A Novel Patient Monitoring Framework and Routing Protocols for Energy & QoS Aware Communication in Body Area Networks

by

Zahoor Ali Khan

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

at

Dalhousie University
Halifax, Nova Scotia
June 2013

DALHOUSIE UNIVERSITY

DEPARTMENT OF ENGINEERING MATHEMATICS AND INTERNETWORKING

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled “A Novel Patient Monitoring Framework and Routing Protocols for Energy & QoS Aware Communication in Body Area Networks” by Zahoor Ali Khan in partial fulfilment of the requirements for the degree of Doctor of Philosophy.

Dated: June 20, 2013

External Examiner: ___________________________

Research Co-Supervisors: ___________________________

____

Examining Committee: ___________________________

____

Departmental Representative: ___________________________________

DALHOUSIE UNIVERSITY

DATE: June 20, 2012

AUTHOR: Zahoor Ali Khan

TITLE: A Novel Patient Monitoring Framework and Routing Protocols for Energy & QoS Aware Communication in Body Area Networks

DEPARTMENT OR SCHOOL: Department of Engineering Mathematics and Internetworking

DEGREE: PhD CONVOCATION: October YEAR: 2013

Permission is herewith granted to Dalhousie University to circulate and to have copied for non-commercial purposes, at its discretion, the above title upon the request of individuals or institutions. I understand that my thesis will be electronically available to the public.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

The author attests that permission has been obtained for the use of any copyrighted material appearing in the thesis (other than the brief excerpts requiring only proper acknowledgement in scholarly writing), and that all such use is clearly acknowledged.

Signature of Author

DEDICATIONS

I dedicate this dissertation to the memory of my loving son Aaqib Zahoor Khan who passed away while I was working on this thesis.

This thesis is also dedicated to my beloved parents, my wife Sumera, and my loving children (Aasim, Aatif, and Areebah).

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Significant challenges to patient monitoring systems in a hospital environment include the reliable and energy-efficient transmission of data and their real-time display. This thesis proposes innovative and novel mechanisms for the reliable transmission of patient data in Body Area Network (BAN) communication, which simultaneously ensure high throughput, low data latency, and low energy consumption by implementing energy and QoS aware routing protocols. Five main contributions are made in this regard. Firstly, a novel patient monitoring system (ZK-BAN peering framework) is proposed for real-time hospital BAN communication that displays patient data on the display units by considering data privacy, low energy consumption, better control on the devices, and patient mobility. Secondly, a novel energy-aware peering routing protocol (EPR) is introduced in which the choice of next hop is based on the residual energy and geographic information of the neighbor nodes. EPR contains three main components: a Hello protocol, a neighbor table constructor algorithm, and a routing table constructor algorithm. Thirdly, a new modular QoS-aware routing protocol (QPRD) is designed to handle the ordinary and delay-sensitive data for BAN communication in hospitals. QPRD provides an end-to-end path delay mechanism to calculate the path delays of all possible paths from a source to destination and then chooses the best path with the lowest path delay for delay-sensitive packets. Fourthly, a novel modular QoS-aware routing protocol (QPRR) is developed to handle ordinary and reliability-sensitive data for BAN communication in hospitals. The modular architecture of QPRR includes five modules: a reliability module, a packet classifier, a Hello protocol module, a routing services module, and a QoS-aware queuing module. The proposed mechanisms for end-to-end path reliability calculation and data transmission using redundant paths ensure more reliable BAN communication. Finally, a new integrated energy and QoS aware routing protocol (ZEQoS) is designed to deal with ordinary, delay-sensitive, and reliability-sensitive data packets. Extensive simulations in the OMNeT++ based Castalia 3.2 simulator show that EPR, QPRD, QPRR, and ZEQoS perform better than other similar energy and QoS aware routing protocols.

LIST OF ABBREVIATIONS AND SYMBOLS USED

ρ_d	Average weighting factor for node delay calculation
ρ_r	Average weighting factor for link reliability calculation
$\bar{X}_i$	Average probability of successful transmission
σ	Electrical conductivity of tissue
ρ	Tissue density
$\|E\|$	Total number of edges
$\|V\|$	Order of the graph

A	Adjacency matrix
ADC	Analog to Digital Converter
AES	Advanced Encryption Standard
AID-N	Advanced Health and Disaster Aid Network
a_{ij}	Wireless link between the two nodes *i* and *j*
ALTR	Adaptive Least Temperature Routing
AODV	Ad-hoc On Demand distance Vector
API	Application Programming Interface
A^t	Transpose of adjacency matrix
AZR-LEACH	Advanced Zonal Rectangular LEACH

BAN	Body Area Network
BANC	Body Area Network Coordinator
BASN	Body Area Sensor Network
BER	Bit Error Rate
BP	Blood Pressure
BS	Base Station
BSD	Berkeley Software Distribution

CCR	Corner-Cube Retro-reflector
CH	Cluster-Head
CICADA	Cascading Information Retrieval by Controlling Access with Distributed
C_j	Communication cost
CPs	Critical Packets
CPU	Central Processing Unit
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance

$D_{(i,j)}$	Distance between node *i* to neighbor node *j*
$D_{(j,Dst)}$	Distance between neighbor node *j* and destination Dst
DAC	Digital to Analog Converter
DARPA	Defense Advance Research Project Agency
$DL_{channel}$	Delay due to capturing the channel
DL_{MAC_queue}	MAC layer queuing delay
$DL_{node(i)}$	Time delay within the node *i*
$DL_{path(i,Dst)}$	Path delay from node *i* to destination *Dst*
$DL_{queue+channel}$	Delay due to the MAC & network layers Queues and capturing the channel
DL_{req}	Required path delay for delay-sensitive packets
DL_{trans}	Transmission time
DM	Delay Module
DMQoS	Data centric Multiobjective QoS-aware routing protocol
DSNs	Distributed Sensor Networks
DSPs	Delay-Sensitive Packets
DSSS	Direct Sequence Spread Spectrum
Dst	Destination
Dst	Destination node (i.e. NSC, MDCs, BAN)

E	Edges of Graph G
ECG	Electrocardiography

EEG	Electroencephalography
E_j	Residual energy of node j
EPR	Energy-aware Peering Routing protocol
ER	Emergency Room
EWMA	Exponentially Weighted Moving Average

f_c	Carrier Frequency
FCC	Federal Communications Commission
FDA	Food and Drug Authority
FFD	Full Function Device
FFSN	Full Function Sensor Node
FFSN	Full Function Sensor Node
FHSS	Frequency Hopping Spread Spectrum
FIFO	First-In-First-Out

GAF	Geographic Adaptive Fidelity
GBR	Gradient-Based Routing
GEAR	Geographic and Energy Aware Routing
GPS	Global Positioning System
GW	Gateway

HIT	Hybrid Indirect Transmissions
HPM	Hello Protocol Module
HPR	Hotspot Preventing Routing Algorithm for Delay-Sensitive Biomedical Sensor Networks
HR-WPANs	High Rate Wireless Personal Area Networks

i	Source node
ICU	Intensive Care Unit
ID	Identification
ID_{Dst}	Destination ID

ID_j	Neighbor node j ID
IEEE	Institute of Electrical and Electronics Engineers
IEEE-SA	The IEEE Standards Association
IG-BAN	BAN Interest Group
IP	Internet Protocol
ISM	Industrial Scientific and Medical
IWK	Izaak Walton Killam

j	Neighbor node of source node i

KB	Kilo Byte
Kbps	Kilobits per second

L2CAP	Logical link Control and Adaptation Protocol
LAN	Local Area Network
L_{Dst}	Destination Location
LEACH	Low Energy Adaptive Clustering Hierarchy
L_j	Neighbor node j location
LMP	Link Manager Protocol
LO	Lexicographic Optimization-based
LOCALMOR	LOCALized Multi-Objective Routing
LR-WPANs	Low Rate Wireless Personal Area Networks
LTR	Least Temperature Routing

MAC	Media Access Control
MAN	Metropolitan Area Network
MANETs	Mobile Ad-hoc Networks
Mbps	Mega bits per second
MCFA	Minimum Cost Forwarding Algorithm
MDC	Medical Display Coordinator
MECN	Minimum Energy Communication Network

MHz	Mega Hertz
mmWave	millimeter Wave
MSP	Mixed-Signal Processor
MWE	Multiple Winner Algorithm
N_{Acks}	Total number of acknowledgements received
N_{bit}	total number of bits in each packet
$NH_{(i,Dst)}$	Next Hop between node *i* and destination *Dst*
NH_{D}	Next Hop for delay-sensitive packets
NH_{E}	Energy-aware Next Hop
NH_{R1}	1st reliable Next Hop
NH_{R2}	2nd reliable Next Hop
NH_{R3}	3rd reliable Next Hop
NS	Nova Scotia
NSC	Nursing Station Coordinator
N_{Trans}	Total number of packets transmitted
OPs	Ordinary Packets
OR	Operation Room
OSI	Open Systems Interconnection
P2P	Peer-to-Peer
PAN	Personal Area Network
PC	Packet Classifier
PDA	Personal Digital Assistants
P_{error}	Error probability
PHY	Physical layer
p-mp	Point-to-multipoint
PMU	Pediatric Medical Unit
PMU	Pediatric Medical Unit
p-p	Point-to-point

PR	Patient Room
QoS	Quality of Service
QPRD	QoS-aware Peering Routing protocol for Delay-sensitive data
QPRR	QoS-aware Peering Routing protocol for Reliability-sensitive data
QQM	QoS-aware Queuing Module
RAIN	Routing Algorithm for network of homogeneous and Idless biomedical sensor Nodes
RAM	Random Access Memory
rb	Impedances of the receivers
R_{bit}	Data rate
RF	Radio Frequency
RFD	Reduced Function Device
RFSN	Reduced Function Sensor Node
RFSN	Reduced Function Sensor Node
$R_{link(i,j)}$	Link reliability from node *i* to neighbor node *j*
RL-QRP	Reinforcement Learning based routing protocol with QoS support
RM	Reliability Module
$R_{option1(i,Dst)}$	1st option reliability for sending reliability-sensitive packets
$R_{option2(i,Dst)}$	2nd option reliability for sending reliability-sensitive packets
$R_{option3(i,Dst)}$	3rd option reliability for sending reliability-sensitive packets
$R_{path(i,Dst)}$	Path reliability from node *i* to destination *Dst*
$R_{path(j,Dst)}$	Path reliability between neighbor j and destination
$R_{path1(i,Dst)}$	1st Path reliability from node *i* to destination *Dst*
$R_{path2(i,Dst)}$	2nd Path reliability from node *i* to destination *Dst*
$R_{path3(i,Dst)}$	3rd Path reliability from node *i* to destination *Dst*
R_{req}	Required reliability of reliability-sensitive packets
RSM	Routing Services Module
RSPs	Reliability-Sensitive Packets
RSSI	Received Signal Strength Indication

SAR	Specific Absorption Rate
SMART	Scalable Medical Alert and Response Technology
SOSUS	Sound Surveillance System
SPIN	Sensor Protocols for Information via Negotiation
SWE	Single Winner Algorithm
TARA	Thermal Aware Routing Algorithm
tb	Impedances of the transmitter
TBF	Trajectory-Based Forwarding
TDMA	Time Division Multiple Access
TI	Texas Instrument
TICOSS	TImezone COordinated Sleep Scheduling
T_j	Device type of node *j*
TSHR	Thermal-aware Shortest Hop Routing algorithm for in vivo biomedical sensor networks
UCLA	University of California Los Angeles
UWB	Ultra-Wideband
V	Vertices of Graph G
WANET	Wireless Ad-hoc Network
WASP	Wireless Autonomous Spanning tree Protocol for multihop wireless body area networks
WBANs	Wireless Body Area Networks
WBSNs	Wireless Body Sensor Networks
WG	Working Group
WINS	Wireless Integrated Network Sensors
WLAN	Wireless Local Area Network
WMAN	Wireless Metropolitan Area Network
WMNs	Wireless Mesh Networks

WPANs	Wireless Personal Area Networks
WSNs	Wireless Sensor Networks
WSSN	Wireless Self-Sustaining Sensor Network
WWAN	Wireless Wide Area Network
X_{Dst}	X coordinate of destination node *Dst*
X_i	X coordinate of source node *i*
X_j	X coordinate of neighbor node *j*
Y_{Dst}	Y coordinate of destination node *Dst*
Y_i	Y coordinate of source node *i*
Y_j	Y coordinate of neighbor node *j*
ZDO	ZigBee Device Object
ZEQoS	Zahoor Energy and QoS aware routing protocol
ZK-BAN	Zahoor Khan Body Area Network

ACKNOWLEDGEMENTS

First and foremost I would like to express my sincere gratitude to Dr. Bill Robertson, Dr. William Phillips, and Dr. Shyamala Sivakumar for their endless guidance and patience. Their excellent feedback made this thesis complete and interesting. Special thanks to Dr. Sivakumar for her valuable ideas and suggestions. Without the motivation, help, and support of Dr. Robertson, it is doubtful I would have ever finished.

It is my pleasure to thank the examining committee members, Dr. Srinivas Sampalli and Dr. Serguei Iakovlev.

I wish to express my thanks to the Internetworking program for giving me the opportunity to gain precious professional experience. I would like to acknowledge the support of its staff, especially the program manager Mrs. Shelley Caines.

Thanks to the faculty and staff of the Department of Engineering Mathematics and Internetworking for their academic assistance. Mrs. Shelley Caines and Mrs. Claire Chisholm had all the answers to any non-thesis related worries.

I wish to express my deepest gratitude to my parents, brothers, sisters, and friends for their prayers and support.

Finally, I would like to express my special thanks to my wife for her great patience and invaluable support.

Above all, I thank almighty God for giving me the strength and resources to complete this dissertation successfully.

CHAPTER 1

INTRODUCTION

Body Area Networks (BANs) are small wireless networks consisting of sensors placed inside or outside the human body. The body implant or wearable sensors transmit their data to a central device called a Body Area Network Coordinator (BANC). Typically, the BANC is a computationally more powerful device than the body sensors. The BANC is responsible for reliably transferring the sensors' data to the next node or destination.

Important issues in BAN data transmission include ensuring high reliability of data delivery at the destination, low data latency, low energy consumption, and enabling patient mobility. The goal of this research is to propose innovative and novel mechanisms for the real-time display of patient's data while simultaneously minimizing the overall BAN energy consumption by implementing energy and QoS aware routing protocols for reliable transmission of delay-sensitive BAN data. The proposed patient monitoring system and the associated routing protocols aim to provide higher data throughput, increased reliability, lower network traffic and packets forwarded by intermediate nodes, fewer dropped packets due to buffer overflow, and overall lower energy consumption for low, medium, and high offered traffic loads when compared to similar protocols. The remaining part of this chapter is organized as follows: Section 1.1 discusses the motivation and background for BAN; Section 1.2 describes the contributions; Section 1.3 presents the thesis outline.

1.1. MOTIVATION AND BACKGROUND

Wireless Sensor Networks (WSNs) help to monitor the physical or environmental conditions in different areas including industry, commercial sector, or military fields. The monitoring of physiological and biochemical parameters for different diseases in the human body is one of the major challenges in the healthcare field. To overcome this challenge, Body Area Network (BAN) is suggested as a new sub-field of WSN. Other terms commonly used for BANs include Wireless Body Sensor Networks (WBSNs), Wireless Body Area Networks (WBANs) or Body Area Sensor Networks (BASNs).

Implantable and wearable sensors are used inside and outside the body to monitor both the physiological and biochemical parameters of patients. These wearable and implanted sensors collect the data from the body and transfer these data to the healthcare center which helps the healthcare professionals to monitor the patient's vital signs.

BAN has many applications related to a person's health and wellness. Figure 1-1 illustrates some of the BAN applications in different areas. BAN can be used in medical, emergency services, consumer electronics, health fitness and lifestyle monitoring, defence, entertainment, and personal health applications [1].

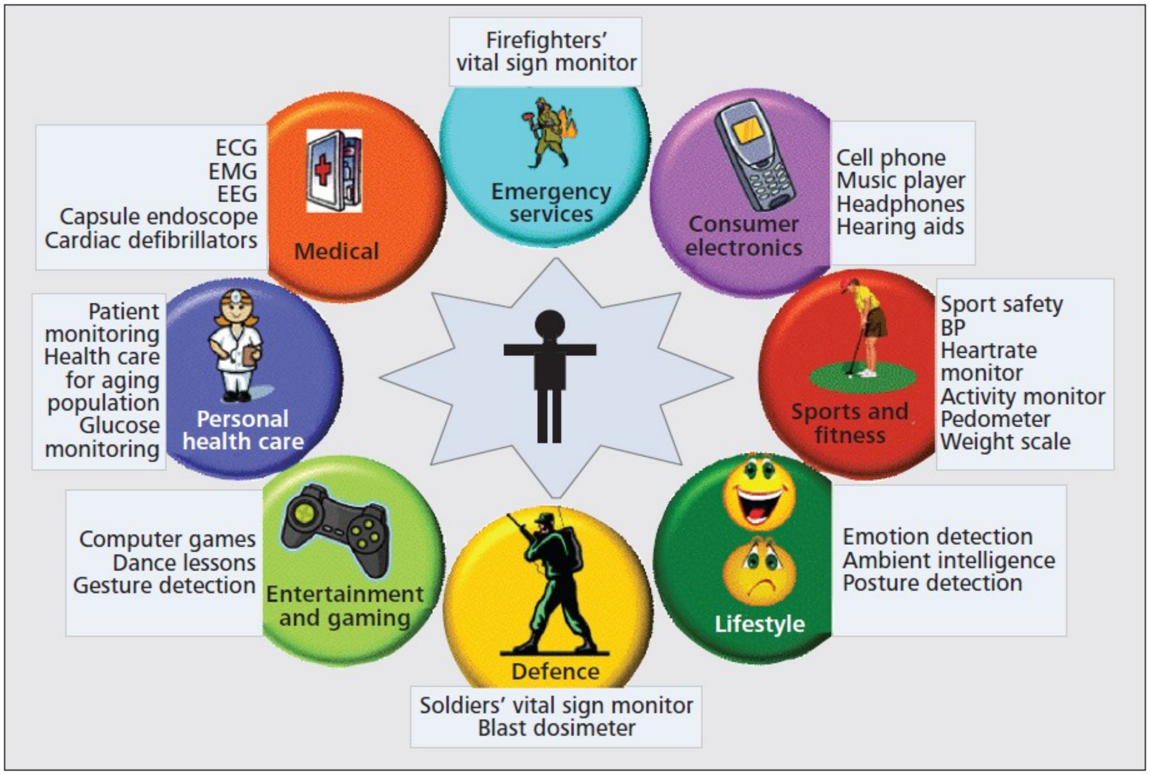


Figure 1-1: BAN applications [1]

BAN communication factors include short range transmission, low data rates, low energy consumption, and non-interference with other electronic/medical devices in addition to the reliable transmission of data with minimal delay. These specific needs of BAN communication are not fulfilled by the existing Personal Area Network (PAN) standards [2]. The IEEE task group 6 has been assigned the job of proposing a BAN communication standard, the IEEE Standard 802.15.6, by considering the short range transmission and QoS (i.e. reliability and latency) requirements of data, and low energy consumption of such devices [3]. An automated monitoring of BAN data in a hospital environment is required to address the challenges faced by the healthcare team during the process of collecting and managing delay-sensitive medical information [4].

