Communications (ICWMC), July 23rd – 27th 2017, Nice, France.
- Rantisi M, Mapp G, and Gemikonakli, O: ***Simulation of an Event-Driven Wireless Sensor Network Protocol for Environmental Monitoring****:* 4th International Conference on Innovative Computing Technology (INTECH 2014), University of Bedfordshire, Luton, August 13th – 15th 2014.
- Rantisi M, Maqousi A, Mapp G, and Gemikonakli O: ***Simulation of an Event-Driven Wireless Sensor Network Protocol for Environmental Monitoring:*** Presented at the 6th International Conference on Information Technology, Amman Jordan, May 8th-12th 2013.
- Rantisi M, Maqousi A, Mapp G, and Gemikonakli O: ***The Development of a Dynamic and Robust Event-Based Routing Protocol in Wireless Networks for Environmental Monitoring***: Presented at the First Global Conference on Communication, Science and Information Engineering, 25th – 27th July 2011, Middlesex University, London.

Contents

List of Figures

List of Tables

Chapter 1: Introduction

Wireless Sensor Networks (WSNs) comprise autonomous, distributed nodes that work together to monitor factors such as temperature, humidity, and pressure. The nodes of a WSN are used to sense, gather, process and communicate data. It is assumed that a typical WSN network would contain a large number of nodes. WSNs have the ability to sense a variety of parameters permanently or temporarily. WSNs are part of a growing technology that has been designed to support a wide range of applications in wireless environments [1, 3]. WSN has been used for many applications, such as animal tracking [5], structural monitoring [6], environmental monitoring [9] and resource monitoring in offices and homes [11]. WSNs may be used in a variety of everyday activities or services. For example, a common application of WSNs is monitoring. In the area of monitoring, the WSN is deployed over a region in order to monitor some phenomenon. A practical use of such a network could be a military use where sensors can detect enemy intrusion. In that case, the sensors must detect an event and then the event should be immediately reported to the base station, which decides on the appropriate action. Another area for WSNs usage may be the monitoring of air pollution, where the sensors are deployed in several cities to monitor the concentration of dangerous gases for citizens. Moreover, a WSN may be used for the detection of forest fires to discover when a fire has started. The nodes will be equipped with sensors to detect temperature, humidity, and gases which are produced by fire in the trees or in vegetation. In addition to the above, an important area of use is in the healthcare sector and where WSNs may offer significant cost savings and enable new functionalities that will assist elderly people living alone or people with chronic diseases with their activities.

Although sensor networks have many applications, environmental monitoring is a domain in which they may have a huge impact. Recent climate change-related catastrophes have illustrated the importance of a detailed understanding of our environment and its evolution for human beings. The capacity of researchers to improve this knowledge is mainly limited by current data collection techniques, which are based on very expensive stations [2]. WSNs are particularly useful in remote or dangerous environments whose behaviors have rarely been studied due to their inaccessibility.

The recent research and development in wireless technologies and the advancements in wireless network services have made wireless communications a pervasive means of monitoring the natural environment. There are many innovations made in the field of communications which are transforming to the Internet of Things (IoT). In the domain of IoT, WSN is employed as independent sensing devices to monitor physical and environmental conditions along with thousands of applications in other fields. One of fastest growing applications of WSN in the IoT domain and in the urban context is the smart city that promises to improve the quality and performance of urban services by the use of Information and Communications Technology (ICT). Together they have improved the lifestyle of the citizens by providing better facilities and reduced the administration, management efforts of the city enabling effective utilization of resources and better quality of services.

There are many services in a smart city for which the quality can be improved such as waste management, traffic management, air quality management, parking management, energy consumption management, weather monitoring, noise monitoring and automation buildings. Temperature, humidity, and CO_2 are the basic parameters for services like weather monitoring for future agricultural actions and air quality management for reduction of pollution and healthy environment. To achieve this, a wireless sensor node is required to collect and monitor the data remotely. There have been numerous efforts on microclimate monitoring using Wireless Sensor Network (WSN). Hence the architecture of Wireless Sensor Networks (WSNs), offers numerous potential applications where a WSN can be deployed. Therefore environmental monitoring is an important subject for applying wireless sensor networks.

This thesis examines the problem of deploying WSNs in one of such harsh environments in the Middle East region. One of the major problems in the Middle East region is the limitation of water resources. Many research efforts have tried to find the solution to this problem. These efforts have been partially successful [4]. Contrary to previous efforts, we will try to find out the suitable land for agricultural purposes through measuring the soil moisture of different regions. Since there is vast spatial distribution of variables to be monitored (e.g., temperature and soil moisture) we need a special WSN for such environments as in Middle East region where the sun is present for at least 12 hours per day which makes the environment very dry, and hence the soil moisture needs to be regularly and accurately measured to allow targeted irrigation techniques to be implemented.

1.1 Motivation and Problem Statement

In order to deal with the harsh environment, it is necessary to build a system which has two main proprieties, i.e. event-driven and energy aware. A lot of advancements have been recently in both hardware and software for wireless sensor networks. Several routing protocols exist for data dissemination and power management but they suffer from various disadvantages. However, to ensure proper operation and deployment of sensor networks we need to resolve following issues for monitoring remote outdoor environments.

Firstly, the main issue is that the sensor network has the ability to react to the different events correctly; for example the presence and the absence of rainfall and hence the changes in the degree of moisture in the soil.

Secondly, once the event is detected, the affected nodes need to sample the environment at a much faster rate. For example, if the system is set to sample after every few minutes when it is not raining then it should sample every few seconds when it is raining. Hence the available bandwidth should be more at nodes affected by rain. Hence the required information is sent across as fast as possible and also accurately to the monitoring center or base station.

Thirdly, there may be times when there are fewer resources available, hence the system must be able to make a trade-off between requirements based on available resources.

Currently, there is no such energy aware and event-based wireless routing protocol to react to such type of harsh outdoor environments. Resource limitations of the sensor nodes and unreliability of low-power wireless links [8], in combination with various performance demands of different applications, impose many challenges in designing efficient communication protocols for wireless sensor networks [10].

In this thesis, our main focus is to build an energy aware and reactive routing protocol, which is also energy efficient and can be used for monitoring in harsh outdoor environments.

1.2 Research Objectives

The main objectives of this research study, are as follows:

- To design a WSN for Environmental Monitoring that will monitor the soil moisture for agricultural purposes as there is a lack of water resources in the Middle East.
- To develop a reactive, event-driven and energy efficient WSN routing protocol for environmental monitoring.
- To make the WSN routing protocol robust, so that when the rainfall is light or heavy, in all cases the overall the amount of energy consumed is reduced.
- To explore the challenges and describe the issues involved in the design of an event-driven and energy aware routing protocol for environmental monitoring.
- To validate the result of the proposed protocol through simulation and test bed studies.

1.3 Contributions

This thesis makes the following original contributions:

- AOMDV-E, an enhanced AOMDV routing protocol that is reactive, event-driven, energy efficient and robust, that is capable of providing timely and reliable information about soil moisture scenarios.
- The ability of the proposed AOMDV-E protocol to be used for precision agriculture and smart irrigation systems in the future.
- A WSN testbed for monitoring the soil moisture content for irrigation purposes.
- Detailed analysis and comparison of all popularly available routing protocols (Proactive Reactive and Hybrid)
- Performance analysis and evaluation of the proposed AOMDV-E routing protocol using different quantitative metrics including Packet Delivery Fraction, Average End-to-End Delay, Throughput, and Energy consumption.
- Energy measurements in different phases of operation of the protocol.
- A prototype for initial testing of the proposed AOMDV-E protocol.

- Validation of the results of the proposed AOMDV-E protocol through simulation and test bed studies.

1.4 Thesis Structure

The thesis is structured as follows:

Chapter 2 provides the literature review, describing WSNs and popular routing protocols. Chapter 3 outlines the research methodology. Chapter 4 describes the problem analysis and the solution approach taken. Chapter 5 gives the simulation results. Testbed design is discussed and testbed results are presented in Chapter 6. Finally, conclusions and further work are discussed in Chapter 7.

Chapter 2: Literature Review

In this chapter, we will review some of the literature in the field of wireless sensor networks and their applications. We will look at how WSN have been applied to environmental monitoring, we will study the architecture, components and hardware specifications of WSN. We will also look at communication protocols, routing in WSN. We will also review some literature using WSN for precision agriculture and irrigation systems. We will study the Ad-Hoc On-Demand Distance Vector (AODV) routing and Ad-Hoc On-Demand Multipath Distance Vector routing (AOMDV) routing protocols and we will compare between them and other routing approaches, and at last, we will also discuss some work related to our study.

2.1 WSN for Environmental Monitoring

The monitoring of natural environments through observations and study is called Environmental Monitoring. Data collection forms the basis of environmental monitoring, enabling us to understand our natural surroundings in a better way [7].

Environmental Monitoring shows the effects of human behavior on the environment. Some of the applications include the protection of water supplies, air pollution monitoring, natural resource protection, radioactive waste treatment, weather forecasting and monitoring of species [35]. By studying a sample of the environment we can determine the state of the changing environment. Sampling techniques include grab sampling, remote sampling stations, and remote sensing. In Grab sampling, we manually remove a sample from the environment for its analysis. But due to technological progress, it is not frequently used nowadays as it requires a lot of physical human effort. Using remote sampling stations consisting of sensor systems deployed in the environment of interest, we can monitor the surrounding continuously or at defined intervals. These sensor systems significantly reduce the amount of human labor, by storing locally the measurement of samples taken or by transferring them by means of GSM or satellite communication. Remote sensing is the sensing of environments using images or radiation, from a distance by satellite or aircraft. Hence by using remote sensing we can cover larger areas and monitor inaccessible or dangerous environments.

Wireless sensor networks (WSNs) have a lot of advantages in environmental monitoring because of its advanced measurement system capabilities. The biggest advantage being automatic data aggregation. Networks used in traditional sampling methods require labor input to collect large amounts of samples, but by using a WSN, we can observe the environment at multiple locations and automatically transmits the data to a receiving station using its networked architecture. We can autonomously observe and sample harsh locations and extreme conditions [36], without obstructions. Another advantage of WSNs is that we can sample the data in real time hence it becomes extremely useful in a dynamic environment. In addition, the researcher is able to directly interact with the sensors at a monitoring site and thus able to get first-hand information. Therefore environmental monitoring [7] is a natural candidate for applying sensor networks because the variables to be monitored, (e.g., temperature and soil moisture) are usually distributed over a large spatial region. Obtaining real-time, fine-grained data is critical for success, but this is not possible with currently wired data-loggers, which are both expensive, and unable to react to significant events (e.g. to increase sensing rate during a rainstorm) [37]. Wireless Sensor Networks (WSN) is a new technology that promises fine grain monitoring in time and space and at a lower cost. The sensor networks are required to provide a robust service in a hostile environment. They detect and report the fine grain temporal and spatial dynamics of monitored variables across a landscape.

As environmental monitoring is a very broad area having different applications which put different requirement constraints on Wireless Sensor Networks. One of the classifications of the applications of WSNs divides the WSNs into time-driven, event-driven and query-driven sensor networks [38]. In most of the applications, sensing is time-driven in which we sample the attached sensing devices continuously or periodically with respect to time. Data in time-driven applications is usually transmitted periodically, as in most of the data gathered applications [37, 39]. But event-driven sensor networks are more economical and they are used when we need to minimize the traffic and the flooding of the receiving station with meaningless information. Event-driven WSNs sample and observe the area of interest and transmit the sensor information only on the occurrence of particular events, for example on the occurrence of fire [40], rain or volcanic eruption [41]. In query-driven systems information gathered is stored locally and it is only transmitted when requested. Hence it requires a lot of memory and additional power. In environmental monitoring, this type of sensor network is not commonly used.

Hence there are different requirements of the WSN for different applications. In some applications, different classes of WSNs are also used interchangeably. For our work presented in this thesis, we aim for both time driven and event-driven applications of WSNs in environmental monitoring. Below we describe a sensor node, its working principle, and sensor network architecture.

2.2 Wireless Sensor Network Architecture

The WSN design depends mainly on the requirements of different applications and the characteristics of a sensor network. Each of the sensor nodes (also called motes) in a WSN monitor their local environment. These sensor nodes are tiny low-cost, self-powered devices have low transmission power and can only send the sampled data to other, nearby sensor nodes each of which forwards the transmitted information to other sensors in WSN till the data reaches the receiving station. In this way, these tiny sensor nodes form a self-organizing, distributed system.

In the following points we give some of the features of WSNs which make them different to other networks [67, 68]:

- The number of nodes in a wireless sensor network is usually much higher than the number of nodes in other wireless networks.
- In WSN, nodes are deployed in a dense manner.
- The sensor nodes failure rate is high.
- The topology in WSN is not always fixed and changes very frequently.
- Nodes in WSN have limited power, computational capacity, and memory.

The sensor network design is influenced by a number of factors like sensor network topology, hardware constraints, fault tolerance, scalability, transmission media, power consumption, production costs, and operating environment. [42]

The network should continue to function even if some nodes fail, this is known as fault tolerance. Due to harsh environmental conditions and lack of power sensor nodes can fail but it should not affect the overall function of the sensor network.

A typical WSN can have tens, hundreds or thousands of nodes. Hence the protocols should be scalable and be able to handle them efficiently. When sensor nodes are deployed in large numbers in a harsh or extreme environment, they cannot be reused, hence it is important to reduce the cost of sensor nodes to reduce the overall cost of the whole network. A sensor node may fail, move or join other networks which change the node density and also network topology. Therefore, network routing protocols should be able to adapt to such changes.

Sensor nodes also have limited capacity for processing and storage and can perform limited computational functions only. Hardware constraints such as these present many issues and challenges in network protocol design and software development of sensor networks.

Since the sensor nodes are connected by a wireless medium, which can be noisy and error prone which may result in frequent disconnection of the sensor network.

Low power consumption is the most important of these factors when we design a sensor network. Sensor nodes mostly are powered by a battery and it is often difficult to change or recharge the batteries, so it is very important to reduce the power consumption of sensor nodes. This will increase their lifetime and the lifetime of the whole network.

Many protocols have been proposed for wireless ad-hoc networks, but they are not well suited for wireless sensor networks due to the different features and application requirements of WSN [65]. WSNs significantly differ from networks like cellular networks and MANETs in which the tasks organization, routing, and management of mobility is done to enhance the Quality of Service (QoS) and great bandwidth efficiency [66]. These networks provide good throughput and delay characteristics in high mobility. In these networks, energy consumption is not as important as the energy stores can be replaced when needed. But wireless sensor networks consist of a large number of nodes made for unattended functioning. The traffic is also much less as compared to the multimedia data in MANETs and cellular networks. The data rate is also very low when compared to conventional networks. So the main objectives are to prolong the life of the network and to conserve energy as the batteries cannot be easily replaced because WSNs usually operate in hostile or remote environments.

2.3 Energy-Aware Wireless Sensor Networks

WSN nodes have to operate for long periods and have limited energy reserves hence they are energy-constrained. Therefore the WSN hardware and the embedded software should have an energy-aware operation. When WSNs were first developed, energy efficiency was of prime importance but as certain new applications have emerged [69], there is even a greater need for extremely small and long-life sensor nodes. Hence there is greater requirement for continued research in energy-efficient operation, energy efficient hardware, efficient power management circuitry and energy-aware protocols. The energy source provides energy to the node, in the energy harvesting from sources such as solar, wind, or a resource such as the mains supply and replacement of batteries. This energy from the energy source is accumulated in an energy store which is a battery or capacitor. From there the energy is used by the hardware components such as; the microcontroller, radio transceiver, or sensors.

But there is a need for energy stores other than batteries due to the increase in energy usage in nodes [70, 71]. Hence many of the research projects are now using supercapacitors to store the energy in nodes [72]. Energy-aware also means that the embedded software on the nodes knows the state of energy of its components. This may include monitoring the energy used from each source [73], checking the rate of consumption of energy by different components [74], and managing the charging of rechargeable batteries or capacitors. Hence these measurements, enable to adjust the operation of the node accordingly to maximize the lifetime of the WSN.

2.4 WSN Components and Hardware Specification

As shown in Figure 1, a WSN may be connected to the internet or other existing infrastructures. There are three major parts: (i) Sensor Nodes, (ii) Sinks and (iii) Gateways. The area where sensor nodes are placed is called a Sensor Field e.g. a desert, a forest, a volcano, a nuclear plant, or even our home. Each of the sensor nodes is connected to a Gateway via a Sink node. A sink node is also a sensor node having more power than normal sensor nodes. Each sampled data packet is transmitted through the sink node and is relayed to the gateway. The gateway node delivers data to other services at a higher level, such as a WSN middleware. A sensor node act as a source node, as a Forwarding or a Sink node. A Source node samples data such as temperature, pressure, etc., and transmits it to its neighbors. The node is in sleep mode or inactive most of the time to save energy and only wakes up periodically to sample and transmit data. A forwarding node or routing node performs some intermediate data processing like data aggregation and relays data from other sensor nodes. A sensor node can act as both source node and a forwarding node. The function of a sink node is to receive all the data generated in the WSN and forward them to the gateway. As all the traffic goes through the sink node, it needs more power and resources than a normal sensor node. The gateway node acts as a bridge between the WSN and other base station and user applications. It can either store data locally or forward it. The task of a base station is to serve as an interface between the end users and the network, as it receives all the information from the network and it sends control information back to the network. The base station can be either a laptop or a workstation.

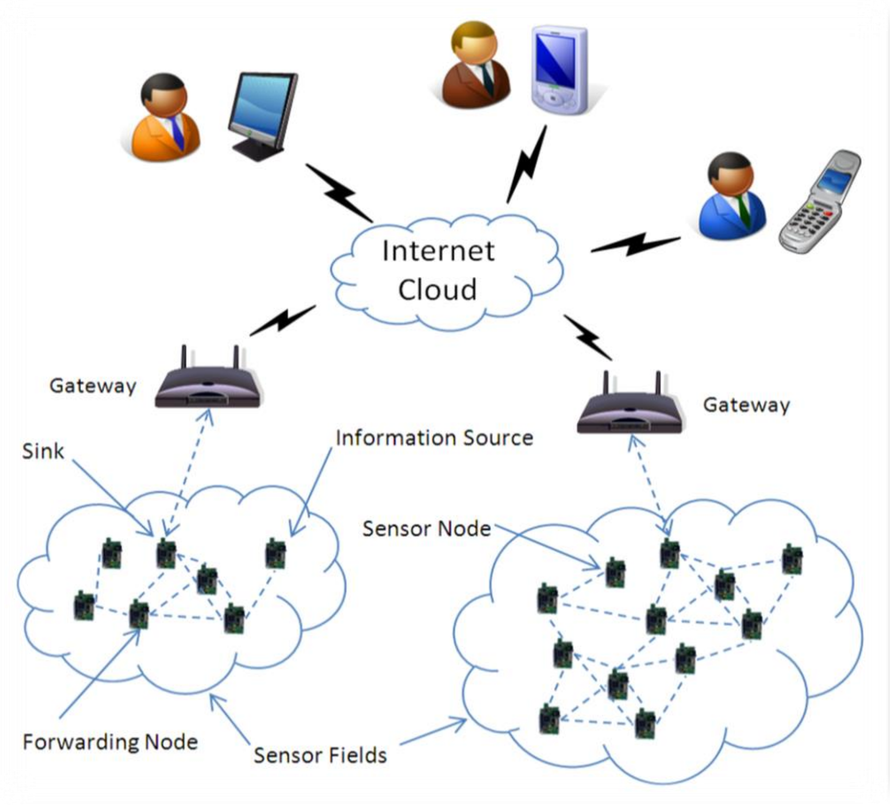


Figure 1 WSN Architecture

A sensor node consists of four basic components (Figure 2): a sensing unit, a processing unit, a radio transceiver and a power unit. Additional components may include location finding systems like GPS, mobilizers and power generators. The sensing unit contains a sensor and an analog to digital converter (ADC). The analog signals measured by the sensors are converted to digital signals by the ADC. Afterward, they are used as input for the processing unit, which manages different mathematical and logical operations and also allows the nodes to collaborate with other nodes and helps it to carry out the different sensing tasks. The processing unit has a limited storage unit associated with it. This can be a type of flash memory, using which the remote nodes sense and store data when they are commanded by a base station, or when they sense an important event. The microprocessor carries out a number of functions such as management of data collected from the sensors, management of power, interfacing the sensor data to the physical radio layer, management of the radio network protocol.

It is important for any sensor node to minimize the power consumed by the whole sensor network. The wireless radio system consumes the most amount of power. Hence data is sent over the radio network only when it is required. The routing protocol determines when to send data based on the sensed event. The sensor network should also minimize the power consumed by the sensor node itself. Therefore, the hardware should be designed to allow the microprocessor to cautiously manage and control power to the radio, sensor, and transceiver. The transceiver is used to connect the node to the overall wireless network. The power unit is the most important component of the sensor node because it implicitly determines the lifetime of the entire network. Power is a resource hard to maintain, basically for two reasons: a) because of the size limitations of the nodes and b) because of the possible harsh conditions of the observed phenomenon can make the nodes inaccessible but on the other hand, it is possible to extend the lifetime of the sensor network by energy scavenging, e.g. using solar cells.

As an example, we show a Crossbow MICAz Sensor Board1 in Figure 3. This sensor node has 128 KiB of memory, its power comes from 2xAA batteries, it consumes 19.7 [mA] in reception and 14 [mA] in transmission. We see that sensor nodes have little energy and computational power at their disposal. The main requirement for working with sensor nodes, therefore, becomes optimizing its energy consumption.

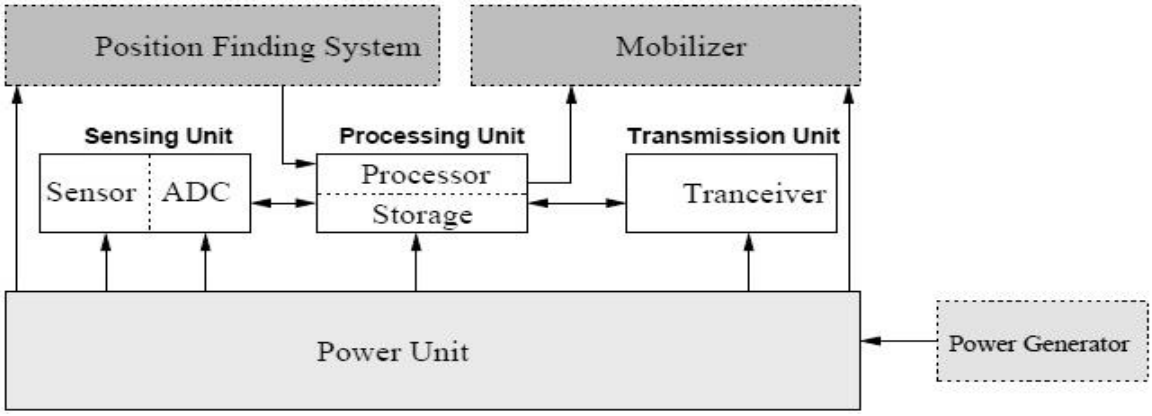


Figure 2 Wireless Sensor Node Components

Figure 3 Mote

2.5 Communication Protocols for WSN

Popular general purpose WSN communication protocols include Bluetooth, IEEE 802.15.4, ZigBee, 6LoWPAN, and RPL.

A. Bluetooth

Bluetooth is an open wireless communication protocol used in the Personal Area Networks for the applications that require the transfer of data over short distances. It operates at frequencies between 2400 and 2483.5 MHz. Bluetooth wireless devices have limited battery life compared to other wireless protocol standards. Since Bluetooth devices have a range of only 10-30 meters depending on the practical circumstances, it hampers the deployment of Bluetooth technology in wider areas [75, 76].

B. IEEE 802.15.4

IEEE 802.15.4 provides lower network layers for Wireless Personal Area Networks and has low power consumption and lower cost. IEEE 802.15.4 has physical layer medium access control. The physical layer maintains the frequency, power, modulation, wireless conditions of the link, and the MAC layer defines data format [77]. There are many technologies such as ZigBee and WirelessHART that have been proposed using the IEEE 802.15.4 as link layer technology. These further extend it by introducing the upper layers which are not defined in the IEEE 802.15.4. IEEE 802.15.4 standard supports Full Function Device (FFD) which has network initialization and acts as a router and Reduced Function Device (RFD) in which nodes only form a star topology and cannot become network coordinator and has a very simple implementation. At least one FFD is should act as the network coordinator. The RFD is generally powered by a battery, while the FFDs have line power. RFD searches for available networks, transmit their data, receive data from the network coordinator, and turn to sleep mode to save power. The FFD can be a network coordinator, routers, or end devices.

C. 6LoWPAN

The 6LoWPAN is a low power wireless PAN system aimed at wireless sensor networks and which is designed on IEEE 802.15.4 technology.

The 6LoWPAN system is used for many applications and also wireless sensor networks. The 6LoWPAN sends data as packets using IPv6 over Low power Wireless Personal Area Networks. The 6LoWPAN uses the IEEE 802.15.4 standard to provide the lower layer elements of this wireless sensor network and defines the encapsulation and compression mechanisms that enable to carry the IPv6 data over the wireless network. To send packet data, IPv6 over 6LowPAN converts the packet data into a format that can be managed by lower layer system of the IEEE 802.15.4. With many low power wireless sensor networks and other forms of ad hoc wireless networks, only 6LoWPAN addresses this area of using IPv6 to carry the data.
Hence 6LowPAN system provides wireless connectivity at low data rates and with a low duty cycle. It has applications in automation, entertainment, control and sensor networks in manufacturing applications or monitoring for industrial or other locations.
6LoWPAN is not as popular as some other standards such as Zigbee but only 6LoWPAN uses IPv6 and this has a distinct advantage as many systems are changing to packet data.

D. RPL

RPL offers a solution for routing challenges in low-power and lossy networks (LLNs). LLNs have limited resources in terms of memory, battery life and processing power. LLN's Wireless Sensor Networks (WSNs), Wireless Personal Area Networks (WPANs), and low power Line Communication networks (PLC). Other routing protocols like OSPF, in general, were not suited for LLNs and RPL was specifically designed for them.
RPL is a distance-vector and a source routing protocol that is for IEEE 802.15.4 physical and MAC layers. RPL is generally used in networks, where nodes periodically send measurements to a collection point. An important feature of RPL is that it has a specific routing solution for low power and lossy networks. The protocol is highly adaptable to network conditions and provides alternate routes when default routes are not accessible. It can also send information over the dynamically formed network topology.

E. ZigBee

ZigBee which uses low power and the low data rate is a low-cost wireless communication protocol which is primarily used in remote control applications [78]. ZigBee devices have a range of 10-70 meters, based on environment and application. It has a highest data rate of 250kbps at 2.4GHz. ZigBee supports all topologies and its nodes can perform three main roles

of ZigBee Coordinator, Router or End-device. It is mostly utilized for very low power operations. The ZigBee network layer supports star, tree, and mesh topologies. In a star topology, the ZigBee coordinator controls the network, and Zigbee end devices communicate with the ZigBee coordinator. In mesh and tree topologies, the ZigBee coordinator creates the network manages it, but ZigBee routers can extend the network. In tree networks, Zigbee routers use hierarchical routing to transfer data and control messages.
Both 802.15.4 and ZigBee have been improved to ensure low power consumption. The low power design has made possible for the battery life to be measured in years, and the network to no longer require constant maintenance.

2.6 WSN Environment Monitoring Systems, Precision Agriculture, and Irrigation Systems

A number of researchers are working on the design, development, and deployment of WSN for environmental monitoring systems. In this section, we discuss some of the existing WSN based environment monitoring systems. Researchers in [79] designed and deployed an indoor WSN real-time monitoring system for indoor air quality by measuring the concentration of CO, temperature, and humidity. In [80] the authors deployed WSN systems for indoor air quality monitoring to measure temperature, humidity, gaseous pollutants, and provide Air Quality Index to access air quality. In [81] the authors developed a WSN for monitoring indoor air quality, access for people’s comfort and health. This system contains a metal oxide semiconductor gas sensor and pyroelectric infrared send or to detect the people. An industrial monitoring system based on WSN was presented in [82] to monitor temperature, humidity, and gas. It used the ARM embedded microprocessor and XBee. In [83] the authors collaborated in a project called SensorScope to provide an efficient and low cost WSN-based environmental monitoring system. The authors addressed the problem of energy by developing an efficient data gathering algorithm to reduce the energy consumption for sensor nodes. In [84] another WSN based environmental monitoring system was proposed for air quality assessment and gas leakages monitoring using catalytic gas sensors and uses techniques to give low energy consumption. Each node is connected to gaseous and meteorological sensors. Four sensor nodes were deployed over an area of 1 km square. An environmental monitoring system SECOAS [85] was developed to study the impact of wind arm on coastal processes. Sensor hardware is based on MCU PIC 18F452, each node is connected to pressure, temperature, and salinity sensors. ASWP [86] is WSN based system for environmental monitoring of a forested outdoor environment in Pennsylvania, USA. The proposed system also considers data duplication, loss, maintenance. In [87] authors implemented a General Packet Radio Service (GPRS) sensors array for air pollution monitoring. The data gathering unit consists of single-chip microcontroller, air pollution sensors array, GPRS modem, and a Global Positioning System. The data gathering unit gathers air pollutants levels (CO, NO2, and SO2. In [88] researchers developed a WSN based marine environmental monitoring system which is energy aware and robust with a large number of sensor nodes. It is based on Ad-Hoc WSN and star topology, having sensor nodes with the local transmission, and control center and data storage for real-time processing. In [89], authors proposed a WSN system, to analyze the effect of climate change on crops. The system had two base stations and numerous sensor nodes powered by solar panels. The climate data acquired from sensor nodes is transmitted to the base station, which transmits it to remote data server using GPRS network. They used simulations in NS-2 to validate the power consumption. In [90] researchers proposed a WSN based environment monitoring systems using ZigBee and GPRS as communication protocols. They built pollution maps using sensor equipment and published the experimental results using Google Maps web services. In [91] the researchers developed a very wide WSN based network to monitor changes in environmental conditions which consisted of 25 base stations spread over 5 countries and sensed data from vineyards. A WSN based environmental monitoring system is proposed in [92] which senses the SO2, NO2, and NO concentration, and also measures temperature, humidity, and air pressure. The system uses an ARM9 to improve the processing speed and reduce the power consumption. In [93] researchers proposed an online WSN based greenhouse gas monitoring system, using XBee and Arduino microcontrollers. A WSN based embedded system was proposed in [94] using ZigBee protocol for remotely controlling greenhouse parameters. The system consists of user laptop and XBee modules, which collect the sensed data from the sensor nodes and send the gathered data to a GUI system for real-time monitoring of various greenhouse gases. Each sensor node has sensors arrays consisting of temperature, humidity, soil, light, and moisture sensors, to monitor greenhouse environment.