This thesis proposes a novel Body Area Network peering framework, the ZK-BAN, in Chapter 4. The ZK-BAN provides mechanisms for the real-time display of patient data in an indoor hospital scenario even when the patient is mobile. More importantly, the ZK-BAN and associated routing protocols have taken into account the IEEE 802.15.6 standard requirements, and the ZK-BAN is compatible with this BAN standard. Associated energy and QoS aware routing protocols are required for the proposed ZK-BAN peering framework.

In this thesis, the proposed energy and QoS aware routing protocols consider all possible devices (i.e. NSC, MDCs, and BANCs) used in the hospital environment. The different communication devices employed in a real hospital environment include the Nursing Station Coordinator (NSC), Medical Display Coordinator (MDC), BANC, and the various implanted/wearable body sensors. The computational, energy, and memory capabilities of these devices are different and must be considered while routing BAN data in an indoor hospital environment. The motivation factor for my research is my own personal experience in the Children's hospital, IWK Health Centre, Halifax, NS, Canada. My son "Aaqib Khan (2006-2011)" was a brain tumor patient who spent about 3 years in the Hematology-Oncology and Pediatric Medical units of the IWK Health Centre. During his treatment, he had many surgeries and procedures. The wired sensors were connected to his body to monitor his health. His mobility was limited due to these sensors. But as a child, he wanted to move freely. His movement with the sensors and medical display unit was not easy. This has further motivated me to work on a patient monitoring framework and associated routing protocols that facilitate wireless patient monitoring in an indoor hospital environment with some patient mobility.

The scope of this research is to propose an indoor hospital based real-time patient monitoring system with the associated routing algorithms for energy-efficient, reliable, and QoS-aware communication. This research proposes a novel patient monitoring system, the ZK-BAN peering framework, based on communication devices that are typically found in real hospital environments across Canada. For the ZK-BAN framework to be of practical use it requires associated routing protocols with energy and QoS aware based capabilities. Lastly, an integrated routing protocol is proposed that combines all three capabilities to route regular, delay-sensitive, and reliability-sensitive data over real-life indoor hospital inspired networks that employ the ZK-BAN framework.

1.2. Contributions

In this thesis, five major contributions are made in regard to the ZK-BAN patient monitoring system and associated energy and QoS aware routing protocols in Body Area Networks. The detailed discussion of these contributions is provided in the remaining chapters of the thesis and is summarized below.

1. *Proposal of a novel patient monitoring framework named ZK-BAN peering framework for indoor hospital environments, previously published in* [5, 6]. The ZK-BAN peering framework emphasizes the real-time display of BAN data and discusses the different communication scenarios in real hospital environments. The ZK-BAN peering framework classifies the hospital sensor devices into three types (BANs, MDCs, and NSC) with the consideration of their energy levels. Both centralized and distributed modes of communication are used in the proposed framework. The centralized mode ensures data privacy and provides better control on the devices; whereas,the distributed mode reduces overall network traffic load and energy consumption in addition to accommodating BAN node mobility.

2. *Development of algorithms for energy-aware peering routing protocol (EPR) associated with ZK-BAN peering framework, previously published in* [5, 6]. EPR consists of a new Hello protocol in addition to the neighbor table algorithm and routing table construction algorithm. The main feature that distinguishes EPR from other similar protocols is the ability to choose the next hop that considers both residual energy and geographic information of the neighbor nodes, which helps to reduce the traffic load, number of packets forwarded by intermediate

nodes, end-to-end delay (latency), and energy consumption while simultaneously increasing the successful packet transmission rate without any packet being dropped due to buffer overflow. Extensive simulations in the OMNeT++ based Castalia-3.2 simulation environment have been performed, which show that the proposed protocol has better performance characteristics than DMQoS and noRouting protocols.

3. *Design of a novel modular routing architecture and associated QoS-aware routing protocol named QoS-aware Peering Routing protocols for Delay-sensitive data (QPRD) to handle the ordinary and delay-sensitive data for hospital BAN communication, previously published in* [7]. The modular architecture includes seven modules: the MAC receiver, the Delay Module (DM), the Packet Classifier (PC), the Hello Protocol Module (HPM), the Routing Services Module (RSM), the QoS-aware Queuing Module (QQM), and the MAC transmitter. The main capability that makes QPRD better than similar protocols is the choice of the best next hop by considering end-to-end path delays of all the paths from source to destination. The performance of the protocol is evaluated for both stationary and mobile source nodes scenarios. Simulations performed in the OMNeT++ based Castalia 3.2 simulator show that QPRD performs better than DMQoS and noRouting protocols.

4. *Design of a novel modular QoS-aware routing protocol (QPRR) for ordinary and reliability-sensitive data in hospital BAN communication, previously published in* [8]. The architecture of QPRR consists of five modules: the reliability module, the packet classifier, the Hello protocol module, the routing services module, and the QoS-aware queuing module. In the proposed QPRR mechanism, the choice of the next hop(s) depends upon the end-to-end path reliabilities and the use of multiple redundant paths. The simulations considered five different network topologies inspired by real hospital environments:

   Case 1) small-sized network (mimics an ICU) with stationary nodes;

   Case 2) small-sized network (mimics an ICU) with mobile source node;

Case 3) medium-sized network (mimics a Hematology-Oncology Unit) with 49 stationary nodes;

Case 4) medium-sized real hospital network (mimics a Pediatric Medical Unit) with 49 nodes and with mobile source node;

Case 5) large-sized network (mimics a Pediatric Medical Unit) with 93 nodes.

Extensive simulations using OMNeT++ based simulator Castalia 3.2 revealed that QPRR outperforms DMQoS and noRouting in terms of increased successful transmission rate, reduced number of reliability packets dropped, lower MAC buffer overflow, reduced number of Hello packets, lower overall energy consumption, and lower end-to-end delay (latency) for all five cases.

5. *Design a new integrated Energy and QoS aware routing protocol (ZEQoS) to deal with Ordinary Packets (OPs), Delay-Sensitive Packets (DSPs), and Reliability-Sensitive Packets (RSPs) simultaneously.* ZEQoS provides a mechanism to combine the functionalities of EPR, QPRD, and QPRR protocols in a modular but integrated routing framework. All data types (OPs, DSPs, and RSPs) are used for the testing of this protocol. The simulation results show that ZEQoS performs well with all data types.

1.3. THESIS OUTLINE

The nine chapters of this thesis cover the background work and the details of the contributions listed in the previous section. The remainder of the thesis is organized as follows.

Chapter 2: Body Area Networks

This chapter provides the background of Body Area Networks (BANs). The discussion includes the development process of BAN, the comparison of BAN with Wireless Sensor Networks (WSNs), the components and topologies of BAN, design considerations in BAN, a brief discussion of current BAN standards, different BAN application environments, and the use of BAN in healthcare.

Chapter 3: Literature Review

This chapter starts with a discussion of BAN routing protocols, then it sheds light on the classification of BAN routing protocols, their subcategories, and their advantages and disadvantages. At the end of this chapter the discussion of different suggested types of patient monitoring systems is provided.

Chapter 4: ZK-BAN Peering Framework

In this chapter the motivation, architecture, and advantages of the proposed patient monitoring framework is described in detail.

Chapter 5: Energy-aware Peering Routing Protocol (EPR)

This chapter presents the architecture and working mechanism of the energy-aware peering routing protocol (EPR). The simulation parameters and results are also discussed.

Chapter 6: QoS-aware Peering Routing Protocol for Delay-sensitive Data (QPRD)

To ensure the delivery of delay-sensitive data, a QoS-aware routing protocol (QPRD) is proposed in this chapter. The protocol results are compared with the other similar protocols with the extensive simulations performed by using the OMNeT++ based Castalia 3.2 simulator.

Chapter 7: QoS-aware Peering Routing Protocol for Reliability-sensitive Data (QPRR)

This chapter introduces the QoS-aware routing protocol (QPRR) for the reliable transmission of critical data. Redundant paths are used with the goal of improving end-to-end path reliability which also helps to maximize throughput. The simulation results are considered with different network topologies that simulate real hospital environments such as the Intensive Care Unit, Hematology-Oncology Unit, and Pediatric Medical Unit.

Chapter 8: Zahoor Energy and QoS aware Routing Protocol (ZEQoS)

This chapter discusses the routing protocol which provides the mechanism of combining all the functionalities of the three proposed routing protocols (i.e. EPR, QPRD, and QPRR) in a single integrated routing framework. It is shown that ZEQoS can handle all the three types of data packets (ordinary packets, delay-sensitive packets, and reliability-sensitive packets).

Chapter 9: Conclusions and Future Work

This chapter provides conclusions and insights into future directions for research.

CHAPTER 2

BODY AREA NETWORKS

Body Area Networks (BANs) provide a variety of services in different fields including medical and consumer electronics. These services are possible due to a set of tiny, sensitive, and smart sensors connected to the body. The major function of BAN in the medical field is inexpensive and real-time un-interrupted health monitoring with the help of implanted and wearable bio-sensors. In consumer electronics, BAN facilitates the functions of hearing aids, emotion detection, posture detection, computer games, music players, dance lessons, and activity monitoring. A general overview of BAN and its difference from Wireless Sensor Network (WSN) are discussed in this chapter. This chapter contains eight sections. Section 1 discusses the development process of BAN. Section 2 gives a comparison of BAN with the WSN highlighting their differences. Sections 3 and 4 explain BAN components and network topologies, respectively. Section 5 discusses BAN requirements. Section 6 explains BAN standards. Sections 7 and 8 discuss different uses of BAN in healthcare and its possible application environments, respectively.

2.1. DEVELOPMENT PROCESS OF BAN

The research area of BAN is an emerging field. BAN technology is a subfield of existing research in the field of Wireless Sensor Networks (WSNs), and can be considered to be a specialization of biomedical Engineering. The concept of BAN technology was first introduced in 1995 when Zimmerman [9] presented an idea of information exchange between electronic devices placed inside, on or near the human body. In this work [9], Wireless Personal Area Networks (WPANs) technology was employed to enable communication between electronic devices. The electrical properties of the communication channel, establishing a reliable link, and connection of PAN devices to specific applications was done on layers 1, 2, and 3 of the OSI model, respectively. A low carrier frequency (f_c<1MHz) is used to reduce energy consumption, minimize interference, and secure the communication from eavesdropping. Figure 2-1 shows how the battery operated WPAN transmitter and WPAN receiver work while connected with the human body. A biological conductor is formed where the displacement current flows through the body. The "earth ground" is used to prevent the shorting of the communication circuit [9].

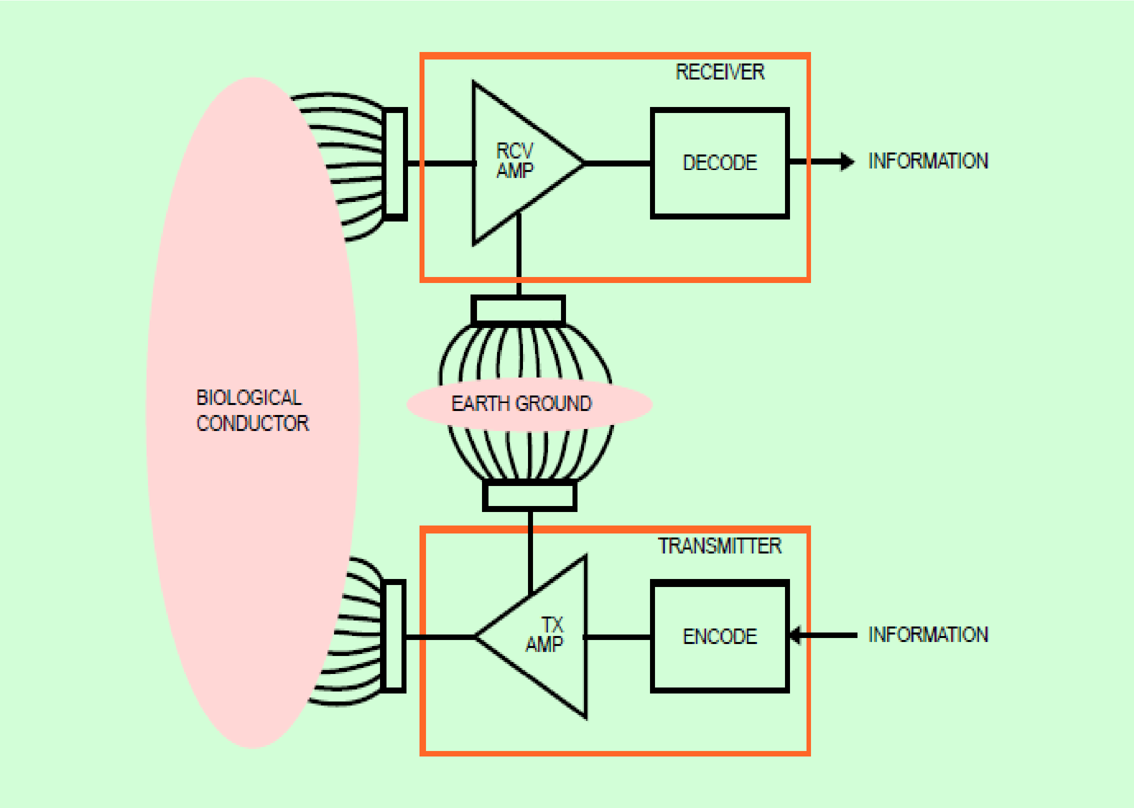


Figure 2-1: Block diagram of a PAN system [9]

The electric field model explains the communication between WPAN transmitter T and WPAN receiver R as shown in Figure 2-2. The human body was characterized in terms of electric fields. The electric fields D and G are created due to the transmitter and receiver respectively. It is observed that the transmitter creates more electric fields D than the receiver generated fields G. The connection of the transmitter and receiver is through D and G. The notations tb and rb are the impedances of the transmitter and receivers to the body. The transmitter causes higher values of electric fields and impedance to the body due to the direct attachment of the transmitter with the body. Figure 2-2 shows the electric fields A, B, C, D, E, F, and G caused by the transmitter and receiver electrodes. The electric field model with the typical component values for watch based PAN devices is shown in Figure 2-3.

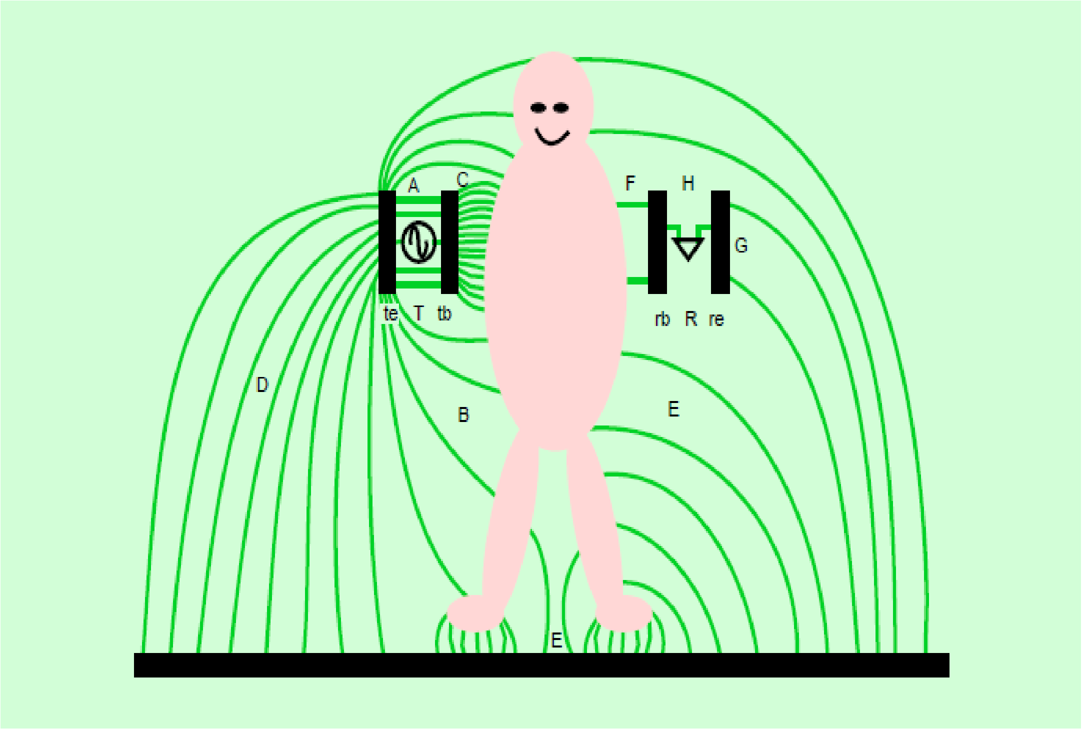


Figure 2-2: Electrical fields produced by transmitter (T) [9]

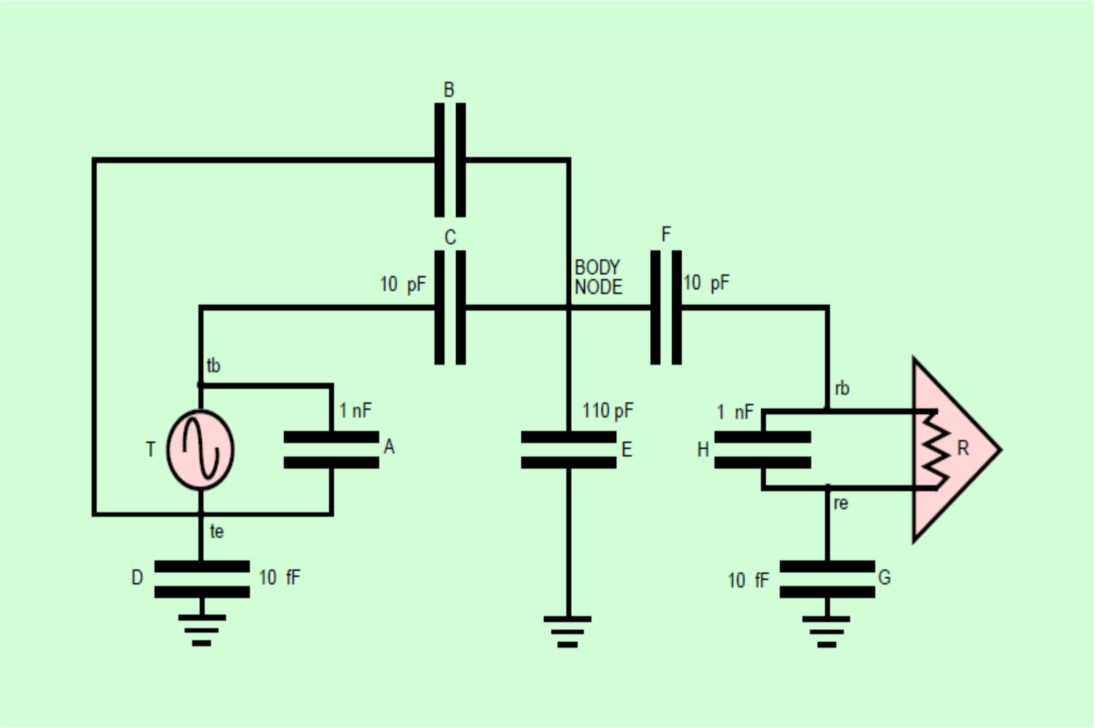


Figure 2-3: Electrical model of PAN system [9]

The circuit model reveals that the earth ground is very important for the WPAN devices which are attached to the human body. The best location of these devices is closer to the feet. The large size of the electrodes also plays a significant role to make the communication quality better [9].

In 2001, the term Body Area Network (BAN) was introduced for the first time (instead of WPAN) in applications and communications of the electronic devices used on, in and around a human body [10].

2.2. Comparison of BAN with WSN

Body Area Networks (BANs) have specific characteristics as compared to Wireless Sensor Networks (WSNs) and Wireless Personal Area Networks (WPANs). Energy efficiency, cost structures, reliability, multi-hop communication, and node density are the common features of BAN and WSN. The types of challenges for BAN and WSN are similar; however, the sensitivity of these challenges is different for these two fields. BAN differs from WSN in seven major areas. These seven areas are network size, node characteristics, operational environment, resource limitations, mobility, accessibility, and context awareness [11, 12]. The details of these areas are given below.

2.2.1. Network Size

The BAN communication is between the wearable and implanted sensors connected with the human body. Generally, the number of nodes used in BAN is less than a few dozen; whereas, the number of sensor nodes used in WSN varies widely from as few as a dozen to as large as several thousands. The transmission range of sensors used in a BAN is limited to the height of the human body. Generally, all the body sensors send their data to a central node, also known as BAN Coordinator (BANC). The BANC forwards these data towards the destination, and is similar to a Cluster-Head (CH) in WSN. Due to the low transmission range, the network area of inter-BAN communication is normally within a few meters; whereas, the area for WSN is in the 100's meters or a few kilometers. The low transmit power reduces body tissue damages and electromagnetic interference effects on sensitive hospital equipment. Figure 2-4 shows the network size of a typical BAN.

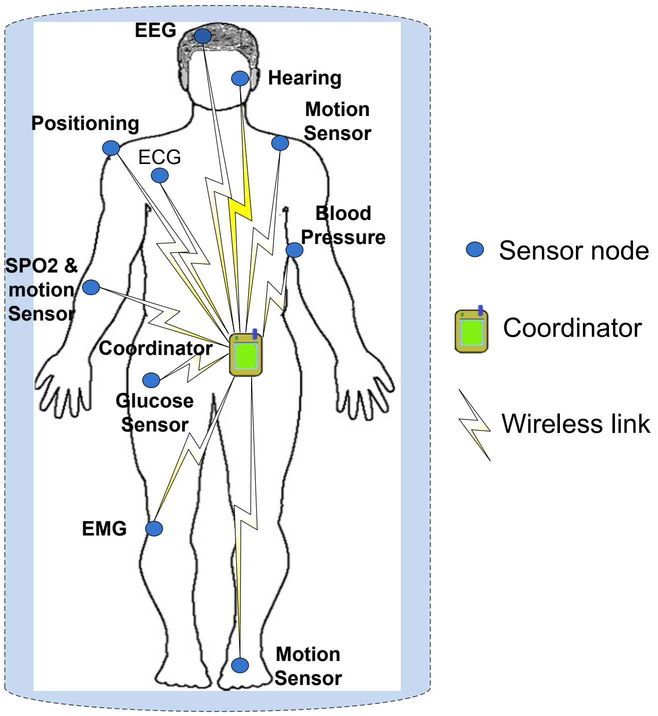


Figure 2-4: Network size of BAN

2.2.2. Node Characteristics

Though sensor nodes are used in both BAN and WSN, the characteristics of these nodes are different for these two types of sensor networks. The use of WSN in larger and remote areas leads to different requirements for the kind of WSN nodes than the nodes used in BANs. As noted previously BANs are employed over a smaller area i.e. around the human body. Some of the important differences in nodes characteristics are given below.

2.2.2.1. Node Identification

The node identification is the method of assigning a unique node ID for each node in a network. The node identifiers used for BAN and WSN nodes have local significance. Unlike WSN where the number of sensor nodes used may be, at times, indiscriminate, BAN requires accurate numbers of sensor nodes used on or around the body. The size of BAN nodes are also smaller than the nodes used in WSN. The number of bits used for a node ID in BAN is much less than the bits used for WSN node identification (as there are fewer BAN nodes in a BAN). More bits are employed to identify a WSN node as WSNs typically employ hundreds or even thousands of sensor nodes. The lower number of bits in a BAN node ID, shorter communication distances, and lower power levels help to keep the energy required to send or receive data packets low.

2.2.2.2. Node Functionality

Each node in a WSN performs a dedicated task. Due to remoteness and inaccessibility of areas, multiple nodes with similar functionality are used for redundancy. The use of redundant nodes helps to continue monitoring the area in case of the failure of a few WSN nodes. On the other hand, each node in BAN performs multiple tasks and there is no redundancy of the nodes.

2.2.2.3. Node Accuracy

The use of redundant nodes in WSN helps to compensate the accuracy [13] and allows validation of results. BAN uses sensor nodes that are more robust and accurate in nature. This is due to the limited number of nodes and the sensitive nature of the human body.

2.2.2.4. Node Size

WSN can have sensors of any size selected according to circumstance. On the other hand the smaller size of sensor nodes in BAN is one of the key requirements. The implant nodes are placed inside the human body so the size of these nodes must be as small as possible. Figure 2-5 shows the Band-Aid-like circuits introduced by the team of scientists from the University of Illinois [14]. The electronic skin sensor is thin like a mini-tattoo which can be worn easily on the patient's skin as shown in Figure 2-5a. This smart skin helps to monitor the muscle activity, brain waves, and heart rate. The lightweight, tiny size and flexibility of this sensor provide comfort to the patient. Figure 2-5b shows the sensor's flexibility and removal is shown in Figure 2-5c [14, 15]. A new touch-hear speech recognition system helps readers to get more information about text. The function of finger implant sensors is to get the text, convert it to voice signals, and then transmit it to the ear wearable sensor receiver as shown in Figure 2-6. This system also provides different pronunciations of the words [14].

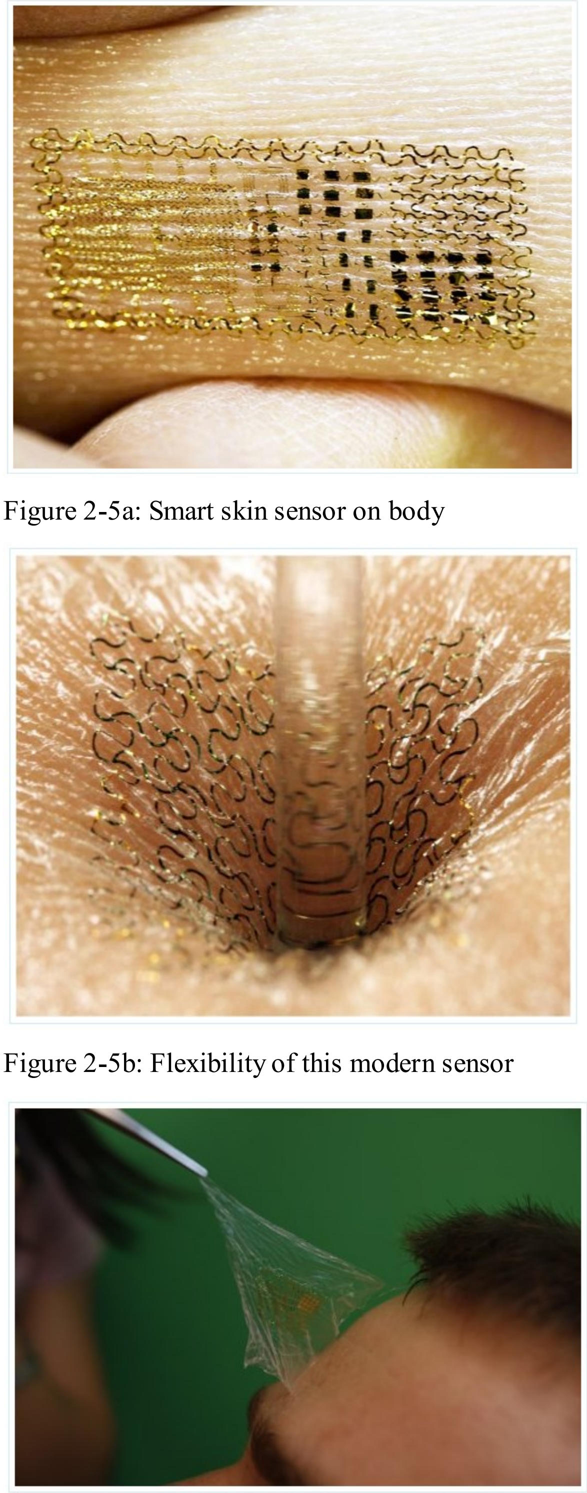


Figure 2-5a: Smart skin sensor on body

Figure 2-5b: Flexibility of this modern sensor

Figure 2-5c: Deformation of electronic skin

Figure 2-5: Band-Aid-like circuit sensor [14]

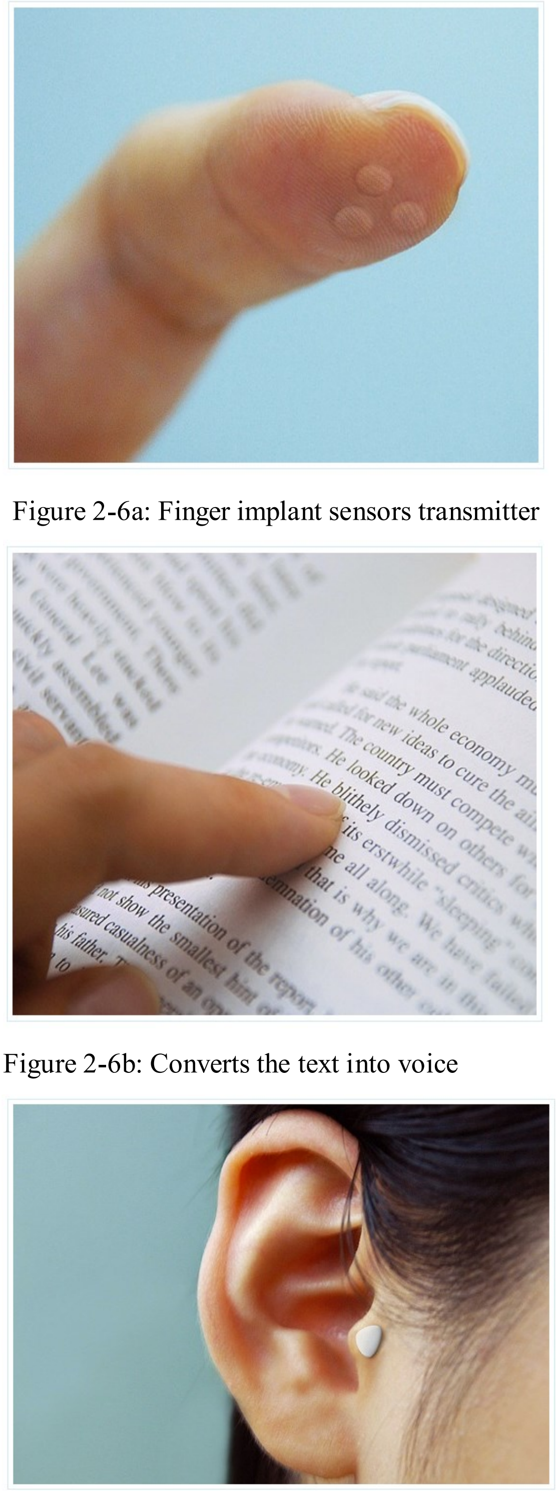


Figure 2-6a: Finger implant sensors transmitter

Figure 2-6b: Converts the text into voice

Figure 2-6c: Wearable receiver sensor

Figure 2-6: Touch-hear speech recognition system [14]

2.2.3. Resource Limitations

The sensor nodes in BAN are tiny in size as compared to the nodes used in WSN. This is one of the major causes of resource limitations. The limitations of BAN nodes include lower bandwidth, low energy source, slower processing speed, and smaller memory as compared to the WSN node.

2.2.4. Mobility

The BAN structure has biological variations and complexity. This is due to the more variable structure of a human body. The protocols and techniques used in BAN communication need to be compatible with mobility of the human body. The physical topology (structure) of a WSN is mostly fixed or static. Other than the structure of the body, there are different requirements for BAN communication in indoor and outdoor environments. The indoor BAN environments are in hospital or at home. The outside environment includes scenarios such as a patient on the road.

2.2.5. Accessibility

The WSN sensor nodes are more easily accessible which helps in node replacement or even in disposal of the node. On the other hand the accessibility of implant nodes in BAN is difficult. A surgical operation may be required for the replacement of an implant node. This process of node replacement can cause damage to internal organs and blood vessels during surgery.

2.2.6. Context Awareness

Due to the relatively static placement of WSN nodes, and their known operational environment, context awareness is not very important in WSN. On the other hand the mobility of the human body and the sensitive nature of the human body require the feature of context awareness in BAN.

The comparison of BAN with WSN is summarized in Table 2-1.

Table 2-1: Comparison of BAN with WSN

	BAN	WSN
Network Size	Smaller (within few meters)	Large area (meters to kilometers)
Node Identification	Less bits are used due to the smaller (< 64) number of nodes	More bits are required to provide a unique ID to each node. Number of nodes in WSN is from a dozen to a thousand
Node Functionality	No redundancy and multiple tasks are required from each node	Redundant nodes used. Each node performs a single dedicated task
Node Accuracy	Highly robustness and accuracy is required	Redundant nodes help to compensate the accuracy
Node Size	Smaller node size required	Smaller size preferable but not a limitation
Operational Environment	Operational area is in, on or around a human body	Stable to inaccessible. Can be extreme in weather or noise
Resource Limitaions	Smaller nodes support less bandwidth, low energy source, slower processing, and less memory size	More resources than BAN node due to the larger acceptable size
Mobility	Depends upon the mobility of a human body. Biological variations and complexity	Mostly fixed or static nodes
Accessibility	Wearable nodes accessible but implant sensors are not easily accessible. Surgery is required to access implant sensors	Depends upon the operational area. Normally accessible. Easy node replacement
Context Awareness	Required due to mobile nature of human body	Not important due to the static nodes used in known environment

2.3. BAN Components

Generally, the basic architecture of the technologies used in different types of WSNs is similar. A typical sensor node structure is shown in Figure 2-7. The brief description of the main components used in a BAN is given below.

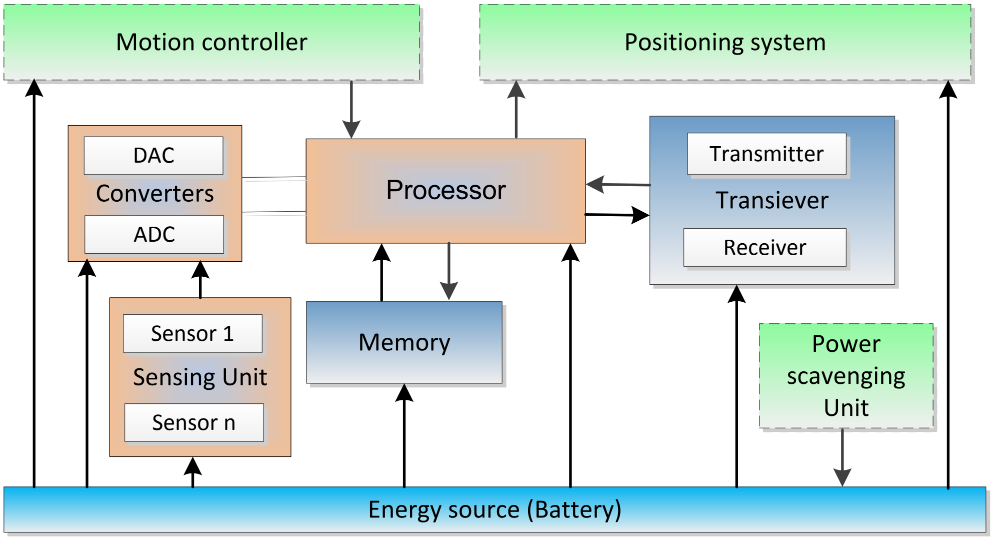


Figure 2-7: BAN sensor node structure

- **Energy Source:** Generally, the size of the batteries limits the source of energy available to BAN nodes. The tiny size of sensor node batteries allows very low power levels as compared to the larger batteries used in the WSN nodes.

- **Processor:** This is the brain of the sensor node. The processor handles all computations in the node. The MSP430 from Texas Instrument (TI) is an example of a processor used in some of the BAN nodes [16]. The world's Ultra-low power MSP430 is a 16-bit microcontroller platform. The speed of this processor is 8MHz to 15 MHz and the number of pins is 14 to 113 [17].

- **Memory:** Different kinds of memories are used in BAN. A typical BAN node with MSP430 processor contains 128B to 64KB RAM and 0.5KB to 512KB Flash memory [17].

- **Transceiver:** The transceiver is used to send or receive the data from or to the node. Chipcon CC2420 is used for low power and low voltage wireless communication in a BAN node [18]. The current consumption of receiver and transmitter of CC2420 are 19.7 mA and 17.4 mA respectively [19].

- **Sensors:** The sensing unit in the sensor node contains multiple sensors. These sensors are used to monitor the physiological or biochemical parameters of the disease processes in the human body. The ECG sensors determine the heartbeat rate and any damage of the heart. The sensing unit in Blood Pressure (BP) sensor measures pressure of circulating blood on the blood vessels.

- **Actuators:** Actuators are used to take action after getting the data from the sensor or from the user. Actuators convert motion into energy or energy into motion. An actuator placed with a body sensor allows the healthcare professional to inject the insulin in case it is required for a diabetic patient.

- **Operating Systems:** TinyOS is used as the operating system in the BAN node. TinyOS, a BSD licensed open source operating system, is specifically designed for all kinds of WSN platforms. TinyOS is ideally suited to the BAN nodes due to its special design for low-power devices [20].

2.4. BAN Topologies

The term network topology is used to define the structure for data communication between the different devices in the same network. The sensor nodes are directly connected to each other in Peer-to-Peer (P2P) topology as shown in Figure 2-8. Peer-to-Peer, star, mesh, and cluster tree are the basic topologies used in BAN [12]. A hybrid of these basic topologies is also commonly used in BAN. In order to get the unique characteristics and performance requirements of an application, the main factors to be considered by the application developer for choosing the appropriate topology are the sensor node costs, battery drain, complexity of routing, robustness, scalability, latency, mobility, and spatial coverage [12].