The authors in [95] proposed a WSN for crop monitoring in the paddy fields which are salty and acidic. This acidity of the soil is considered a problem which slows the growth of rice. So this system was designed to rinse the soil regularly by water which reduces the acidity and increases the production of rice. Electro-mechanical sensors were used with ZigBee communication systems to monitor the water level and regulate the water. The system sends messages to the farmers. They demonstrated that Zigbee technology has low-cost, low-power consumption, low data-rate. In [96] authors developed a nitrogen management system for paddy rice fields for accessing the correct rate of fertilizer nitrogen to be applied. This system simulated the growth stages, nitrogen used, yield based on soil nitrogen. In [97] the authors developed a system for precision irrigation using WSN to monitor soil moisture, soil temperature, and relative humidity. The authors of [98] proposed a Wi-Fi-based smart WSN for the agricultural environment to monitor temperature, humidity, light, air pressure and soil moisture. They claimed to have reduced the cost and to have enhanced the flexibility and mobility of the system. The authors in [99] developed smart sensing systems using ZigBee for monitoring environmental parameters such as temperature, relative humidity, pressure, and sunlight. They used a microcontroller as a smart weather station. They used XBee modules to increase range and reduced the energy consumption. In [100] the authors proposed a remote sensing and control automated irrigation system using distributed WSN for automated irrigation, real-time in field sensing, to minimize the use of water. In [101] authors developed a WSN based real-time system to monitor water content of the soil. The system was composed of a base station unit, valve units, and sensing units and was used to monitor cherry trees. They showed that the system had low cost and it was reliable and prevented moisture stress in trees, minimized excessive use of water and also ensured rapid growth. In [102] the researchers proposed a routing algorithm that worked by accessing water level and by computing transmission range. Hence the algorithm was based on distances from source to sink node. Their system was optimized by the use of genetic and neural network algorithms.

2.7 Routing in Wireless Sensor Network

The main responsibility of the sensor nodes in each application is to sense the targeted area and transmit their collected information to the sink node for further operations. Resource limitations of the sensor nodes and unreliability of low-power wireless links [12], in combination with various performance demands of different applications, impose many challenges in designing efficient communication protocols for wireless sensor networks [13]. Meanwhile, designing suitable routing protocols to fulfill different performance demands of various applications is now considered an important and hot issue in wireless sensor networking. In this context, researchers have proposed numerous routing protocols to improve performance demands of different applications through the network layer of wireless sensor networks protocol stack [14, 15]. Most of the existing routing protocols in wireless sensor networks are designed based on a single-path routing strategy without considering the effects of various traffic load intensities. In this approach, each source node selects a single path which can satisfy performance requirements of the intended applications for transmitting its traffic towards the sink node.

Although route discovery through single-path routing approach can be performed with minimum computational complexity and resource utilization, the limited capacity of a single path significantly reduces the achievable network throughput [16, 17]. Furthermore, the low flexibility of this approach against node or link failures may significantly reduce the network performance in critical situations. For instance, whenever the active path fails to transmit data packets (as a result of the limited power supply of the sensor nodes, high dynamics of wireless links and physical damages), finding an alternative path to continue data transmission process may cause extra overhead and delay in data delivery. Therefore, due to the resource constraints of sensor nodes and the unreliability of wireless links, single-path routing approaches cannot be considered effective to meet the performance demands of various applications.

In order to cope with the limitations of single-path routing techniques, another type of routing strategy, known as multipath routing has become a promising technique in wireless sensor and ad hoc networks. Dense deployment of the sensor nodes enables a multipath routing approach to construct several paths from individual sensor nodes towards the destination [18].

Discovered paths can be utilized concurrently to provide adequate network resources in intensive traffic conditions. Alternatively, each source node can use only one path for data transmission and only switch to another path upon node or link failures. The latter one is mainly used for fault-tolerance purposes, and this is known as alternative path routing.

In the past decade, multipath routing approach has been widely utilized for different network management purposes such as improving data transmission reliability, providing fault-tolerance routing, congestion control and Quality of Service (Qu's) support in traditional wired and wireless networks.

However, the unique features of wireless sensor networks (e.g., constrained power supply, limited computational capability, and low-memory capacity) and the characteristics of short-range radio communications (e.g., fading and interference [19, 20]) introduce new challenges that should be addressed in the design of multipath routing protocols. Accordingly, existing multipath routing protocols for traditional wireless networks (such as ad hoc networks) cannot be used directly in low-power sensor networks. During the past years, this issue has motivated the research community of wireless sensor networks to develop multipath routing protocols which are suitable for sensor networks.

In [14], routing challenges and design issues in wireless sensor networks were presented. They classified all the existing routing strategies based on the network structure and protocol operation. Researchers in [21], provided a brief overview on the existing fault-tolerant routing protocols in wireless sensor networks and categorized these protocols into retransmission-based and replication-based protocols. The authors in [22] and [23], classified the existing multipath routing protocols in ad hoc networks based on the primary criterion used in their design. Accordingly, the principal motivation of conducting this research was lack of a comprehensive survey on the proposed multipath routing protocols for wireless sensor networks. To the best of the author's knowledge, this work is first of its kind that is being conducted to classify and investigate the operation as well as benefits and drawbacks of the existing multipath routing protocols in sensor networks.

2.7.1 Ad-hoc On-Demand Distance Vector (AODV) Routing

AODV is a reactive protocol that discovers routes only on the basis of demand using a route discovery mechanism. It uses traditional routing tables with one entry per destination. Without using source routing, AODV relies on its routing table entries to propagate an RREP (Route Reply) back to the source and also to route data packets to the destination. AODV uses sequence numbers maintained at each destination to determine the freshness of routing information and to prevent routing loops [24]. All routing packets carry these sequence numbers.

AODV maintains timer-based states in each node for utilization of individual routing table entries, whereby older unused entries are removed from the table. Predecessor node sets are maintained for each routing table entry, indicating the neighboring nodes sets which use that entry to route packets. These nodes are notified with RERR (Route Error) packets when the next-hop link breaks. This packet gets forwarded by each predecessor node to its predecessors, effectively erasing all routes using the broken link. Route error propagation in AODV can be visualized conceptually as a tree whose root is the node at the point of failure and all sources using the failed link as the leaves [24]. The main advantage of AODV compared to other routing protocols is that less memory space is required as information of only active routes are maintained, which, in turn, increases the performance. While on the other side, the disadvantage is; this protocol is not scalable and in large networks, it does not perform well and is not capable to support asymmetric links.

2.7.2 Ad-hoc On-demand Multipath Distance Vector Routing (AOMDV)

Ad-hoc On-demand Multipath Distance Vector Routing (AOMDV) [25] protocol is an extension to the AODV protocol for computing multiple loop-free and link disjoint paths [24]. The routing entries for each destination contain a list of the next-hops along with the corresponding hop counts. All the next hops have the same sequence number. This helps in keeping track of a route. For each destination, a node maintains the advertised hop count, which is defined as the maximum hop count for all the paths, being used for sending route advertisements of the destination. Each duplicate route advertisement received by a node defines an alternate path to the destination. Loop freedom is assured for a node by accepting alternate paths to the destination if it has a less hop count than the advertised hop count for that destination. Because the maximum hop count is used, the advertised hop count, therefore, does not change for the same sequence number [24]. When a route advertisement is received for a destination with a greater sequence number, the next-hop list and the advertised hop count are reinitialized.

AOMDV can be used to find node-disjoint or link-disjoint routes. To find node-disjoint routes, each node does not immediately reject duplicate RREQs. Each RREQ arriving via a different neighbor of the source defines a node-disjoint path. This is because nodes cannot broadcast duplicate RREQs, so any two RREQs arriving at an intermediate node via a different neighbor of the source could not have traversed the same node. In an attempt to get multiple link-disjoint routes, the destination replies to duplicate RREQs, the destination only replies to RREQs arriving via unique neighbors. After the first hop, the RREPs follow the reverse paths, which are node-disjoint and thus link-disjoint. The trajectories of each RREP may intersect at an intermediate node, but each takes a different reverse path to the source to ensure link disjointness [24].

The advantage of using AOMDV is that it allows intermediate nodes to reply to RREQs, while still selecting disjoint paths. But, AOMDV has more message overheads during route discovery due to increased flooding because it is a multipath routing protocol, the destination replies to the multiple RREQs which results in longer overheads.

2.7.3 Routing Protocols Comparison

Many routing protocols have been proposed for the Mobile Ad Hoc Network and classified as proactive or Table-Driven Routing Protocol, Reactive or on Demand Routing Protocol and Hybrid Routing Protocol.

Proactive or Table-driven routing protocols

In proactive protocols, each node maintains an individual routing table containing routing information for every node in the network. Each node maintains consistent and current up-to-date routing information by sending control messages periodically between the nodes which update their routing tables. The proactive routing protocols use link-state routing algorithms which frequently flood the link information about its neighbors. The drawback of proactive routing protocols is that all the nodes in the network have to maintain an updated table which will consume already scarce resources such as memory and CPU usage. Some of the popular proactive routing protocols are Destination-Sequenced Distance-Vector Routing (DSDV) [26] and Optimized Link State Routing Protocol (OLSR) [27], [29].

Reactive or On-Demand Routing Protocol:

In Reactive routing protocols, when a source wants to send packets to a destination, it invokes the route discovery mechanisms to find the route to the destination. The route remains valid till the destination is reachable or until the route is no longer needed. Unlike table-driven protocols, all nodes need not maintain up-to-date routing information. Some of the commonly used on-demand routing protocols are Dynamic Source Routing (DSR) [27] and Ad hoc On-Demand Distance Vector (AODV) [26].

Hybrid Routing Protocol:

Hybrid routing protocol combines the advantages of both proactive and reactive routing protocols. The routing is initially established with some proactively prospected routes and then serves the demand from additionally activated nodes through reactive flooding. Some of the existing hybrid protocols are Zone Routing *Protocol (*ZRP) [28] and Temporally-Ordered Routing Algorithm (*TORA*) [30]. In this section we present a detailed performance and energy consuming comparison of important routing protocol for mobile and ad hoc wireless networks.

To evaluate the performance of routing protocols, we use four different quantitative metrics to compare the performance of the selected protocols. They are

- **Packet Delivery Fraction:** The fraction of packets sent by the application to the packets received by the receiver [31].
- **Average End-to-end delay:** End-to-end delay indicates how long it took for a packet to travel from the source to the application layer of the destination [32].
- **Throughput:** The throughput is defined as the total amount of data a receiver receives from the sender divided by the time it takes for the receiver to get the last packet [33].
- **Energy consumption:** the total energy that each node consumes

We consider two comparisons; i.e. The 1st one is routing protocol properties or performance constraints and the 2nd is performance and energy consumptions. AOMDV, OLSR and TORA have been chosen for this comparison.

Table 1 shows the performance constraints or protocol properties for each protocols

Performance Constraints	OLSR	AOMDV	TORA
Category	Table Driven or Proactive	On Demand or Reactive	Hybrid
Protocol Type	Link State scheme	Distance Vector	Link Reversal
Route Maintained in	Route Table	Route Table	Route Table
Loop Freedom	Yes	Yes	Yes
Rout Philosophy	Flat	Flat	Flat
Multiple Routes	No	Yes	Yes
Multicast	Yes	Yes	No
Message Overhead	Minimum	Moderate	Moderate
Periodic Broadcast	No	Yes	Yes

Table 1 Comparison of Ad Hoc Routing Protocols

Table 2 shows Routing protocol performance and energy consumption for each protocol

Protocol	End to End Delay	Packet Delivery Ratio	Throughput	Energy Consumption
OLSR	Low	Average	Average	Average
AOMDV	Low	High	High	Average
TORA	High	Low	Average	High

Table 2 Protocol performance and energy consumptions

Based on the detailed comparison being performed, AOMDV seems to be the perfect candidate to meet the objectives of our project.

2.8 Related Work (AOMDV)

In [43] researchers improved the performance of standard AOMDV in conditions like mobility or multi-communication. They proposed link reliability in route choice. They modified the Route request process to enable reliable paths using Bit Error Rate (BER). They tested the effectiveness of the new protocol by considering these improvements under realistic conditions and the result was compared to standard AOMDV and AODV protocols to show improved performance.

Researchers in [44] modified AOMDV protocol by proposing a new fuzzy logic based scheme. The proposed protocol was shown to select better paths and increase network survivability. The proposed protocol considered many selection criteria, where some represented network status and others were issued by preventive, reactive and tolerant saving lines. They carried out simulations to compare the modified AOMDV protocol with AODV and AOMDV. They showed the proposed protocol's survivability under different conditions.

An AOMDV based method (E-AOMDV) was proposed by [45], to conserve energy, find the shortest path and for load balancing. In order to conserve energy, they defined an energy factor as selection criteria, which was defined as the product of energy factors of all nodes on different paths. The status of energy was denoted by Energy factor. They compared the performance of AOMDV and E-AOMDV to show that the lifetime of proposed E-AOMDV protocol was limited but it showed improved routing compared to AOMDV without including the energy factor. So they concluded that performance of the proposed method was better in a limited lifetime and showed better results with the new E-AOMDV protocol.

In [46], the authors proposed a modified AOMDV protocol called Network Coding-based AOMDV (NC-AOMDV) routing algorithm for MANET. The proposed method tried to increase data transmission reliability and ensure load balancing. They compared NC-AOMDV routing protocol to AOMDV routing protocol in simulation based on packet overhead, packet delivery ratio, and average end-to-end delay during packet transmission. The results in simulations showed that NC-AOMDV routing protocol was accurate and efficient and provided route stability in dynamic MANET.

An algorithm finds maximal nodal remaining energy was proposed in [47], called as Delay Remaining Energy for AOMDV (DRE-AOMDV) routing protocol. It claimed to get a solution for finding maximal nodal remaining energy for all routes in selecting a path for the end-to-end requirement. The protocol was specifically for route failures caused by lack of energy. The proposed protocol showed significant network performance improvement in terms of energy consumption, packet delivery and network lifetime.

An extension to AOMDV routing protocol was proposed by [48]. It was channel adaptive routing protocol to accommodate channel fading. The proposed algorithm was called Channel-Aware AOMDV (CA-AOMDV). It used channel average nonfading duration as routing metric. Using this, it selected stable links for path discovery by applying a pre-emptive handoff strategy. By doing this, it was able to maintain reliable connections and exploit channel state information to ensure that the paths were reused when available, instead of being discarded. Simulation results were provided for downtime and lifetime multiple path systems. The authors also provided theoretical expressions for network performance measures and evaluated the differences in performance between CA-AOMDV and AOMDV. The Simulation results showed that CA-AOMDV gave better network performance than AOMDV.)

A Modified AOMDV Routing Protocol for Maritime Inter-ship Communication [49] was proposed that provides routing recovery mechanism when a link breaks in an active route to reduce lost packets, this will reduce packet loss ratio and delay time.

M-AOMDV applied to ship communications that are using UHF and VHF, these modifications use the existing AOMDV protocol and enhance the inter-ship communication by adding new recovery methods for a broken link. The normal AOMDV protocol uses one recovery method to recover from broken links; so if a link breaks, this will lead to packet loss and increases the end to end packet delay. In this modification, there are two new recovery methods that will try to increase the recovery process and decrease packet loss and end to end delay.

M- AOMDV had good performance results as the delay of the AOMDV protocol is longer than M-AOMDV, and the overall delays of these two protocols are directly related to their speeds of execution. The link repair mechanism of M-AOMDV can reduce delays, to a certain extent, by requesting routing entities forward data packets as soon as possible in order to complete link repair to the destination node when the node detects that the link has been interrupted. The results show that M-AOMDV reduces average delay and packet loss ratio. Also, this proposed method increases the potential for the ad-hoc network mode to be applied to ship networks on shoreline areas.

Another modified version of AOMDV was proposed in [50], called ant-AOMDV or ant colony optimization (ACO) modification for AOMDV in MANET, in this research the writers used the modified AOMDV for multipath routing using ant colony for mobile ad hoc networks (MANETs). The final result was a comparison between ant-AODV and ant-AOMDV.

The idea behind the working of ant-AODV and ant-AOMDV is that the RREQ message packets are sent via a single path in the case of Ant-AODV based routing and to multiple paths in the case of Ant-AOMDV based routing. RREQ message packets can be termed as a pheromone in terms of the standard algorithm of ant colony optimization (ACO) used by the ants.

Parameters like Quality of Service (QoS) will suffer unless special schemes are developed to sustain such networks because in multi-hop routing no default route is available. Each node acts as a router and forwards each other’s packets to enable information sharing between mobile nodes.

Ant Colony optimization falls into a class of biologically inspired algorithms that have recently been developed. The Ant Algorithm mimics the behavior of ants in nature while they are searching for food. Particle swarm optimization is inspired by the behavior of flocks of birds as they fly in search of food. These nature-inspired techniques share a common characteristic, the whole information about the state of the system is contained not in a single entity, but rather some part of the information is stored in many of the entities.

The main idea behind the proposed algorithm is that nodes in the network periodically and asynchronously send out artificial ants towards possible destination nodes of data. These ant agents are small control packets, which have the task to find a path towards their destination and gather information about it. This pheromone takes the form of routing tables maintained locally by all the nodes of the network. They indicate the relative quality of different routes from the current node towards possible destination nodes.

In Ant-AOMDV multipath routing is done and it sees all possible routes. So, packets can be sent to all the selected paths and this helps in load balancing. Routing overhead is also minimized which is shown with the help of parameters Network Routing Load (NRL) and Routing Overhead respectively. The other parameters such as Packet Delivery Fraction (PDF), number of sent packets, number of received packets and number of route request packets sent (RREQ) also shows a vast improvement in Ant-AOMDV.

Another researcher [51] published a paper about AOMDV-PAMAC, they started their paper explaining that power consumption of nodes in ad-hoc networks is a critical issue because they operate on batteries, they suggested a new link layer algorithm knows as Power Aware

Medium Access Control (PAMAC) protocol, which enables the network layer to select a route with minimum total power requirement among the possible routes between a source and a destination. The first few nodes whose battery power is drained to the threshold value are pushed to the exterior part of the network and the nodes in the exterior are brought to the interior. So they used AOMDV using PAMAC as the mac layer protocol and the average power consumption.

AOMDV-PAMAC uses the basic ideas of Power Efficient Battery Capacity Routing (PEBCR) and it incorporates these features into the MAC layer as it is essential to minimize the total transmission power consumption. PAMAC protocol is incorporated in the link layer and simulation works are carried out using GloMoSim by considering thirty nodes randomly distributed in an area of 2000 x 2000 m. The AOMDV routing protocol was incorporated in the network layer and its performance was evaluated under CBR traffic with PAMAC as link layer protocol. Multipath routing protocols compute multiple paths during route discovery to avoid high overhead and latency. It was observed the performance of AOMDV, which is a multipath routing protocol relative to AODV, and the Link layer protocol PAMAC which is the modification to the Multiple Access with Collision Avoidance (MACA) Protocol, when these two protocols are applied simultaneously, it is produced good results compared to the other protocol combination.

There are multipath transport layer protocols such as multipath TCP [111], which efficiently uses several Internet paths between a pair of hosts while showing a single TCP connection to the application layer. It offers benefits like better resource utilization, better throughput and smoother reaction to failures. Hence it can be used in the above-discussed modifications of AOMDV, to improve their reliability and performance.

Based on the detailed comparison being performed, AOMDV performance was found to be the best and the perfect candidate to meet the objectives of our project. So we studied many research works which tried to improve AOMDV so that it can be applied to WSN and Environmental monitoring. The above research works tried to improve accuracy, performance, effectiveness, packet delivery, end-to-end delay, energy-aware operation, reliability, packet delivery and network lifetime. But none of the above studies address all these issues simultaneously. For successful operation of WSN for environmental monitoring, we need a protocol that is reactive, event-driven, energy efficient and robust that is capable of providing timely and reliable information. Since our study is targeting the Middle East regions where we have sudden outpours of inconsistent rainfall, we need event-based protocol for these regions. Also as in these regions, we have a scarcity of water, we need a system to utilize the irrigation water to the best. For all these reasons we need event-based, reactive, robust and energy aware routing protocol. This research gap needs to be filled to improve the performance of WSN for rainfall monitoring, particularly in the Middle East regions.

2.9 Chapter Summary

In this chapter, we reviewed wireless sensor networks and their applications in environmental monitoring and in other fields. We studied how the WSN have been applied to environmental monitoring. The energy-aware usage of WSN was also briefed. We also studied common WSN architecture, the different WSN components, and the hardware specifications.

We looked at some latest WSN communication protocols which include Bluetooth, IEEE 802.15.4, 6LoWPAN, RPL, ZigBee. We reviewed some latest WSN based environment monitoring systems, Precision agriculture, and Irrigation systems. Then we discussed routing mechanisms in WSN and some popular routing protocols which fall in the classes of Proactive, Reactive and Hybrid routing protocols. We also compared the Ad-Hoc On-Demand Distance Vector (AODV) routing and Ad-Hoc On-Demand Multipath Distance Vector routing (AOMDV) routing protocols with some other popular routing protocols. Performance evaluation of the protocols was using quantitative metrics such as End-to-End Delay, Packet Delivery Ratio, Throughput, and Energy consumption. Finally, we discussed some latest studies that are closely related to our work and we highlighted the research gaps that need to be filled to apply WSN for environmental monitoring.

Chapter 3: Research Methodology

In this third chapter, we will discuss the research methodology used in this thesis. This can be divided into two large mechanisms, which involve using simulation and building a small prototype of the system.

Therefore we will look at several simulation packages to determine which simulation would be most suitable for our work. We will also look at different hardware sensor systems to build our prototype.

3.1 Simulation

This thesis intends to carry out the experimental analysis of routing heuristics and protocols by the use of simulation. The exploring of sensor networks is usually done by using one of the following three techniques: (1) analytical methods, (2) practical implementations, and (3) computer simulations. The analytical methods are unsuitable or inaccurate due to the constraints and complexity of WSN. Analysis through practical implementation is comparatively less prevalent because of deployment cost and application dependence of WSNs. Hence simulation is the most widely used method for analyzing WSN as it allows rapid evaluation, optimization, and modification of proposed algorithms and protocols. Some areas of network operation can be simplified for example assuming that packet collisions, interference, and noise do not occur and that nodes are always perfectly synchronized with one another etc. Hence simulation makes the process of development and evaluation quick and easy, but sometimes algorithms are not practically realizable. Therefore a simulation is only as realistic as the models and assumptions that it is based upon.

3.2 Simulators

Due to different design aims and strategies of different simulators they having different advantages and disadvantages. We need to carefully select a simulator, based on these properties. Simulators can be classified into two main categories: those that provide extensions to existing network simulators like the SensorSim [104], NS-2 [103], and others that have been developed specially for the WSN simulation like J-Sim [109]. Here, we discuss an overview of the design and architecture of some of the popular WSN simulators:

NS-2 [103] is very popular simulation tool for sensor networks. Its main features are that it is an object-orientated discrete event network simulator which is written in C++. NS-2 is more complicated to use than other tools, but as it is popular, researchers still learn it. It is also extensible, which is another main advantage, with implementations being widely developed by the research community. NS-2 also allows its users to define simulation parameters which allow the user to specify what happens to the network over a period of time. For example, the occurrence of an event, the movement of nodes. Hence the user can examine the effect of a routing protocol under particular conditions without writing code for this event in the application code. NS-2 requires parameter files to be written in TCL. It does not have Graphical User Interface (GUI).

NS2 consists of two key languages: C++ and Object-oriented Tool Command Language (Tcl) while the C++ defines the internal mechanism (i.e., a backend) of the simulation objects, the OTcl sets up simulation by assembling and configuring the objects as well as scheduling discrete events (i.e., a frontend). The C++ and the OTcl are linked together using TCL after simulation.

SensorSim [104] is an extension of NS-2 built for the simulation of WSNs. It has advanced models and the ability to interact with external systems like sensor network hardware but it has scalability problems.

OMNeT++ [105] also is a discrete-event network simulator like the NS-2 but it is modular in nature having modules like layers of a protocol stack which contain algorithms and more complex modules like such as a sensor node. Hence these modules can communicate with each other using messages. It has good Graphical User Interface (GUI) which makes the user interact easily with the network.

SenSim [106] is WSN simulator which is an extension for OMNeT++. It is modular in nature with modules used to represent each protocol layer, the hardware, and a coordinator.

There are modules outside of the nodes which represent a sensor channel and a network channel. However, SenSim is not popular with simulators due to steep learning curve. There is also small user base hence there are not many developed protocols available for it.

OPNET [107] is an object-orientated network simulator, which was developed for military use, but now is popular as a commercial network simulation tool. It enables to simulate

heterogeneous networks with different protocols. The system behavior is simulated by modeling all the events in the system and then processing it by user-defined processes.
OPNET is not free for commercial use but offers free licenses for educational use. It consists of C and C++ source code blocks with a library of OPNET functions. But it has disadvantages due to scalability and extensibility issues and hence not widely used for WSN simulation.
J-Sim [109] is a simulator developed to counter scalability issues in object-orientated models. It has a component oriented structure which increases its scalability, and it is implemented in Java to make it platform independent. It is also complicated to use and thus has a limited code base.

To validate the result of the proposed protocol in this phase of the project we have to use one of the known simulation tools. Having examined these possible simulators, NS-2 was chosen to carry out the simulations, since it is an event-driven simulation tool that has proved useful in studying the dynamic nature of communication networks. Simulation of wired as well as wireless network functions and protocols (e.g., routing algorithms, TCP, UDP) can be done using NS2. It consists of two simulation tools. The network simulator (ns) contains all commonly used IP protocols. The network animator (NAM) is used to visualize the simulations. It is also easier to modify protocols in NS-2. It also does not require costly equipment, hence it is cheap. In NS-2 Complex scenarios can be easily tested, it is also fast and results can be quickly obtained. It supports many protocols and multiple platforms. It has a modular approach. Because of these advantages over other simulators, we chose NS-2 for the current work.
In this simulation, we use the TCL language to build the scenario and use C++ language to build AOMDV-E by modifying the AOMDV routing protocol source code. The source code of the simulation is written in the TCL language and runs the simulation example on ns-2.35 Linux Ubuntu version 12. The simulations have been done in three scenarios the 1st one consists of 5 nodes with one of them affected by rain, the 2nd one consists of 7 nodes 2 two of them affected by rain the last one, which is the complex one consists of 30 nodes 5 of them affected by rain.

3.3 Building Prototype

Since it is the starting point in our research, we are building a prototype first. Afterward with more research when it achieves stability it will be used in the fields. The architecture is also general, so it is easy to integrate the driver of other sensor probes, thus collecting more types of data for different research purposes. Limited battery supply is a major concern when utilizing WSNs to build real applications. Therefore, we utilize solar panels and rechargeable battery to address this challenging problem. We illustrate our detailed design to packaging WSN nodes for outdoor development and present collected data to validate our design. Specifically, using sand soils with different water content, we demonstrate the result of our system. With enough details for researchers to understand and thus improve our design, our system could be a good starting point for their own research. Hence this prototype is the starting point of the current research.

The proposed wireless sensor network could be used to monitor spatial variations in soil moisture as well as to detect the presence of rainfall over time. Our experimental Smart Grid testbed is set up in a garden to monitor surface soil moisture and presence of rainfall. Our long-term objective is to monitor soil moisture regularly so as to implement targeted irrigation techniques. The data we gather from our system will also be used in future for managing the irrigation resources throughout Jordan by providing improved guidelines for the Jordan irrigation management model.

The reactivity of the system should be also very high to detect the occurrence of the event as soon as possible, as in this case, rainfall. When such an event is detected the data or sensor readings should be sent to the base station reliably and quickly and for this reason, we need to select the right routes. This data should be sent to the base station at a higher priority than all other data being sent [54].

Wireless systems are likely to face errors as the environment in which these sensors are deployed are harsh environments which might have severely dry conditions and can affect their performance. Hence we need a very reliable system which ensures that information is correctly and accurately routed to the base station. The sensor network must also ensure heavy data flow in the event of rain so that no routes or links are overwhelmed with data. We need to manage the information flow and to also reduce redundancy from data to reduce traffic.

Finally, since there is always energy crises in the case of wireless sensor networks thus we need to carefully plan each activity to maximize battery life and also take advantage of the high duration of sunlight available in the Middle East Region. We still need power management as a lot of data will be sent when a wireless node is experiencing rainfall and it needs to send a lot of information and a lot of power will be required at this time. Hence the sensor network needs to strike a perfect balance and trade-off between the power left and the information required to be sent.

We need an extended WSN lifetime for the soil moisture monitoring system because we cannot afford a breakdown of the data routing operation during a rainfall event when we need to respond quickly to the change in the soil moisture level. We are using ZigBee medium range communication module standard, which is based on the IEEE 802.15.4 [55], which has a lot of advantages such as it consumes less power, also saves power as it has sleep mode, has a large capacity network, is more reliable and has low cost of related components [56, 57]. The authors in [58] have proposed to reduce energy consumption by using a routing algorithm which is a combination of Ad hoc On-Demand Distance Vector Routing Junior and the Cluster-Tree technique. But there is no practical implementation for all these research initiatives. In our system, power is directly saved by using event-based routing algorithm hence reducing the data at times of raining when a lot of data is produced. However even after raining, data should be sampled more frequently in order to keep track of the soil moisture.