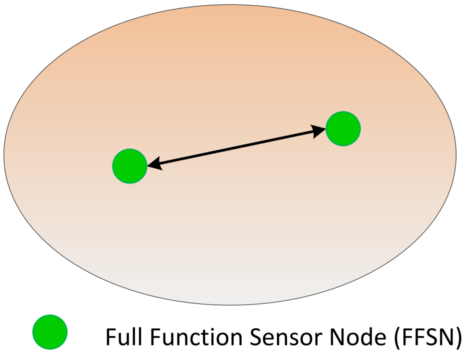


Figure 2-8: Peer-to-Peer (P2P) topology

The smaller size sensor nodes with limited resources require low power consumption communication. Like IEEE 802.15.4, the sensor nodes used in BAN can be divided into two main types: Reduced Function Sensor Nodes (RFSNs) and Full Function Sensor Nodes (FFSNs) [21]. RFSNs can perform only P2P communication without routing capability. RFSNs are used in a place where the energy consumption is a major concern. For example, the battery life of a typical implant node should be at least three years. The implant sensor nodes are mostly RFSNs. On the other hand, FFSNs are capable of network routing functions in addition to their usual data communication capabilities. Their network routing capability allows FFSN to route the data received from one node to the other nodes by following the paradigm of the routing scheme used. The advantages, disadvantages, and diagrams of BAN topologies are discussed below.

2.4.1. Star Topology

In a star topology, all the Reduced Function Sensor Nodes (RFSNs) are connected with a central Full Function Sensor Node (FFSN) as shown in Figure 2-9. The communication between the nodes is only possible via the central FFSN. The advantages of star topology are simplicity, lower network cost, less energy consumption, low latency, and high bandwidth. The major drawback of a star topology is a single point of failure. The whole communication fails in case of central node failure. The other drawbacks are poor scalability, inefficient RFSN to RFSN communication, higher power consumption of the central node, and limited spatial coverage [12].

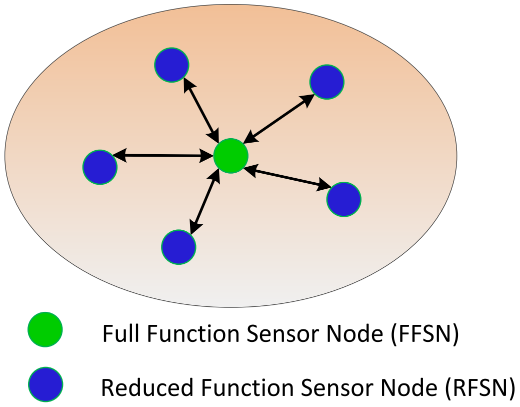

Full Function Sensor Node (FFSN)
Reduced Function Sensor Node (RFSN)

Figure 2-9: Star topology

2.4.2. Mesh Topology

The nodes used in mesh topology are all Full Function Sensor Nodes (FFSNs) as shown in Figure 2-10. Each node in this topology is capable of doing all the routing operations as well as data communication between the nodes. The main advantage of this topology is the multiple paths provided to each node which help to continue the communication processes in case of the failure of one or more nodes. The other features of this topology are scalability, large spatial coverage, fault tolerance, distributed processing, medium complexity, and balanced energy consumption. The disadvantages of mesh topology are higher node costs, complex routing operations, high latency, and low bandwidth [12].

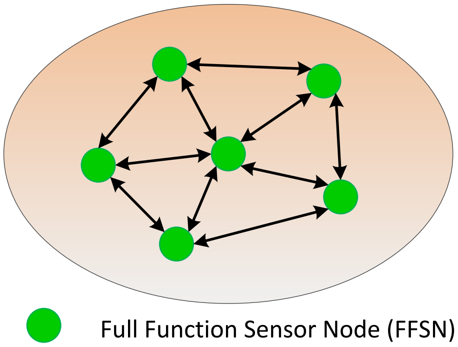


Figure 2-10: Mesh topology

2.4.3. Cluster Tree Topology

Multiple star topologies form a cluster tree topology as shown in Figure 2-11. The nodes from one star topology can communicate to the nodes of other star topologies via their central nodes. The advantages of a cluster tree topology are large spatial area coverage, low power consumption of leaf nodes, increased scalability and medium complexity. The drawbacks of this topology include low reliability, high latency, and low bandwidth [12].

2.4.4. Hybrid Topology

A hybrid topology is a combination of both star and mesh topologies. An example of hybrid topology is shown in Figure 2-12. The main advantage of hybrid topology is the increased scalability. The new nodes can be added at any point. The other features of hybrid topologies are large spatial coverage, high potential reliability, and scalability. The disadvantages are high complexity, high latency, asymmetrical power consumption, and low bandwidth [12].

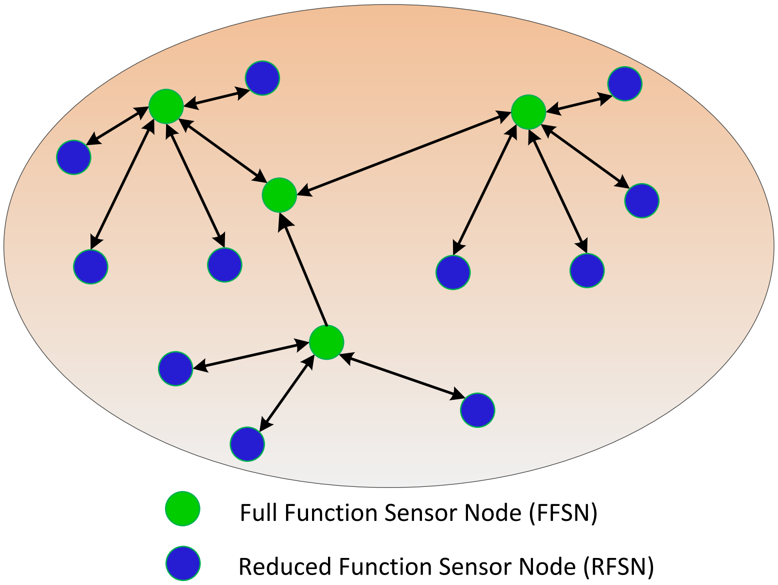

● Full Function Sensor Node (FFSN)
● Reduced Function Sensor Node (RFSN)

Figure 2-11: Cluster tree topology

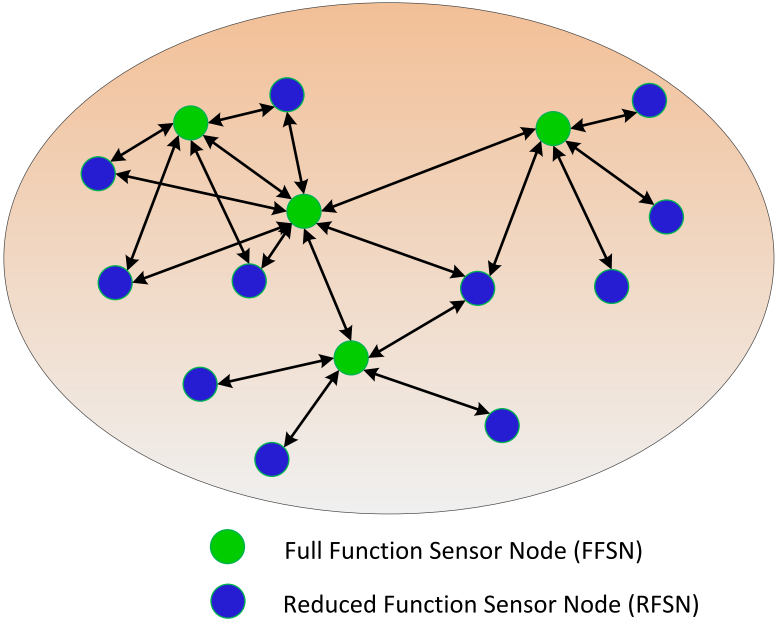

● Full Function Sensor Node (FFSN)
● Reduced Function Sensor Node (RFSN)

Figure 2-12: Hybrid topology

2.5. BAN Requirements

Even though, the advancements of WSNs are significant, BAN face unique technical challenges mainly due to the diversity of applications and their tough environmental requirements. The requirements of BAN are different than the other existing wireless sensor technologies. Table 2-2 shows the technical requirements for a typical BAN [1, 22].

Table 2-2: BAN requirements

	Requirement	Expected Range
Data Rate	Scalable	From few kbps to 10Mbps
Effective Area	In, on, or around the body	Within 0-5 meters
Lifetime	Long for wearable and ultra-long for implant sensors	About five years for implants and one week for wearable sensors
Security	Different levels but light weight	Privacy, Authentication, Confidentiality, Message integrity, Encryption, Authorization, Authentication
Setup Time	Should be fast with minimum delays	Less than three seconds
Biocompatibility	Must be compatible with human body physiology	Meet the regulations of Food and Drug Authority (FDA) and other regulatory agencies
Fault Management	Mechanism for node failure detections	Self-healing capability of sensor nodes
Customization	Reprogrammable and configurable nodes	Remotely accessibility of nodes and context awareness feature
Topology	Star, Tree or Mesh	Centralized and distributed modes with multi-hop features
Quality of Service	Efficient communication with maximum throughput	Consideration of reliability and delay control mechanisms
Energy and Power	Least energy consumption during communication operations, Controlled power consumption w.r.t. the operations	Power consumption upto 0.1mW for standby mode and upto 30mW for fully active mode
Medium Access Control	Self-control, scalable, and reliable	Lower power during listening and wakeup modes
Ergonomic Concerns	Shape, size, weight, form factor and path loss restricted by organ and location in the body	Small size, light weight, harmless, and non-invasive
Compatibility	Compatible with other electronic consumer devices and BANs	Able to communicate with other devices around body and simultaneous co-located operation of upto ten other BANs

Frequency Bands	Bands assigned for medical devices and other global unlicensed	UWB, ISM, Med Radio and WMTS

2.5.1. Security & Privacy

Privacy, confidentiality, authentication, authorization, and integrity are the basic requirements of the BAN applications because of its legal, financial, privacy, and safety implications. The limited processing power, low memory, low energy, lack of user interface, unskilled users, longevity of devices, and global roaming, make it difficult to apply conventional security and privacy mechanisms in BAN. New lightweight and energy-efficient security methods are required for BAN [23].

2.5.2. Data Transmission

The different BAN applications require different values of data transmission parameters. The important parameters include data rate, number of nodes used, BAN topology, setup time, latency, Bit Error Rate (BER), and battery lifetime [1]. The requied values of these parameters with respect to the selected BAN applications are given in Table 2-3.

Table 2-3: Selected BAN applications requirements [1]

Applications	Data Rate	Nodes Used	BAN Topology	Setup Time	Latency	Bit Error Rate	Battery Lifetime
EEG	86.4kbps	<6	Star	< 3Sec	< 250ms	$<10^{-10}$	>7 days
ECG	72 kbps	<6	Star	< 3Sec	< 250ms	$<10^{-10}$	>7 days
Positioning	<10kbps	2	Star	<3 sec	< 250ms	$<10^{-10}$	>7 days
Hearing	200kbps	3	Star	< 3sec	< 250ms	$<10^{-10}$	>2 days
Motion Sensor	<10kbps	3	Star	<3 sec	< 250ms	$<10^{-10}$	>7 days
SPO2/temp/BP	<10kbps	3	Star	<3 sec	< 250ms	$<10^{-10}$	>7 days
Drug Dosage	<1kbps	2	P2P	<3 sec	< 250ms	$<10^{-10}$	>1 day
Glucose Sensor	<10kbps	2	Star	<3 sec	< 250ms	$<10^{-10}$	>7 days
EMG	1.536Mbps	<6	Star	<3 sec	< 250ms	$<10^{-10}$	>7 days
Imaging	<10Mbps	2	P2P	<3 sec	< 100ms	$<10^{-3}$	>0.5 day
Implant	<1 Mbps	2	P2P	<3 sec	< 250ms	$<10^{-3}$	>3 years

2.5.3. Power Consumption

The tiny battery size employed in a typical BAN node means that these batteries store very little energy. It is fact that the wireless transmission operations consume much more power as compared to wired transmission. Different techniques are used to save power during wireless communication. Some of these power saving techniques include avoiding unnecessary retransmissions, reducing the frequency of sending network control messages, reducing the size of message headers, and using the standby or sleep mode whenever possible [23]. Figure 2-13 shows the expected position of BAN technology as compared to the other standards in terms of power consumption.

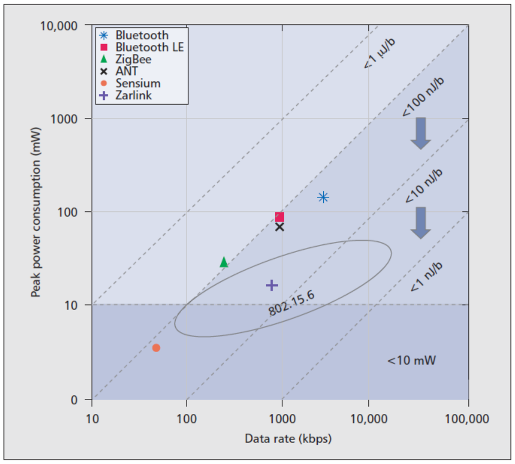


Figure 2-13: Expected position of BAN technology [1]

The average power consumption used for the continuous monitoring in different BAN applications is shown in Figure 2-14.

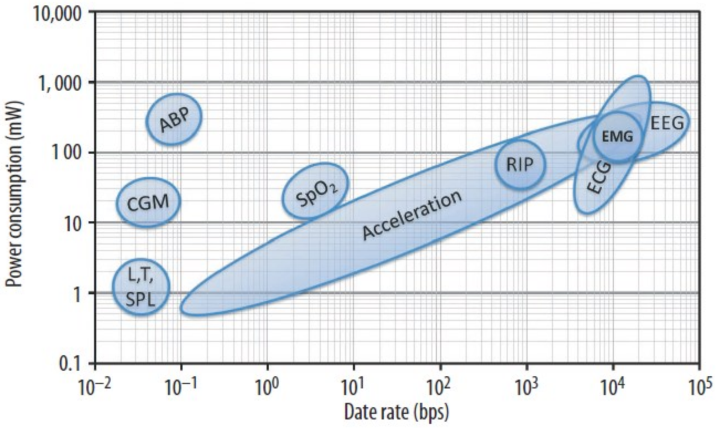


Figure 2-14: Average power consumption of continuous monitoring [24]

2.5.4. Quality of Service (QoS)

Quality of service is an important requirement for reliable and energy-efficient BAN communications. A false signal or alarm can cause a severe problem in patient monitoring. The response of an actuator to faulty data from a sensor can create complications. The smaller size of memory and energy-source also restrict the number of retransmissions. So a very efficient fault diagnoses system provides better reliability which is required to improve the BAN quality of service [22].

2.5.5. Compatibility

There are many standards involved in BAN transmission. For example, bio-sensors need biocompatibility, interface between biomedical equipment needs Food and Drug regulatory Authority (FDA) compatibility, and radio frequency transmission needs communication standard compatibility [1]. Regulatory compliance becomes more complicated when the device user is able to move globally.

2.6. BAN STANDARDS

The Wireless Personal Area Network (WPAN) protocols were commonly used for the implementations of BAN communication before the development of BAN standards. The important WPAN protocols are ZigBee (IEEE 802.15.4) and Bluetooth (IEEE 802.15.1) [25, 21, 26]. There are several candidate wireless technologies that can be used for BAN communications. The characteristics of the candidate technologies for BAN are given in Table 2-4.

The IEEE 802.15 Working Group was formed to develop the standards for WPANs or short distance wireless networks. The communication between the portable and mobile computing devices like PCs, Personal Digital Assistants (PDAs), peripherals, cell phones, tablets, and consumer electronics devices are supported by WPAN standards.

The IEEE 802.15 group is a sub-group of the 802 local and metropolitan area network standards committee of the IEEE computer society. One of the IEEE divisions, IEEE-SA, is responsible for the standardization of the new protocols.

Table 2-4: Characteristics of candidate technologies for BAN [1]

	Frequency	**Modulation**	**Topology**	**Channels**	**Operating Range**	**Peak Power**	**Data Rate**	**Setup Time**
Bluetooth Classic	2.4 GHz	GFSK	Scatternet	79	1-10 m on-body	~45mA	1-3 Mbps	~3 sec
Bluetooth Low Energy	2.4 GHz	GFSK	Piconet Star	3	1-10 m on-body	~28mA	1 Mbps	<100 sec
ZigBee	2.4 GHz	O-QPSK	Star, Mesh	16	10-100 m on-body	~16.5mA	250 kbps	30 msec
ANT	2.4 GHz	GFSK	Star, Tree, or Mesh	125	10-30 m on-body	~22mA	1 Mbps	
Senium	868 MHz 915 MHz	BFSK	Star	16	1-5 m on-body	~3mA	50 kbps	<3 sec
Zarlink ZL70101	402-405 MHz 433-434 MHz	2FSK/4FSK	P2P	10 MedRadio, 2ISM	1-5 m on-body	~3mA	50 kbps	<3 sec

The suitability and non-suitability of the candidate technologies for BAN are summarized in Table 2-5.

Table 2-5: Suitability and non-suitability of other technologies with respect to BAN [1]

Technology	**Suitability**	**Non-suitability**
Bluetooth Classic	Low cost, satisfactory data rate, use in cell phones and laptops, used in health monitoring devices	Higher power consumption, non-support to implant devices, limited QoS and scalability, less secure
Bluetooth Low Energy	Lower power than Bluetooth, Interoperable with Bluetooth	Limited scalability and QoS, non-support to implant devices, less secure
ZigBee	Scalable, lower power than Bluetooth, used in health monitoring devices	Limited QoS, non-support to implant devices, low data rate
ANT	Simple protocol, lower power consumption	Limited throughput and QoS, non-support to implant devices
Sensuim	Custom designed for BANs, ultra-low-power consumption	Low data rate, limited QoS, proprietary
Zarlink ZL70101	MedRadio compliant, support to implant devices, ultra-low-power consumption	Support only implant devices, proprietary

Figure 2-15 shows the status of IEEE 802.15 until February 2013. A number of WPAN standards from 802.15.1 to 802.15.9 had been developed by the IEEE 802.15 Working Group (WG). The recent standard developed from WG15 is IEEE 802.15.6 which was approved in February 2012. The current in progress standards are 802.15.8, 802.15.9, 802.15.4j, 802.15.4k, 802.15.4m, 802.15.4n, 802.15.4p, and 802.15.4q [27].