3.4 WSN Architecture

The nodes will be installed in fixed locations. The distance between each node will be around 150 meters and the connection between the nodes and base station will be handled by ZigBee. At each sensor node, we have a soil moisture sensor and rainfall sensor attached to it. Our sensor network is based on Libelium Waspmotes sensor nodes and ATmega1281rds Microcontroller [59]. It is connected to following components:

- Wireless Interface (Waspmote XBee based on IEEE 802.15.4 standard)
- Soil moisture sensor(Watermark 200SS)
- Rainfall sensor(Libelium Weather Station WS 3000)
- USB-PC Interface (Waspmote Gateway)
- Solar Panel (Waspmote rigid solar panel 7V-500mA)
- 6600mA Li-Ion rechargeable batteries

All the sensor nodes are connected to the base that is a computer using a USB-PC gateway. All the nodes act as sampling nodes and hence have both soil moisture sensor as well as rainfall sensor attached to them. The current network deployment has four sampling nodes, but more generally the architecture supports many nodes [59].

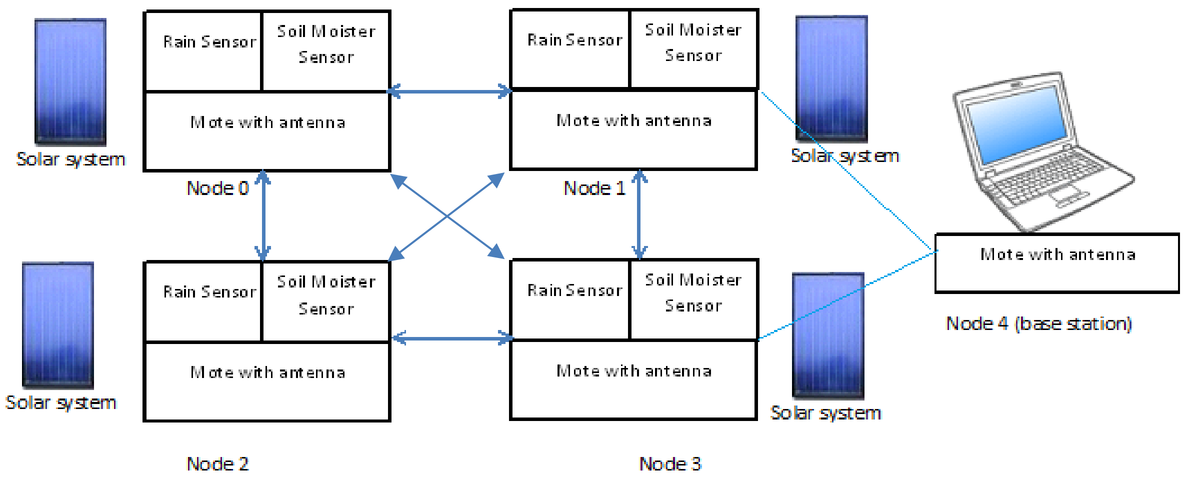


Figure 4 Testbed Block Diagram

Zigbee RF nodes are used as communication models to transmit the sampled data to the gateway, these can have the best average lifetime of 8 years using dual batteries 3V, at 3000mAh. Some studies like [55], showed that ZigBee model based on IEEE 802.15.4 standard with correct topology can last up to 10 years. Hence we are using ZigBee communication model for our research.

3.5 Testbed Hardware

(1) Sensor Nodes. We are using Waspmote WSN which is an open source wireless sensor platform and has speciality in implementing sensor nodes having low power consumption capability [55] and hence our sensor nodes can work autonomously, working on battery power and having a maximum lifetime of 7-8 years [57] based on the cycle of operation, communication protocol and the communication module used. The architecture of Waspmote is based on the Atmel ATMEGA 1281 microcontroller. The Atmega1281 microprocessor is based on the high-performance group of Atmel 8-bit AVR RISC based microcontrollers [59]. Its advantages are low power, high performance and advanced architecture based on RISC which has a maximum of 20 MIPS throughput and 23 I/O lines.

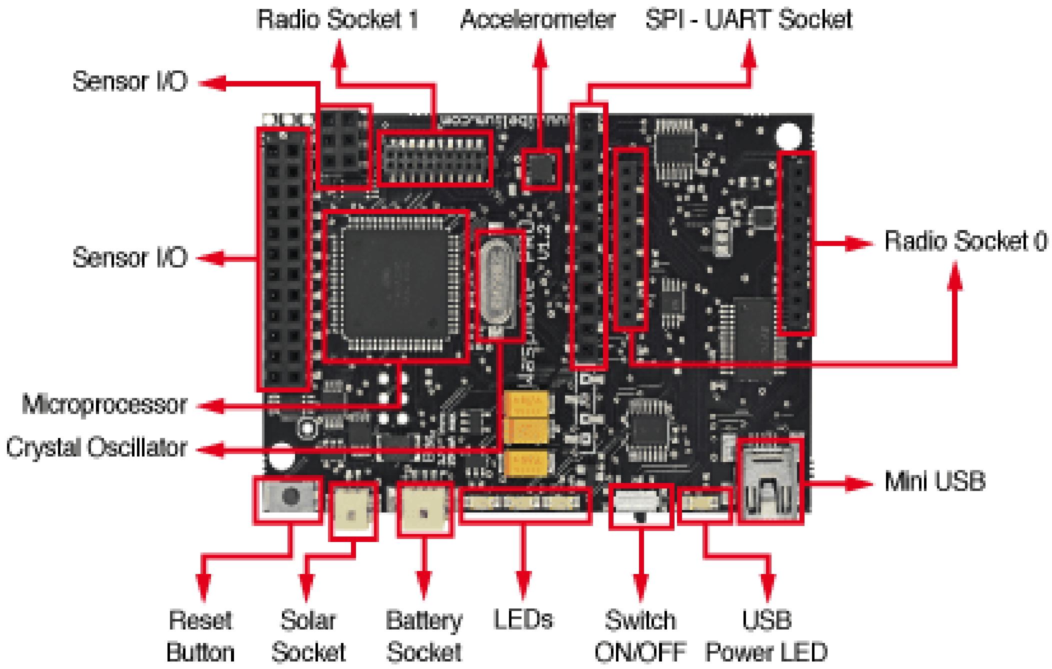


Figure 5 Waspmote Sensor Node

The technique used by Waspmote to save energy is called blocking in which the program is blocked in sleep mode until an event occurs which causes interrupts and then the program is executed. This way when an interrupt occurs, only then are the associated functions executed, thus saving energy.

Waspmote has the ability to operate in very adverse conditions containing noise due to electromagnetic radiations hence it can provide very stable communication at all times. It has different types of communication modules connected by a serial line to the XBee, GPRS, USB and it can offer a great transmission speed of about 115200bps for these interfaces at a 100% success rate.

As shown in Figure 5, Waspmote has a mechanism to set an absolute time by using real-time clock (RTC) which runs at 32 KHz. We have total control over when the mote wakes up from sleep or hibernate mode to record values and perform other programming actions. This RTC also allows the Waspmote to work in the maximum energy saving modes when in Deep Sleep and the Hibernate modes and to wake up right at the correct moment. We are using the XBee-802.15.4 RF communication modules [53] which have a firmware (DigiMesh) with the help of this firmware they create mesh networks instead of the usual Star topology networks.

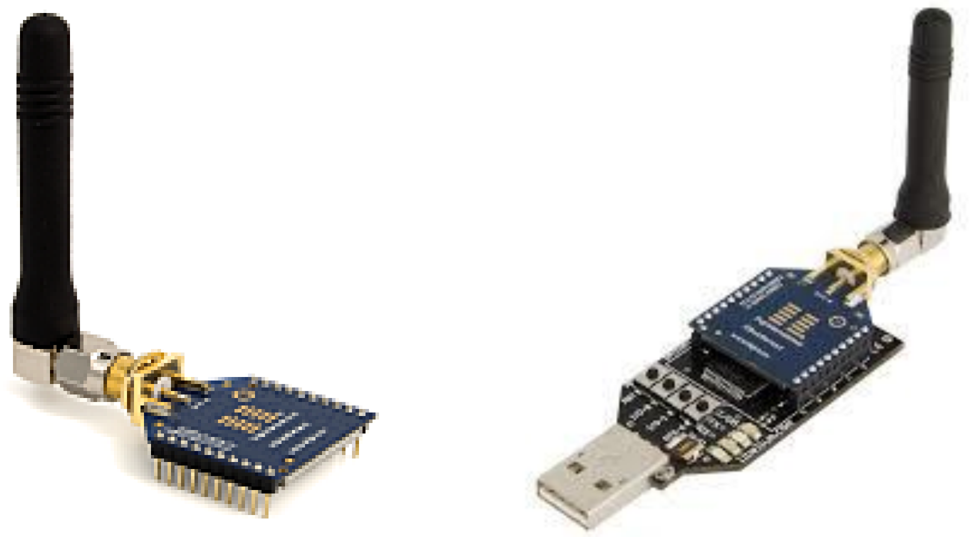


Figure 6 XBee RF Module and USB PC Gateway

Following are the some of the characteristics of the firmware protocol implemented Digi for XBee-802.15.4: It has Self-Healing mechanism: which makes it possible for any node to join or leave the network at any point of time. It has Peer-to-Peer architecture: which means that all nodes are equal without any hierarchies, coordinator node and any father-son relationships. It uses the Silent protocol: which can considerably reduce the routing overhead due to using a reactive protocol similar to AODV (Ad hoc On-Demand Vector Routing). Dynamic Route finding: by using this characteristic the module does not keep the route map but finds the route dynamically when they are needed. Sending fewer Acknowledgements: only the recipient of the message responds to route messages. Increased reliability of data transmission is obtained by sending acknowledgments. Sleep Modes: low energy consumption modes with synchronization are used to wake at the same time.

This RF module has a long range of 7000m but lower power consumption than Wi-Fi (802.11). It operates at a very low data rate of 250 kbps at 2.4 GHz [61, 62]. These modules have a mechanism to extend battery life by using sleep modes.

(2) Soil Moisture Sensor. The Soil Moisture Sensor [60] is one of the most important components in our research on which the efficiency of our study heavily relies. There were many soil moisture sensors available and each had its advantages based on accuracy, and soil texture, cost, and reliability, and we chose, the Watermark 200SS [60] developed by Irrometer since it satisfied most of the above criteria. It has very satisfactory features like stable calibration, a measurement range from 0 to 239 cb (kPa), it has no soil dissolvability, can withstand freezing temperatures, low cost, low maintenance [60]. Some of its features are, extreme low cost with volume pricing, not conductivity based, insensitive to salinity, probe does not corrode over time, rugged design for long-term use, small size, consumes less than 7mA for very low power operation, precise measurement, measures volumetric water content (VWC) or gravimetric water content (GWC), output voltage is proportional to moisture level, wide supply voltage range, can be buried and is waterproof.

The WATERMARK sensor uses a resistance device that measures response to changes in soil moisture. After it is planted in the soil it works by exchanging water with the soil to stay in equilibrium. Since the water in the soil has electrical conductivity. When there is more moisture than the water increases in the sensor thereby decreasing resistance and when the soil is dry, water in the sensor becomes low thereby increasing resistance measurement. This feature enables the sensor to have a stable and consistent calibration and it does not require recalibration for every installation. Figure 7 also shows the soil moisture sensor used in this study.

Figure 7 Watermark Soil Moisture Sensor

(3) Rainfall Sensor. The rainfall sensor is a part of the Weather Station WS-3000 which comprises of an anemometer, a wind vane, and a pluviometer [63]. The Weather Station is placed above ground on a pole and connected to the WSN sensor node. Our study is only concerned about pluviometer readings which provides a digital signal whose frequency is directly proportional to the intensity of rainfall. We use the rainfall intensity in our research to control the operation of the WSN. The pluviometer has a small bucket that, starts to fill when the rainfall starts, it has a capacity of approximately 0.28mm of water. When it is completely filled, its switch is closed and it is automatically emptied. A digital signal is produced as a result whose frequency which is proportional to rainfall intensity. It has an inbuilt function which returns rain intensity per minute. An interrupt is triggered in the microprocessor as soon as the start of rainfall is detected. The library defines several functions to measure from the pluviometer.

readPluviometerCurrent() calculates the precipitations in mm for the current period. For example, it is 10:02 am, so this function returns the mm of rainfall that took place in the last 2 minutes.

In the event of rain, the system measures rainfall every minute if there is no rainfall then there no rainfall measurement. It converts the reading into rainfall per hour, and classifies it as light rain or heavy rain according to the following criteria:

Light rain — when the precipitation rate is < 7 mm per hour, it is light rain.

Heavy rain — when the precipitation rate is > 7 mm per hour, it is heavy rain.

Figure 8 Weather Station having Rainfall Sensor

(4) Battery and Solar Panel. Our WSN Waspmote nodes will be deployed in remote areas hence they need to be self-sustained in power. To fulfill our power requirements we use rechargeable Li-ion batteries and external solar panels to power each Waspmote deployed in the fields. The Li-ion battery has the capacity of 6600 mAh and the voltage of 3.7 volts. The external solar panel has a rating of 7V- 500mA which is used to recharge the batteries. Solar cells have vastly differing characteristics from batteries. There are several kinds of photovoltaic cells available and all of them are relatively inefficient under indoor light conditions [52]. Since amorphous cells are much more expensive than their monocrystalline counterparts, it was decided to use monocrystalline cells as the source of natural energy. 3.7500 · 2.500, 4-4.0-100 monocrystalline solar panels from Solar World Inc. were chosen for the system due to ease of availability and low cost.

Figure 9 Solar Panel

3.6 Software Module

We use Digi XCTU v.6.3.0 software to implement our WSN testbed. It is a free, multi-platform, Next Generation Configuration platform for XBee/RF Solutions [64]. Apart from these, it has a lot of other advantages like simple graphical network view for network architecture and configuration which also shows signal strength for each connection. It also has API frame builder for building XBee API frames. It is easy to set up, test and conFigure for XBee. It has following features [64]: Management and Configuration of multiple RF devices or remote devices which are connected to the air. It has a firmware update feature which automatically restores all module settings, handling all the changes. We can save our console sessions and load them in any PC running XCTU at any time.

We can deploy XCTU on multiple platforms as it is compatible with the most-used operating systems, like Microsoft Windows, Linux, and Mac OS. We can discover our local and remote radio modules automatically which are connected to our PC, regardless of their port connections or conFigured settings. We can explore our complete RF network by visualizing its topology and by displaying all network nodes and connections graphically or in a table.

3.7 WSN Working Principle

The objective of our research is to provide a dynamic and reactive event-driven protocol AOMDV-E, for soil moisture monitoring so that the results can be used in the future to manage irrigation resources throughout Jordan. The overall data flow of our system comprises of (1) sampling soil moisture and rainfall data readings at predefined time intervals based on particular events, (2) sending sampled data to the gateway node, (3) sending the data from the gateway node to the base station, (4) moving to sleep mode, and (5) waking up and running whole process again. We use a mesh topology for network connectivity because of its advantages in our application scenario. Mesh topologies are able to route data and messages to the final gateway node through several different nodes. Even if the connection to an RF node is lost because of an environmental tragedy or loss of power, etc. still all the important data can reach to the gateway node and the base station due to the mesh topology. In a star network, there is a single base station can send and/or receive a message to a number of remote nodes. The remote nodes are not connected to each other. This type of network is simple, has low power consumption and allows low latency communications between the remote node and the base station but the disadvantage of such a network is that the base station must be within radio transmission range of all the individual nodes and is not as robust as mesh topology. However, we use mesh network as it can transmit data to one node to another node in the network that is within its radio transmission range. Hence we can have multi-hop communications, a node can use an intermediate node to forward the message to the desired node which is out of its transmission range. Hence the network will have redundancy and scalability. If at any point of time a node fails, then a remote node is still able to communicate to any other node which is inside its communication range, which in turn, forwards the message to the destination node or the hub. Also in a mesh topology, the range of the sensor network is not necessarily limited by the range in between individual nodes, it can be extended by adding more nodes to the sensor network. The only disadvantage of using a mesh topology is that it consumes more power to implement the multi-hop communications. Also as multiple communication hops to a destination increase, the time to deliver the message also increases. But due to its overall advantages and robustness, we use mesh topology, and the routing protocol is designed to save power.

Each sensor node is directly connected to the other nodes and also to the gateway node and also synchronized to each other so that they send data simultaneously after three hours. When there is rainfall, the sensor node which is affected by the event sends STATUS (send rain()) message to all nodes indicating that it is receiving rainfall, and hence the priority of the messages of this node is increased and this gives it precedence to use routes until there is no longer any rain.

3.8 Chapter Summary

In this chapter, we discussed the research methodology used in this study. We discussed the various simulation environments and about the NS-2 simulator used in this study and why it is chosen. We looked in detail at the building of the WSN prototype and its architecture. We also had a detailed look into the hardware components which include wireless interface (Waspmote XBee based on IEEE 802.15.4 standard), soil moisture sensor (Watermark 200SS), rainfall sensor (Libelium Weather Station WS 3000), USB-PC Interface (Waspmote Gateway), solar panel (Waspmote rigid solar panel 7V-500mA), 6600mA Li-Ion rechargeable batteries. The software that we use for the WSN setup, Digi XCTU v.6.3.0 was also discussed in detail. Finally, the working principle of the WSN setup was discussed giving details about objective, the data flow for the complete system, mesh topology, its advantages from the perspective of this research and how the nodes communicate with each other.

Chapter 4: Proposed Method

In this chapter, we will first analyze the problem at hand and we will then discuss the solution approach we will choose to solve the problems. We will then give details of the proposed AOMDV-E protocol and the changes we make to the standard AOMDV protocol.

4.1 Problem Analysis

Figure 10 shows the wireless sensor network is deployed over a small geographical area. Each node is able to measure the amount of soil moisture in the ground as well as detect the presence of rain. The latter is necessary as the amount of soil moisture may not, by itself, be enough to detect the presence of rain. In addition, the system must quickly react in order to discover over which part of the sensor network is experiencing the event: in this case rain. When rain is detected as falling in a given part of the system, the data must be collected from the relevant nodes and sent to a central administrative server or base station. This means that it will be necessary to change the routing so that information on the change in the soil content in the affected area could be sent to the server as quickly as possible. This data should be routed at a higher priority than other information being sent. The key thing to note is that these changes must be dynamically setup as there will be little foreknowledge involved and so changes in the normal routing protocol are necessary to facilitate this. Hence the need for reactivity at the routing level. Wireless systems are prone to errors associated with the environment in which these systems are deployed. Harsh environments such as very dry conditions can affect their performance and these systems must be carefully tuned to overcome environmental hindrances.

However, this also means that a severe change in the environment such as going from a situation when it is dry to a situation when it is raining, sometimes very heavily, will likely affect how these wireless nodes operate and may introduce more errors. It will, therefore, be necessary to boost the reliability of the system to ensure that the required information reaches the server. Furthermore, depending on the amount of rain being experienced, the amount of data moving towards the central server may be substantial and could overwhelm certain links. In this context, some management of information flow may be necessary. Hence, there will be the temptation to remove all redundancy from the data, but given the fact that reliability is needed some redundancy of the data is in fact recommended in this context. So a balance must be achieved and maintained.

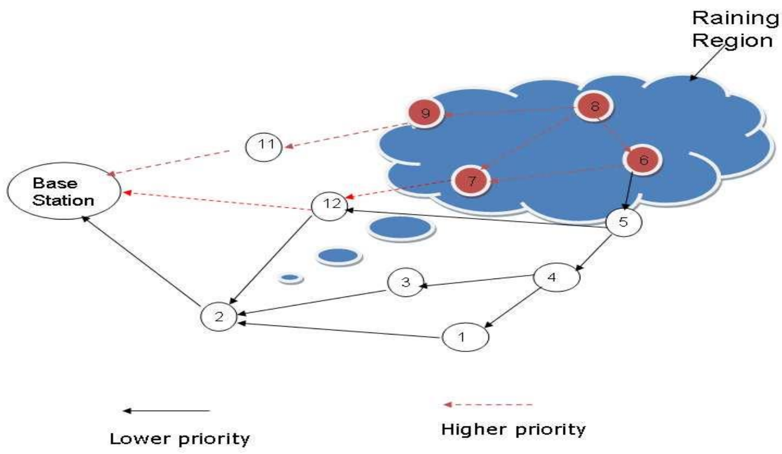


Figure 10 Sensor Network Measuring Moisture

Finally, there are energy considerations as Wireless Sensor Networks need access to power which must be carefully managed in outdoor environments. A lot of analysis of energy and power management has to do with the careful scheduling of activity to maximize battery life. As in this study, the proposed protocol that will be developed is an energy aware routing protocol, and the need for careful power management is lessened with the use of solar-powered WSN nodes; but the problem does not entirely go away. In the given scenario, a lot of data will be sent when a wireless node is in a rain affected area and hence a lot of power could be consumed at this time. So the wireless node may have to balance between the power left and the information being sent. In addition, if such a node is also routing information from other nodes, it may be necessary to decide which information stream is more important in order to convey the information required to give the big picture.

4.2 Solution Approach

4.2.1 Proposed AOMDV-E protocol and changes to AOMDV

We propose AOMDV-E protocol by enhancing AOMDV to move towards supporting a dynamic event-driven system. The first issue is that the routes may change due to the change in the status of the individual nodes, it is necessary to allow HELLO messages to not just monitor links but to fully indicate the status of the node at the other side of the link. So in this case, HELLO messages would also indicate whether or not it is raining on the other end, the network load as well as the power left in the system. A key piece of data is whether or not a node is able to route data on behalf of other nodes. As explained before, if a node is at the center of a downpour, its data is probably more important than the data of its neighboring nodes so it should not route data on behalf of these nodes. In such a situation, the node should also not respond to routing request (RREQ) messages. In addition, though HELLO messages will be sent periodically, it is necessary to have another message type which can be sent immediately in response to sudden environmental, link, or node changes. These messages are called STATUS (send rain) messages and are sent to neighboring nodes. For example, when a node first detects rain, it will send a STATUS message which is picked up by other nodes. By storing this information from various nodes, it would be possible to detect where the rain is falling in the sensor network. STATUS messages must be sent by each node to its neighbors when affected by rain, then the neighbor nodes have to update its routing table to remove the affected node from its routing table.

In a normal situation without any nodes affected by rain, the WSN will use Standard AOMDV else it will use our AOMDV-E protocol.

STATUS (send rain()) messages will cause routes to the central server to be re-evaluated. Let’s consider the case depicted in Figure 11. If node A that is not affected by rain was using a route to the central server via node B, now node A receives a STATUS message from node B saying that it has detected rain. Then node A, which is not rain-affected, should no longer send data through node B if possible since the data being generated by node B is more important.

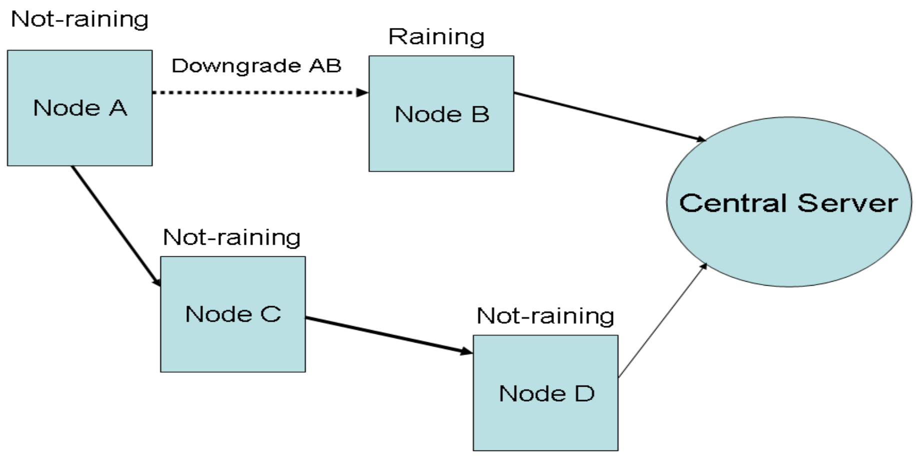


Figure 11 Routing decisions made based on external events

Node A should look for another route back to the server using other non-affected nodes. This would suggest that routes may have priorities based on the importance of the data being routed relative to the data being generated. In this case, node A will downgrade its route through node B, resulting in other routes through non-affected nodes being favored. This is illustrated in Figure 11.

If rain is detected by both nodes, A and B, both nodes will send each other their absolute measurements as well as relative changes in the soil moisture content. Nodes with less relative soil moisture content changes which indicate less rain will downgrade routes through regions with high relative soil moisture content changes. So data will be routed away from the most rain-affected areas to the least rain-affected areas.

In addition, this work is attempting to decrease the energy consumption of each node. By the above protocol modifications, we make the system event-based and energy aware. This is the main objective of this research.

There are some issues and problem scenarios which will also be solved in this work. For example, what happens when the nodes which are having rainfall are directly connected to the base station, such nodes would stop sending the HELLO messages and hence the routes to the base station would be cut off if all the nodes directly connected nodes are having rainfall. Since we are using mesh topology such a situation would not arise. The nodes are placed in such a manner that there are multiple nodes which are inside the communication range of the base station. In a real situation, we could also place the base station at the center of the sensor network, in such a manner that most of the nodes are in the communication range of the base station. There are also multiple protocols available for the Waspmotes which have a maximum range of 15.5 km which is quite enough for Middle Eastern countries at least. Hence the there are multiple nodes in communication range of each node and the range is also more than the area that could receive rainfall, hence even if multiple nodes fail then too the sensor network would find alternative routes to reach the base station. One could also argue that if multiple nodes are in communication range of each node there would be many routes which would result in wastage of power and deficiency. To solve this issue this protocol would only take the nearest route available to each node, in case an intermediate node fails then it can find an alternative route.

We use a mesh topology for network connectivity because of its advantages in our application scenario. Mesh topologies are able to route data and messages to the final gateway node through several different nodes. Even if the connection to an RF node is lost because of an environmental tragedy or loss of power, etc. still all the important data can reach to the gateway node and the base station due to the mesh topology. In a star network, there is a single base station can send and/or receive a message to a number of remote nodes. The remote nodes are not connected to each other. This type of network is simple, has low power consumption and allows low latency communications between the remote node and the base station but the disadvantage of such a network is that the base station must be within radio transmission range of all the individual nodes and is not as robust as mesh topology. However, we use mesh network as it can transmit data to one node to another node in the network that is within its radio transmission range. Hence we can have multi-hop communications, a node can use an intermediate node to forward the message to the desired node which is out of its transmission range. Hence the network will have redundancy and scalability. If at any point of time a node fails, then a remote node is still able to communicate to any other node which is inside its communication range, which in turn, forwards the message to the destination node or the hub. Also in a mesh topology, the range of the sensor network is not necessarily limited by the range in between individual nodes, it can be extended by adding more nodes to the sensor network. The only disadvantage of using a mesh topology is that it consumes more power to implement the multi-hop communications. Also as multiple communication hops to a destination increase, the time to deliver the message also increases. But due to its overall advantages and robustness, we use mesh topology, and the routing protocol is designed to save power.

4.3 AOMDV-E (AOMDV Modifications)

C++ language compiler has been used to modify the source code of AOMDV, this modification consists of several enhancements for the current AOMDV protocol the following pseudo-code describes the modification: (See *Appendix A*)

```
if (event == NO_RAIN)
 begin
        //use standard AOMDV
end
else
if ( event == RAIN)
begin
   sendBrodcastRainPacket(node_id);
   Disable_Hello();
   limit_response();
   setPacketPriority(1);
end
else if (event == END_RAIN)
begin
   Enable_Hello();
   replyto_response();
   setPacketPriority(0);
end
```

When a node is affected by rain it will receive a rain event, this event triggers a special broadcast rain packet. Every node that is adjacent to this node will receive this packet. After that the affected node disables sending the HELLO packets; this will prevent the neighbors nodes from seeing it and send their data through the affected node, this node will stop putting itself in the path for the RREQ packets this will also prevent other nodes from sending packets through it. Finally, we implemented a simple priority for the packets that have originated from the affected node; all packets that are sent from this node will have a greater value for its QoS value in the IP header.

If the event ends, the node will return to its normal activity, it will send HELLO packets normally and continue to respond on the RREQ packets and finally sets the QoS value to its normal value.

```
if (packet.type == RAIN_PACKET)
begin
   for each(r in routing_table)
   begin
      if (r.nextHop == packet.source)
         remove(packet.source);
   end
end
```

When an adjacent node receives a rain packet it will immediately remove all next hops that this node is involved in. This will prevent any node from sending any data through this node. The data of the node experiencing rainfall has its priority increased, so now the data from this node. In addition, the rain-affected node disables the sending of Hello messages but it continues to respond to the Hello messages received from its neighbors. Therefore it does not have to compete with data from other nodes, this affected node now cannot be used as an intermediate router as long as it is raining. All these steps are for removing the affected nodes from the routes of the non-affected nodes, but when the affected nodes want to send the data it can still send RREQ packets and all other active nodes will respond to it resulting in the formation of the route to the destination.

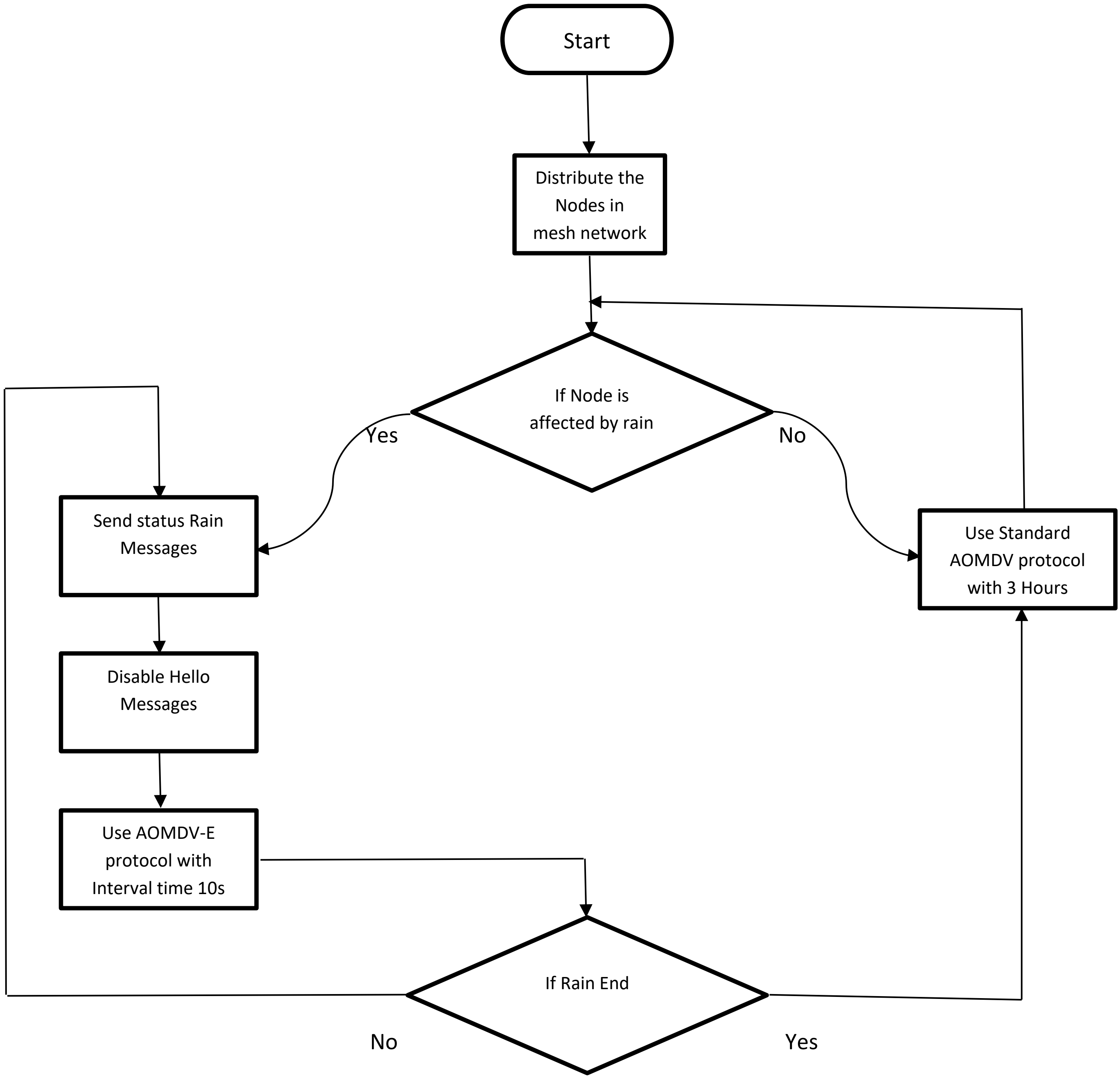


Figure 12 Flowchart showing the data flow for the proposed WSN system

As a result of these changes transit time decreases for data from the rain affected nodes. Also when the data from the rain affected nodes reaches intermediate nodes and at the gateway node, it takes less time as it has a higher priority than data from non-affected nodes. Hence our proposed changes result in overall reduction in the transmission and send energies required whenever the nodes are affected by rain.

In this scheme described above it seems like the rain-affected nodes cannot send data to other nodes as routes are broken and cut off from the affected nodes as they are not sending HELLO messages. But this is not the case, when the affected nodes want to send data, they can still broadcast RREQ packets to all other nodes and can have all the routes to the destination. All other nodes cannot send their data to the affected nodes but the affected nodes can still find routes to the destination as all other nodes are visible. Hence the affected nodes will send rain packets to tell other nodes that they are having rainfall, as well as RREQ packets to find routes to a destination and send data. When any of the nodes are dead they cannot send any messages and other nodes can eventually find out about them.

4.4 Chapter Summary

In this chapter, we analyzed the current problems in the moisture and rainfall monitoring using WSN, and the solution approach that this study would take to solve these problems. We also discussed details of the proposed AOMDV-E protocol and the changes we made to the standard AOMDV protocol. The modified AOMDV-E protocol C++ code was also given along with the flowchart of the complete system. Then we also Figured out some real-time issues that the proposed WSN system could face when using the modified protocol, for all the issues solutions were discussed and possible steps were looked into to negate the failing of nodes and making the WSN system robust in any possible scenario.

Chapter 5: Simulation Results

In this chapter, we will first explore the simulation setup giving details about different simulation configurations and setup of the simulation environment and its variables. We will then provide the routing tables and the simulation results for each type of configuration. We will also evaluate the performance of each configuration setup. Finally, we will also evaluate the WSN setup in terms of energy measurements.

In this simulation, we use the TCL language to build the scenario and use C++ language to build AOMDV-E by modifying the AOMDV routing protocol source code. The source code of the simulation is written in the TCL language and runs the simulation example on ns-2.35 Linux Ubuntu version 12. The simulations have been done in three scenarios the 1st one consists of 5 nodes with one of them affected by rain, the 2nd one consists of 7 nodes 2 two of them affected by rain the last one, which is the complex one consists of 30 nodes 5 of them affected by rain.

5.1 Simulation setup

In this section we examine 3 scenarios the 1st one consists of 5 nodes with one of them affected by rain (which is node 1) the 2nd one consists of 7 nodes with two of them (2, 6) affected by rain the last one, which is the complex one consists of 30 nodes with five of them (1, 12, 16, 21, 25) affected by rain. There were many reasons why we chose these three scenarios. First of all, with the simulation setup we had also planned to set up a prototype to test the proposed WSN protocol and the complete system. Since the nodes are costly, hence the first scenario which we would also implement physically is comprising of five nodes, one out of which would be affected by rain. Then just to prove that our network can also work if more than one node is affected by rainfall, we included a scenario comprising of seven nodes, two of which would be affected by rain. We increased a number of nodes from five to seven because if two nodes out of five would be affected by rain then it would cause the problem to test the system as one out of remaining three nodes is the base station. At last, we also wanted to prove that our WSN protocol is scalable, so the third scenario comprises of thirty nodes, out of which five nodes experience rainfall. These nodes can have a maximum range of approx. 15 km each so as far as Middle East climate is concerned it does not rain at such a vast area simultaneously to require more than thirty nodes, hence thirty nodes are more than enough. Furthermore, the area under cultivation or irrigation is also not so huge in any one location so that it needs more than this number of nodes to monitor it.

5.1.1 Five node simple simulation

The setup of this scenario consists of 5 wireless nodes from node 0 to node 4 distributed with fixed locations across the grid as shown in Figure 12. One of them is affected by rain, which is node 1. We will have two constant bit rates (cbr) streams one from node 0 and the other from node 1 the destination for both streams will be node 2 we will call them cbr0 and cbr1 respectively. According to the AOMDV protocol, cbr0 traffic can go from node 0 to node 2 using node 1 or node 4, and cbr1 traffic will go to node 2 directly. AOMDV is a multipath algorithm which finds multiple paths between source and destination, and the path having the least number of hops to reach the destination is given a priority. If more than one shortest path are found in the table then the first path is taken in order. That means traffic from node 0 can go through 2 paths, since both paths have the same number of hops. Simulation time is 2000s, node 0 is set to send data after every 100ms with agent name cbr0 and stops at 1500s; node 1 is set to send data after every 200ms with agent name cbr1 and stops at the end of the simulation. Packet size for cbr0 is 1000 byte with the rate of 0.05Mbps, as for cbr1 packet size is 700 byte with the rate of 0.05Mbps. During sending packets exactly at 900ms node 1 affected by RAIN (which is the event). The parameter values like simulation time, packet size, bandwidth rate, etc. are just experimental values chosen to test the system as suggested by NS-2. One can take different values of these parameters. For example, the simulation time can be extended, but for our system, we just need to show that our proposed protocol is working, so even a short simulation time is good enough. Nodes can send data with different packet sizes and at different intervals so for node 0 and node 1 we take different values. As for the data rate of 0.05 Mbps, we will take different data rates based on if the rainfall is light, heavy or very heavy. When more than one nodes are affected by rain (as in 7 nodes and 30 node scenarios), then in heavy and very heavy rainfall we increase the bandwidth rate, because more traffic would be generated. Hence all these simulation parameters values are experimental and could be changed. The effect and implications of taking different values of these parameters are out of the scope of the thesis.

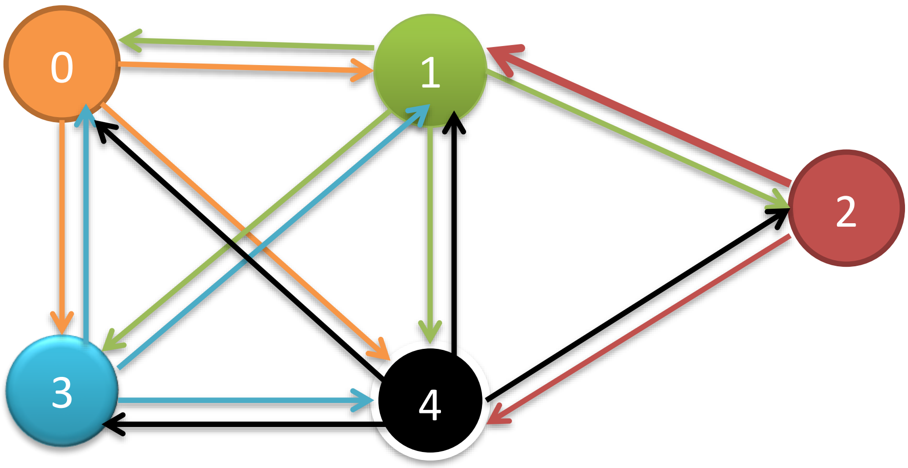


Figure 12 Simulation Setup with 5 nodes

5.1.2 Seven node simple simulation

The setup of this scenario consists of 7 wireless nodes from node 0 to node 6 distributed with fixed locations across the grid as shown in Figure 13. Two of them are affected by rain node 1 and node 6. We have three constant bit rates (cbr) streams one from node 0, one from node 1 and one from node 6 the destination for all streams will be node 3 we will call them cbr0, cbr1, and cbr2 respectively. AOMDV is a multipath algorithm which finds multiple paths between source and destination, and the path having the least number of hops to reach the destination is given a priority. If more than one shortest path are found in the table then the first path is taken in order. According to AOMDV protocol cbr0 traffic can go from node 0 to node 3 using multiple paths, cbr1 traffic can use node 2 or 6 to reach node 3 and cbr6 traffic will go directly from node 6 to node 3; the shortest possible both for node 0 will be through node 1 and through node 4 also node 1 will go through 2 paths one is through node 2 and the other through 5, node 6 has 1 path directly to node 3. Simulation time is 2500s, we setup the constant bit rate (cbr) with variable rate values depending on rain levels for example when its light raining the (cbr) rate will be 0.05Mbps, when rain becomes heavier (cbr) rate changed to 0.1Mbps at the last when flood accoutred (cbr) rate changed to 0.5 Mbps. Packet size for all (cbr) is 1000 byte.

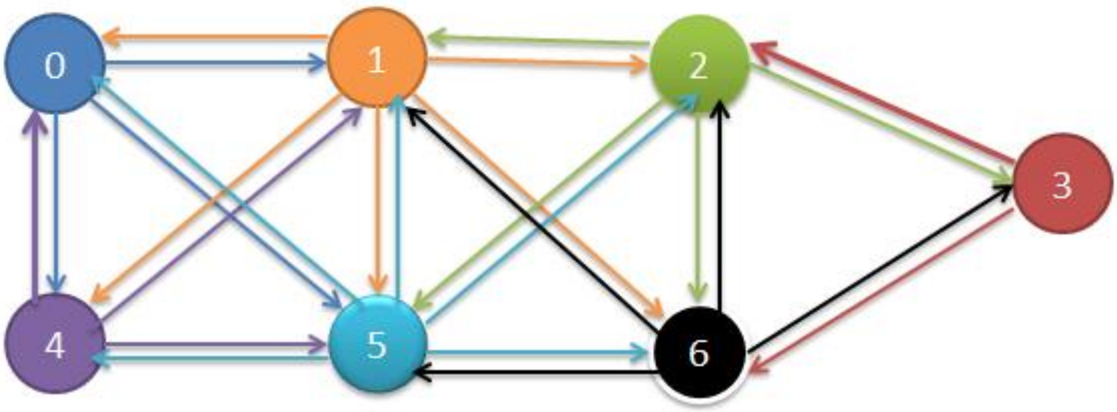


Figure 13 Simulation Setup with 7 nodes

5.1.3 Complex Simulation

In the section, we examine a more complex scenario consisting of 30 wireless nodes from node 0 to node 29 distributed with fixed locations across the grid as shown in Figure 14. We will have two constant bit rates (cbr) streams one from node 0 and one from node 1 the destination for both streams will be node 29 we will call them cbr0 and cbr1 respectively. According to the AOMDV protocol cbr0 traffic will go through the path node 0 -> node 1 -> node 2 -> node 13 -> node 4 -> node 15 -> node 6 -> node 17 -> node 18 then node 29, and cbr1 traffic will go through the same path.

Five nodes affected by rain 1, 12, 16, 21, 25 this means that node 0 or 1 will use secondary paths when any node in the primary path affected by RAIN.

Simulation time is 2500s, we set up the constant bit rate (cbr) with variable rate values depending on rain levels for example when its light raining the (cbr) rate will be 0.05Mbps, when rain becomes heavier (cbr) rate changed to 0.1Mbps at the last when flood accoutred (cbr) rate changed to 0.5 Mbps. Packet size for all (cbr) is 1000 byte.

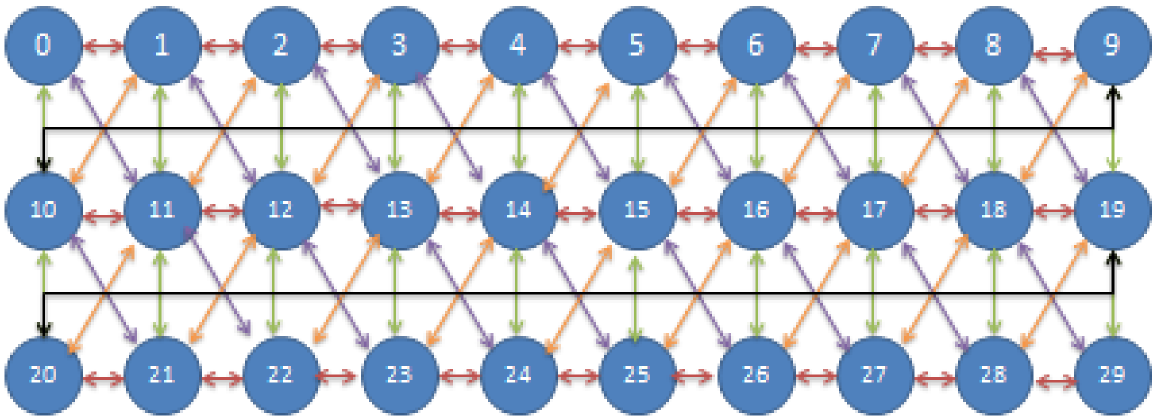


Figure 14 30-Nodes Location

5.2 Simulation Environment

Table 3 shows the simulation environment parameters that conFigured in the scenarios set up.

To determine the distance between nodes by NS2 simulation we did the following steps:-

- Compiled threshold.cc program
- Used the command that comes from the compilation called threshold

- # ./threshold -m Shadowing -r 0.95 150 the output of this command set the distance between each node will be 150m. If the distance between nodes greater than 150m the nodes won't see each other.

S#	Parameter Variables	Parameters Description	5 nodes Associated Values	7 nodes Associated Values	30 nodes Associated Values
1	Simulation Tool	Operating System And simulator	Ubuntu 12 + NS 2.35	Ubuntu 12 + NS 2.35	Ubuntu 12 + NS 2.35
2	Propagation	The Two-ray ground reflection model considers both the direct path and a ground reflection path. Which means both transmitter and receiver node is assumed to be in the line of sight. It has been shown that this model is more accurate than other models	TwoRayGround	TwoRayGround	TwoRayGround
3	MAC Type	Mac Protocol	Mac/802_11	Mac/802_11	Mac/802_11
4	Interface Queue	Queueing method at the interface	Queue/DropTail/PriQueue	Queue/DropTail/PriQueue	Queue/DropTail/PriQueue
5	Physical Wireless Data speed	The Bandwidth Rate	11Mbps	11Mbps	11Mbps
6	Antenna Type	Antenna Type	Antenna/OmniAntenna	Antenna/OmniAntenna	Antenna/OmniAntenna
7	Packet Queue Size	Packet Queue Size	50	50	50
8	Total Number of nodes	Number of nodes distributed	5	7	30
9	Base Protocol	Routing Protocol	AOMDV	AOMDV	AOMDV
10	Optimization Algorithm	The event	Rain Optimization	Rain Optimization	Rain Optimization
11	X dimension of Topography	X-axis of the covered area	797	797	797
12	Y dimension of Topography	Y-axis of the covered area	595	595	595
13	Simulation Time	Time in seconds	2000	2500	2500
14	No of rain affected Nodes	No of rain affected Nodes	1	2	5
15	Packet size	Packet size in bytes	1000	1000	1000
16	CBR Bandwidth rate	CBR Bandwidth rate	0.05Mbps	0.05Mbps, 0.1 Mbps and 0.5 Mbps	0.05Mbps, 0.1 Mbps and 0.5 Mbps
17	traffic generation starts at	Traffic generator objects generate traffic and can be of four types, namely, exponential, pareto, CBR and traffic trace.	100.0ms	100.0ms	100.0ms

Table 3 Simulation Scenario Environment

5.3 Routing Tables

5.3.1 Five Node Routing Table Changes

a- Before rain being detected at node 1

N0

Dest No	Seq#	Advhop	Nxthop	Hopcnt	Lsthop
3	2	16383	3	1	0
4	2	16383	4	1	0
1	2	16383	1	1	0
2	2	16383	1	2	1
			4	2	4

N1

Dest No	Seq#	Advhop	Nxthop	Hopcnt	Lsthop
3	2	16383	3	1	1
4	2	16383	4	1	1
2	2	16383	2	1	1
0	4	16383	0	1	1

N2

4	2	16383	4	1	2
1	2	16383	1	1	2
0	5	16383			

N3

4	2	16383	4	1	3
1	2	16383	1	1	3
0	4	1	0	1	3

N4

3	2	16383	3	1	4
1	2	16383	1	1	4
2	2	1	2	1	4
0	4	1	0	1	4

Table 4 Routing Table for each node before rain being detected at node 1

b- Routing table after rain is detected at node 1

N0

Dest No	Seq#	Advhop	Nxthop	Hopcnt	Lsthop
3	2	16383	3	1	0
4	2	16383	4	1	0
2	2	16383	4	2	4

N1

Dest No	Seq#	Advhop	Nxthop	Hopcnt	Lsthop
3	2	16383	3	1	1
4	2	16383	4	1	1
2	2	16383	2	1	1
0	4	16383	0	1	1

N2

4	2	16383	4	1	2
0	5	16383			

N3

4	2	16383	4	1	3
0	4	1	0	1	3

N4

3	2	16383	3	1	4
2	2	1	2	1	4
0	4	1	0	1	4

Table 5 Routing Table for each node after rain is detected at node 1

Tables 4 and 5 show the result of routing table before and after rain is detected at node 1, the routing table for node 0 has been changed and node 0 takes another route through node 4 instead of the primary one through node 1 to send data to node 2 (destination) because node 1 affected by RAIN then it has very important data (soil moisture) has to send first node 1 should take the highest priority to send data.

5.3.2 Seven Node Routing Table Changes

a- Before rain being detected at node 1 and node 6

Note: The route path marked by RED will be removed after protocol change

N0	Route Table Before Dynamic Change				
Dest No	**Seq#**	**Advhop**	**Nxthop**	**Hopcnt**	**Lsthop**
4	2	16383	4	1	0
1	2	16383	1	1	0
5	2	16383	5	1	0
3	2	16383	1	3	6

N1	Route Table Before Dynamic Change				
Dest No	**Seq#**	**Advhop**	**Nxthop**	**Hopcnt**	**Lsthop**
4	2	16383	4	1	1
6	2	16383	6	1	1
2	2	16383	2	1	1
5	2	16383	5	1	1
3	2	2	6	2	6
			2	2	2
0	4	1	0	1	1

N2	Route Table Before Dynamic Change				
1	2	16383	1	1	2
6	2	16383	6	1	2
5	2	16383	5	1	2
3	2	1	3	1	2
0	4	2	1	2	1
			5	2	5

N3	Route Table Before Dynamic Change				
6	2	16383	6	1	3
2	2	16383	2	1	3
0	4	16383	6	3	1

N4	Route Table Before Dynamic Change				
1	2	16383	1	1	4
5	2	16383	5	1	4
0	4	1	0	1	4

N5	Route Table Before Dynamic Change				
4	2	16383	4	1	5
1	2	16383	1	1	5
6	2	16383	6	1	5
2	2	16383	2	1	5
0	4	1	0	1	5

N6	Route Table Before Dynamic Change				
1	2	16383	1	1	6
2	2	16383	2	1	6
5	2	16383	5	1	6
3	2	1	3	1	6
0	4	2	1	2	1
			5	2	5

Table 6 Routing Table for each node before rain being detected at node 1 and node 6

b- Routing table after rain is detected at node 1 and node 6

N0	Route Table After Dynamic Change				
Dest No	**Seq#**	**Advhop**	**Nxthop**	**Hopcnt**	**Lsthop**
4	2	16383	4	1	0
5	2	16383	5	1	0
3	2	16383			

N1	Route Table After Dynamic Change				
Dest No	**Seq#**	**Advhop**	**Nxthop**	**Hopcnt**	**Lsthop**
4	2	16383	4	1	1
2	2	16383	2	1	1
5	2	16383	5	1	1
3	2	2	2	2	2
0	4	1	0	1	1

N2	Route Table After Dynamic Change				
6	2	16383	6	1	2
5	2	16383	5	1	2
3	2	1	3	1	2
0	4	2	5	2	5

N3	Route Table After Dynamic Change				
2	2	16383	2	1	3
0	4	16383			

N4	Route Table After Dynamic Change				
5	2	16383	5	1	4
0	4	1	0	1	4

N5	Route Table After Dynamic Change				
4	2	16383	4	1	5
2	2	16383	2	1	5
0	4	1	0	1	5

N6	Route Table After Dynamic Change				
2	2	16383	2	1	6
5	2	16383	5	1	6
3	2	1	3	1	6
0	4	2	5	2	5

Table 7 Routing Table for each node after rain is detected at node 1 and node 6

Tables 6 and 7 show the result of routing table before and after rain is detected at node 1, the routing table for node 0 has been changed and node 0 takes another route through node 4 to node 5 then node 2 instead of the primary one through node 1 to send data to node 3 (destination) because node 1 and node 6 are affected by RAIN then it has very important data (soil moisture) has to send first node 1 should take the highest priority to send data.

5.3.3 Complex Simulation Result – Router Table Changes

a- Before rain being detected at nodes 1, 12, 16, 21 and 25 (colored by RED)

N0 — Route Table Before Dynamic Change

Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
1	2	16383	1	1	0
11	2	16383	11	1	0
10	2	16383	10	1	0
29	2	16383	1	9	28
			11	9	28

N15 — Route Table Before Dynamic Change

Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
14	2	16383	14	1	15
24	2	16383	24	1	15
26	2	16383	26	1	15
5	2	16383	5	1	15
6	4	16383	6	1	15
4	3	16383	4	1	15
25	2	16383	25	1	15
16	4	16383	16	1	15
29	2	4	6	4	28
			16	4	28
			26	4	28

N1 — Route Table Before Dynamic Change

Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
12	2	16383	12	1	1
2	2	16383	2	1	1
11	2	16383	11	1	1
10	2	16383	10	1	1
0	4	1	0	1	1
29	2	8	2	8	28
			12	8	28

N16 — Route Table Before Dynamic Change

Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
17	2	16383	17	1	16
15	2	16383	15	1	16
5	3	16383	5	1	16
7	2	16383	7	1	16
6	4	16383	6	1	16
25	2	16383	25	1	16
26	2	16383	26	1	16
27	2	16383	27	1	16

N2 — Route Table Before Dynamic Change

Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
13	2	16383	13	1	2
11	2	16383	11	1	2
3	2	16383	3	1	2
12	2	16383	12	1	2
1	2	16383	1	1	2
29	2	7	13	7	28
			3	7	28

N3 — Route Table Before Dynamic Change

Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
14	2	16383	14	1	3
4	2	16383	4	1	3
2	2	16383	2	1	3
12	2	16383	12	1	3
13	2	16383	13	1	3

N17 — Route Table Before Dynamic Change

Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
6	2	16383	6	1	17
28	2	16383	28	1	17
8	2	16383	8	1	17
7	2	16383	7	1	17
18	2	16383	18	1	17

N4	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
13	2	16383	13	1	4
5	2	16383	5	1	4
14	2	16383	14	1	4
15	2	16383	15	1	4
3	2	16383	3	1	4
29	2	5	15	5	28
			5	5	28

N5	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
14	2	16383	14	1	5
6	4	16383	6	1	5
4	2	16383	4	1	5
15	6	16383	15	1	5
16	4	16383	16	1	5

N6	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
17	2	16383	17	1	6
15	2	16383	15	1	6
5	3	16383	5	1	6
7	2	16383	7	1	6
16	4	16383	16	1	6
29	2	3	17	3	28
			7	3	28

16	4	16383	16	1	17
26	2	16383	26	1	17
27	2	16383	27	1	17
29	2	2	28	2	28
			18	2	28

N18	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
19	2	16383	19	1	18
17	2	16383	17	1	18
9	2	16383	9	1	18
8	2	16383	8	1	18
7	2	16383	7	1	18
28	2	16383	28	1	18
27	2	16383	27	1	18
29	2	1	29	1	18

N19	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
9	2	16383	9	1	19
8	2	16383	8	1	19
29	2	16383	29	1	19
18	2	16383	18	1	19
28	2	16383	28	1	19

N7	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
6	2	16383	6	1	7
17	2	16383	17	1	7
8	2	16383	8	1	7
18	2	16383	18	1	7
16	4	16383	16	1	7

N8	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
19	2	16383	19	1	8
17	2	16383	17	1	8
9	2	16383	9	1	8
7	2	16383	7	1	8
18	2	16383	18	1	8

N9	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
19	2	16383	19	1	9
8	2	16383	8	1	9
18	2	16383	18	1	9

N10	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
20	2	16383	20	1	10
1	2	16383	1	1	10
11	2	16383	11	1	10
21	2	16383	21	1	10
0	4	1	0	1	10

N20	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
11	2	16383	11	1	20
10	2	16383	10	1	20
21	2	16383	21	1	20

N21	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
10	2	16383	10	1	21
22	2	16383	22	1	21
20	2	16383	20	1	21
12	2	16383	12	1	21
11	2	16383	11	1	21

N22	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
11	2	16383	11	1	22
12	2	16383	12	1	22
13	2	16383	13	1	22
23	2	16383	23	1	22
21	2	16383	21	1	22

N23	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
14	2	16383	14	1	23
22	2	16383	22	1	23
24	2	16383	24	1	23
12	2	16383	12	1	23
13	2	16383	13	1	23

N11	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
22	2	16383	22	1	11
20	2	16383	20	1	11
12	2	16383	12	1	11
1	2	16383	1	1	11
2	2	16383	2	1	11
10	2	16383	10	1	11
21	2	16383	21	1	11
0	4	1	0	1	11

N12	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
1	2	16383	1	1	12
13	2	16383	13	1	12
11	2	16383	11	1	12
3	2	16383	3	1	12
22	2	16383	22	1	12
2	2	16383	2	1	12
23	2	16383	23	1	12
21	2	16383	21	1	12

N13	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
22	2	16383	22	1	13
4	2	16383	4	1	13
2	2	16383	2	1	13
14	2	16383	14	1	13
24	2	16383	24	1	13
12	2	16383	12	1	13
23	2	16383	23	1	13
3	2	16383	3	1	13
29	2	6	4	6	28
			14	6	28
			24	6	28

N24	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
23	2	16383	23	1	24
13	2	16383	13	1	24
14	2	16383	14	1	24
25	2	16383	25	1	24
15	2	16383	15	1	24

N25	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
14	2	16383	14	1	25
24	2	16383	24	1	25
15	6	16383	15	1	25
16	4	16383	16	1	25
26	2	16383	26	1	25

N26	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
17	2	16383	17	1	26
15	2	16383	15	1	26
25	2	16383	25	1	26
16	4	16383	16	1	26
27	2	16383	27	1	26

N27	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
28	2	16383	28	1	27
17	2	16383	17	1	27
18	2	16383	18	1	27
16	4	16383	16	1	27
26	2	16383	26	1	27

N14	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
25	2	16383	25	1	14
5	2	16383	5	1	14
23	2	16383	23	1	14
24	2	16383	24	1	14
13	2	16383	13	1	14
4	2	16383	4	1	14
15	2	16383	15	1	14
3	2	16383	3	1	14

N28	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
17	2	16383	17	1	28
18	2	16383	18	1	28
27	2	16383	27	1	28
29	2	1	29	1	28

N29	Route Table Before Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
19	2	16383	19	1	29
18	2	16383	18	1	29
28	2	16383	28	1	29

Table 8 Routing Table for each node before rain being detected at nodes 1, 12, 16, 21 and 25

b- Routing table after rain is detected at nodes 1, 12, 16, 21 and 25 (Using AOMDV-E)

N0	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
1	2	16383	1	1	0
11	2	16383	11	1	0
10	2	16383	10	1	0
29	2	16383	11	9	28

N1	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
12	2	16383	12	1	1
2	2	16383	2	1	1
11	2	16383	11	1	1
10	2	16383	10	1	1
0	4	1	0	1	1
29	2	8	2	8	28

N2	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
13	2	16383	13	1	2
11	2	16383	11	1	2
3	2	16383	3	1	2
12	2	16383	12	1	2
1	2	16383	1	1	2
29	2	7	13	7	28
			3	7	28

N3	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
14	2	16383	14	1	3
4	2	16383	4	1	3
2	2	16383	2	1	3
13	2	16383	13	1	3

N15	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
14	2	16383	14	1	15
24	2	16383	24	1	15
26	2	16383	26	1	15
5	2	16383	5	1	15
6	4	16383	6	1	15
4	3	16383	4	1	15
29	2	4	6	4	28
			26	4	28

N16	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
17	2	16383	17	1	16
15	2	16383	15	1	16
5	3	16383	5	1	16
7	2	16383	7	1	16
6	4	16383	6	1	16
26	2	16383	26	1	16
27	2	16383	27	1	16

N17	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
6	2	16383	6	1	17
28	2	16383	28	1	17
8	2	16383	8	1	17
7	2	16383	7	1	17
18	2	16383	18	1	17
26	2	16383	26	1	17
27	2	16383	27	1	17
29	2	2	28	2	28
			18	2	28

N4	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
13	2	16383	13	1	4
5	2	16383	5	1	4
14	2	16383	14	1	4
15	2	16383	15	1	4
3	2	16383	3	1	4
29	2	5	15	5	28
			5	5	28

N5	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
14	2	16383	14	1	5
6	4	16383	6	1	5
4	2	16383	4	1	5
15	6	16383	15	1	5

N6	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
17	2	16383	17	1	6
15	2	16383	15	1	6
5	3	16383	5	1	6
7	2	16383	7	1	6
29	2	3	17	3	28
			7	3	28

N7	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
6	2	16383	6	1	7
17	2	16383	17	1	7
8	2	16383	8	1	7
18	2	16383	18	1	7

N18	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
19	2	16383	19	1	18
17	2	16383	17	1	18
9	2	16383	9	1	18
8	2	16383	8	1	18
7	2	16383	7	1	18
28	2	16383	28	1	18
27	2	16383	27	1	18
29	2	1	29	1	18

N19	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
9	2	16383	9	1	19
8	2	16383	8	1	19
29	2	16383	29	1	19
18	2	16383	18	1	19
28	2	16383	28	1	19

N20	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
11	2	16383	11	1	20
10	2	16383	10	1	20

N21	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
10	2	16383	10	1	21
22	2	16383	22	1	21
20	2	16383	20	1	21
11	2	16383	11	1	21

N8	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
19	2	16383	19	1	8
17	2	16383	17	1	8
9	2	16383	9	1	8
7	2	16383	7	1	8
18	2	16383	18	1	8

N9	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
19	2	16383	19	1	9
8	2	16383	8	1	9
18	2	16383	18	1	9

N10	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
20	2	16383	20	1	10
11	2	16383	11	1	10
0	4	1	0	1	10

N11	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
22	2	16383	22	1	11
20	2	16383	20	1	11
2	2	16383	2	1	11
10	2	16383	10	1	11
29	4	8	22	8	28
0	4	1	0	1	11

N22	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
11	2	16383	11	1	22
13	2	16383	13	1	22
23	2	16383	23	1	22
29	4	7	23	7	28

N23	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
14	2	16383	14	1	23
22	2	16383	22	1	23
24	2	16383	24	1	23
29	4	6	24	6	28
13	2	16383	13	1	23

N24	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
23	2	16383	23	1	24
13	2	16383	13	1	24
14	2	16383	14	1	24
29	4	5	15	5	28
15	2	16383	15	1	24

N25	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
14	2	16383	14	1	25
24	2	16383	24	1	25
15	6	16383	15	1	25
26	2	16383	26	1	25

N12	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
13	2	16383	13	1	12
11	2	16383	11	1	12
3	2	16383	3	1	12
22	2	16383	22	1	12
2	2	16383	2	1	12
23	2	16383	23	1	12

N13	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
22	2	16383	22	1	13
4	2	16383	4	1	13
2	2	16383	2	1	13
14	2	16383	14	1	13
24	2	16383	24	1	13
23	2	16383	23	1	13
3	2	16383	3	1	13
29	2	6	4	6	28
			14	6	28
			24	6	28

N14	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
5	2	16383	5	1	14
23	2	16383	23	1	14
24	2	16383	24	1	14
13	2	16383	13	1	14
4	2	16383	4	1	14
15	2	16383	15	1	14
3	2	16383	3	1	14

N26	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
17	2	16383	17	1	26
15	2	16383	15	1	26
27	2	16383	27	1	26

N27	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
28	2	16383	28	1	27
17	2	16383	17	1	27
18	2	16383	18	1	27
26	2	16383	26	1	27

N28	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
17	2	16383	17	1	28
18	2	16383	18	1	28
27	2	16383	27	1	28
29	2	1	29	1	28

N29	Route Table After Dynamic Change				
Dest No	Seq#	Advhop	Nxhop	Hopcnt	Lsthop
19	2	16383	19	1	29
18	2	16383	18	1	29
28	2	16383	28	1	29

Table 9 Routing Table for each node after rain is detected at nodes 1, 12, 16, 21 and 25

Tables 8 and 9 show the result of routing table before and after rain is detected at nodes 1, 12, 16, 21, 25, the routing table for node 0 has been changed and node 0 takes another route through node 11 instead of the primary one through node 1 to send data to node 29 (destination) because node 1 affected by RAIN then it has very important data (soil moisture) has to send, so node 1 should take the highest priority to send data.