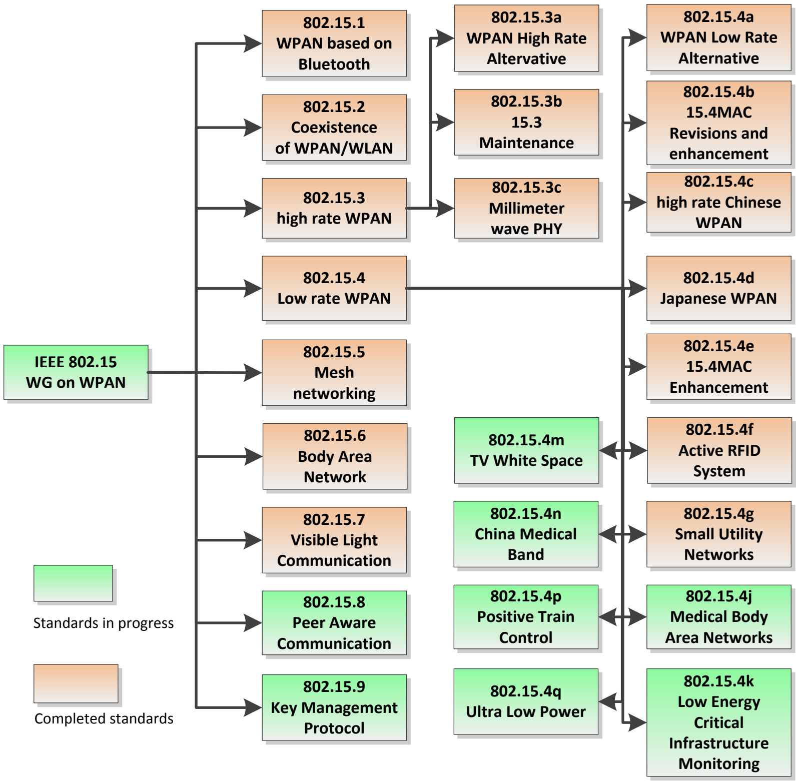

IEEE 802.15 WG on WPAN
802.15.1 WPAN based on Bluetooth
802.15.2 Coexistence of WPAN/WLAN
802.15.3 high rate WPAN
802.15.4 Low rate WPAN
802.15.5 Mesh networking
802.15.6 Body Area Network
802.15.7 Visible Light Communication
802.15.8 Peer Aware Communication
802.15.9 Key Management Protocol
802.15.3a WPAN High Rate Altervative
802.15.3b 15.3 Maintenance
802.15.3c Millimeter wave PHY
802.15.4m TV White Space
802.15.4n China Medical Band
802.15.4p Positive Train Control
802.15.4q Ultra Low Power
802.15.4a WPAN Low Rate Alternative
802.15.4b 15.4MAC Revisions and enhancement
802.15.4c high rate Chinese WPAN
802.15.4d Japanese WPAN
802.15.4e 15.4MAC Enhancement
802.15.4f Active RFID System
802.15.4g Small Utility Networks
802.15.4j Medical Body Area Networks
802.15.4k Low Energy Critical Infrastructure Monitoring
Standards in progress
Completed standards

Figure 2-15: Current status of IEEE 802.15 Working Group (WG)

Some important issues of BAN data transmission are to ensure high reliability, low latency, compatibility with movable sensors, and low energy consumption. The specific needs of BAN communication are not fulfilled by the existing Personal Area Network (PAN) standards [22]. The development process of IEEE 802.15.6 is shown in Figure 2-16. The first BAN interest group (IG-BAN) was formed in a meeting held at Jacksonville, FL, USA, in May 2006. IEEE Task Group 6 was assigned the job in November 2007 to suggest a BAN communication standard IEEE 802.15.6 by considering short range transmission, reliability & latency requirements of QoS, and less energy consumption [3].

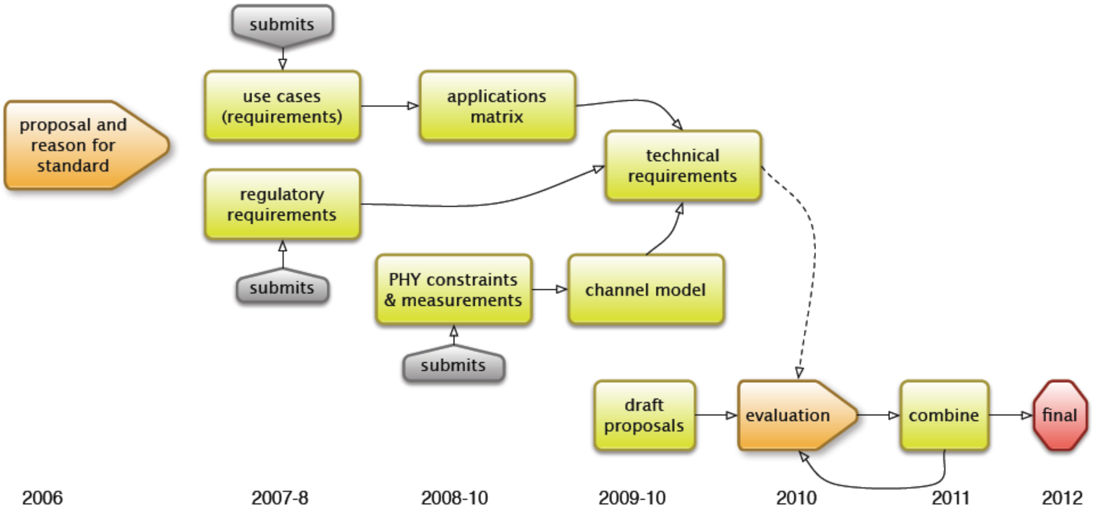


Figure 2-16: IEEE 802.15.6 development process [28]

2.7. BAN IN HEALTHCARE

According to the World Health Organization survey in 2005, the deaths of 17.5 million people were due to cardiovascular disease. These deaths were the 30 percent of all the deaths during 2005. Currently 180 million people are affected by diabetes worldwide and these numbers are expected to increase to 360 million by 2030. More than 2.3 billion people will be overweight by 2015 [1].

The number of elderly people and chronic disease patients increases rapidly. The quality and quantity of healthcare services are required to improve with respect to this increase. The most important application of BAN is to monitor the patient’s medical data in the healthcare environment. The advancements of the medical field also bring new specialties of different areas in the healthcare. The continuous monitoring of the patient in indoor (hospital, home) and outdoor environments help physicians to get useful information that can be used in developing better treatment plans. Hospital, ambulatory, PeriOperative, ER/Trauma Units, Rescue, Maternity/Ob, and Nursing Homes are some of the healthcare environments where BANs can be used.

2.7.1. Chronic Disease Patient Monitoring

For a chronic disease patient, the formal procedure of routine visits is required to monitor the progress, development of complications or relapse of the disease. The choice of what to monitor, when to monitor, and how to adjust will affect the treatment plan. Poor choices can have a severe effect on the patient's health. Specialized bio-sensors may be used to monitor the different physiological or biochemical parameters of different disease processes. Some of the examples are provided in Table 2-6 [12].

Table 2-6: Sensor types used for physiological/biochemical parameters to monitor the different disease processes

	Physiological Parameters	**Biochemical Parameters**
Cancer	Weight loss, body fat sensors (implantable/wearable mechanoreceptor)	Tumor markers, blood detection (implantable bio-sensor)
Hypertension	BP (implantable/wearable mechanoreceptor)	Adrenocorticosteroids (implantable bio-sensor)
Heart Disease	ECG, heart rate, BP, (implantable/wearable ECG sensors)	Implantable bio-sensors
Asthma	Respiration, peak expiratory flow, oxygen saturation (implantable/wearable mechanoreceptor)	Oxygen pressure (implantable/wearable bio-sensors)
Stroke	Impaired speech, memory, Gait, activity (wearable sensors)	
Diabetes	Visual impairment, sensory disturbance (wearable accelerometer)	Blood glucose, glycated hemoglobin (implantable bio-sensors)
Rheumatoid Arthritis	Reduced function, joint stiffness (wearable accelerometer, thermistor)	Inflammatory and auto-immune markers (implantable bio-sensor)
Renal Failure	Urine output (implantable sensors)	Urea, potassium, creatinine (implantable bio-sensors)
Vascular Diseases	Peripheral perfusion, blood pressure (implantable/wearable sensors)	Hemoglobin level (implantable bio-sensor)
Infectious Diseases	Temperature (wearable thermistor)	Inflammatory markers, white cell count (implantable bio-sensor)
Post-operation Monitoring	Blood pressure, ECG, oxygen, temperature (implantable/wearable sensors)	Operation spot monitoring, blood glucose, hemoglobin level (implantable bio-sensor)

2.7.2. Elderly Patient Monitoring

The rapid increase in the elderly population will generate a severe shortage of healthcare professionals in the near future. BAN provides a very cost effective solution to monitor the health parameters of the elderly people without disturbing their daily activities. Only a few wearable sensors and a small central device will enable caretakers or healthcare professionals to get information in case of any problem.

2.7.3. Hospital Patient Monitoring

Various levels of monitoring are necessary for the treatment of a patient in the hospital environment. The patient with stable conditions may require monitoring only four to six times per day in terms of vital signs measurement (blood pressure, heart rate, ECG, respiratory rate, and temperature), visual appearance (assessing their level of consciousness), and verbal response (asking them about the pain). The level of monitoring is very high for patients who are in an Intensive Care Unit (ICU). The monitoring of pre- and post-surgery is also very important. Different bio-sensors are used to monitor the different vital signs of a patient. Figure 2-17 shows the location of wearable sensors used with the human body. BAN can be the most cost effective solution for the continuous monitoring of a patient in the hospital environment.

2.8. BAN APPLICATION ENVIRONMENTS

Different possible BAN scenarios can be deployed depending on the BAN applications. In these scenarios, the collection of sensor nodes may work alone or with the combination of other positioning or cellular devices. The following are the three types of BAN environments.

2.8.1. Stand Alone BAN

Star and mesh topologies are used in standalone BAN [12]. Communication between the nodes is possible via a central personal device called coordinator in star BAN. On the other hand the data is transferred directly from one node to another node in mesh BAN. The star and mesh based BANs are shown in Figure 2-18. There is no connectivity of

BAN with other networks in the standalone case. The results can be analyzed/seen directly on the central device.

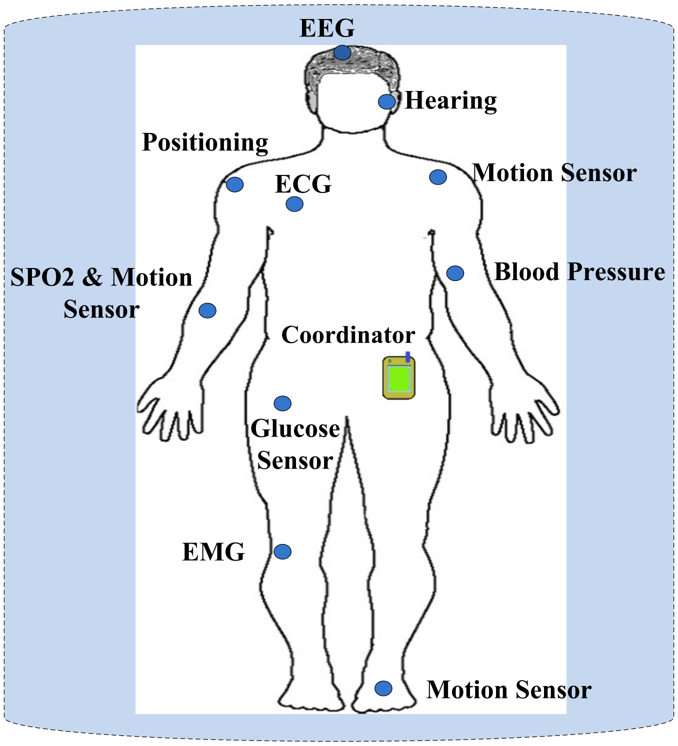
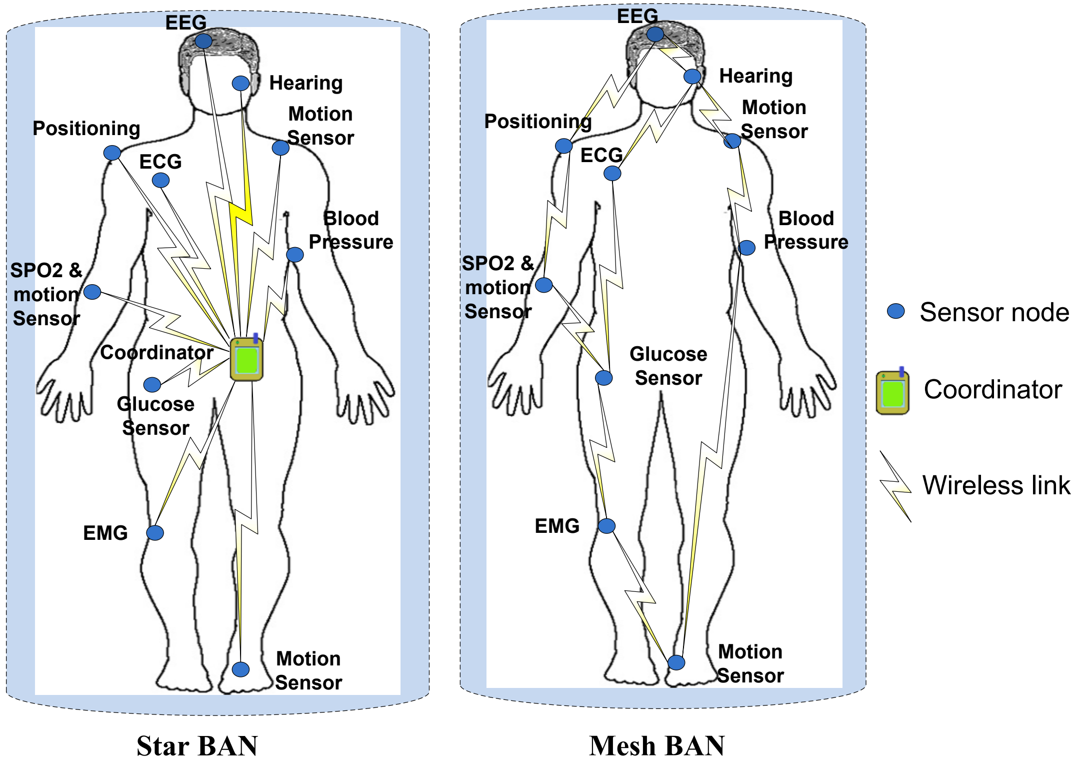


Figure 2-17: Location of wearable sensors on a human body



Star BAN

Mesh BAN

Figure 2-18: Star and Mesh based stand alone BAN

2.8.2. BAN Global Connectivity

The MobiHealth [29] and HealthService24 [30] projects developed the health service platform. A mobile phone is used as a mobile base station or gateway to collect the data from the sensors of the BAN. The BAN data from mobile base station is then sent to the healthcare servers. A feedback message is transmitted to the mobile device as an acknowledgement of correct data delivery. In home or hospital scenario, instead of using the mobile device, existing networks (WLAN, LAN) can be used to transfer the data from BAN to the other required locations.

2.8.3. BAN Connectivity via Ambient Sensors

In future, when all types of wireless fields (WPAN, WLAN, WMAN, and WWAN) are available reliably and the use of sensors become common place, then data from BAN can be transferred from sensor nodes to the other ambient sensors. The data from ambient sensors is then forwarded to the healthcare providers via Internet, as shown in Figure 2-19.

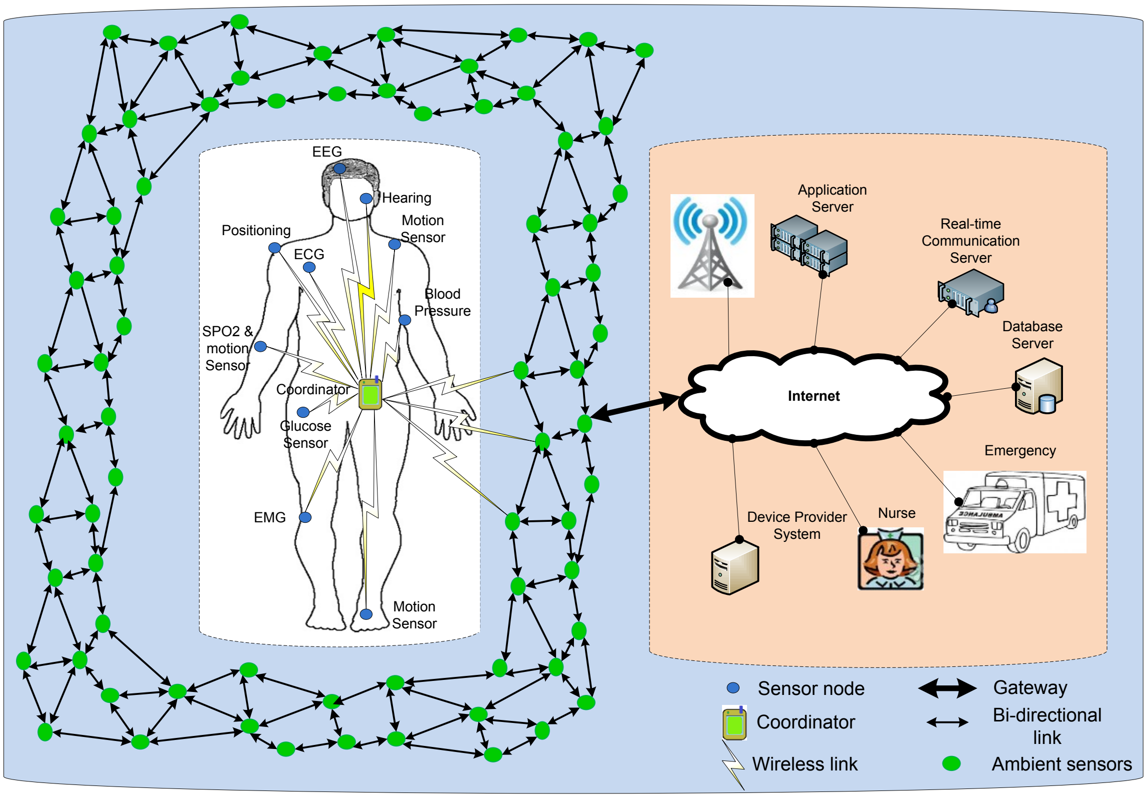


Figure 2-19: BAN connected to ambient sensor network

2.9. Summary

This chapter presented a brief overview of Body Area Networks (BANs). Section 1 highlighted the development process of BANs. As BAN is a subfield of WSN, Section 2 provided a comparison of BAN with WSN. Sections 3 and 4 discussed the components of BAN and its topologies, respectively. BAN faces unique technical challenges due to its use in, on, or around the human body. Section 5 discussed the requirements of BAN communication. The communication standard for BAN, IEEE 802.15.6, has been recently developed. Section 6 explained the development process of this standard. One of the major applications of BAN is the monitoring of patients in the healthcare field. Section 7 discussed the monitoring of chronic disease patients, elderly patients, and hospital patients. Section 8 provided a brief description of different BAN application environments.

CHAPTER 3

LITERATURE REVIEW

Various advanced and valuable state-of-the-art applications of Body Area Networks (BANs) help to enhance the patient's healthcare monitoring and their quality of life. The BAN devices are used to monitor the patients' health related concerns such as changes in blood pressure (BP), heart rate, or body temperature. One of the BAN features is to facilitate the physical mobility of the patient; this means now the patients are not required to stay in the hospital at all times. Routing protocols are required to route a patient's data towards the required destination even when a patient moves. Numerous routing protocols have been proposed for reliable and energy-efficient communication between sensor nodes [31, 32, 33]. The literature survey given in this chapter is divided into four sections. Section 3.1 provides the general overview of BAN routing protocols. BAN routing protocols are divided into four main classes based on the type of routing they employ: QoS based routing, thermal based routing, cluster based routing, and cross layer routing. Section 3.2 discusses the classification of BAN routing protocols and the routing protocols proposed in each class. A framework of the patient monitoring system is required for the implementation of the BAN routing protocol. Many frameworks for the patient monitoring system in the hospital environment have been suggested [34, 35, 36]. Section 3.3 deals with a detailed study of these patient monitoring systems. Our proposed ZK-BAN peering framework is discussed in Chapter 4. Section 3.4 provides the summary of the chapter.