5.4 Performance Measurements before and after AOMDV protocol is modified

5.4.1 Five Node Simulation Results

In this example, we use standard AOMDV in the first period from 0 to 900s, in the second one from 900 to 2000 we use the AOMDV-E.

An AWK script was built to calculate the average of Throughput and End to End Delay on node 0 and node 1 before and after node 1 affected by rain. The highest priority data stream will be granted to cbr1 coming from node 1 because it is affected by rain and it has important data that should be sent to the destination node 2. The cbr0 data coming from node 0 should routing to node 4 which has data with low priority. In this way, we reduce the forwarding load on node 1 that means the throughput and end to end delay will be better on cbr1 after node 1 affected by rain.

The results listed in Table 10 clearly show improvement of Throughput and End to End Delay on cbr1 coming from node 1 which validate our assumptions and it showing that the event-driven approach does improve overall network efficiency.

Type/protocol and Time	standard AOMDV From 0 To 900 s Before Rain	AOMDV-E From 900 To 2000 s After Rain
Total Sent Size From CBR0	5000000 bit	3749000 bit
Total Sent Size From CBR1	4375000 bit	6874700 bit
End-to-End Delay (cbr0)	299.987661 s	399.315447 s
End-to-End Delay (cbr1)	19.987827 s	0.004115 s
Throughput (cbr0)	5.555556 kbps	3.408182 kbps
Throughput (cbr1)	4.861111 kbps	6.249727 kbps

Table 10 Performance measurements before and after node 1 affected by rain

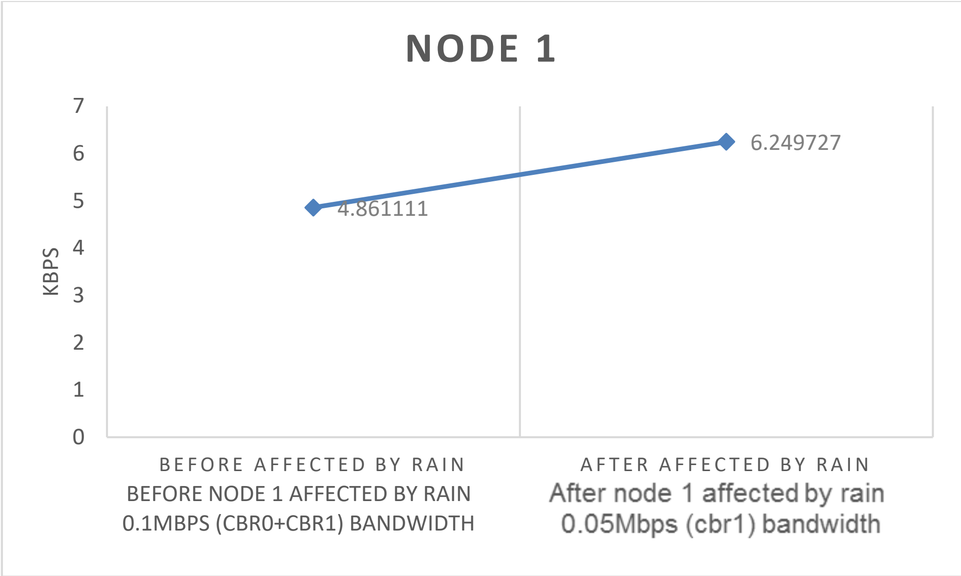


Figure 15 Throughput on Node 1 before and after affected by Rain

In Figure 15, we present the throughput of node 1 in kbps (on Y-axis) against the effect of rain (on the x-axis). It clearly shows that the throughput on Node 1 after it affected by rain with AOMDV-E protocol is more than before affected by rain with standard AOMDV protocol.

The main reason behind this is, "when node 1 is affected by rain, only stream cbr1 will send through node 1 otherwise cbr0 and cbr1 streams will sending to the destination node through node 1. Also, the sensor sampling the rain parameters at a faster rate results in more data being generated, consequently increasing the throughput". However, since the speed of the link is 11 Mbps the link is never saturated even in the very heavy rain.

5.4.2 Seven Node Simulation Results

In this example, we divided the simulation period depending on raining levels, for instance with no rain the period from 0 to 900s, light rain period from 900 to 2000s and heavy rain from 2000 to 2500s.

An AWK script was built to calculate the average of Throughput and End to End Delay on node 1 and node 6 before and after they were affected by rain.

Node 1 and node 6 affected by two levels of rain light rain and heavy rain, we compared the Throughput and End to End Delay in two levels.

The results listed in table 11 clearly show that after raining both Throughput and End to End Delay has been improved. It shows that when the rain level increases the performance of the affected nodes also improves. The reason behind this increase in throughput is that at the light and heavy rain, we use the AOMDV-E protocol.

readPluviometerCurrent() calculates the precipitations in mm for the current period. For example, it is 10:02 am, so this function returns the mm of rainfall that took place in the last 2 minutes.

In event of rain, the system measures rainfall every minute, if there is no rainfall than the no rainfall measurement. It converts the reading into rainfall per hour, and classifies it as light rain or heavy rain according to the following criteria:

Light rain — when the precipitation rate is < 7 mm per hour, it is light rain.

Heavy rain — when the precipitation rate is > 7 mm per hour, it is heavy rain.

Type/Rain-Level (Time)	**Standard AOMDV No Rain - 0.05Mbps (0 – 900) s**	**AOMDV-E Light Rain - 0.1Mbps (900 – 2000) s**	**AOMDV-E Heavy Rain - 0.5 Mbps (2000 – 2500) s**
Total Sent Size From CBR1	5000000 bit	68749000 bit	31250000 bit
Total Sent Size From CBR6	4375000 bit	13748700 bit	31250100 bit
End-to-End Delay (cbr1)	0.004124 s	0.004103 s	0.004015 s
End-to-End Delay (cbr6)	0.002239 s	0.002204 s	0.002182 s
Throughput (cbr1)	4.861111 kbps	12.498818 kbps	62.500200 kbps
Throughput (cbr6)	4.861111 kbps	12.499091 kbps	62.490000 kbps

Table 11 Performance measurements before and after node 1 and 6 affected by rain

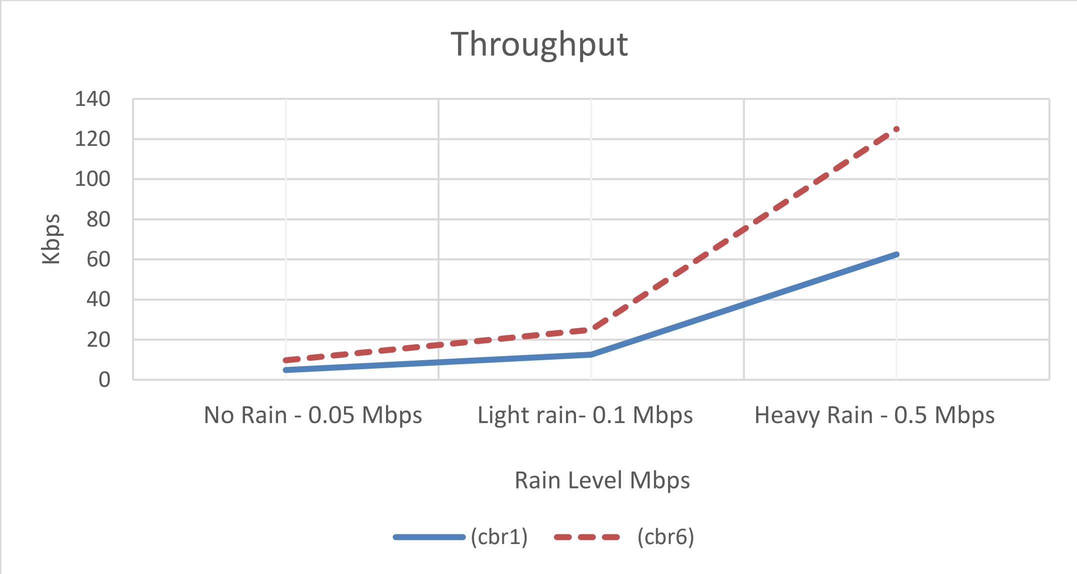


Figure 16 Throughput (node 1 and node 6)

In Figure 16 we can clearly notice that for no rain with standard AOMDV, the throughput is less. But in contrast to this even at the light and heavy rain, the throughput is more with our AOMDV-E protocol.

5.4.3 Complex 30 nodes Simulation Result

To prove that our WSN protocol is scalable and can work in a complex scenario, the third scenario comprises of thirty nodes. Out of 30 nodes, five nodes had rainfall. These nodes can have a maximum range of approx. 15 km each so as far as Middle East climate is concerned it does not rain at such a vast area simultaneously to require more than thirty nodes, hence thirty nodes are more than enough. Furthermore, the area under cultivation or irrigation is also not so huge in any one location so that it needs more than this number of nodes to monitor it.

In this example, we divided the simulation period depending on raining level, for instance with no rain the period from 0 to 900s, light rain period from 900 to 2000s and heavy rain from 2000 to 2500s.

An AWK script was built to calculate the average of Throughput and End to End Delay on node 0 and node 1 before and after node 1 affected by rain. The highest priority data stream will be granted to cbr1 coming from node 1 because it is affected by rain and it has important data that should be sent to the destination node 29. The cbr0 data coming from node 0 should routing to node 10 which has data with low priority. In this way we reduce the forwarding load on node 1 this means that the throughput and end to end delay will be better on cbr1 after node 1 affected by rain.

Node 1 was affected by two levels of rain light rain and heavy rain, we compared the Throughput and End to End Delay in two levels.

The results listed in table 12 clearly show improvement of Throughput and End to End Delay on cbr1 coming from node 1 which validates our assumptions and it shows that the event-driven approach does improve overall network efficiency. The reason behind this increase in throughput is that at the light and heavy rain, we use the AOMDV-E protocol.

Type/Rain-Level (Time)	**No rain - 0.05Mbps (0 – 900) s**	**Light Rain - 0.1Mbps (900 – 2000) s**	**Heavy Rain - 0.5 Mbps (2000 – 2500) s**
Total Sent Size From CBR0	5000100 bit	6875400 bit	18746000 bit
Total Sent Size From CBR1	4375000 bit	13748700 bit	18750200 bit
End-to-End Delay (cbr0)	0.018779 s	0.018787 s	4.465589 s
End-to-End Delay (cbr1)	0.01662 s	0.016501 s	0.016415 s
Throughput (cbr0)	5.555667 kbps	6.250364 kbps	3.749200 kbps
Throughput (cbr1)	4.861111 kbps	12.498818 kbps	37.500400 kbps

Table 12 Performance measurements before and after node 1 affected by rain

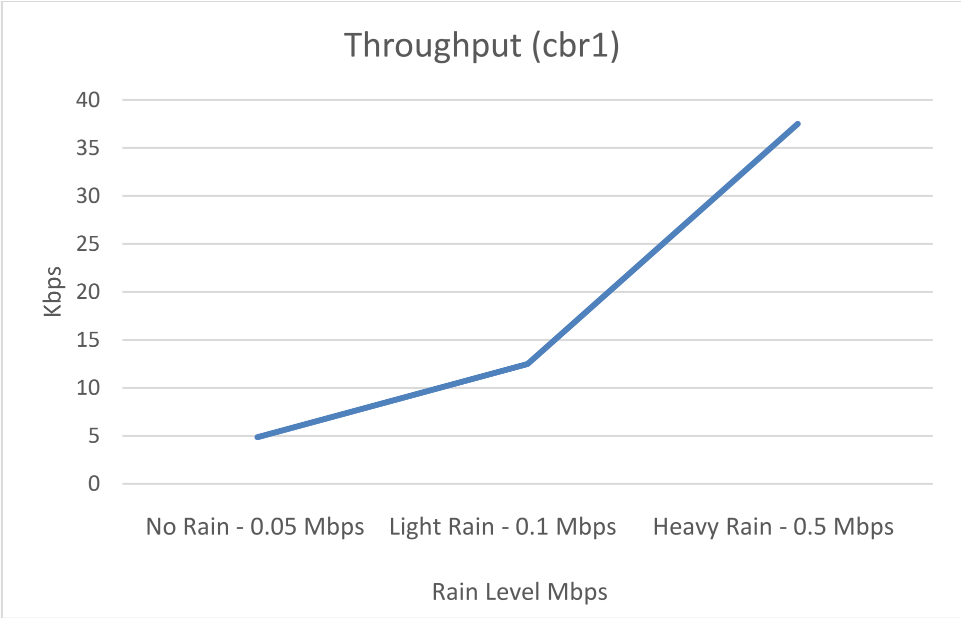


Figure 17 Throughput differences between no rain, light and heavy rain on node 1

Figure 17 shows the throughput of Node 1 separately being affected by rain. The reason behind the increase in throughput at the light and heavy rain is the modified AOMDV protocol. As we can see the result is almost similar to the result of 7 node simulation, but 30 node simulation shows that the proposed protocol is scalable and works in complex situations also.

5.5 Energy Measurements before and after nodes is affected by RAIN

We measured the energy consumption for the simulation scenario of WSN setup. Before rainfall, we used the standard AOMDV protocol and after the rainfall, we used our AOMDV-E protocol to compare the energy consumption of our proposed protocol with the standard AOMDV. Before rainfall, all the nodes send the sampled data after every 3 hours and are using multiple routes to send data to the gateway node. When there is no rainfall, HELLO messages are being sent but after rainfall is detected at any node, it sends STATUS messages to other nodes and stops sending HELLO messages, as a result, no data is now sent to the affected node, and its priority to send data also increases. By rerouting, the data sent to the affected node is greatly reduced hence the receiving energy is also reduced significantly. The affected node now sends data after every second, this operation is continued as long as it is raining. Hence when it is raining the affected nodes are receiving much less data and have just their own data to send, the overall energy for the affected nodes also decreases significantly.

We measure the energy consumption in three modes for each sensor node, idle mode, receiving mode and transmitting mode. The energy consumed in idle mode is sum of the energy consumed while the node is sleeping and while it is doing nothing and it is called idle energy (Idle_Energy). The energy consumed by a node in the transmission of data is called transmission energy (Trans_Energy) and the energy consumed by a node to receive data is called receiving energy (Rec_Energy).

When a node is affected by rain it will receive a rain event from its sensor, this event triggers a special broadcast rain packet called STATUS message. Every node that is adjacent to this node will receive this packet. The affected nodes will stop sending HELLO messages and also stop responding to RREQ messages, this will prevent the neighboring nodes from seeing it and sending their data through the affected node. The data of the node experiencing rainfall has its priority increased, so now the data from this node does not have to compete with data from other nodes, this affected node now cannot be used as an intermediate router as long as it is raining. As a result of these changes, transit time decreases for data from the rain-affected nodes. Also when the data from the rain-affected nodes reaches intermediate nodes and at the gateway node, it takes less time as it has a higher priority than data from non-affected nodes. Hence our proposed changes result in overall reduction in the transmission and send energies required whenever the nodes are affected by rain.

Hence the nodes affected by rainfall will not receive any packets from other nodes and hence they will use less receive energy, on the other hand, they will just have their own data to transmit, hence the transmission energy also decreases because it no longer acts like an intermediate router.

The total consuming energy has been calculated for all nodes before and after the nodes affected by rain for the three simulations. Also, we calculated the rate of consuming decreasing energy before and after the nodes affected by rain for each simulation

Energy = Power * Time where the power in watt, time in seconds and initial energy in joules

The total energy for the node (k) =

$$idle\_energy\ (k) + trans\_energy(k) + recv\_energy(k)$$

The total energy for all nodes =

$$\sum_{k=0}^{n} \mathrm{idle\_energy(k)} + trans\_energy(k) + recv\_energy(k)$$

5.5.1 Five Node Energy Measurements before and after node 1 is affected by RAIN

Table 13 and 14 show the total energy consumptions in each node before and after rain.

Node No.	Idle_Energy (joules)	Trans_Energy (joules)	Recv_Energy (joules)	Total_Energy (joules)
0	6082.3600	0.0040	159.2260	6241.5900
1	121271000.0000	1100760.0000	2326700.0000	124698460.0000
2	86630600.0000	437705.0000	2010770.0000	89079075.0000
3	8517.6100	0.0100	223.0000	8740.6200
4	13708.3000	0.0170	369.9540	14078.2710

Table 13 Energy Measurements before node 1 affected by Rain

Node No.	Idle_Energy (joules)	Trans_Energy (joules)	Recv_Energy (joules)	Total_Energy (joules)
0	993.9160	0.0020	6.1060	1000.0240
1	12122000.0000	1100720.0000	2326530.0000	15549250.0000
2	86626600.0000	437683.0000	2010690.0000	89074973.0000
3	3429.3100	0.0080	69.8800	3499.1980
4	5293.2400	0.0090	118.8940	5412.1430

Table 14 Energy Measurements after node 1 affected by Rain

As we can see in the tables above, the idle energy is not only non-zero but it is higher than other energy components. This is because these nodes must run the software and hence can still consume some amount of energy even in the sleep mode. Most of the time the node remains in sleep mode in which the consumed energy is less but the overall sum of the energy consumed in sleep mode is more than other energy components. For example, before rainfall, the nodes are in sleep mode most of the time, and only wake up after every three hours to record and send data for few seconds, and then again sleep.

Tables 13 and 14 clearly show that of consumption energy for affected nodes before is greater than energy consumption after rainfall starts. As we can see from the 'Total' columns in both tables, the total energy consumed decreases for all the nodes and the decrease in energy is much greater for the nodes affected by rainfall. The idle energy, transmission energy as well received energy decreases for all the nodes. The largest decrease in total energy is due to the decrease in transmission energy because it no longer acts like an intermediate router.

We calculate total energy consumed by all the nodes that are affected by rainfall.

So total consumed energy for all nodes before rains = T-energy_before = 213806595.4810

And total consumed energy for all nodes after rains = T-energy_after = 104634134.3650

Therefore the consumed energy decreasing rate = (T-energy_before/ T_energy_after) %

Consumed energy decreasing rate overall networks = 51%

We can clearly observe that the energy consumption of node 1 have been decreased sharply, also the energy consumption of all nodes have been reduced. The main reason for that we stop sending data from any neighbors through the node that affected by rain. Hence the nodes affected by rainfall will not receive any data from other nodes and hence they will use less receive energy, on the other hand, they will just have their own data to transmit, hence the transmission energy also reduces.

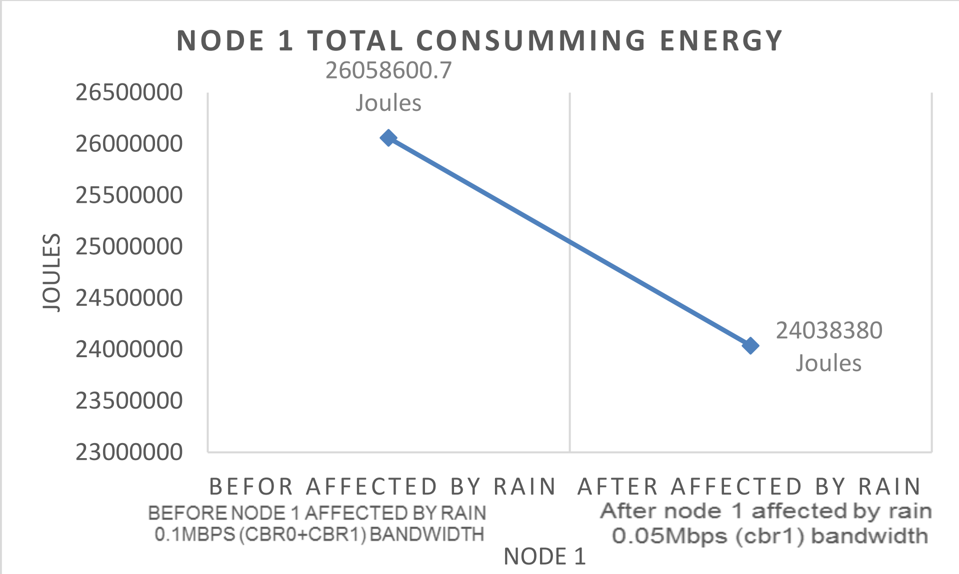


Figure 18 nodes consuming energy before and after node1 affected by rain

We can clearly observe in Figure 18 that the energy consumption of node 1 is high with standard AOMDV even without affected by rain. But with AOMDV-E, its energy consumption has extensively reduced even after affected by rain.

The energy consumption for all the non-affected nodes and nodes neighboring the affected nodes has also decreased. This is due to the fact that before rainfall there were many routes to send data. But after rainfall the affected nodes do not receive data from neighboring nodes, hence there is less traffic from the neighboring nodes around the affected nodes. So the energy consumption for the non-affected nodes and the nodes neighboring the affected nodes also decreases, as they send data through other nodes.

When we compare the result of Figure 15 and Figure 18, we find that the throughput of node 1 after it is affected by rain with AOMDV-E protocol is more than before being affected by rain with standard AOMDV protocol, while from Figure 18 we come to know that the overall energy consumption of node 1 has decreased. The reason for these two observations is that the after rainfall the neighboring traffic from all the nodes to node 1 is stopped as it is affected by rain, hence it has only its own data to send, and its own data rate is increased due to increased sampling in rainfall, resulting in increased throughput. At the same time, its overall energy consumption is decreased because node 1 does not receive any data from any other node, nor does it need to send Hello packets as before. Routes are now reduced as well, which results in overall energy being reduced. Hence both transmission, as well as received energy, is reduced after rainfall.

5.5.2 Seven Node Energy Measurements before and after node 1, 6 are affected by RAIN

Tables 15 and 16 show the total energy that consumed in each node before and after rain

Node No.	Idle_Energy (joules)	Trans_Energy (joules)	Recv_Energy (joules)	Total_Energy (joules)
0	582027000.0000	359133000.0000	1176360000.0000	2117520000.0000
1	701713000.0000	534323000.0000	1313660000.0000	2549696000.0000
2	711970000.0000	547011000.0000	1328500000.0000	2587481000.0000
3	355334000.0000	87519600.0000	848444000.0000	1291297600.0000
4	244580000.0000	62561000.0000	588576000.0000	895717000.0000
5	5375970.0000	6431500.0000	13960100.0000	25767570.0000
6	5447700.0000	6460100.7000	14150800.0000	26058600.7000

Table 15 Energy Measurements before node 1 affected by Rain

Node No.	Idle_Energy (joules)	Trans_Energy (joules)	Recv_Energy (joules)	Total_Energy (joules)
0	580992000.0000	358422000.0000	1151990000.0000	2091404000.0000
1	47977100.0000	3320310.0000	100964000.0000	152261410.0000
2	61373600.0000	5228600.0000	134840000.0000	201442200.0000
3	350340000.0000	87096300.0000	809392000.0000	1246828300.0000
4	242831000.0000	61458700.0000	580063000.0000	884352700.0000
5	6571700.0000	4587440.0000	12909300.0000	24068440.0000
6	6563460.0000	4523520.0000	12951400.0000	24038380.0000

Table 16 Energy Measurements after node 1 affected by Rain

As we can see in the tables above, the idle energy is not only non-zero but it is higher than other energy components. This is because these nodes must run the software and hence can still consume some amount of energy in the sleep mode. Most of the time the node remains in sleep mode in which the consumed energy is less but the overall sum of the energy consumed in sleep mode is more than other energy components. For example, before rainfall, the nodes are in sleep mode most of the time, and only wake up after every three hours to record and send data for few seconds, and then again sleep.

Table 15 and 16 clearly show that of consumption energy for affected nodes before is greater than energy consumption after rainfall starts. As we can see from the 'Total' columns in both tables, the total energy consumed decreases for all the nodes and the decrease in energy is much greater for the nodes affected by rainfall. The idle energy, transmission energy as well received energy decreases for all the nodes. The largest decrease in total energy is due to the decrease in transmission energy.

So total consumed energy for all nodes before rains = T-energy-before = 9493537770.7000

And total consumed energy for all nodes after rains = T-energy-after = 4624395430.0000

Therefore the consumed energy decreasing rate = (T-energy-before/ T-energy-after) %

Consumed energy decreasingrate = 51%.

We can clearly observe that the energy consumptions for affected nodes have been decreased sharply, also the energy consumption of all nodes have been decreased. The main reason for that we stop sending data from any neighbors through the node that affected by rain. Hence the nodes affected by rainfall will not receive any data from other nodes and hence they will use less receive energy, on the other hand, they will just have their own data to transmit, hence the transmission energy also decreases.

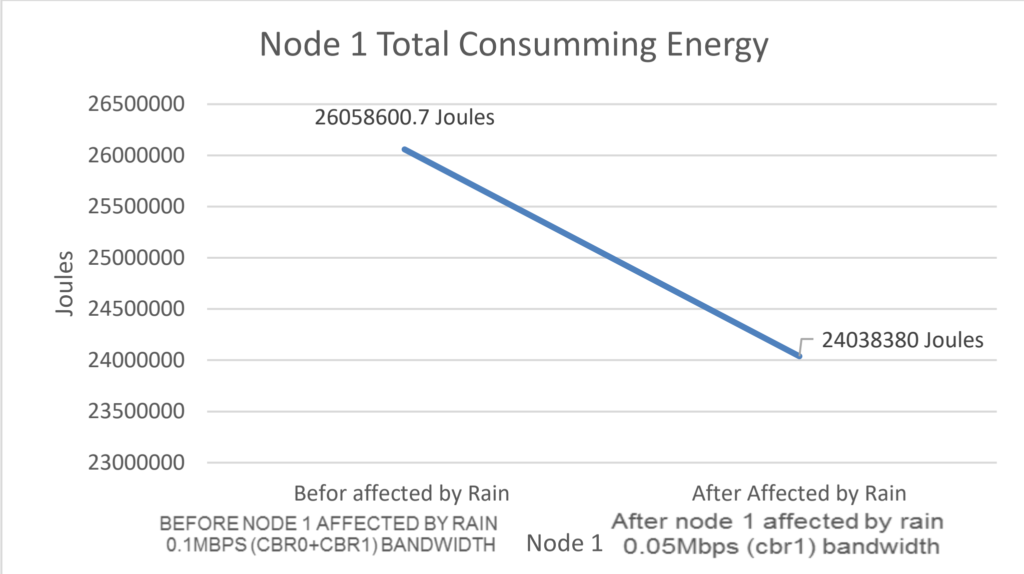


Figure 19 nodes consuming energy before and after node1 affected by rain

We can clearly observe in Figure 19 that the energy consumption of node 1 is high with standard AOMDV even without affected by rain. But with AOMDV-E, its energy consumption has extensively reduced even after affected by rain. The logic behind this is, "The total energy drop through node 1 is due to the drop in traffic as node 0 is routed elsewhere"

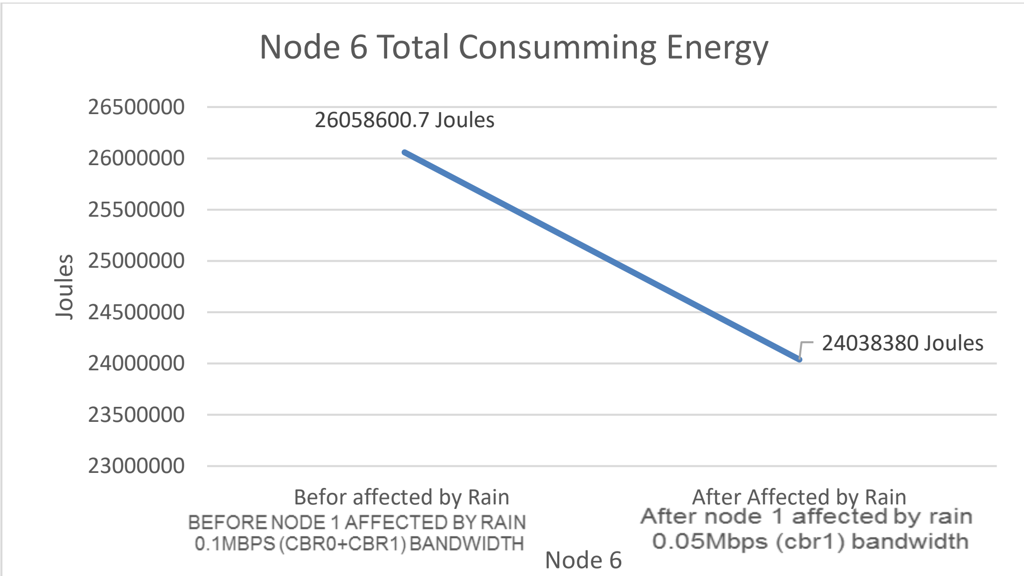


Figure 20 nodes consuming energy before and after node6 affected by rain

Again, we can again notice in Figure 20 that the energy consumption of node 6 is high with standard AOMDV even without affected by rain. But with AOMDV-E, its energy consumption has extensively reduced after affected by rain.

The energy consumption for all the non-affected nodes and nodes neighboring the affected nodes has also decreased. This is due to the fact that before rainfall there were many routes to send data. But after rainfall the affected nodes do not receive data from neighboring nodes, hence there is less traffic from the neighboring nodes. So the energy consumption for the non-affected nodes and the nodes neighboring the affected nodes also decreases.

5.5.3 Complex Simulation Result: Energy Measurements before and after node 1 is affected by RAIN

Tables 17 and 18 shows that of consumption energy for all nodes before is greater than after rain

Node No.	Total_Energy (joules)	Idle_Energy (joules)	Tran_Energy (joules)	Rec_Energy (joules)
0	65510893.00	58800000.00	520893.00	6190000.00
1	218940000.00	191000000.00	4140000.00	23800000.00
2	271310000.00	230000000.00	6310000.00	35000000.00
3	23657457.40	19500000.00	17457.40	4140000.00
4	271470000.00	218000000.00	627 0000.00	47200000.00
5	23817465.40	19100000.00	17465.40	4700000.00
6	270560000.00	222000000.00	6260000.00	42300000.00
7	23847777.30	20100000.00	17777.30	3730000.00
8	23867648.60	20700000.00	17648.60	3150000.00
9	15931799.90	14200000.00	11799.90	1720000.00
10	23877587.00	21400000.00	17587.00	2460000.00
11	35556242.50	31000000.00	26242.50	4530000.00
12	35506380.00	30100000.00	26380.00	5380000.00
13	283580000.00	234000000.00	6580000.00	43000000.00
14	35396011.00	28400000.00	26011.00	6970000.00

15	282940000.00	227000000.00	6540000.00	49400000.00
16	35726423.20	29300000.00	26423.20	6400000.00
17	282240000.00	238000000.00	6540000.00	37700000.00
18	35756510.60	31000000.00	26510.60	4730000.00
19	23887672.00	21300000.00	17672.00	2570000.00
20	15951791.90	14300000.00	11791.90	1640000.00
21	23737540.90	20700000.00	17540.90	3020000.00
22	23707469.40	20100000.00	17469.40	3590000.00
23	23637474.40	19500000.00	17474.40	4120000.00
24	23667335.30	19000000.00	17335.30	4650000.00
25	23817550.60	19100000.00	17550.60	4700000.00
26	23887561.70	19600000.00	17561.70	4270000.00
27	23847623.80	20100000.00	17623.80	3730000.00
28	270960000.00	235000000.00	6260000.00	29700000.00
29	139060000.00	124000000.00	1160000.00	13900000.00
Total	**2875654215.9**	**2416300000**	**50964215.90**	**408390000**

Table 17 Energy Measurements before node 1 affected by Rain

Node No.	Total_Energy (joules)	Idle_Energy (joules)	Tran_Energy (joules)	Rec_Energy (joules)
0	61478769.00	55300000.00	488769.00	5690000.00
1	140710000.00	123000000.00	1310000.00	16400000.00
2	194990000.00	166000000.00	2790000.00	26200000.00
3	19814613.80	16400000.00	14613.80	3400000.00
4	201370000.00	162000000.00	2870000.00	36500000.00
5	19874744.50	16000000.00	14744.50	3860000.00
6	20853751.70	17300000.00	43751.70	3510000.00

7	19774622.20	16700000.00	14622.20	3060000.00
8	23837627.60	20700000.00	17627.60	3120000.00
9	16011897.80	14300000.00	11897.80	1700000.00
10	15921647.00	14300000.00	11647.00	1610000.00
11	93520284.00	81800000.00	920284.00	10800000.00
12	23851969.88	20300000.00	1969.88	3550000.00
13	210210000.00	174000000.00	3010000.00	33200000.00
14	31533292.90	25400000.00	23292.90	6110000.00
15	274560000.00	221000000.00	6360000.00	47200000.00
16	27722152.14	22800000.00	2152.14	4920000.00
17	32767147.80	27800000.00	67147.80	4900000.00
18	35706614.70	31000000.00	26614.70	4680000.00
19	23967682.10	21400000.00	17682.10	2550000.00
20	11908834.09	10700000.00	8834.09	1200000.00
21	15771225.74	13800000	1225.74	1970000
22	85239388.00	72600000.00	839388.00	11800000.00
23	89074818.00	73800000.00	874818.00	14400000.00
24	88871212.00	71700000.00	871212.00	16300000.00
25	16011241.87	12900000	1241.87	3110000.00
26	261640000.00	215000000.00	5740000.00	40900000.00
27	265430000.00	224000000.00	5830000.00	35600000.00
28	270570000.00	235000000.00	6270000.00	29300000.00
29	138960000	124000000.00	1160000.00	13800000.00
Total	**2731953536.82**	**2301000000**	**39613536.82**	**391340000**

Table 18 Energy Measurements after node 1 affected by Rain

As we can see in the tables above, the idle_energy is not only non-zero but it is higher than other energy components. This is because these nodes must run the software and hence can still consume some amount of energy in the sleep mode. Most of the time the node remains in sleep mode in which the consumed energy is less but the overall sum of the energy consumed in sleep mode is more than other energy components. For example, before rainfall, the nodes are in sleep mode most of the time, and only wake up after every three hours to record and send data for few seconds, and then again sleep.

Tables 17 and 18 clearly show that of consumption energy for affected nodes before is greater than energy consumption after rainfall starts. As we can see from the 'Total' columns in both tables, the total energy consumed decreases for all the nodes and the decrease in energy is much greater for the nodes affected by rainfall. The idle energy, transmission energy as well received energy decreases for all the nodes. The largest decrease in total energy is due to the decrease in transmission energy.

So total consumed energy for all nodes before rains = T-energy-before = 337727894.70

And total consumed energy for all nodes after rains = T-energy-after = 224066589.63

Therefore the consumed energy decreasing rate = (T-energy-before/ T-energy-after) %

Consumed energy decreasing rate = 51%

We can clearly observe that the energy consumptions for all the affected nodes have been decreased sharply, also the energy consumption of all nodes have been decreased. The main reason for that we stop sending data from any neighbors through the node that affected by rain. Hence the nodes affected by rainfall will not receive any data from other nodes and hence they will use less receive energy, on the other hand, they will just have their own data to transmit, hence the transmission energy also decreases.

Node ID.	Energy_Before (joules)	Energy_After (joules)
1	218940000.00	120710000.00
12	35506380.00	21051969.88
16	35726423.20	19812212.14
21	23737540.90	11771225.74
25	23817550.60	12011241.87

Table 19 the energy consuming an all nodes affected by rains

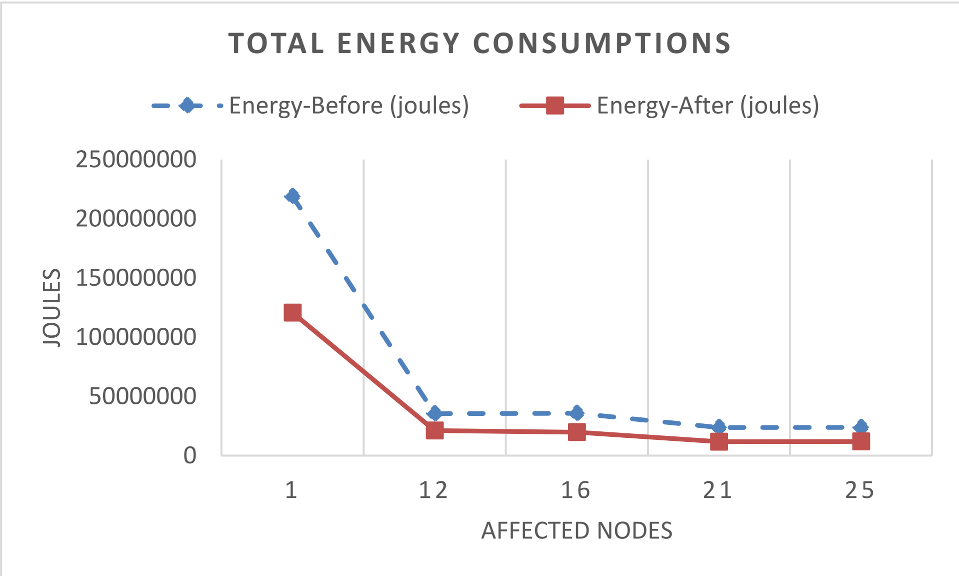


Figure 21 nodes consuming energy after and before affected by rains

The above Figure 21 is showing energy consumption by 5 nodes before and after they are affected by rain. Again our AOMDV-E clearly outperforms the standard AOMDV in term of energy consumption.

The energy consumption for all the non-affected nodes and nodes neighboring the affected nodes has also decreased. This is due to the fact that before rainfall there were many routes to send data. But after rainfall the affected nodes do not receive data from neighboring nodes, hence there is less traffic from the neighboring nodes. So the energy consumption for the non-affected nodes and the nodes neighboring the affected nodes also decreases.

5.6 Chapter Summary

In this chapter, we looked at the simulation setup and the different simulation configurations. We set up five nodes, seven node and thirty node network simulation to test our system comprehensively and to also check the scalability of the proposed protocol. We also defined the simulation environment and its variables. We provided the routing tables and the simulation results for all the WSN configuration. We checked seven and thirty node configurations with having rainfall at more than one node and having light and heavy rainfall with different bandwidth rates. After performing all the simulation experiments we evaluated the WSN performance for each configuration setup and also calculated the energy measurements. For performance evaluation, the throughput and the delay time were analyzed and for energy measurements, idle energy, receiving energy and transmission energy was analyzed.

Chapter 6: WSN Testbed Setup and Results

This chapter will describe the WSN testbed setup and give its routing tables. We will also look into the performance measurements and the energy consumption for the testbed in different configurations and conditions. We will also compare the testbed results with standard AOMDV protocol.

6.1 WSN Testbed Setup

We set up our testbed in a garden. The current network deployment has five nodes, which are 150 meters apart from each other, but more generally the architecture supports many nodes [20]. In our protocol, the sensor data are sampled for 10 seconds after every 3 hours, when there is no rainfall. During these 3 hours the sensors are in sleep mode to save energy and after the data is sampled they again go into sleep mode for another 3 hours. In case we have rainfall event the data is sampled after every 10 seconds for the sensor node experiencing the rainfall event. That means the affected nodes are up for 10 seconds and down for 10 seconds. In our protocol, we also increase the priority of this affected node to transfer data to the base station through the gateway. All the other sensor nodes where there is no rainfall still keep sampling after every 3 hours.

The setup of this scenario consists of 5 wireless nodes from node 0 (N0) to node 3 (N3), as shown in Figure 22 and one gateway node (GW) which is connected to the server. We will have two constant bit rates (cbr) streams one from node 0 and the other from node 2. The destination for both streams will be gateway node (GW). We will call them cbr0 and cbr2 respectively. According to AOMDV protocol, cbr2 traffic will go from node 2 to node 0 then node GW, and cbr0 traffic will go to node GW directly. Node 2 traffic, cbr2 will go through two paths, the primary one through node 0 and the secondary through node 1. After three hours of sending packets, node 0 is affected by RAIN (which is the event).

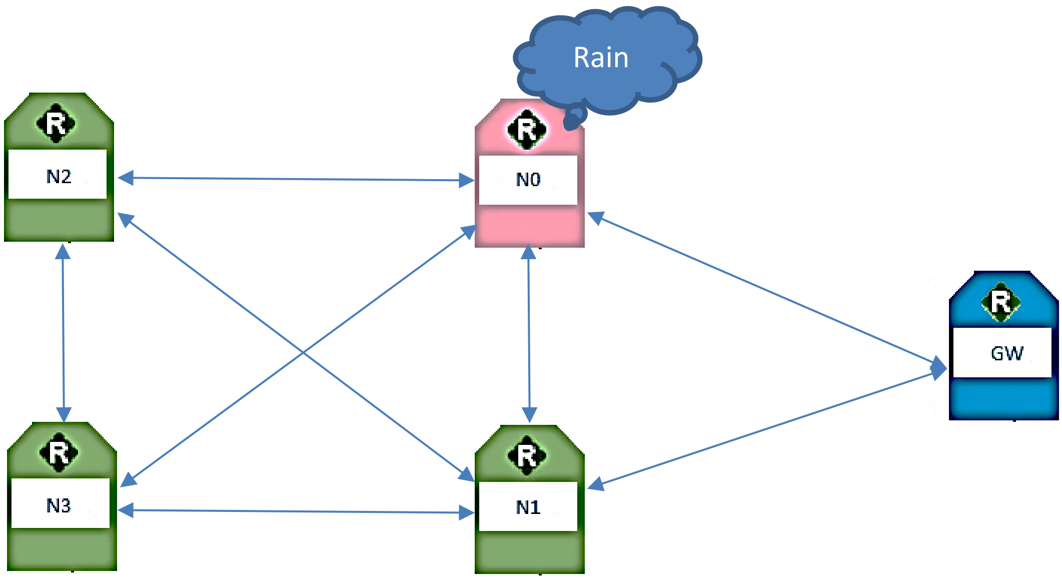


Figure 22 Five Node WSN Test Bed Setup

Tables 20 and 21 show the results of routing table before and after rain is detected at node 0, the routing table for node 2 and 3 has been changed which take another route through node 1 instead of the primary one through node 0 to send data to GW (destination) because node 0 is affected by RAIN. Hence node 0 has very important data (soil moisture) and it has to send it first and should take the highest priority to send data.

N0		
Dest No	Nxt hop	Hop cnt
3	3	1
GW	GW	1
1	1	1
2	2	1

N1		
Dest No	Nxt hop	Hop cnt
3	3	1
GW	GW	1
2	2	1
0	0	1

N2		
Dest No	Nxt hop	Hop cnt
3	3	1
1	1	1
0	0	1
GW	1	2
	0	2

N3		
Dest No	Nxt hop	Hop cnt
1	1	1
2	2	1
0	0	1
GW	1	2
	0	2

GW		
Dest No	Nxt hop	Hop cnt
0	0	1
1	1	1
2	0	2
	1	2
3	0	2
	1	2

Table 20 Routing Table before rain being detected at node 0

N0		
Dest No	**Nxt hop**	**Hop cnt**
3	3	1
GW	GW	1
1	1	1
2	2	1

N1		
Dest No	**Nxt hop**	**Hop cnt**
3	3	1
GW	GW	1
2	2	1

N2		
Dest No	**Nxt hop**	**Hop cnt**
3	3	1
1	1	1
GW	1	2

N3		
Dest No	**Nxt hop**	**Hop cnt**
1	1	1
2	2	1
GW	1	2

GW		
Dest No	**Nxt hop**	**Hop cnt**
1	1	1
2	1	2
3	1	2

Table 21 Routing Table after rain being detected at node 0

Tables 20 and 21 show the result of routing table before and after rain is detected at node 0, the routing table for node 2 and 3 has been changed which take another route through node 1 instead of the primary one through node 0 to send data to GW (destination) because node 0 is affected by RAIN. Hence node 0 has very important data (soil moisture) and it has to send it first and should take the highest priority to send data.

6.2 Performance Measurements

6.2.1 Performance Measurements for 5 node WSN Testbed:

An AWK script was built to calculate the average of Throughput and End to End Delay on node 0 and node 2 before and after node 0 was affected by rain. Before rainfall, we used the standard AOMDV protocol and after the rainfall, we used our AOMDV-E protocol to compare its performance with the standard AOMDV. The highest priority data stream will be granted to cbr0 coming from node 0 because it is affected by rain and it has important data that should be sent to the destination GW. Re-routing cbr2 coming from node 2 traffic through node 1 which has data with low priority. In this way we reduce the forwarding load on node 0, that means the throughput and end to end delay will be better on cbr0 after node 0 is affected by rain because less traffic is now going through that node.

The results listed in Table 22 clearly show improvement of Throughput and End to End Delay on cbr0 coming from node 0 which validate our assumptions and it showing that the event-driven approach does improve overall network efficiency.

Type/protocol and Time	Standard AOMDV 3 hours Before Rain Starts	AOMDV-E 3 hours After Rain Starts
Total Sent Size From cbr2	5000000 bit	3749000 bit
Total Sent Size From cbr0	4375000 bit	6874700 bit
End-to-End Delay (cbr2)	256.91 s	349.41 s
End-to-End Delay (cbr0)	16.21 s	0.00567 s

Throughput (cbr2)	2.65 kbps	3.7 kbps
Throughput (cbr0)	4.2 kbps	6.2 kbps

Table 22 Performance Measurement before and after node 0 is affected by rain

6.3 Energy Consumption

We measured the energy consumption for the WSN testbed and also for simulation networks. Before rainfall, we used the standard AOMDV protocol and after the rainfall, we used our AOMDV-E protocol to compare the energy consumption of our proposed protocol with the standard AOMDV. Before rainfall, all the nodes sent the sampled data after every 3 hours and are using multiple routes to send data to the gateway node. When there is no rainfall, HELLO messages are being sent but after rainfall is detected at any node, it sends STATUS messages to other nodes and stops sending HELLO messages, as a result, no data is now sent to the affected node and its priority to send data also increases. By rerouting, the data sent to the affected node is greatly reduced hence the receiving energy is also reduced significantly. The affected node now sends data after every 10 seconds, this operation is continued as long as it is raining. Hence when it is raining the affected nodes are receiving much less data and have just their own data to send, the overall energy for the affected nodes also decreases significantly.

We measure the energy consumption in three modes for each sensor node, idle mode, receiving mode and transmitting mode. The energy consumed in idle mode is called idle energy (Idle_Energy). The energy consumed by a node in the transmission of data is called transmission energy (Trans_Energy) and the energy consumed by a node to receive data is called receiving energy (Rec_Energy).

When a node is affected by rain it will receive a rain event, this event triggers a special broadcast rain packet called STATUS message. Every node that is adjacent to this node will receive this packet. The affected nodes will stop sending HELLO messages and stop responding to RREQ messages, this will prevent the neighboring nodes from seeing it and sending their data through the affected node.

Hence the nodes affected by rainfall will not receive any packets from other nodes and hence they will use less receive energy, on the other hand, they will just have their own data to transmit, hence the transmission energy also decreases.

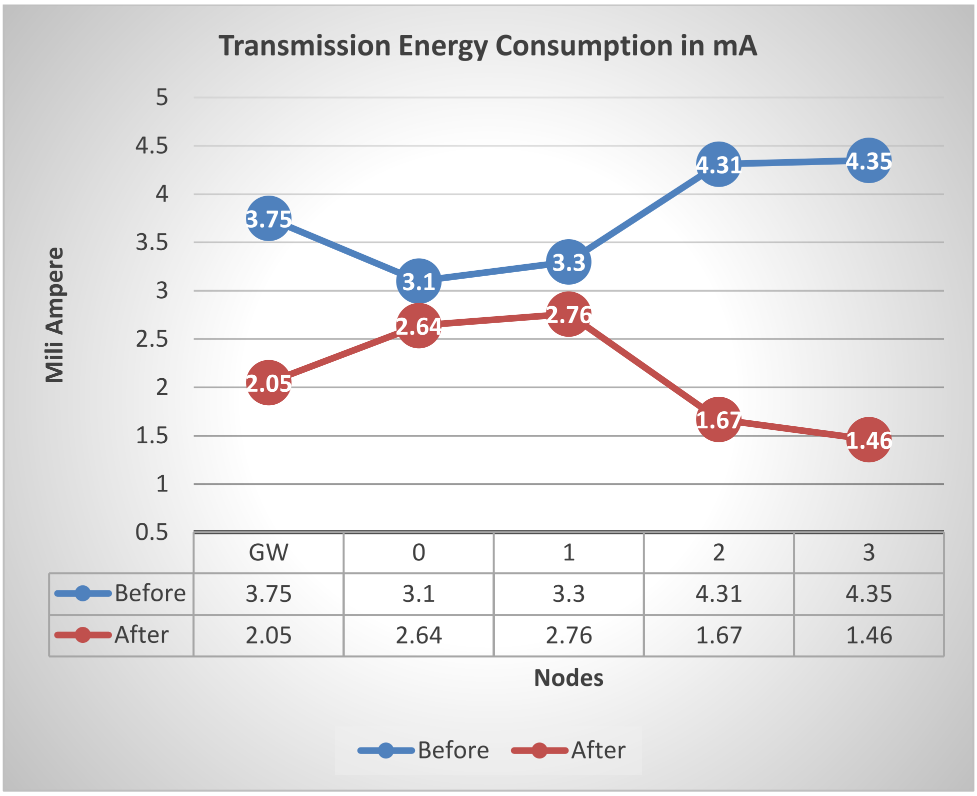
6.3.1: Energy Consumption for 5 node WSN Testbed:

The Waspmotes consumes 15 mA of energy per hour in switched on state and just consumes 55µA of energy per hour in sleep state [53]. We have designed our protocol to be energy-efficient. In our protocol sensor data under normal events are sampled for 10 seconds after every 3 hours and sent to the server. When we measured energy consumption for our protocol it was found that it only consumes 0.004 mA of energy per hour. In case we have rainfall event we implement scheme 2 under which we sample the data for 10 seconds after every 10 seconds, only for the sensor node experiencing the rainfall event, since the soil moisture may change very rapidly. In scheme 2, when there is a rainfall our protocol consumes only 0.042 mA of energy per hour for the affected node, which is also very less than the default energy consumption of 15 mA per hour for the Waspmote [53]. Hence our protocol is also energy efficient.

As shown in Tables 23 and 24 the energy measurements in mA taken after rainfall when we use AOMDV-E, are less than those taken before rainfall when we use standard AOMDV, for both received energy (Rec_Energy) as well as transmission energy (Trans_Energy) because the nodes affected by rainfall will not receive any data from other nodes and hence they will use less receive energy, on the other hand they will just have their own data to transmit, hence the transmission energy also decreases. We also show these results in Figs. 23 and 24.

Node No	Idle_Energy mili Ampere	Rec_Energy mili Ampere	Trans_Energy mili Ampere
GW	0.0275	3.75	3.75
0	0.0272	4.35	3.1
1	0.0271	4.33	3.3
2	0.0269	3.21	4.31
3	0.027	3.31	4.35

Table 23 Energy Measurements before node 0 affected by rain (Standard AOMDV)



	GW	0	1	2	3
Before	3.75	3.1	3.3	4.31	4.35
After	2.05	2.64	2.76	1.67	1.46

Figure 23 Transmission Energy Consumption in mA

As shown in Figure 23 above, the transmission energy for gateway node, node 2 and node 3 decreases significantly during rainfall but for node 0 and node 1 the reduction in transmission energy is by small amounts, this is because during rainfall the GW node, node 2 and node 3 do not send messages to the node 0 neither do they receive messages from node 0, hence they have less data to send. The transmission energy of node 0 and node 1 is reduced because node 0 is having rainfall and stops receiving any data from other nodes but it has its own data to send to the GW node every 10 seconds so its transmission energy reduces by little amount, for node 1 it does not have to send data to node 0 but now has extra traffic from other nodes which was before going through node 0, hence its transmission energy also recuces by small amount.

Node No	Idle_Energy mili Ampere	Rec_Energy mili Ampere	Trans_Energy mili Ampere
GW	0.0275	4.21	2.05
0	0.0265	1.02	2.64
1	0.0260	3.01	2.76
2	0.0245	1.82	1.67
3	0.0210	1.79	1.46

Table 24 Energy Measurements when node 0 is affected by rain (AOMDV-E)

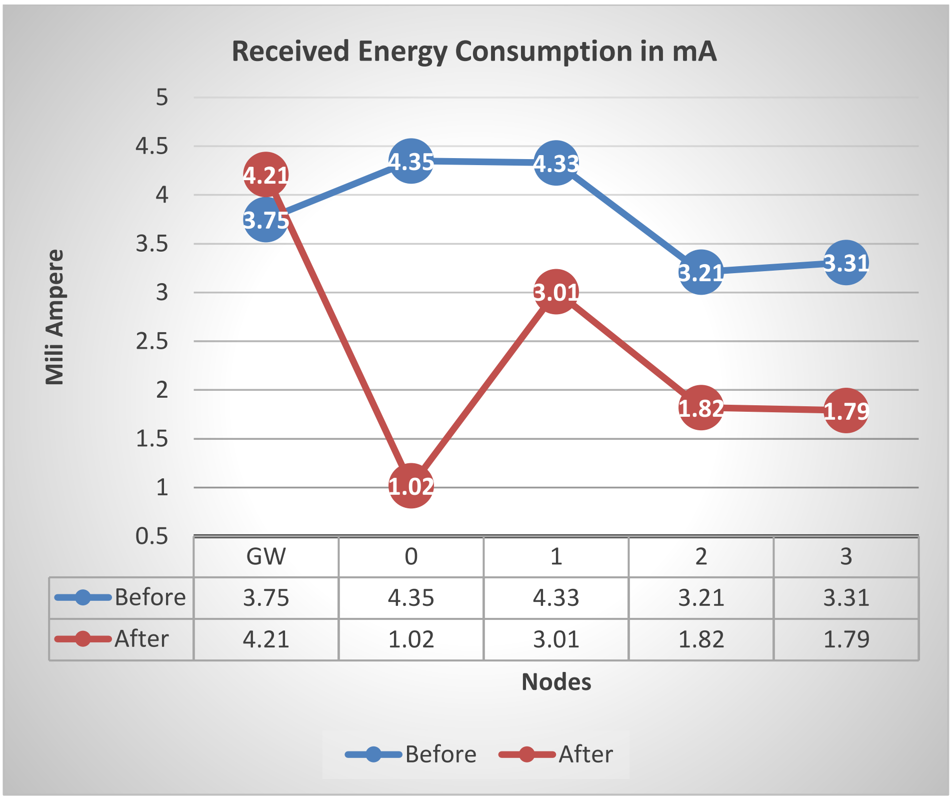


Figure 24 Received Energy Consumption in mA

As we can see in Figure 24 above, the received energy for the gateway does show an increase. This is because when raining starts the nodes sample data after every 10 seconds and send towards gateway node, hence the receiving energy for the gateway node increases. Therefore while it is raining, there is more work being done at the gateway node because the number of packets at the node has increased as sampling rate has increased.

We can clearly observe that the energy consumptions for all nodes affected by rain have decreased sharply, also the energy consumption of all nodes have decreased. The main reason for this decrease is that we stop sending data from any neighbors through the node that is affected by rain. Hence the nodes affected by rainfall will not receive any data from other nodes and hence they will use less receive energy, on the other hand, they will just have their own data to transmit, hence the transmission energy also decreases.

The energy consumption is considerably reduced for the nodes experiencing the rainfall and also in general because we used the sleep mode when it is not raining. We had shown that the energy consumptions for all nodes affected by rain had decreased sharply, also the energy consumption of all nodes had decreased since the overall transmission energy also decreased. Simulation results also showed that as the rainfall became heavier, then also the amount of energy consumed decreased which showed that our proposed AOMDV-E protocol was robust in terms of the amount of rainfall. Furthermore, the energy of the Gateway increased due to more packets being sent to it by rain affected nodes.

In our system, power is directly saved by using priority-based routing algorithm hence reducing the data at times of raining event when a lot of data is produced. However even after raining, data should be sampled more frequently in order to keep track of the soil moisture. When an adjacent node receives a rain packet it will immediately remove all next hops that this node is involved in. This will prevent any node from sending any data through this node. In addition, the data of the node experiencing rainfall will also have its priority increased, so now the data from this node does not have to compete with data from other nodes, this affected node now cannot be used as an intermediate router as long as it is raining. As a result of these changes, transit time decreases for data from the rain-affected nodes. Also when the data from the rain-affected nodes reaches intermediate nodes and at the gateway node, it takes less time as it has a higher priority than data from non-affected nodes. Hence our proposed changes result in overall reduction in the transmission and send energies required whenever the nodes are affected by rain.