3.1. BAN ROUTING PROTOCOLS

Routing is an issue for the sensor nodes due to the limited availability of resources including ultra-low computation power, lower memory, and reduced energy source. The Radio Frequency (RF) portion of the sensor nodes in BAN plays a major role in the consumption of energy. MAC protocols can reduce the energy consumption by controlling the duty cycle of the RF part. MAC protocols are also helpful to effectively control the other sources that are the cause of energy waste, such as collision, idle listening, overhearing, and packet overhead. In short, an ideal MAC protocol increases error-free data transmission, maximum throughput, medium access management, and minimizes transmission delay, thereby increasing network lifetime. Despite the fact that MAC protocols are helpful in resolving many problems, the issues of end-to-end packet delivery, logical-physical address mapping, frame fragmentation, addressing techniques, and route determination methods are not in the scope of MAC protocols. These issues can be more easily handled by the network layer. As a result, it is important to consider the network layer routing protocols to resolve these issues [37].

The challenges and features of BAN are different than WSN due to the specific needs of the wireless environment on the human body. The development of an efficient routing protocol in BAN requires more careful considerations than WSN. Some of the important factors to consider for the BAN routing protocols are their limited bandwidth, node and link heterogeneity, energy efficiency, coverage area, data aggregation, quality of service (QoS), transmit power, and mobile flexibility [37, 38].

The effects of fading, noise, and interference limit effective bandwidth. The bandwidth available for BAN also varies due to these effects. The routing protocol can have only limited network control. The placement of sensor nodes during the formation of BAN is possible by a manual process. The nodes are placed manually on the predefined locations of the body where the data transmission is minimally disturbed by noise or interference. Ideally each node sends its own data and forwards the data received from other nodes towards the required destination. But in case of BAN, the implanted sensors, due to their tiny size and limited energy resources, only send the data to the central node or coordinator. The coordinator and other wearable nodes are capable of multi-hop communication, which helps to route the data towards the desired destination. With the consideration of these facts, the routing protocol should be able to find and manage alternate routing paths in case of node failure.

Most of the nodes used in BAN are heterogeneous in terms of their capabilities including available energy, computational power, and communication capability. An example of heterogeneous nodes in BAN is the use of different wearable sensors to monitor body temperature, blood pressure, and other important vital signs of a patient. The link speeds of different implanted and wearable sensor nodes are not similar. The heterogeneity of the nodes should be considered by routing protocols.

The sensor nodes are placed on a human body that can be in motion. The node functionality may be affected due to mobility of the patient. This is because, the sensing capability of the mobile node can place increased energy demands on an application in different scenarios, e.g. vital sign monitoring of a mobile patient indoor in the hospital is different than a patient in the outside environment of the hospital. With the mobility of the nodes, the routing protocol should be able to provide a suitable solution for the reliable communication. Our proposed routing protocols are for the indoor hospital environment with the enhanced capability of handling mobile node communications.

Different data reporting techniques are used to send the data from the sensor nodes to the coordinator. These techniques, including time driven, event driven, query driven, and hybrid methods, are used as per application requirement. The time driven data reporting is used for continuous monitoring applications. Event driven is the technique when the sensor nodes send the data at the occurrence of a critical event; for example, priority is given to the sensors used for monitoring the heartbeat (ECG) at the time of heart attack. In query driven reporting, the data from a sensor node transfers when the node receives a query request from the base station. The query driven technique is used in BAN when the doctor wants to see the readings of a specific sensor. The patient’s monitoring in hospital environment mostly requires the hybrid method in which more than one technique is used; for example healthcare professionals are interested in observing the effected part of the patient body more frequently in addition to continuous monitoring of the other vital signs. The routing protocol should manage the communication properly by any data reporting methods. In this thesis, the proposed routing protocols will employ a hybrid method including both event driven and time driven data reporting.

The BAN sensor nodes are, typically, required to use an extremely low transmission power to reduce health concerns and avoid tissue heating [39]. The low transmit power restricts the BAN transmission range to three meters. Periodic updates exchanged by

Hello packets are used by routing protocols to maintain the routing table of the nodes. Additional energy used during the flooding process of these Hello packets is a major cause of node energy depletion. Therefore, it is an essential requirement that minimum energy consumption is required during the computations of the routing protocols. In this thesis, the proposed Hello protocol addresses these issues, and mechanisms are employed that overcome the problems of flooding. A detailed discussion of the Hello protocol is given in Chapter 5.

Each sensor node can sense a limited range of area due to energy and other constraints. The deployment of nodes on a body needs to be made in a way that can provide maximum coverage. The routing protocol needs to consider the location and available energy of the neighbor node for the selection of a route. This thesis proposes an Energy-efficient Peering Routing protocol (EPR) [6] that considers the geographical locations and energy levels of the neighbor nodes for route determination. A detailed description of EPR is provided in Chapter 5.

The data aggregation techniques, like duplicate suppression, median, and minima-maxima, are required in routing protocols to reduce redundant transmissions and minimize traffic load. The energy efficiency and throughput can also be improved by these techniques [38]. In this thesis, the proposed routing protocols provide the routes for intra-BAN communication which is from a BAN coordinator (similar to a cluster-head) of one BAN to the coordinator of another BAN. The BAN coordinator, which acts like a cluster-head, receives the data from the body nodes and then forwards these data towards the sink via other coordinators.

Quality of service is one of the important factors in BAN communication. The reliability of associated algorithms improves the successful delivery of critical reliability-sensitive data from sensor nodes to the base station. The routing protocols fulfill the QoS demand of different BAN applications by using the delay-control algorithms. These QoS-aware protocols help to monitor the patient’s health during a critical situation [7, 8]. In this thesis, three routing protocols based on delay-sensitive and reliability-sensitive data are proposed. The proposed QoS-aware routing protocols QPRD [7], QPRR [8], and ZEQoS are discussed in Chapters 6, 7, and 8 respectively.

3.2. Classifications of BAN Routing Protocols

Researchers proposed numerous BAN routing protocols during the last few years [37]. The BAN routing protocols can be divided into four major classes: QoS based routing, thermal based routing, cluster based routing, and cross-layer based routing. Figure 3-1 shows the classification of BAN routing protocols. The overview of the different BAN routing protocol classes is given below.

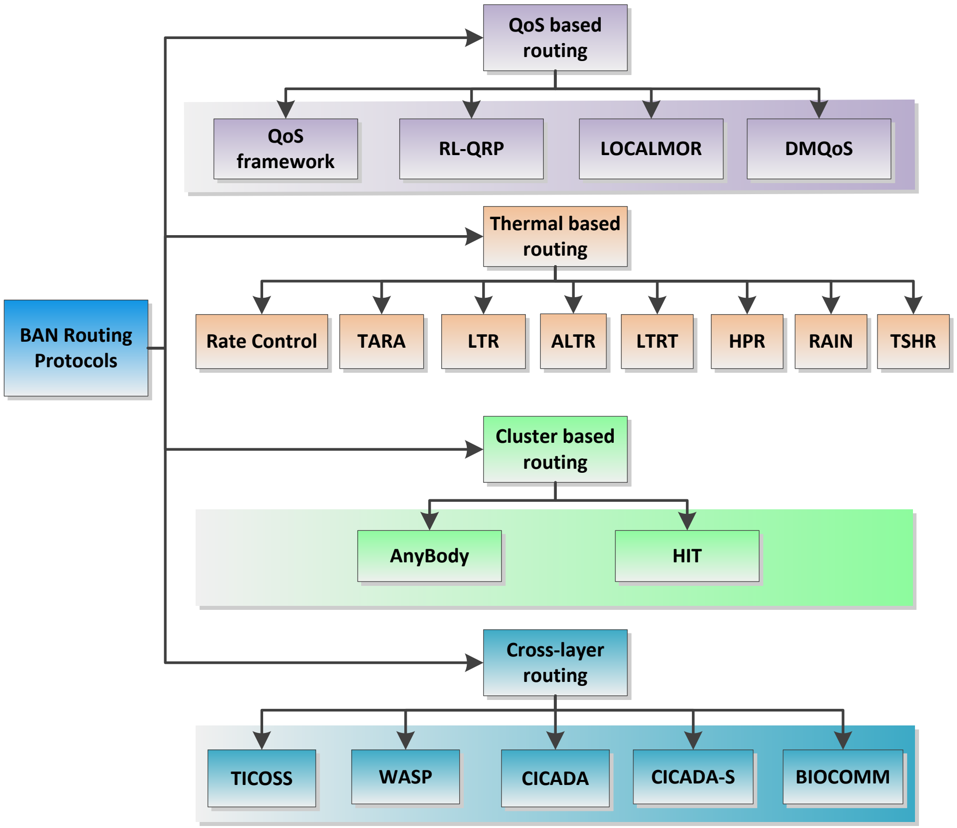


Figure 3-1: Classifications of BAN routing protocols

3.2.1. QoS Based Routing

The consideration of Quality of Service (QoS) is an important, but challenging task for the designers of BAN routing protocols. An ideal BAN routing protocol should provide an efficient and reliable path to route the patient's ordinary and critical data. The two important QoS routing protocols are reliability and delay-tolerant based protocols. The reliability-aware routing protocols ensure the delivery of maximum data packets to the destination. The transmission delay is not an issue for the reliability packets' delivery. For achieving the maximum throughput, data packets are sent on multiple redundant paths in some of the techniques used in reliability-aware protocols.

The delay-tolerant based routing protocols deal with the packets that are required to be delivered within a deadline. The route determination for the traffic of video streaming is one of the examples of this kind of routing. The end-to-end packet delay must be less than a specific delay; otherwise, the quality of overall data monitoring will be effected. Many routing protocols are proposed by researchers to address this issue. Researchers have proposed different QoS-aware based routing protocols. Some of the important QoS-aware routing protocols such as QoS-aware framework [31], RL-QRP [32], LOCALMOR [40], and DMQoS [33] are briefly discussed below.

3.2.1.1. QoS-aware Framework

In [31], a QoS-aware routing service framework for biomedical sensor networks is proposed based on a cross layered modular approach. The metrics considered for the determination of routes are wireless channel status, packet priority level, and sensor node's willingness to behave as a router. The proposed framework contains four main modules: an Application Programming Interfaces (APIs) module, a routing service module, a packet queuing and scheduling module, and a system information repository module. The architecture of the QoS-aware framework is shown in Figure 3-2.

The APIs module works as an interface between the user application and the routing service module. The components of APIs are QoS metrics selection, packet sending/receiving, packet priority level setting, and admission control & service level control. The QoS metrics are end-to-end delay, delivery ratio, and power consumption. The sensed data sent by user application for sink or other nodes is received by the packet sending/receiving component of APIs. These data packets contain destination ID, source ID, priority level, and payload. The data packets are received from the network layer. The payloads are forwarded to the user application for aggregation after separation from the data packets.

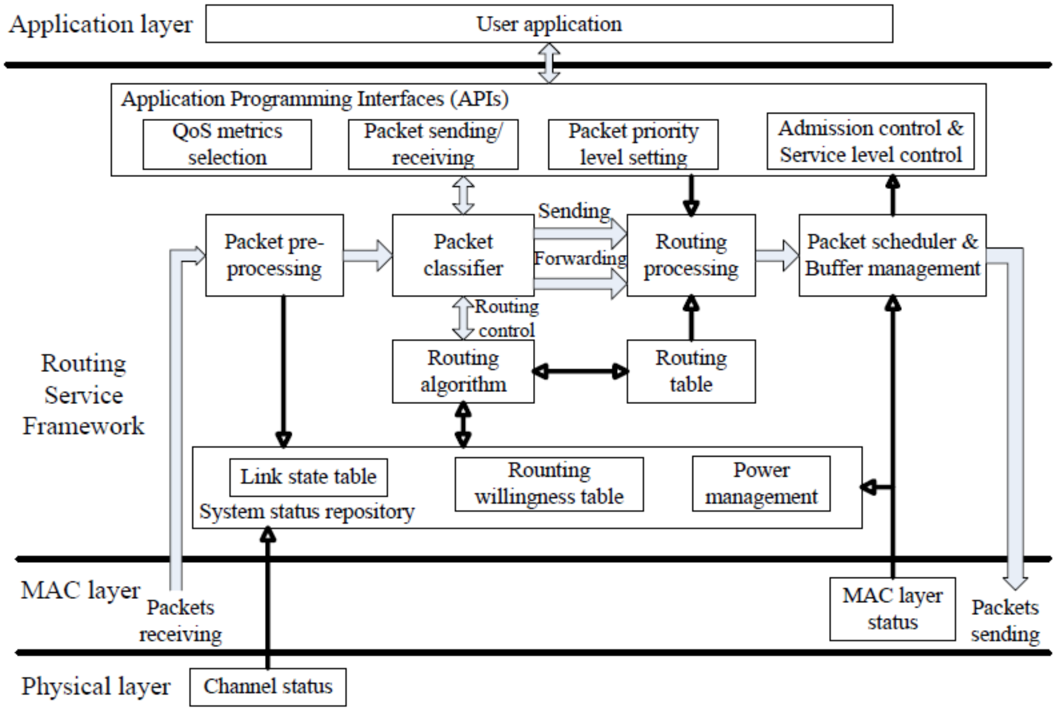


Figure 3-2: Architecture of QoS-aware routing service framework [31]

The routing service module is responsible for constructing a routing table with the help of the routing algorithm. The routing table is updated with the received neighbor's status information.

The data and control packets are divided into eight priority categories to ensure prioritized packet routing. The control packets are granted higher priorities than the data packets. Different types of data packets have different priority levels. For example, the priorities of long term monitoring, real-time data, and vital signs are 4, 5, and 6, respectively.

The system status information repository module consists of link state table and willingness table. The parameters used in link state table are link quality, end-to-end delay, communication bandwidth, and average packet delivery ratio. Each sensor node considers values of the link state, buffer status, and power supply level for adjusting its willingness level to become a router. The willingness table contains the information of all nodes which can perform the functions of a router.

The QoS-aware framework [31] is based on a modular technique that addresses QoS related issues for BAN. However, the suggested design is neither scalable nor adaptive to dynamic environment. The newer routing techniques that consider the geographic location of neighbor nodes prove very effective. The benefits of using geographic based routing include scalability, routing decisions based on neighborhood information, and being adaptive to dynamic environments. These protocols are also effective for mobile nodes. In this thesis, the proposed protocols use a similar modular approach, but with the additional enhancements of location and energy aware routing.

3.2.1.2. RL-QRP

RL-QRP [32] is a reinforcement learning based routing protocol with QoS support for biomedical sensor networks. The protocol focuses on two types of QoS requirements: packet delivery ratio and end-to-end delay. The machine learning approach used in this protocol uses optimal routing policies. These optimal routing policies can be found through experiences and rewards without the requirement of keeping precise network state information. The reinforcement learning based routing model is shown in Figure 3-3.

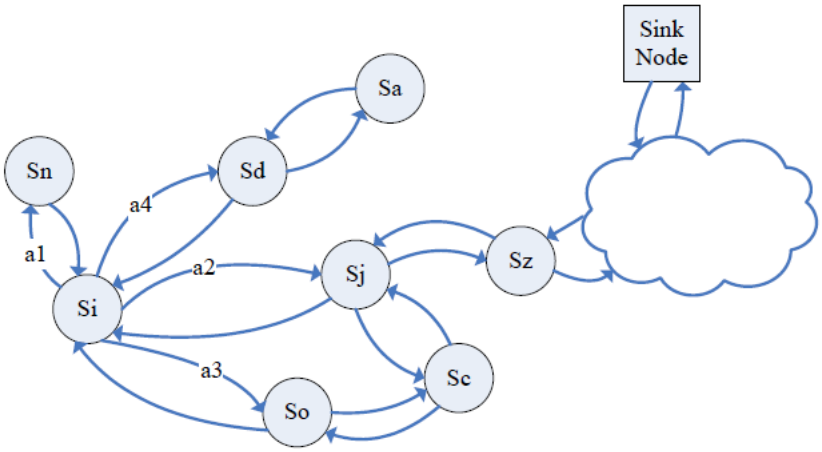


Figure 3-3: Reinforcement learning based routing model [32]

Each sensor node is considered as a state s∈ S. The corresponding action of the sensor node with its neighbor s' is represented by a $(P(s'|s,a) = 1)$. The quality values of the action “a” at state “s” is denoted by Q(s, a). These values are used in a routing table to find the appropriate path for the data packets.

RL-QRP [32] considers the neighborhood node's Q-values and location information for the determination of a QoS route. Energy is one of the major constraints in sensor nodes. The drawback of RL-QRP [32] is not considering energy at all. The proposed routing protocols, in this thesis, consider the residual energy and geographic location of the next hop node, which helps to improve the node life-time.

3.2.1.3. LOCALMOR

LOCALMOR [40] is a QoS based BAN routing protocol that relies on the traffic diversity of biomedical applications and guarantees differentiated routing, based on using QoS metrics. The three different QoS requirements 1) energy efficiency, 2) reliability, and 3) latency are considered in this protocol. The data traffic of biomedical applications is divided into four classes.

Regular Traffic: This traffic includes the regular measurements of the patient's vital signs e.g. temperature, heartbeat, blood pressure, etc. This type of traffic does not have any specific QoS requirement.

Reliability-sensitive Traffic: The maximum successful transmission of this traffic is required. The increased traffic load or delay in this kind of traffic can be acceptable.

Delay-sensitive Traffic: The requirement of this kind of traffic is to deliver the data packets within a defined delay. The example delay-sensitive traffic is video streaming.

Critical Traffic: The critical traffic needs maximum throughput with minimum delay. This kind of traffic is generated when the patient has a severe condition e.g. heart attack, brain stroke, etc.

A modular approach used in LOCALMOR consists of four modules: a power-efficiency module, a reliability-sensitive module, a delay-sensitive module, and a neighbor manager. The power efficiency module deals with the regular traffic and ensures the minimum energy consumption during the transmission of this traffic. The reliability-sensitive module calculates the path reliability and routes the sensitive packets towards the best path. The path required to route the delay-sensitive packets is the responsibility of the delay-sensitive module. Hello packets are used to update the neighbor's information in the neighbor table. The neighbor manager module is responsible to send/receive the Hello packets and manage the update information of neighbors. The system architecture of this QoS and geographical routing protocol is shown in Figure 3-4.

The data from body sensor nodes transfer to the primary and secondary sinks via routers. LOCALMOR [40] provides a QoS-aware modular solution for different packet types. However, all packets are blindly duplicated toward both the sinks. The primary and secondary sinks, which are connected to the high quality networks, transfer the data to the healthcare servers. Due to the sending of too many duplicate packets this protocol is non-scalable, and it increases the network traffic.