6.4 Chapter Summary

This chapter described the WSN testbed setup and calculated the routing tables. We also looked into the performance measurements and the energy consumption for the testbed in different configurations and conditions. Performance evaluation was done based on throughput and end-to-end delay, energy evaluation was done based on the summation of idle energy, receiving energy and transmission energy. Finally, we compared the testbed results with standard AOMDV protocol. AOMDV protocol was used before rainfall and our proposed protocol was used after rainfall to show that when there is rainfall event our method gives better performance and consumes less energy than the standard AOMDV.

Chapter 7: Conclusion and Future Scope

In Chapter 2, we reviewed wireless sensor networks and their applications in environmental monitoring and other fields. We studied common WSN architecture, components, and hardware. We looked at some latest WSN communication protocols and compared the ad-hoc on-demand distance vector (AODV) routing and ad-hoc on-demand multipath distance vector routing (AOMDV) routing protocols with some other popular routing protocols. We also explored some latest studies that are closely related to our work.

In Chapter 3, we gave the research methodology and discussed the various simulation environments. We looked in detail at building the WSN prototype and its architecture. We also gave a detailed summary of the hardware components and software that we used for the WSN setup.

In Chapter 4, we analyzed the problem and the solution approach in our study. We discussed details of the proposed AOMDV-E protocol and the changes we made to the standard AOMDV protocol.

In Chapter 5, we looked at the simulation setup and the different simulation configurations used in our study. We provided the routing tables and the simulation results for all the WSN configuration. After performing all the simulation experiments we evaluated the WSN performance for each configuration setup and also calculated the energy measurements.

At last in Chapter 6, we described the WSN testbed setup and calculated the routing tables. We evaluated the performance measurements and the energy consumption for the testbed for the different configurations and conditions.

7.1 Conclusion

This thesis outlined the development of AOMDV-E, an enhanced AOMDV routing protocol that is reactive, event-driven, energy efficient and robust that is capable of providing timely and reliable information about soil moisture scenarios. The proposed AOMDV-E protocol has the ability to be used for precision agriculture and smart irrigation systems in the future. This thesis described the design, deployment, and evaluation of our reactive, energy aware and enhanced protocol. The thesis presents detailed simulation scenarios and testbed setup to test the proposed protocol comprehensively. Detailed analysis and comparison of all popularly available routing protocols (Proactive Reactive and Hybrid) were also presented. Performance analysis and evaluation of the proposed AOMDV-E routing protocol is done using different quantitative metrics including Average End-to-End Delay, Throughput, and Energy consumption. Energy measurements have been calculated in different phases of operation of the protocol.

Our event-driven enhancements made our protocol not only increase the performance but also made it energy efficient as the energy consumption is considerably reduced for the nodes experiencing the rainfall and also in general because we used the sleep mode when it is not raining. We had shown that the energy consumptions for all nodes affected by rain had decreased sharply, also the energy consumption of all nodes had decreased since the overall transmission energy also decreased. Simulation results also showed that as the rainfall became heavier, then also the amount of energy consumed decreased which showed that our proposed AOMDV-E protocol was robust in terms of the amount of rainfall. Furthermore, the energy of the Gateway increased due to more packets being sent to it by rain affected nodes.

In our system, power is directly saved by using priority-based routing algorithm hence reducing the data at times of raining event when a lot of data is produced. However even after raining, data should be sampled more frequently in order to keep track of the soil moisture. When an adjacent node receives a rain packet it will immediately remove all next hops that this node is involved in. This will prevent any node from sending any data through this node. In addition, the data of the node experiencing rainfall will also have its priority increased, so now the data from this node does not have to compete with data from other nodes, this affected node now cannot be used as an intermediate router as long as it is raining. As a result of these changes, transit time decreases for data from the rain-affected nodes. Also when the data from the rain-affected nodes reaches intermediate nodes and at the gateway node, it takes less time as it has a higher priority than data from non-affected nodes. Hence our proposed changes result in overall reduction in the transmission and send energies required whenever the nodes are affected by rain.

We also successfully showed in simulation results that the enhanced AOMDV-E protocol was scalable. We compared its performance with the normal AOMDV protocol. For our proposed protocol the results clearly show that it reduced the average delay while at the same time increased the throughput of the nodes that were affected by rain. Our AOMDV-E protocol clearly outperformed the standard AOMDV in term of energy consumption as well as in terms of throughput. The enhanced throughput and low delay clearly indicated the proposed changes to AOMDV to support events such as rain would be a significant and meaningful addition to AOMDV which in future can be used in several WSN applications.

7.2 Future Scope

In this work, we had applied our AOMDV-E protocol for soil moisture monitoring, but it is not limited to this application, and we chose this application based on its importance to the local climate and area. In future, we want to deploy and evaluate our reactive, energy aware and enhanced version of AOMDV protocol for other applications like agricultural monitoring, natural habitat monitoring, greenhouse monitoring, wildlife monitoring, climate monitoring, forest monitoring, etc. Since our event-driven enhancements make our protocol energy efficient it can be used in dangerous and inaccessible areas where providing or replenishing energy is a big problem.

In the future, we need to make use of transport layer protocols such as multipath TCP which efficiently uses several Internet paths between a pair of hosts, while showing a single TCP connection to the application layer. Multipath TCP offers benefits like better resource utilization, better throughput and smoother reaction to failures. Hence it would be interesting to investigate how the performance of the proposed AOMDV-E protocol can be increased in more complex environments by using it with Multipath TCP. Multipath TCP is a good fit to improve reliability and performance of AOMDV-E since it is multipath already so Multipath TCP should just work.

We also want to investigate in future how our modified algorithm reacts to different attacks like the flooding attack or the refusal of service attacks.

In future for WSN energy optimization, we should further investigate into different node platforms, behavioral studies in real-world deployments. There is a need to investigate hybrid architectures for node platforms, as most of the high-performance data processing is handled by the sensor, and the system and communication control is handled by the central server. Therefore there is unequal energy distribution which should be addressed. Furthermore, we need to investigate the increased power at the gateway.

We need long-term studies of WSN at the deployment stage which may uncover data that helps us relate environmental conditions to WSN behavior. We need to further harness different renewable forms of energy like the wind energy to make the WSN system more energy-efficient to increase the system lifetime.

As shown in simulation results our enhanced AOMDV-E protocol is also scalable it can be used in large-scale WSN applications. Our WSN testbed was tested with just 4 nodes, so for future work, a large-scale deployment is proposed in order to assess the ability of modified AOMDV-E protocol in handling numerous queries from the in-field sensors.

It is further proposed that future deployments should focus on improving water application efficiency for irrigation. In this way, both water and energy used in irrigation water pumping will be conserved. This could result in promoting installations of low-capacity solar photovoltaic water pumping systems for irrigation to suit the socioeconomic conditions of small-scale farmers in developing countries. It would be interesting to explore the practical performance of WSN based soil moisture monitoring systems in areas other than Jordan where there are sudden and heavy showers which tend to destroy the crops.

We also need to test the proposed WSN system in future using different topology other than mesh or make the nodes autonomous so that they can group in a configuration which are most beneficial in terms of energy efficiency and throughput.

Future work will focus on addressing the limitations of the current prototype in the robustness of packet delivery and network longevity, and in guaranteeing network response to events of interest. We also plan to generalize our event-condition-action protocol for programmable reactive sensor networks.

References

[1] Matin M. A. and Islam M. M. Overview of Sensor Networks Computer, Sep 2012

[2] Steere D., Baptista A., McNamee, C. Pu, and J. Walpole. Research challenges in environmental observation and forecasting systems, in Proceedings of the 6th International Conference on Mobile Computing and Networking (MOBICOM), Sep 2011

[3] Martinez K., Hart J., and R. Ong. Environmental Sensor Networks. Computer, Nov 2010.

[4] Jayapriya .P, M, S.V. (2018). Precision Agriculture Using Wireless Sensor Network System: Opportunities and Challenges. International Journal Of Engineering And Computer Science, 5(11). http://ijecs.in/index.php/ijecs/article/view/3070.

[5] Jain N, Madan S, Malik SK. Review of Multiparameter Techniques for Precision Agriculture Using Wireless Sensor Network. Journal of Network Communications and Emerging Technologies (JNCET) 2018.

[6] Prabhu, Boselin and Pradeep, M. and Gajendran, E., Monitoring Climatic Conditions Using Wireless Sensor Networks (January 25, 2017). A Multidisciplinary Journal of Scientific Research & Education, 3(1), January-2017, Volume: 3, Issue: 1.

[7] Mompoloki Pule, Abid Yahya, Joseph Chuma, Wireless sensor networks: A survey on monitoring water quality, Journal of Applied Research and Technology, Volume 15, Issue 6, 2017, Pages 562-570, ISSN 1665-6423.

[8] Kumar N, Singh Y, Niranjan KR, Kumar TA. Routing protocols in wireless sensor networks. Handbook of Research on Advanced Wireless Sensor Network Applications, Protocols, and Architectures' (IGI Global, 2016). 2017:86-128.

[9] Kumar A, Shwe HY, Wong KJ, Chong PH. Location-Based Routing Protocols for Wireless Sensor Networks: A Survey. Wireless Sensor Network. 2017 Jan 17;9(01):25.

[10] Hasan MZ, Al-Rizzo H, Al-Turjman F. A survey on multipath routing protocols for QoS assurances in real-time wireless multimedia sensor networks. IEEE Communications Surveys & Tutorials. 2017 Jan 1;19(3):1424-56.

[11] Ridha OA, Jasim DM. Design and Implementation of Enhanced Smart Energy Metering System. Journal of Engineering. 2017 Feb 28;23(3):61-79.

[12] Yuan D, Kanhere SS, Hollick M. Instrumenting wireless sensor networks—a survey on the metrics that matter. Pervasive and Mobile Computing. 2017 Jun 1;37:45-62.

[13] Hasan MZ, Al-Rizzo H, Al-Turjman F. A survey on multipath routing protocols for QoS assurances in real-time wireless multimedia sensor networks. IEEE Communications Surveys & Tutorials. 2017 Jan 1;19(3):1424-56.

[14] Al-Karaki, J.N.; Kamal, A.E. Routing Techniques in Wireless Sensor Networks: A Survey. IEEE Wirel. Commun. 2011, 11, 6-28.

[15] Akkaya, K.; Younis, M. A Survey on Routing Protocols for Wireless Sensor Networks. Ad Hoc Netw. J. 2010, 3, 325-349.

[16] Son, D.; Krishnamachari, B.; Heidemann, J. Experimental Study of Concurrent Transmission in Wireless Sensor Networks. In Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (SenSys '06), Boulder, CO, USA, 31 October-3 November 2011; pp. 237-250.

[17] Kafi MA, Ben-Othman J, Ouadjaout A, Bagaa M, Badache N. REFIACC: Reliable, efficient, fair and interference-aware congestion control protocol for wireless sensor networks. Computer Communications. 2017 Mar 15;101:1-1.

[18] Lou, W.; Liu, W.; Zhang, Y. Performance Optimization Using Multipath Routing in Mobile Ad Hoc and Wireless Sensor Networks. Combinator. Optim. Commun. Netw. 2012, 2, 117-146.

[19] Kafi MA, Othman JB, Badache N. A survey on reliability protocols in wireless sensor networks. ACM Computing Surveys (CSUR). 2017 May 25;50(2):31.

[20] Zamalloa, M.Z.; Krishnamachari, B. An Analysis of Unreliability and Asymmetry in Low-Power Wireless Links. ACM Trans. Sens. Netw. 2011, 3, doi:10.1145/1240226.1240227.

[21] Yetgin H, Cheung KT, El-Hajjar M, Hanzo LH. A survey of network lifetime maximization techniques in wireless sensor networks. IEEE Communications Surveys & Tutorials. 2017 Jan 1;19(2):828-54.

[22] Tarique, M.; Tepe, K.E.; Adibi, S.; Erfani, S. Survey of Multipath Routing Protocols for Mobile Ad Hoc Networks. J. Netw. Comput. Appl. 2009, 32, 1125-1143.

[23] Mueller, S.; Tsang, R.; Ghosal, D. Multipath Routing in Mobile Ad Hoc Networks: Issues and Challenges. Lect. Note. Comput. Sci. 2010, 2965, 209-234.

[24] Pawar, Lokesh, Rohit Bajaj, and Geetika Sharma. "Evaluation of Mobile Ad Hoc Network Routing Protocols and Associated Mobility Models." (2017).

[25] Marina MK, Das SR. On-demand multipath distance vector routing in ad hoc networks. InNetwork Protocols, 2001. Ninth International Conference on 2001 Nov 11 (pp. 14-23). IEEE.

[26] Meera SN, Kumar DS, Rao S. Ad Hoc Networks: Route Discovery Channel for Mobile Network with Low Power Consumption. In Proceedings of 2nd International Conference on Micro-Electronics, Electromagnetics and Telecommunications 2018 (pp. 665-671). Springer, Singapore.

[27] Saini A, Nath R. Performance Evaluation of AODV and DSR Routing Protocol on Varying Speed and Pause Time in Mobile Ad Hoc Networks. InNext-Generation Networks 2018 (pp. 313-322). Springer, Singapore.

[28] Yadav AK, Kush A. TCP-and UDP-Based Performance Evaluation of AODV and DSR Routing Protocol on Varying Speed and Pause Time in Mobile Ad Hoc Networks. InNext-Generation Networks 2018 (pp. 323-332). Springer, Singapore.

[29] Jacquet P., Laouiti A.,Minet P. and L. Viennot "Performance Analysis of OLSR Multipoint Relay Flooding in Two Ad Hoc Wireless Network Models." Research Report-4260. INRIA, September 2001. RSRCP journal special issue on Mobility and Internet.

[30] Abd Rahman A. H., Zulkarnain Z. A., Performance Comparison of AODV, DSDV and I-DSDV Routing Protocols in Mobile Ad Hoc Networks, European Journal of Scientific Research Vol. 31, No. 4, pp. 556-576 (June 2009).

[31] Bhat R. Performance Evaluation of OLSR and DSDV Protocols in Military Operations. International Journal. 2017 May 1;8(5).

[32] Gowrishankar S., Basavaraju T.G., Subir Kumar Sarkar. Simulation Based Overhead Analysis of AOMDV, TORA and OLSR in MANET Using Various Energy Models, Proceedings of the World Congress on Engineering and Computer Science 2010 Vol.I , October 2010.

[33] Kuppusamy P., Thirunavukkarasu Dr.K. . A Study and Comparison of OLSR, AODV and TORA Routing Protocols in Ad Hoc Networks, 2010.

[34] Kumar A, Shwe HY, Wong KJ, Chong PH. Location-Based Routing Protocols for Wireless Sensor Networks: A Survey. Wireless Sensor Network. 2017 Jan 17;9(01):25.

[35] BF Spencer. A study on building risk monitoring using wireless sensor network mica mote. In First International Conference on Structural Health Monitoring and Intelligent Infrastructure, Japan, pages 353-363, 2003.

[36] Noel AB, Abdaoui A, Elfouly T, Ahmed MH, Badawy A, Shehata MS. Structural health monitoring using wireless sensor networks: A comprehensive survey. IEEE Communications Surveys & Tutorials. 2017 Jan 1;19(3):1403-23.

[37] Aboelela E, Atta R, Aloufi K. Design and implementation of a wireless sensor network based scalable monitoring system (WiSe-SMS). Journal of Computers. 2018 Mar 1;13(3):244-62.

[38] Song S, Hou Y, Guo M, Wang L, Tong X, Wu J. An investigation on the aggregate-shape embedded piezoelectric sensor for civil infrastructure health monitoring. Construction and Building Materials. 2017 Jan 30;131:57-65.

[39] Hongki Jo, Sung-Han Sim, Kirill A Mechitov, Robin Kim, Jian Li, Parya Moinzadeh, BF Spencer Jr, Jong Woong Park, Soojin Cho, Hyung-Jo Jung, et al. Hybrid wireless smart sensor network for full-scale structural health monitoring of a cable-stayed bridge. In Pro-ceedings of the SPIE Smart Structures/NDE Conference, 2011.

[40] Nicholas Brigman. Structural Health Monitoring in Commercial Aviation. 2012.

[41] Cheung WF, Lin TH, Lin YC. A Real-Time Construction Safety Monitoring System for Hazardous Gas Integrating Wireless Sensor Network and Building Information Modeling Technologies. Sensors. 2018 Feb 2;18(2):436.

[42] Mehta K, Pal R. Energy Efficient Routing Protocols for Wireless Sensor Networks: A Survey. International Journal of Computer Applications. 2017;165(3).

[43] Yelemou, T. (2013, March). New approach to improve the robustness of AOMDV protocol. In Systems, Signals & Devices (SSD), 2013 10thInternational Multi-Conference on (pp. 1-6). IEEE.

[44] Lima, M. N., da Silva, H. W., Dos Santos, A., & Pujolle, G. (2008, April). Survival multipath routing for MANETs. In Network Operations and Management Symposium, 2008. NOMS 2008. IEEE (pp. 425-432). IEEE.

[45] Sharma, B., Chugh, S., & Jain, V. (2014, April). Energy Efficient Load Balancing Approach to Improve AOMDV Routing in MANET. In Communication Systems and Network Technologies (CSNT), 2014 Fourth International Conference on (pp. 187-192). IEEE.

[46] Yang, F., Ling, S., Xu, H., & Sun, B. (2012, March). Network coding based AOMDV routing in MANET. In Information Science and Technology (ICIST), 2012 International Conference on (pp. 337-340). IEEE.

[47] Tekaya, M., Tabbane, N., & Tabbane, S. (2011, September). Delay Remaining Energy for AOMDV Protocol (DRE-AOMDV). In Wireless Communications, Networking and Mobile Computing (WiCOM), 2011 7th International Conference on (pp. 1-5). IEEE.

[48] Chen, X., Jones, H. M., & Jayalath, D. (2011). Channel-aware routing in MANETs with route handoff. Mobile Computing, IEEE Transactions on, 10(1), 108-121.

[49] Daeseung Yoo, Gwangja Jin and Byungtae Jang. A Modified AOMDV Routing Protocol for Maritime Inter-ship Communication, ICTC 2011

[50] Chintan Kanani and Amit Sinhal. Ant Colony Optimization based Modified AOMDV for Multipath Routing in MANET, International Journal of Computer Applications (0975 – 8887) Volume 82 – No.10, November 2013

[51] Al Rantisi MA, Mapp G, Gemikonakli O. A WSN Testbed to Enhance Irrigation Techniques Using a Novel Event-Based Routing Protocol. ICWMC 2017. 2017 Jul 23:49.

[52] A. A. Gorodetsky, M. Cox, N. J. Tremblay, I. Kymissis, and C. Nuckolls. Solar cell from a solution processable pentacene with improved air stability, Chem. Mater., vol. 21, no. 18, pp. 4090–4092, 2009.

[53] Waspmote Technical Guide, Document version: v6.0 - 01/2016, Libelium Comunicaciones Distribuidas S.L.

[54] Rantisi, M.A, Mapp, Glenford E. and Gemikonakli, Orhan (2011). The development of a dynamic and robust event-based routing protocol in wireless sensor networks for environment monitoring. In: First Global Conference on Communication, Science and Information Engineering (CCSIE 2011), 25 - 27 July 2011, Middlesex University.

[55] Peral J., Merlo E., Labrador R. et al., "Automated meter reading based on IEEE 802.15.4," in Proceedings of the 38th Annual Conference on IEEE Industrial Electronics Society (IECON '12), pp.5996–6001,Montreal,Canada,October2012.

[56] Adler ER, Hufford JR, Fazio JJ, inventors; Flume, Inc., assignee. Apparatus and Methods for Remotely Monitoring Water Utilization. United States patent application US 15/283,927. 2017 Apr 20.

[57] Ahmad A, Hanzálek Z. An energy efficient schedule for IEEE 802.15. 4/zigbee cluster tree WSN with multiple collision domains and period crossing constraint. IEEE Transactions on Industrial Informatics. 2018 Jan;14(1):12-23.

[58] Jiang Y, Bai M. Research on Low Power Consumption Technology of Zigbee Network. DEStech Transactions on Computer Science and Engineering. 2017(icitia).

[59] Raza U, Kulkarni P, Sooriyabandara M. Low power wide area networks: An overview. IEEE Communications Surveys & Tutorials. 2017 Jan 1;19(2):855-73.

[60] IRROMETER Company, WATERMARK Soil Moisture Sensor MODEL200SS, Riverside, Calif, USA, 2016.

[61] Vo HM. Online working condition monitoring system integrated power saving and security using zigbee wireless sensor network. InAdvanced Technologies for Communications (ATC), 2017 International Conference on 2017 Oct 18 (pp. 140-143). IEEE.

[62] Correia FP, Alencar MS, Lopes WT, Assis MS, Leal BG. Propagation Analysis for Wireless Sensor Networks Applied to Viticulture. International Journal of Antennas and Propagation. 2017;2017.

[63] Libelium Inc, Weather Stations Comparative—Weather Monitor (Davis) Vs. WS-3000, Libelium Inc, 2016, http://www.libelium .com/.

[64] www.digi.com/products/xbee-rf-solutions/xctu-software/xctu, "Next Generation Configuration Platform for XBee/RF Solution.

[65] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. Wireless sensor network survey. Computer Networks, 52:2292-2330, 2008.

[66] Qiu T, Chen N, Li K, Qiao D, Fu Z. Heterogeneous ad hoc networks: Architectures, advances and challenges. Ad Hoc Networks. 2017 Feb 1;55:143-52.

[67] Dhiman S, Singh O. Design and Implementation of Energy Efficient Routing Protocol for Wireless Sensor Networks. IJASRE. 2017 Jun;3.

[68] Akyildiz I., Tommaso Melodia, and Chowdhury K.. A survey on wireless multimedia sensor networks. Computer Networks, 51:921{960, 2007.

[69] Jin Y., Wang L., Kim Y., and X. Yang. Eemc: An energy-e_cient multi-level clustering algorithm for large-scale wireless sensor networks. Computer Networks, 3(53):542-562, 2008.

[70] Park C. and Chou P. H.. Ambimax: Autonomous energy harvesting platform for multi-supply wireless sensor nodes. In Sensor and Ad Hoc Communications and Networks (SECON'06), pages 168-177, 2006.

[71] Torah R., Glynne-Jones P., Tudor M., and Beeby S. Energy aware wireless micro-system powered by vibration energy harvesting. In PowerMEMS,pages 323-326, Freiburg, Germany, 2007.

[72] Yoon I, Yi JM, Jeong S, Noh DK. Adaptive sensing and compression rate selection scheme for energy-harvesting wireless sensor networks. International Journal of Distributed Sensor Networks. 2017 Jun;13(6):1550147717713627.

[73] Jiang X, Polastre J, Culler D. Perpetual environmentally powered sensor networks. InProceedings of the 4th international symposium on Information processing in sensor networks 2005 Apr 24 (p. 65). IEEE Press.

[74] Lee WK, Schubert MJ, Ooi BY, Ho SJ. Multi-source Energy Harvesting and Storage for Floating Wireless Sensor Network Nodes With Long Range Communication Capability. IEEE Transactions on Industry Applications. 2018 Jan 30.

[75] Bluetooth. http://www.bluetooth.com.

[76] Wu Q, Cao J, Zhou C, Huang J, Li Z, Cheng SM, Cheng J, Pan G. Intelligent Smoke Alarm System with Wireless Sensor Network Using ZigBee. Wireless Communications and Mobile Computing. 2018; 2018.

[77] IEEE Std 802.15-2006. Part15.4: wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (LR-WPANs). IEEE Stud 802.15.4-2006. 2006.

[78] ZigBee Alliance. http://www.zigbee.org.

[79] Kumar NR, Bhuvana C, Anushya S. Comparison of ZigBee and Bluetooth wireless technologies-survey. InInformation Communication and Embedded Systems (ICICES), 2017 International Conference on 2017 Feb 23 (pp. 1-4). IEEE.

[80] Castell N, Dauge FR, Schneider P, Vogt M, Lerner U, Fishbain B, Broday D, Bartonova A. Can commercial low-cost sensor platforms contribute to air quality monitoring and exposure estimates?. Environment international. 2017 Feb 1;99:293-302.

[81] Shu T, Xia M, Chen J, de Silva C. An Energy Efficient Adaptive Sampling Algorithm in a Sensor Network for Automated Water Quality Monitoring. Sensors. 2017 Nov 5;17(11):2551.

[82] Simbeye DS. Industrial Air Pollution Monitoring System Based on Wireless Sensor Networks. Journal of Information Sciences and Computing Technologies. 2017 Nov 22;6(2):612-24.

[83] Vetterli M. Sensorscope: Sensor networks for environmental monitoring. URL http://lcav. epfl. ch/op/edit/sensorscope-en. 2017.

[84] Masudi JK, Ramotsoela TD, Hancke GP. A wireless communication system for environmental monitoring in smart cities. InAFRICON, 2017 IEEE 2017 Sep 18 (pp. 1541-1546). IEEE.

[85] Britton M. , & Sacks L. "The SECOAS Project: development of a self-organizing, wireless sensor network for environmental monitoring." In Second International Workshop on Sensor and Actor Network Protocols and Applications. 2004.

[86] Navarro M., Davis T. W., Liang Y., & Liang X. "A study of long-term WSN deployment for environmental monitoring." IEEE 24th International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), 2013, pp. 2093-2097. IEEE, 2013.

[87] Ali A. R. , Zualkernan I., & Aloul F. "A mobile GPRS-sensors array for air pollution monitoring." IEEE Sensors Journal, Vol. 10, no. 10: 1666-1671. 2010.

[88] Lippi C., Camplani R., Galperti C. , & M. Roveri. "A robust, adaptive, solar-powered WSN framework for aquatic environmental monitoring." IEEE Sensors Journal, Vol. 11, no. 1 (2011): 45-55.

[89] Nguyen T., Thanh T., Nguyen L., & Huynh H. "On the design of energy efficient environment monitoring station and data collection network based on ubiquitous wireless sensor networks." In Computing & Communication Technologies-Research, Innovation, and Vision for the Future (RIVF), 2015 IEEE RIVF International Conference on, pp. 163-168. IEEE, 2015.

[90] Bagola A., Zennaro M., Inggs G., Scott S., & Gascon. "Ubiquitous sensor networking for development (usn4d): An application to pollution monitoring." Sensors 12, no. 1: 391-414. 2012.

[91] Ghobakhlou A., Zandi S., & Sallis. "Development of environmental monitoring system with wireless sensor networks." 19th International Conference on Modelling and Simulation, Perth, Australia, Dec. 2011.

[92] Petite TD, inventor; SIPCO LLC, assignee. Systems and methods for monitoring conditions. United States patent US 9,691,263. 2017 Jun 27.

[93] Reddy M. A., & Natarajan V., "On-line Monitoring of Green House Gases Storage and Leakage using Wireless Sensor Network", International Research Journal of Engineering and Technology (IRJET), Vol. 2. No. 1. 2015.

[94] Baicker J., Mulla A. "Real time monitoring and control system for green house based on 802.15.4 wireless sensor network." Fourth International Conference on In Communication Systems and Network Technologies (CSNT), pp. 98-103. IEEE, 2014.

[95] Santhosh S, Paulose Jacob K "Wireless Sensor Networks for Paddy Field Crop Monitoring Application in Kuttanad" International Journal of Modern Engineering Research (IJMER), Vol.2, Issue.4, July-Aug 2012 pp-2017-2020.

[96] Nam WH, Kim T, Hong EM, Choi JY, Kim JT. A Wireless Sensor Network (WSN) application for irrigation facilities management based on Information and Communication Technologies (ICTs). Computers and Electronics in Agriculture. 2017 Dec 31;143:185-92.

[97] Hamouda YE. Smart Irrigation Decision Support Based on Fuzzy Logic Using Wireless Sensor Network. InPromising Electronic Technologies (ICPET), 2017 International Conference on 2017 Oct 16 (pp. 109-113). IEEE.

[98] Zulkifli CZ, Noor NN. Wireless Sensor Network and Internet of Things (IoT) Solution in Agriculture. Pertanika Journal of Science & Technology. 2017 Jan 1;25(1).

[99] Rathore P, Rao AS, Rajasegarar S, Vanz E, Gubbi J, Palaniswami M. Real-time urban microclimate analysis using internet of things. IEEE Internet of Things Journal. 2018 Apr;5(2):500-11.

[100] Rodriguez SA, Hamann HF, Klein L, Marianno FJ, inventors; International Business Machines Corp, assignee. Targeted irrigation using a central pivot irrigation system with a sensor network. United States patent US 9,943,046. 2018 Apr 17.

[101] Gajendran E, Vignesh J, Prabhu SB. A Novel Drip Irrigation System Using Wireless Sensor Networks. INTERNATIONAL JOURNAL. 2017 Aug;2(1).

[102] GopalaKrishna Moorthy .K, Dr.C.Yaashuwanth, Venkatesh.K "A Wireless Remote Monitoring Of Agriculture Using Zigbee" International Journal of Engineering and Innovative Technology (IJEIT) Volume 2, Issue 8, February 2013

[103] The Network Simulator - NS-2. www.isi.edu/nsnam/ns; Last accessed: May 2008.

[104] Park S., Savvides A., and Srivastava M.B. Sensorsim: A simulation framework for sensor networks. In Proc. Int'l Workshop Modelling, Analysis and Simulation of Wireless and Mobile Systems, pages 104-111, Boston, MA, August 2000.

[105] Varga A. The omnet++ discrete event simulation system. In Proc. European Simulation Multiconference, pages 319-325, Prague, Czech Republic, June 2001.

[106] Varga A. and Hornig R. An overview of the omnet++ simulation environment. 1st International ICST Conference on Simulation Tools and Techniques for Communications, Networks and Systems, pages 1-10, May 2008.

[107] Ould-Ahmed-Vall E.M., Riley G.F., and Heck B.S. Large-scale sensor networks simulation with gtsnets. Simulation, 83:273-290, 2007.