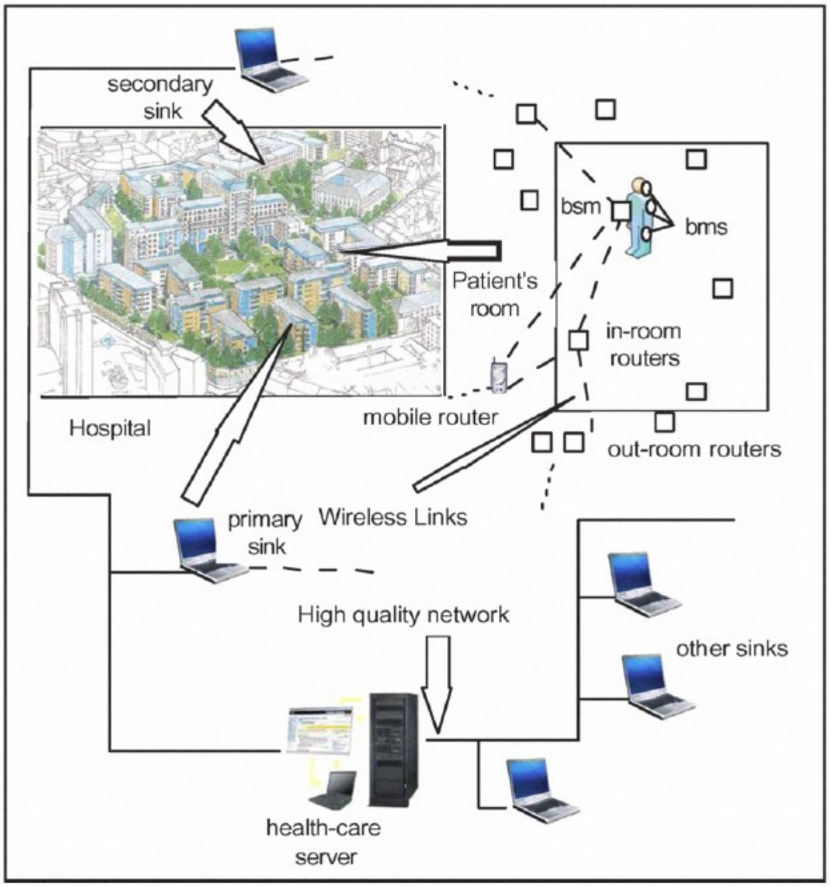
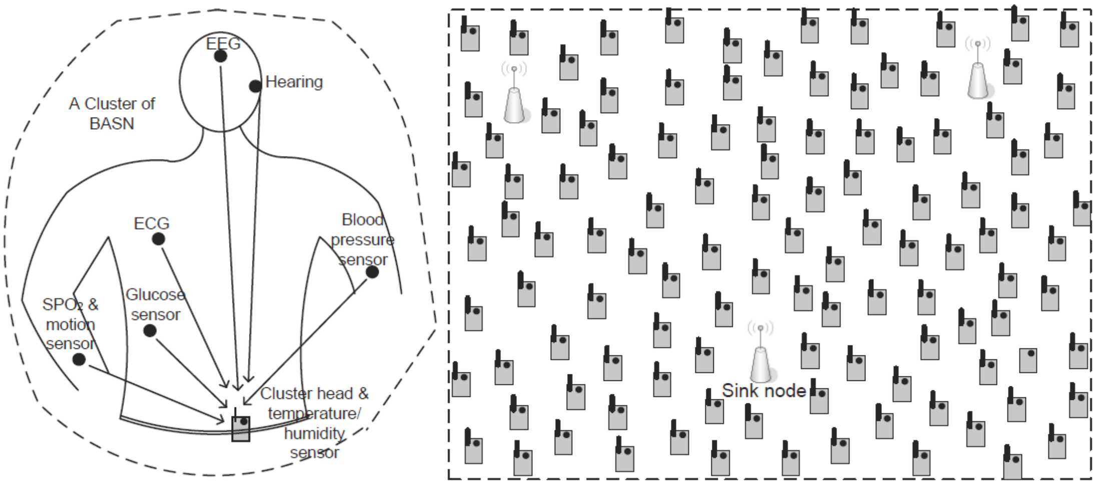


Figure 3-4: System architecture of QoS and geographical routing [40]

A Data-centric Multiobjective QoS-aware routing protocol (DMQoS) [33], discussed in the next section, outperforms the LOCALMOR [40]. The modular based architecture of DMQoS [33] provides the different routing modules to fulfill the QoS services for different packet classes. The purpose of proposed energy and QoS aware routing protocols, in this thesis, is the reliable and energy-efficient routing similar to LOCALMOR and DMQoS. In this thesis, the proposed routing protocols use a similar modular approach and same packet classification as discussed in LOCALMOR and DMQoS. However, the mechanism of Hello protocol and calculation used for end-to-end path delays and end-to-end path reliabilities improves throughput and reduces the network traffic load. The simulation results prove that our protocols (EPR, QPRD, QPRR, and ZEQoS) perform better than these protocols. The detailed discussion of the techniques used in our protocols is given in chapters 5, 6, 7, and 8.

3.2.1.4. DMQoS

DMQoS [33] is a data-centric multiobjective QoS-aware routing protocol for BAN. The assumptions of the network model used in this protocol are to have several nodes attached to the human body, and the data from these nodes are sent to a cluster-head or central node. The central node (also called the coordinator) has relatively high energy and better computing power. Several sink nodes are also available in the network. DMQoS protocol is designed for the communication between the coordinators. The assumed network model is shown in Figure 3-5.



(a) Cluster of a BASN

(b) Architecture of a Body Area Sensor Network (BASN)

Figure 3-5: Network model used by DMQoS [33]

It is also assumed that the location of the coordinators is known. Each node knows its coordinates and can determine the distance between itself and another node if the coordinates of that node are given. Hello packets are used to broadcast the information of a node to its neighbor nodes. After receiving the Hello packets, a node updates its routing table with the help of the information received by Hello packets. However, a disadvantage of the method used for broadcasting the Hello packets is that it increases network traffic which results in higher BAN energy consumption. As the next hops, DMQoS considers only BAN Coordinators (BANCs) in BAN communication, and every node broadcasts its Hello packets after a certain period of time. In a real BAN communication scenario, the next hop can be a different device like a Nursing Station Coordinator (NSC), a Medical Display Coordinator (MDC), or a BANC. The features and requirements of NSC, MDC, and BANC are different in a hospital environment. A detailed study of NSC, MDC, and BANC is given in Chapter 4. In this thesis, the proposed routing protocol EPR [6] discussed in Chapter 5 addresses these shortcomings, with the consideration of all possible next hop devices (i.e. NSC, MDCs, and BANCs) in the hospital environment, by controlling the broadcast of Hello packets. The mechanism of Hello protocol is discussed in Chapter 5.

Figure 3-6 shows the architecture of the DMQoS protocol. Like LOCALMOR [40], the modular approach used in DMQoS also handles four types of data packets: Ordinary Packets (OPs), Reliability-Sensitive Packets (RSPs), Delay-Sensitive Packets (DSPs), and Critical Packets (CPs). The CP carries the critical information in case of a criticial event like heart attack being detected. Due to the important information in CPs, the highest priority is given to these packets. The requirement of these CPs is to consider both delay and reliability constraints. However, DSPs contain information that has a delay requirement, but the loss of a part of the data is tolerable. The packets generated for video streaming are an example of DSPs. Next to be considered is a third category of packets, namely, RSPs. Maximum transmission rates are required for RSPs. RSP carries the data of vital sign monitoring or respiratory monitoring, which can tolerate some delay but require minimal packet loss. In this thesis, DSPs are given a higher priority than RSPs. The least priority is assigned to OPs. Applications that generate OPs include regular measurements of patient physiological parameters like blood pressure or body temperature.

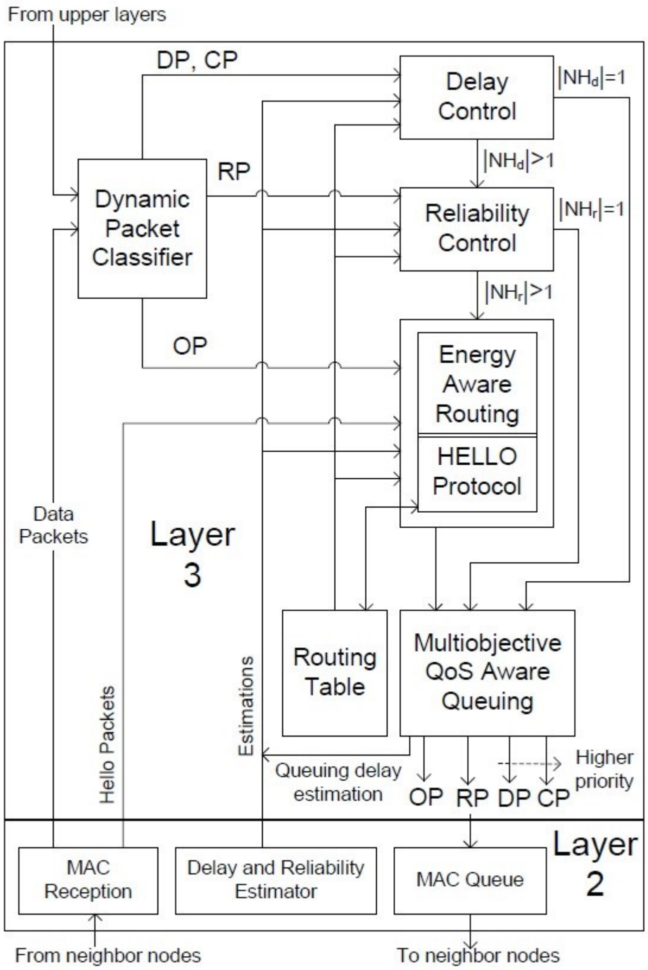


Figure 3-6: Data-centric multiobjective QoS-aware routing architecture [33]

DMQoS contains five modules: dynamic packet classifier, delay control, reliability control, energy-aware routing, and multiobjective QoS-aware queuing. The data packets from neighbor nodes or upper layers are received by the dynamic packet classifier. The dynamic packet classifier separates the different types of data packets. The packets are sent in a first-come-first-served pattern towards their respective modules. The delay-sensitive and critical packets are forwarded to the delay control module. The reliability-sensitive and ordinary packets are sent to the reliability control module and energy-aware module, respectively.

The use of multiobjective Lexicographic Optimization-based (LO) geographic forwarding techniques in DMQoS ensures a homogeneous energy dissipation rate for all routing nodes in the network. However, the choice of LO is not effective because LO ignores all less important objective functions when the most important one provides a unique solution.

The delay control module determines the next hop by considering the next hop device with lowest delay. The reliability control module finds the next hop which has the highest reliability. DMQoS employs a hop-by-hop approach to determine the next hop. The reliability and delay control modules introduced in [33] result in better performance than several state-of-the-art approaches [41, 32, 42, 43, 44, 45, 46, 47] in terms of lower bit error rates, traffic load, and operation energy overload. The disadvantage of this localized hop-by-hop routing proposed in DMQoS is that the source node depends only on the neighbor node's reliability or delay information. In case the neighbor node does not find any upstream next hop node with the required reliability or delay, the data packets are dropped. In this case, the packet does not reach the destination, but the source node assumes the packet is successfully received by the destination. Furthermore, the hop-by-hop approach used in DMQoS causes an increase in traffic load and the required end-to-end latency or reliability may not be guaranteed. In this thesis, the proposed routing protocol QPRD [7] discussed in Chapter 6 addresses these shortcomings by selecting the next hop device based on the lowest end-to-end path delay from the source node to the destination. The third proposed routing protocol QPRR [8] given in Chapter 7 explains the technique used for reliability-sensitive data. It is shown that the use of end-to-end path reliability and the choice of redundant paths result in higher throughput than DMQoS. The fourth proposed routing protocol, ZEQoS, provided in Chapter 8 discusses the technique used to handle the three kinds of data types (i.e. ordinary, delay-sensitive, and reliability-sensitive).

3.2.2. Thermal-aware Based Routing

The sensors used in BAN are either attached with the body (wearable) or placed inside the body (implant). Electrical and magnetic fields are generated during the wireless communication. The exposure to electromagnetic fields results a rise in temperature and an increase in radiation absorption for the patient. Even a slight heat rise can affect the operations of the sensitive organs; for example, a lack of blood flow due to heating disturbs the lens contract [48]. The coordinator requires a continuous data transmission from the sensors. The continuous data operations of body sensors cause the tissues' temperature to rise. The relationship between radiation and SAR is given in [49].

$$SAR = \frac{\sigma|E|^2}{\rho}\left(\frac{W}{kg}\right) \tag{3.1}$$

where Specific Absorption Rate (SAR) is the amount of absorbed radiation energy by the tissues per unit weight. E, ρ, and σ are the induce electric field, tissue density, and electrical conductivity of tissue, respectively. The acceptable range of SAR is defined by different countries. The experimental results illustrate that the tissues can be damaged when an SAR=8 Weight/kilogram in any gram of head or torso tissue is exposed to the radiations for 15 minutes [49].

The temperature rise and radiation absorption are the two major issues of wireless sensor communication in BAN [50]. A number of thermal-aware routing protocols are suggested to elude the effects of heating and radiation. The thermal-based routing protocols are Rate Control [51], TARA [52], LTR [53], ALTR [53], LTRT [54], HPR [55], RAIN [56], and TSHR [57].

The thermal-aware routing techniques are very important when dealing with the designing of inter-BAN routing protocols. The proposed routing protocols address the issues of intra-BAN communication and the real-time display of BAN data. However, thermal-aware techniques are not considered in this thesis.

3.2.3. Cluster Based Routing

Cluster based routing protocols have proved to be very suitable for minimizing the energy consumption and thereby maximizing the network lifetime. The entire network is divided into clusters in cluster based routing. Each cluster consists of cluster-head and member nodes. Different mechanisms are provided for the cluster-head selection. The communication between the member nodes and base station is only possible via the cluster-head. The functions of a cluster-head are to collect, aggregate, and forward the data from member nodes to the sink. The consumption of energy in the cluster-head is higher than in the member nodes due to the additional functions. LEACH [58] is the first cluster based routing protocol that randomly selects a cluster-head at regular intervals. The latest version of LEACH is AZR-LEACH [59]. The two important cluster based BAN routing protocols are Hybrid Indirect Transmissions (HIT) [60] and AnyBody [61].

3.2.3.1. HIT

Hybrid Indirect Transmissions (HIT) [60] is an improved version of LEACH. HIT is a hybrid of clusters and chains. The technique used in HIT improves the energy efficiency and increases the network lifetime. The following assumptions are considered in this protocol:

- Each node has a unique ID.
- The deployment of sensor nodes is randomly and uniformly distributed.
- All the nodes are static.
- All the nodes are able to communicate by CSMA/CD using a known power-level that is agreed upon apriori.
- The dimensions of the sensor field are given.
- All the nodes are able to calculate the distance from the originators of a particular CSMA/CD signal with the help of Received Signal Strength Indication (RSSI).

HIT works in rounds like LEACH, but it has seven phases in each round [60]:

1. **Cluster-Head Election:** One or more cluster-heads are elected.
2. **Cluster-Head Advertisement:** The cluster-head broadcasts its status to the network.
3. **Cluster Setup:** Clusters of the upstream and downstream relationship are formed.

4. **Blocking Set Computation:** Each node calculates its blocking set.

5. **Route Setup:** Sensors within a cluster form multi-hop routes to the cluster-head.

6. **MAC Schedule Creation:** A TDMA schedule is computed to allow for parallel transmissions.

7. **Data Transmission:** The sensed data is sent to the base station. Usually this is a long steady-state phase.

3.2.3.2. AnyBody

AnyBody [61] is a LEACH based self-organization protocol for BAN. The use of the clustering approach reduces the number of direct transmissions to the sink. Like LEACH, a cluster-head is randomly elected from each cluster at regular time intervals in order to spread the energy dissipation. AnyBody works in five steps:

1. **Neighbor Discovery:** Each node broadcasts a Hello packet to exchange its information with the neighbor nodes. A neighbor table is constructed with the information of two hop (next hop and one node after next hop) nodes.

2. **Density Calculation:** The values from the neighbor table are used to calculate the node densities. According to the formula given in [62], the node density is the ratio between the number of links and the number of nodes within the 2-hop neighborhood i.e.

$$\text{Node density} = (N_{\text{links}}/N_{\text{nodes}})$$

3. **Cluster-Head Contact:** A local node with the highest node density in the cluster is elected as the cluster-head.

4. **Backbone Setup:** After step 3, the independent clusters are set up. In this step they need to be interconnected. To avoid the problem of LEACH, supposing any cluster-head is within the range of the sink, AnyBody introduces the gateway nodes. The cluster-head from each cluster identifies the gateway nodes (GW) of

its cluster. The cluster-heads are now virtually connected with the sinks via gateway nodes.

5. **Routing Paths Setup:** Firstly, clusters are formed; secondly, cluster-heads are selected; and then gateway nodes are selected by cluster-heads. The routing is now done in a hierarchical way. The source node sends the data packets to the cluster-head. The cluster-head directs this data towards the gateway node. The gateway node then sends it to the next appropriate node, which is towards the sink.

Figure 3-7 shows the routing paths in the AnyBody protocol. The dotted blue lines show the paths within a cluster; whereas, orange lines illustrate the path between the cluster-heads of cluster A and cluster B. The routing technique used in AnyBody improves the performance of the LEACH protocol. However, the authors in AnyBody do not thoroughly examine the energy efficiency and reliability is also not considered.

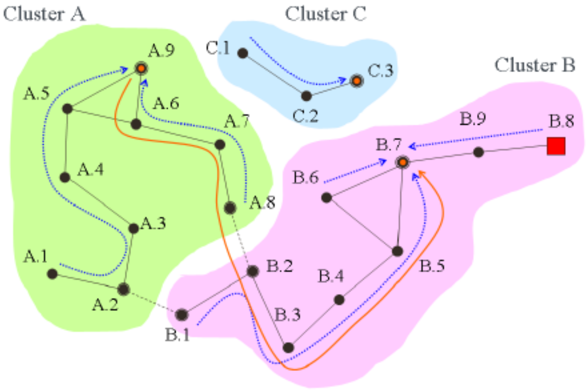


Figure 3-7: Routing in AnyBody [61]

Cluster based routing is a very effective routing technique in WSN; however, due to the specific needs of BAN, as discussed in Chapter 2, it is not very useful in a BAN environment. Hundreds to thousands of nodes are used in a WSN, which are usually in remote areas. The cluster-head needs to change after a certain period of time due to the energy constraints. On other hand, the total number of sensor nodes used in BAN is about 10-20, and the cluster-head is typically the same node all the time. In our proposed ZK-BAN peering framework, to be discussed in Chapter 4, the BAN coordinator works like a cluster-head all the time. Unlike WSN, a cluster-head election is not required in BAN. The proposed routing protocols follow the concepts of the Hello packet and the neighbor table provided by AnyBody, with enhancements.

3.2.4. Cross-layer Based Routing

Most of the existing solutions related to the data communication between different devices are based on the traditional seven OSI layered protocols approach. The design principles provided by the layered protocol approach has been widely used in the implementations and applications of the network systems since 1980. However, the lack of coordination between the layers restricts the performance of the overall architecture. The use of the layered approach in WSN or BAN is limited due to the characteristics of a sensor node such as time varying behavior, limited bandwidth, severe interference, and propagation environment. The limited resources of BAN are one of the major constraints to following the traditional layered approach. Cross-layer is a new approach with the modification of the traditional layered approach. The cross-layer design permits the collaboration, interaction, and optimization of two or more different layers while maintaining the functionalities of the original layers. The efficient communication of BAN due to the cross-layer based protocols attracts researchers to work on this area. Some of the cross-layer based BAN routing protocols are TICOSS [63], WASP [64], CICADA [65], CICADA-S [66], and BIOCOMM [67].