[108] Jinhua G, Weidong X., and Shengquan W.. Reinforce networking theory with opnet simulation. Journal of Information Technology Education, 6:215-226, 2007.

[109] Lattanzi E, Freschi V, Dromedari M, Lorello LS, Peruzzini R, Bogliolo A. A fast and accurate energy source emulator for wireless sensor networks. EURASIP Journal on Embedded Systems. 2017 Jan 1;2016(1):18.

[110] Cao C, Gao Y, Dong W, Bu J. Accurate Per-Link Loss Tomography in Dynamic Sensor Networks. Computer Networks. 2018 Apr 17.

[111] Barré S., Paasch C., Bonaventure O. (2011) MultiPath TCP: From Theory to Practice. In: Domingo-Pascual J., Manzoni P., Palazzo S., Pont A., Scoglio C. (eds) NETWORKING 2011. NETWORKING 2011. Lecture Notes in Computer Science, vol 6640. Springer, Berlin, Heidelberg

Appendix A – New functions in AOMDV-E and AOMDV code modifications

List of modifications by files and functions:-

1- aomdv/aomdv.cc
   - New function AOMDV:: sendRain()
     This function is responsible for sending a broadcast rain packet for all its neighbors and disabling the hello message for this node.

   - New function AOMDV::recvRain()
     This function is responsible for receiving rain packets that are sent from other nodes, after receiving a rain packet this function will delete the path from its routing table.

   - Modifying AOMDV::sendHello() function to disable sending hello messages if the node has rain on it.
   - Modifying AOMDV::command to receive a new event called "Rain" from Tcl .

2- /aomdv/aomdv.h
   - All new function headers are added to this file.
   - New variable pauseHello is added to this file, this variable is used to disable the hello message for this node.

3- aomdv/aomdv_packet.h
   - Define a new constant AOMDVTYPE called AOMDVTYPE_RAIN this is the type of the rain packet.
   - Create a new macro that will create a new rain packet and fill the structure HDR_AOMDV_RAIN
   - Create a new structure hdr_aomdv_rain this is the structure that describe the rain packet.

4- trace/cmu-trace.cc
   - Adding the new packet format AOMDVTYPE_RAIN to CMUTrace::format_aomdv to handle the logging.

1- aomdv/aomdv.cc
   - New function AOMDV:: sendRain()   // STATUS Message
     This function is responsible for sending a broadcast rain packet for all its neighbors and disabling the hello message for this node.

   - New function AOMDV::recvRain()
     This function is responsible for receiving rain packets that are sent from other nodes, after receiving a rain packet this function will delete the path from its routing table.

   - Modifying AOMDV::sendHello() function to disable sending hello messages if the node has rain on it.

- Modifying AOMDV::command to receive a new event called “Rain” from Tcl .

```
void
AOMDV::sendRain()
{
      Packet *p = Packet::alloc();
      struct hdr_cmn *ch = HDR_CMN(p);
        struct hdr_ip *ih = HDR_IP(p);
        struct hdr_aomdv_reply *rh = HDR_AOMDV_REPLY(p);

      printf("sending Rain packet from %d\n",index);

      rh->rp_type = AOMDVTYPE_RAIN;

      rh->rp_hop_count = 0;
        rh->rp_dst = index;
        rh->rp_dst_seqno = seqno;
        rh->rp_lifetime = (1 + ALLOWED_HELLO_LOSS) * HELLO_INTERVAL;

       ch->ptype() = PT_AOMDV;
        ch->size() = IP_HDR_LEN + rh->size();
        ch->iface() = -2;
        ch->error() = 0;
        ch->addr_type() = NS_AF_NONE;
        ch->prev_hop_ = index;          // AODV hack

        ih->saddr() = index;
        ih->daddr() = IP_BROADCAST;
        ih->sport() = RT_PORT;
        ih->dport() = RT_PORT;
        ih->ttl_ = 1;

      pauseHello = 1;
        Scheduler::instance().schedule(target_, p, 0.0);
      printf("Sending rain packet Done\n");
}
void
AOMDV::sendHello() {
      if(pauseHello == 1) return;
      Packet *p = Packet::alloc();
      struct hdr_cmn *ch = HDR_CMN(p);
      struct hdr_ip *ih = HDR_IP(p);
      struct hdr_aomdv_reply *rh = HDR_AOMDV_REPLY(p);

#ifdef DEBUG
      fprintf(stderr, "sending Hello from %d at %.2f\n", index,
Scheduler::instance().clock());
#endif // DEBUG

      rh->rp_type = AOMDVTYPE_HELLO;
      //rh->rp_flags = 0x00;
      // AOMDV code
      rh->rp_hop_count = 0;
      rh->rp_dst = index;
      rh->rp_dst_seqno = seqno;
      rh->rp_lifetime = (1 + ALLOWED_HELLO_LOSS) * HELLO_INTERVAL;
      // ch->uid() = 0;
      ch->ptype() = PT_AOMDV;
```

```
	ch->size() = IP_HDR_LEN + rh->size();
	ch->iface() = -2;
	ch->error() = 0;
	ch->addr_type() = NS_AF_NONE;
	ch->prev_hop_ = index;          // AODV hack

	ih->saddr() = index;
	ih->daddr() = IP_BROADCAST;
	ih->sport() = RT_PORT;
	ih->dport() = RT_PORT;
	ih->ttl_ = 1;

	Scheduler::instance().schedule(target_, p, 0.0);
}
void
AOMDV::recvRain(Packet* p){
	struct hdr_ip *ih = HDR_IP(p);
	struct hdr_aomdv_reply *rp = HDR_AOMDV_REPLY(p);
	printf("receive Rain packet current node id %d source node
%d\n",index,ih->saddr());
	printf("searching for path\n");
	AOMDV_Path* path;
	aomdv_rt_entry *rt;
	rt = rtable.rt_lookup(ih->saddr());

	if(rt == 0)
		printf("Unable to find path\n");

	if(rt && (rt->rt_flags == RTF_UP)){
		printf("path found\n");
		printf("\tremoving path\n");
		path = rt->path_lookup(ih->saddr());
		//path->printPath();
		//rt->path_delete(ih->saddr());
		rt->path_delete(ih->saddr());
		rtable.rt_delete(ih->saddr());
		rt->rt_error = true;
	}
	else
		printf("path not found\n");

	printf("function done\n");
}
```

2- /aomdv/aomdv.h

- All new function headers are added to this file.
- New variable pauseHello is added to this file, this variable is used to disable the hello message for this node.

```
/*
 * Copyright (c) 2008, Marcello Caleffi,
<marcello.caleffi@unina.it>,
 * http://wpage.unina.it/marcello.caleffi
 *
```

```
 * The AOMDV code has been developed at DIET, Department of
Electronic
 * and Telecommunication Engineering, University of Naples "Federico
II"
 *
 *
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License,
 * version 2, as published by the Free Software Foundation.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
along
 * with this program; if not, write to the Free Software Foundation,
Inc.,
 * 59 Temple Place, Suite 330, Boston, MA 02111-1307, USA.
 *
 * The copyright of this module includes the following
 * linking-with-specific-other-licenses addition:
 *
 * In addition, as a special exception, the copyright holders of
 * this module give you permission to combine (via static or

                  * dynamic linking) this module with free software
programs or
 * libraries that are released under the GNU LGPL and with code
 * included in the standard release of ns-2 under the Apache 2.0
 * license or under otherwise-compatible licenses with advertising
 * requirements (or modified versions of such code, with unchanged
                             * license).  You may copy and
distribute such a system following the
 * terms of the GNU GPL for this module and the licenses of the
 * other code concerned, provided that you include the source code
of
 * that other code when and as the GNU GPL requires distribution of
 * source code.
 *
 * Note that people who make modified versions of this module
 * are not obligated to grant this special exception for their
 * modified versions; it is their choice whether to do so.  The GNU
 * General Public License gives permission to release a modified
 * version without this exception; this exception also makes it
 * possible to release a modified version which carries forward this
 * exception.
 *
 */



/*
Copyright (c) 1997, 1998 Carnegie Mellon University.  All Rights
```

```
Reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright
notice,
this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
notice,
this list of conditions and the following disclaimer in the
documentation
and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote
products
derived from this software without specific prior written
permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR
IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES
OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED.
IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT,
INCIDENTAL,
SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS;
OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY,
WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE
OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN
IF
ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The AODV code developed by the CMU/MONARCH group was optimized and
tuned by Samir Das and Mahesh Marina, University of Cincinnati. The
work was partially done in Sun Microsystems.

*/



#ifndef __aomdv_h__
#define __aomdv_h__

//#include <agent.h>
//#include <packet.h>
//#include <sys/types.h>
//#include <cmu/list.h>
//#include <scheduler.h>

#include <cmu-trace.h>
```

```
#include <priqueue.h>
#include <aomdv/aomdv_rtable.h>
#include <aomdv/aomdv_rqueue.h>
// AODV ns-2.31 code
#include <classifier/classifier-port.h>

// AOMDV code
#define AOMDV_PACKET_SALVAGING
#define AOMDV_MAX_SALVAGE_COUNT  10
#define AOMDV_EXPANDING_RING_SEARCH

// AOMDV code
//#define AOMDV_LINK_DISJOINT_PATHS
#define AOMDV_NODE_DISJOINT_PATHS

/*
  Allows local repair of routes
*/
// AOMDV code - could it be uncomment?
#define AOMDV_LOCAL_REPAIR

/*
  Allows AODV to use link-layer (802.11) feedback in determining
when
  links are up/down.
*/
//#define AOMDV_LINK_LAYER_DETECTION

/*
  Causes AODV to apply a "smoothing" function to the link layer
feedback
  that is generated by 802.11.  In essence, it requires that
RT_MAX_ERROR
  errors occurs within a window of RT_MAX_ERROR_TIME before the link
  is considered bad.
*/
#define AOMDV_USE_LL_METRIC

/*
  Only applies if AODV_USE_LL_METRIC is defined.
  Causes AODV to apply omniscient knowledge to the feedback received
  from 802.11.  This may be flawed, because it does not account for
  congestion.
*/
//#define AOMDV_USE_GOD_FEEDBACK


class AOMDV;

#define MY_ROUTE_TIMEOUT        10                      // 100
seconds
#define ACTIVE_ROUTE_TIMEOUT    10          // 50 seconds
#define REV_ROUTE_LIFE          6            // 5  seconds
// AODV ns-2.31 code
#define BCAST_ID_SAVE           6            // 3 seconds
```

```
// No. of times to do network-wide search before timing out for
// MAX_RREQ_TIMEOUT sec.
#define RREQ_RETRIES            3
// timeout after doing network-wide search RREQ_RETRIES times
#define MAX_RREQ_TIMEOUT   1.0 //sec

/* Various constants used for the expanding ring search */
// AOMDV code
#ifdef AOMDV_EXPANDING_RING_SEARCH
#define TTL_START               5          // 5
#define TTL_INCREMENT           2          // 2
#else // NO EXPANDING RING SEARCH
#define TTL_START               30
#define TTL_INCREMENT           30
#endif // NO EXPANDING RING SEARCH
#define TTL_THRESHOLD 7

// Should be set by the user using best guess (conservative)
#define NETWORK_DIAMETER        30             // 30 hops

// Must be larger than the time difference between a node propagates
a route
// request and gets the route reply back.

//#define RREP_WAIT_TIME     (3 * NODE_TRAVERSAL_TIME *
NETWORK_DIAMETER) // ms
//#define RREP_WAIT_TIME     (2 * REV_ROUTE_LIFE)  // seconds
#define RREP_WAIT_TIME         1.0  // sec

#define ID_NOT_FOUND    0x00
#define ID_FOUND        0x01
//#define INFINITY        0xff

// The followings are used for the forward() function. Controls
pacing.
#define AOMDV_DELAY 1.0           // random delay
#define NO_AOMDV_DELAY -1.0       // no delay

// think it should be 30 ms
#define ARP_DELAY 0.01      // fixed delay to keep arp happy


#define HELLO_INTERVAL          60               // 1000 ms
#define ALLOWED_HELLO_LOSS      3               // packets
#define BAD_LINK_LIFETIME       3               // 3000 ms
#define MaxHelloInterval        (1.25 * HELLO_INTERVAL)
#define MinHelloInterval        (0.75 * HELLO_INTERVAL)

// AOMDV code - Could it be removev?
// This should be somewhat related to arp timeout
#define NODE_TRAVERSAL_TIME     0.03             // 30 ms
#define LOCAL_REPAIR_WAIT_TIME  0.15 //sec
```

```
/*
  Timers (Broadcast ID, Hello, Neighbor Cache, Route Cache)
*/
class AOMDVBroadcastTimer : public Handler {
public:
        AOMDVBroadcastTimer(AOMDV* a) : agent(a) {}
        void   handle(Event*);
private:
        AOMDV    *agent;
   Event intr;
};

class AOMDVHelloTimer : public Handler {
public:
        AOMDVHelloTimer(AOMDV* a) : agent(a) {}
        void   handle(Event*);
private:
        AOMDV    *agent;
   Event intr;
};

class AOMDVNeighborTimer : public Handler {
public:
        AOMDVNeighborTimer(AOMDV* a) : agent(a) {}
        void   handle(Event*);
private:
        AOMDV    *agent;
   Event intr;
};

class AOMDVRouteCacheTimer : public Handler {
public:
        AOMDVRouteCacheTimer(AOMDV* a) : agent(a) {}
        void   handle(Event*);
private:
        AOMDV    *agent;
   Event intr;
};

class AOMDVLocalRepairTimer : public Handler {
public:
        AOMDVLocalRepairTimer(AOMDV* a) : agent(a) {}
        void   handle(Event*);
private:
        AOMDV    *agent;
   Event intr;
};

// AOMDV code
/*
  Route List
*/
class AOMDV_Route {
        friend class AOMDVBroadcastID;
 public:
        AOMDV_Route(nsaddr_t nexthop, nsaddr_t lasthop=0) {
```

```
                    nh_addr = nexthop;
                    lh_addr = lasthop;
            }
      protected:
      LIST_ENTRY(AOMDV_Route) route_link;
        nsaddr_t        nh_addr;
        nsaddr_t        lh_addr;
};

LIST_HEAD(aomdv_routes, AOMDV_Route);


/*
  Broadcast ID Cache
*/
class AOMDVBroadcastID {
        friend class AOMDV;
 public:
        AOMDVBroadcastID(nsaddr_t i, u_int32_t b) {
                    src = i;
                    id = b;
                    // AOMDV code
                    count=0;
                    LIST_INIT(&reverse_path_list);
                    LIST_INIT(&forward_path_list);
        }
protected:
        LIST_ENTRY(AOMDVBroadcastID) link;
        nsaddr_t        src;
        u_int32_t       id;
        double          expire;         // now + BCAST_ID_SAVE s

                // AOMDV code
        int                         count;
        aomdv_routes     reverse_path_list;     // List of reverse
paths used for forwarding replies
        aomdv_routes     forward_path_list;     // List of forward
paths advertised already

        inline AOMDV_Route* reverse_path_insert(nsaddr_t nexthop,
nsaddr_t lasthop=0) {
                  AOMDV_Route* route = new AOMDV_Route(nexthop,
lasthop);
                  assert(route);
                  LIST_INSERT_HEAD(&reverse_path_list, route,
route_link);
                  return route;
        }

        inline AOMDV_Route* reverse_path_lookup(nsaddr_t nexthop,
nsaddr_t lasthop=0) {
                  AOMDV_Route *route = reverse_path_list.lh_first;

                  // Search the list for a match of id
                  for( ; route; route = route->route_link.le_next) {
```

```
                        if ( (route->nh_addr == nexthop) &&
(route->lh_addr == lasthop) )
                            return route;
                        }
                    return NULL;
                    }

                inline AOMDV_Route* forward_path_insert(nsaddr_t
nexthop, nsaddr_t lasthop=0) {
                    AOMDV_Route* route = new AOMDV_Route(nexthop,
lasthop);
                    assert(route);
                    LIST_INSERT_HEAD(&forward_path_list, route,
route_link);
                    return route;
                }

                inline AOMDV_Route* forward_path_lookup(nsaddr_t
nexthop, nsaddr_t lasthop=0) {
                    AOMDV_Route *route =
forward_path_list.lh_first;

                    // Search the list for a match of id
                    for( ; route; route = route-
>route_link.le_next) {
                        if ( (route->nh_addr == nexthop) &&
     (route->lh_addr == lasthop) )
                            return route;
                    }
                    return NULL;
                }
};

LIST_HEAD(aomdv_bcache, AOMDVBroadcastID);


/*
  The Routing Agent
*/
class AOMDV: public Agent {

  /*
   * make some friends first
   */

        friend class aomdv_rt_entry;
        friend class AOMDVBroadcastTimer;
        friend class AOMDVHelloTimer;
        friend class AOMDVNeighborTimer;
        friend class AOMDVRouteCacheTimer;
        friend class AOMDVLocalRepairTimer;

 public:
        AOMDV(nsaddr_t id);

        void      recv(Packet *p, Handler *);
```

```
 protected:
        int             command(int, const char *const *);
        int             initialized() { return 1 && target_; }

        /*
         * Route Table Management
         */
        void            rt_resolve(Packet *p);
        void            rt_down(aomdv_rt_entry *rt);
        void            local_rt_repair(aomdv_rt_entry *rt, Packet
*p);
     int        pauseHello;
 public:
        void            rt_ll_failed(Packet *p);
                // AOMDV code
                // void            rt_update(aodv_rt_entry *rt,
u_int32_t seqnum, u_int16_t metric, nsaddr_t nexthop, double
expire_time);
                // void            handle_link_failure(nsaddr_t id);
        void            handle_link_failure(nsaddr_t id);
 protected:
        void            rt_purge(void);

        void            enque(aomdv_rt_entry *rt, Packet *p);
        Packet*         deque(aomdv_rt_entry *rt);

        /*
         * Neighbor Management
         */
        void            nb_insert(nsaddr_t id);
        AOMDV_Neighbor*       nb_lookup(nsaddr_t id);
        void            nb_delete(nsaddr_t id);
        void            nb_purge(void);

        /*
         * Broadcast ID Management
         */

                // AODV ns-2.31 code
                void            id_insert(nsaddr_t id, u_int32_t
bid);
                bool         id_lookup(nsaddr_t id, u_int32_t bid);
         AOMDVBroadcastID*    id_get(nsaddr_t id, u_int32_t bid);
        void            id_purge(void);

        /*
         * Packet TX Routines
         */
        void            forward(aomdv_rt_entry *rt, Packet *p,
double delay);
                // AOMDV code - should it be removed?
        void            forwardReply(aomdv_rt_entry *rt, Packet *p,
double delay);
        void            sendHello(void);
     void       sendRain(void);
```

```
        void            sendRequest(nsaddr_t dst);

                // AOMDV code
                // void            sendReply(nsaddr_t ipdst,
u_int32_t hop_count, nsaddr_t rpdst, u_int32_t rpseq, u_int32_t
lifetime, double timestamp);
        void            sendReply(nsaddr_t ipdst, u_int32_t
hop_count,
                                  nsaddr_t rpdst, u_int32_t rpseq,
                                  double lifetime, double timestamp,
             nsaddr_t nexthop, u_int32_t bcast_id, nsaddr_t
rp_first_hop);
        void            sendError(Packet *p, bool jitter = true);

        /*
         * Packet RX Routines
         */
        void            recvAOMDV(Packet *p);
        void            recvHello(Packet *p);
      void       recvRain(Packet *p);
        void            recvRequest(Packet *p);
        void            recvReply(Packet *p);
        void            recvError(Packet *p);

   /*
    * History management
    */

   double      PerHopTime(aomdv_rt_entry *rt);


        nsaddr_t        index;                  // IP Address of
this node
        u_int32_t       seqno;                  // Sequence Number
        int             bid;                    // Broadcast ID

        aomdv_rtable         rthead;                 // routing
table
        aomdv_ncache         nbhead;                 // Neighbor
Cache
        aomdv_bcache          bihead;                 // Broadcast
ID Cache

        /*
         * Timers
         */
        AOMDVBroadcastTimer  btimer;
        AOMDVHelloTimer      htimer;
        AOMDVNeighborTimer   ntimer;
        AOMDVRouteCacheTimer rtimer;
        AOMDVLocalRepairTimer lrtimer;

        /*
         * Routing Table
         */
        aomdv_rtable          rtable;
```

```
        /*
         *  A "drop-front" queue used by the routing layer to buffer
         *  packets to which it does not have a route.
         */
        aomdv_rqueue         rqueue;

        /*
         * A mechanism for logging the contents of the routing
         * table.
         */
        Trace           *logtarget;

        /*
         * A pointer to the network interface queue that sits
         * between the "classifier" and the "link layer".
         */
        PriQueue        *AOMDVifqueue;

        /*
         * Logging stuff
         */
        void            log_link_del(nsaddr_t dst);
        void            log_link_broke(Packet *p);
        void            log_link_kept(nsaddr_t dst);

                // AOMDV code
                int aomdv_max_paths_;
                int aomdv_prim_alt_path_len_diff_;
                // AODV ns-2.31 code
                /*    for passing packets up to agents */
                PortClassifier *dmux_;
};

#endif /* __aomdv_h__ */
```

3- aomdv/aomdv_packet.h

- Define a new constant AOMDVTYPE called AOMDVTYPE_RAIN this is the type of the rain packet.
- Create a new macro that will create a new rain packet and fill the structure HDR_AOMDV_RAIN
- Create a new structure hdr_aomdv_rain this is the structure that describe the rain packet.

```
/*
 * Copyright (c) 2008, Marcello Caleffi,
<marcello.caleffi@unina.it>,
 * http://wpage.unina.it/marcello.caleffi
```

```
 *
 * The AOMDV code has been developed at DIET, Department of
Electronic
 * and Telecommunication Engineering, University of Naples "Federico
II"
 *
 *
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License,
 * version 2, as published by the Free Software Foundation.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
along
 * with this program; if not, write to the Free Software Foundation,
Inc.,
 * 59 Temple Place, Suite 330, Boston, MA 02111-1307, USA.

 *
 * The copyright of this module includes the following
 * linking-with-specific-other-licenses addition:
 *
 * In addition, as a special exception, the copyright holders of
 * this module give you permission to combine (via static or

                  * dynamic linking) this module with free software
programs or
 * libraries that are released under the GNU LGPL and with code
 * included in the standard release of ns-2 under the Apache 2.0
 * license or under otherwise-compatible licenses with advertising
 * requirements (or modified versions of such code, with unchanged
                             * license).  You may copy and
distribute such a system following the
 * terms of the GNU GPL for this module and the licenses of the
 * other code concerned, provided that you include the source code
of
 * that other code when and as the GNU GPL requires distribution of
 * source code.
 *
 * Note that people who make modified versions of this module
 * are not obligated to grant this special exception for their
 * modified versions; it is their choice whether to do so.  The GNU
 * General Public License gives permission to release a modified
 * version without this exception; this exception also makes it
 * possible to release a modified version which carries forward this
 * exception.
 *
 */
```

```
/*
Copyright (c) 1997, 1998 Carnegie Mellon University.  All Rights
Reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright
notice,
this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
notice,
this list of conditions and the following disclaimer in the
documentation
and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote
products
derived from this software without specific prior written
permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR
IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES
OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED.
IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT,
INCIDENTAL,
SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS;
OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY,
WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE
OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN
IF
ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The AODV code developed by the CMU/MONARCH group was optimized and
tuned by Samir Das and Mahesh Marina, University of Cincinnati. The
work was partially done in Sun Microsystems.
*/



#ifndef __aomdv_packet_h__
#define __aomdv_packet_h__

//#include <config.h>
//#include "aomdv.h"
#define AOMDV_MAX_ERRORS 100
```

```
/*
==============================================================================
=
   Packet Formats...

==============================================================================
= */
#define AOMDVTYPE_HELLO     0x01
#define AOMDVTYPE_RREQ      0x02
#define AOMDVTYPE_RREP      0x04
#define AOMDVTYPE_RERR      0x08
#define AOMDVTYPE_RREP_ACK       0x10
#define AOMDVTYPE_RAIN           0x20
/*
 * AOMDV Routing Protocol Header Macros
 */
#define HDR_AOMDV(p)        ((struct hdr_aomdv*)hdr_aomdv::access(p))
#define HDR_AOMDV_REQUEST(p)     ((struct
hdr_aomdv_request*)hdr_aomdv::access(p))
#define HDR_AOMDV_REPLY(p)  ((struct
hdr_aomdv_reply*)hdr_aomdv::access(p))
#define HDR_AOMDV_ERROR(p)  ((struct
hdr_aomdv_error*)hdr_aomdv::access(p))
#define HDR_AOMDV_RREP_ACK(p)    ((struct
hdr_aomdv_rrep_ack*)hdr_aomdv::access(p))
#define HDR_AOMDV_RAIN           ((struct
hdr_aomdv_rain*)hdr_aomdv::access(p))
/*
 * General AOMDV Header - shared by all formats
 */
struct hdr_aomdv {
        u_int8_t        ah_type;
      /*
        u_int8_t        ah_reserved[2];
        u_int8_t        ah_hopcount;
      */
           // Header access methods
      static int offset_; // required by PacketHeaderManager
      inline static int& offset() { return offset_; }
      inline static hdr_aomdv* access(const Packet* p) {
           return (hdr_aomdv*) p->access(offset_);
      }
};

struct hdr_aomdv_request {
        u_int8_t        rq_type; // Packet Type
        u_int8_t        reserved[2];
        u_int8_t        rq_hop_count;   // Hop Count
        u_int32_t       rq_bcast_id;    // Broadcast ID

        nsaddr_t        rq_dst;         // Destination IP Address
        u_int32_t       rq_dst_seqno;   // Destination Sequence
Number
        nsaddr_t        rq_src;         // Source IP Address
        u_int32_t       rq_src_seqno;   // Source Sequence Number
```

```
        double          rq_timestamp;   // when REQUEST sent;
                            // used to compute route discovery
latency
// AOMDV code
        nsaddr_t        rq_first_hop;  // First Hop taken by the
RREQ

  // This define turns on gratuitous replies- see aodv.cc for
implementation contributed by
  // Anant Utgikar, 09/16/02.
  //#define RREQ_GRAT_RREP 0x80

  inline int size() {
  int sz = 0;
  /*
      sz = sizeof(u_int8_t)       // rq_type
           + 2*sizeof(u_int8_t)   // reserved
           + sizeof(u_int8_t)          // rq_hop_count
           + sizeof(double)       // rq_timestamp
           + sizeof(u_int32_t)    // rq_bcast_id
           + sizeof(nsaddr_t)          // rq_dst
           + sizeof(u_int32_t)    // rq_dst_seqno
           + sizeof(nsaddr_t)          // rq_src
           + sizeof(u_int32_t);   // rq_src_seqno
  */
      sz = 7*sizeof(u_int32_t);
// AOMDV code
   sz += sizeof(nsaddr_t);    // rq_first_hop
      assert (sz >= 0);
      return sz;
  }
};

struct hdr_aomdv_reply {
        u_int8_t        rp_type;        // Packet Type
        u_int8_t        reserved[2];
        u_int8_t        rp_hop_count;           // Hop Count
        nsaddr_t        rp_dst;                 // Destination IP
Address
        u_int32_t       rp_dst_seqno;           // Destination
Sequence Number
        nsaddr_t        rp_src;                 // Source IP Address
        double           rp_lifetime;            // Lifetime

        double          rp_timestamp;           // when
corresponding REQ sent;
                                 // used to compute route discovery
latency
// AOMDV code
        u_int32_t       rp_bcast_id;           // Broadcast ID of
the corresponding RREQ
        nsaddr_t        rp_first_hop;

  inline int size() {
  int sz = 0;
  /*
```

```
      sz = sizeof(u_int8_t)       // rp_type
           + 2*sizeof(u_int8_t)   // rp_flags + reserved
           + sizeof(u_int8_t)          // rp_hop_count
           + sizeof(double)       // rp_timestamp
           + sizeof(nsaddr_t)          // rp_dst
           + sizeof(u_int32_t)    // rp_dst_seqno
           + sizeof(nsaddr_t)          // rp_src
           + sizeof(u_int32_t);   // rp_lifetime
  */
      sz = 6*sizeof(u_int32_t);
// AOMDV code
   if (rp_type == AOMDVTYPE_RREP) {
      sz += sizeof(u_int32_t);   // rp_bcast_id
      sz += sizeof(nsaddr_t);    // rp_first_hop
   }
      assert (sz >= 0);
      return sz;
  }

};

struct hdr_aomdv_error {
        u_int8_t        re_type;                // Type
        u_int8_t        reserved[2];            // Reserved
        u_int8_t        DestCount;                 // DestCount
        // List of Unreachable destination IP addresses and sequence
numbers
        nsaddr_t        unreachable_dst[AOMDV_MAX_ERRORS];
        u_int32_t       unreachable_dst_seqno[AOMDV_MAX_ERRORS];

      u_int8_t   error_source;
  inline int size() {
  int sz = 0;
  /*
      sz = sizeof(u_int8_t)       // type
           + 2*sizeof(u_int8_t)   // reserved
           + sizeof(u_int8_t)          // length
           + length*sizeof(nsaddr_t); // unreachable destinations
  */
      sz = (DestCount*2 + 1)*sizeof(u_int32_t);
      sz += sizeof(u_int8_t);
      assert(sz);
        return sz;
  }

};

struct hdr_aomdv_rain {
        u_int8_t        re_type;                // Type
        u_int8_t        reserved[2];            // Reserved

      u_int8_t   status;
  inline int size() {
  int sz = 0;
  /*
        sz = sizeof(u_int8_t)           // type
```

```
              + 2*sizeof(u_int8_t)       // reserved
              + sizeof(u_int8_t)         // length
              + length*sizeof(nsaddr_t); // unreachable destinations
  */
        sz = sizeof(u_int32_t);
        assert(sz);
        return sz;
  }
};

struct hdr_aomdv_rrep_ack {
     u_int8_t   rpack_type;
     u_int8_t   reserved;
};

// for size calculation of header-space reservation
union hdr_all_aomdv {
  hdr_aomdv          ah;
  hdr_aomdv_request  rreq;
  hdr_aomdv_reply    rrep;
  hdr_aomdv_error    rerr;
  hdr_aomdv_rrep_ack rrep_ack;
};

#endif /* __aomdv_packet_h__ */
```