The cross-layer routing technique is commonly used in sensor networks. In this thesis, the traditional layered technique is used and was found to be more effective than cross-layering to get the best results when considering the QoS-aware routing.

The comparison of all the above stated routing protocols is given below in Table 3-1.

Table 3-1: Comparison of BAN routing protocols

Class	**Protocol Name**	**Year**	**Power efficient**	**Routing metrics**	**Methodology**	**Validation**
QoS based	QoS framework	2007	Medium	Hop count & link state	Deterministic	Castalia
	RL-QRP	2008				
	LOCALMOR	2009	Good	Energy, geometric distance & delay	Deterministic	GloMoSim
	DMQoS	2011	Good	Energy, geometric distance & delay	Deterministic	NS2
Thermal	Rate control	2005				
	TARA	2005	Very low	Body temperature	Deterministic	Matlab
	LTR	2006	Low	Body temperature & hop count	Greedy	C
	ALTR	2006	Low	Body temperature, hop count & routing delay	Greedy	C
	HPR	2006	Low	Body temperature, hop count & routing	Greedy	C
	LTRT	2007	Medium	Body temperature	Deterministic	Java
	RAIN	2008	Medium	Body temperature, hop count & routing	Problalistic	C++
	TSHR	2009				
Cluster based	HIT	2005	Good	Geometric distance	Deterministic	Theoretical
	AnyBody	2007	Medium	Node densities	Deterministic	
Cross layer	WASP	2006	Medium	hop count & routing	Deterministic	Nsclick
	TICOSS	2007	Good	hop count & routing	Deterministic	OMNeT++
	CICADA	2007	Medium	hop count & routing	Deterministic	Nsclick
	CICADA-S	2008				
	BIOCOMM	2009	Medium	Body temperature, hop count & delay	Deterministic	C++

3.3. Patient Monitoring Systems

Different systems based on BAN are proposed to monitor the patient data. Some of the important monitoring systems are ALARM-NET [34], AID-N [36], SMART [35], and CareNet [68]. A brief description of each system is given below.

3.3.1 ALARM-NET

ALARM-NET [34] combines the environmental and wearable sensors to provide a solution of continuous monitoring for assisted-living and residential monitoring. Heterogeneous devices are used with the integration of mobile body networks, wireless environmental sensors, and IP-networks as shown in Figure 3-8.

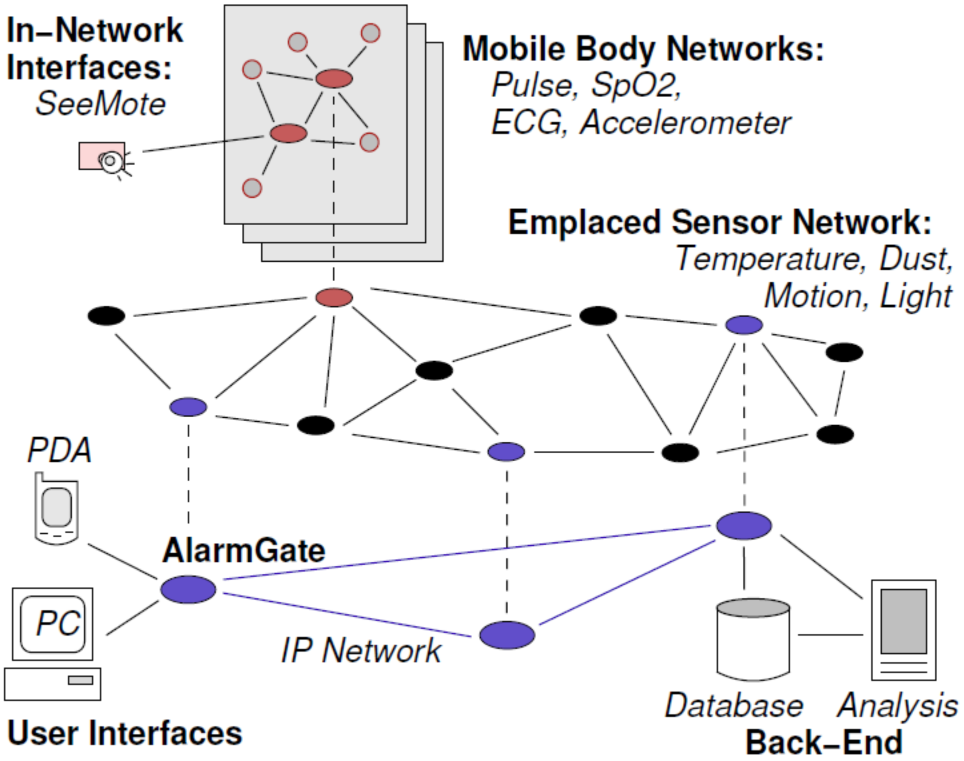


Figure 3-8: Architecture of ALARM-NET [34]

The mobile body networks contain the sensors used for getting the information of patient's vital signs such as pulse, ECG, accelerometers, etc. The environmental sensors monitor the changes in temperature, dust, light, etc. The IP-network is the combination of devices that are compatible with the IP addresses like PDA, PC, etc. The backbone nodes are used to connect the three networks. The patient's medical history is created with the implementation of WSN while maintaining the privacy of the patients. Approved healthcare professionals can monitor the patient's health and activity patterns related to the changes in healthcare needs. This automatic patient monitoring system reduces the labor costs and provides better efficiency.

The goal of ALARM-NET [34] is to collect and analyze BAN data. The general BAN architecture used in this project provides a mechanism to send the data to the central database for monitoring. However, the display of real-time BAN data in the hospital environment is not addressed. Traffic congestion and database server or link failure can cause delay or stop displaying the patient's data, which can affect the patient's treatment. In this thesis, the proposed ZK-BAN peering framework emphasizes on the real-time display of BAN data.

3.3.2 AID-N

The Advanced Health and Disaster Aid Network (AID-N) [36] suggests the applications targeted at dealing with mass casualty incidents. The inter-BAN communication concept of AID-N is similar to the mesh structure used in CodeBlue [69]; however, its application scenario is different. CodeBlue is a distributed wireless sensor network, which is used to sense and transmit the vital signs and geolocation data. Wireless repeaters are placed on the predefined emergency routes as a substitute for deploying APs on the wall. The proposed fault tolerant mechanism monitors the physiological characteristics of each patient. The AID-N system consists of three levels, as shown in Figure 3-9. The first layer contains the network between the patient's sensors. The functions of these lightweight sensors are limited due to their limited computational power and lower memory. The major functions of these sensors are to collect the data and send these data to the level two devices. The laptops and PDAs work as personal servers on the second level. These servers transfer the received data to level three devices via Internet. The central servers are located in level three. The authenticated users are allowed to logon to the central servers and analyze the critical information from the field [36].

AID-N [36] provides a patient monitoring system for an outdoor emergency situation. It uses the approach of first sending the data to the server, and then the authenticated users analyze the patient data from the servers. Link or server failure can stop the monitoring process. On the other hand, in this thesis, the proposed monitoring system provides a real-time monitoring system for an indoor (hospital) environment. The proposed ZK-BAN peering framework is discussed in Chapter 4.

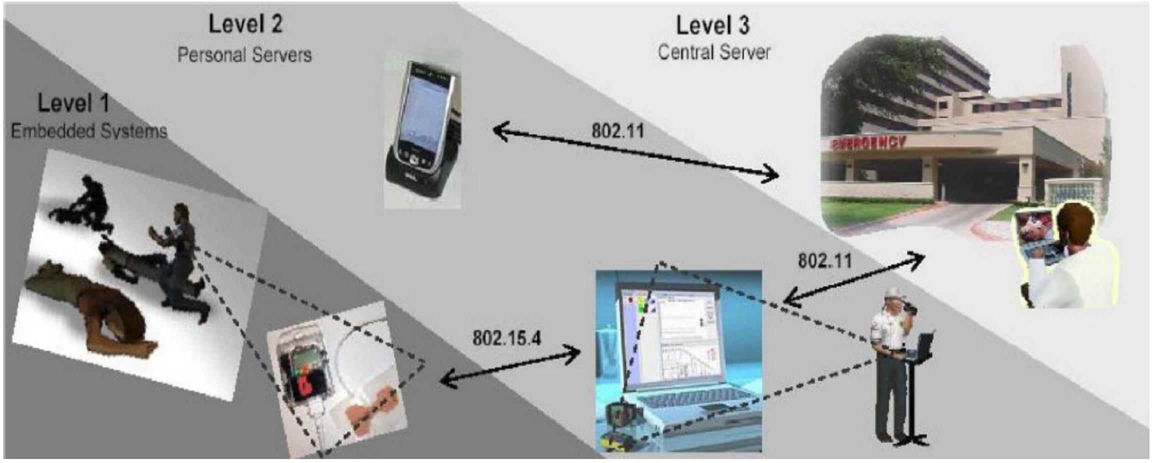


Figure 3-9: Architecture of AID-N system [36]

3.3.3 SMART

The Scalable Medical Alert and Response Technology (SMART) [35] suggests the framework of a patient monitoring system in the waiting areas of hospital emergency rooms. Healthcare professionals report many of cases in which it was found that the patient's health worsens rapidly during the waiting time in the emergency room. The patients in the emergency rooms need immediate care. The lack of resources and care can cause the lives of patients to be at risk. Figure 3-10 shows the components of the SMART system. The major components used are caregiver PDAs, location sensors, and patient PDAs with ECG and SpO2 sensors. The SMART System can be used to collect the patient's data from an emergency room, and transfer these data wirelessly to a central computer for collection and analysis of the data. The central computer performs the necessary operations to determine if a patient needs urgent support. An alert signal is issued if a particular patient's health deteriorates. This way, patients can receive treatment before the condition worsens [35].

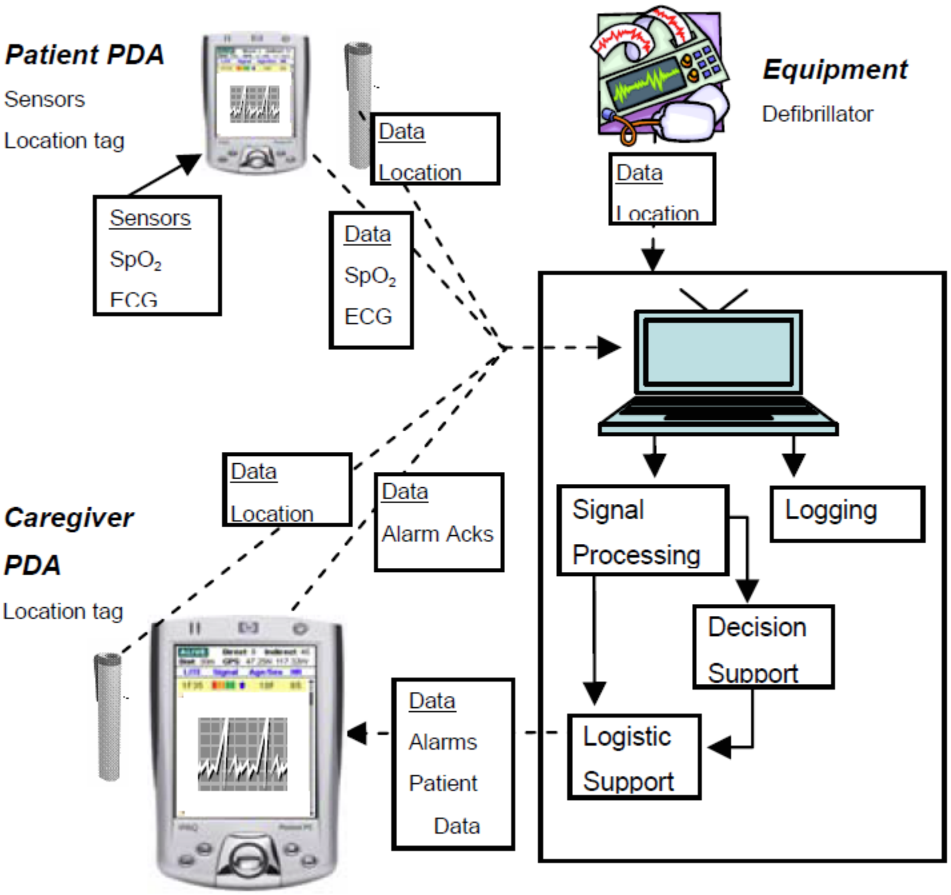


Figure 3-10: Architecture of SMART [35]

SMART [35] provides a monitoring system for the indoor hospital environment, but it only covers the emergency rooms. The patient data is displayed on the PDAs of the patient and healthcare professional. SMART [35] is not implementable in the areas like ICU or ORs, where highly sensitive equipment is used. This is due to the possible disturbances of high transmitting power devices, such as PDAs, on the highly sensitive hospital devices. IEEE 802.15.6 is the newly proposed standard for BAN. In accordance with the IEEE 802.15.6 standard, the transmission range of BAN is limited to a maximum of 3 meters. This distance was used in the standard, to limit ElectroMagnetic Interference (EMI) from BANs on other sensitive electronic equipment used in a hospital setting. In this thesis, the proposed ZK-BAN peering framework is compatible with BAN standard for transmission distance of about 3 meters. Unlike SMART [35], the use of centralized and distributed approaches in the proposed framework also ensures patient data privacy and reduces overall BAN energy consumption. The ZK-BAN peering framework is discussed in Chapter 4.

3.3.4 CareNet

The CareNet [68] project develops a 2-tier remote healthcare system to sense, collect, and transfer the data from the sensors to the web based servers. Figure 3-11 shows the whole process, which can be divided in four parts: sensors sense the data from the human body; the healthcare gateway routers collect these data via backbone routers; the data is then sent to the web based server; finally the data is available to the healthcare professionals for monitoring patient data.

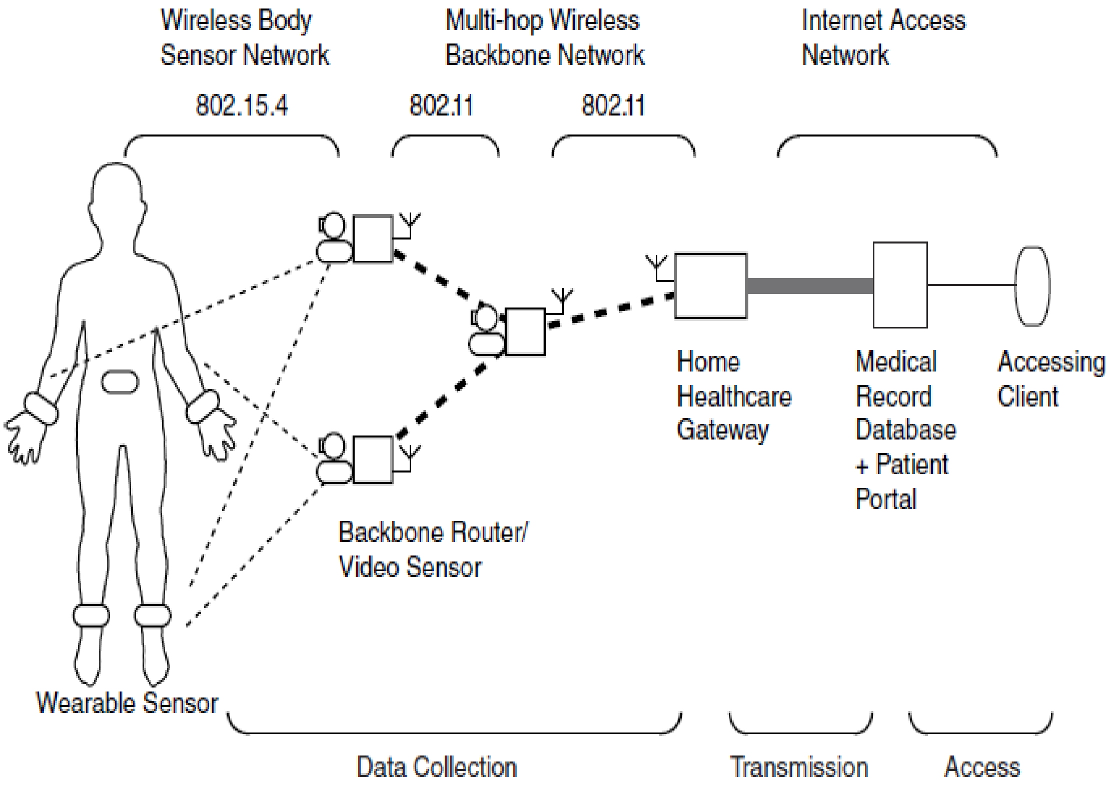

Wireless Body Sensor Network
Multi-hop Wireless Backbone Network
Internet Access Network
802.15.4
802.11
802.11
Home Healthcare Gateway
Medical Record Database + Patient Portal
Accessing Client
Backbone Router/ Video Sensor
Wearable Sensor
Data Collection
Transmission
Access

Figure 3-11: System architecture of CareNet [68]

The proposed integrated wireless environment offers features such as high reliability and performance, scalability, security, and integration with web based portal systems. The wireless standards IEEE 802.15.4 and IEEE 802.11 are used for body area sensor networks and multi-hop wireless backbone networks, respectively. High reliability is attained by using the 2-tier architecture. The use of a web portal with the patient's medical record system provides efficient access to the healthcare professionals [68].

In this thesis, the proposed ZK-BAN peering framework provides a similar architecture as given in CareNet [68]; however, the proposed ZK-BAN peering framework employs three tiers instead of two. CareNet [68] sends the patient's data first to the medical record database, and then the users are able to access the data via patient portal service of the server. The monitoring process fails in case of link or server failure. On the other hand, the proposed monitoring system provides a real-time monitoring system in the indoor (hospital) environment. The data from BAN is directly displayed on the medical display unit. The detailed discussion of the proposed ZK-BAN peering routing protocol can be found in Chapter 4.

3.4. SUMMARY

The literature overview of routing protocols and patient monitoring frameworks is provided in this chapter. The first section explains the general overview of the BAN routing protocol. Due to the specific requirements, BAN routing protocols face different challenges such as limited bandwidth, node and link heterogeneity, energy efficiency, data aggregation, and transmit power; they also require additional features such as low coverage area, QoS, and mobile flexibility as compared to WSN routing protocols. The BAN routing protocols are divided into different classes. The classification of BAN is discussed in the second section. The four types of BAN routing protocols, QoS based routing, thermal based routing, cluster based routing, and cross-layer based routing, are briefly discussed. A number of patient monitoring systems based on BAN are suggested. A brief discussion of important patient monitoring systems, ALARM-NET, AID-N, SMART and CareNet, is given in the third section. The goal of these monitoring systems is to collect and analyze BAN data. The general BAN architecture used in these projects provides a method to send the data to the central database for monitoring. Traffic congestion and database server or link failure can cause delay or stop displaying the patient's data, which can affect patient treatment. The proposed ZK-BAN peering framework provides a mechanism to display real-time BAN data in the hospital environment.

CHAPTER 4

ZK-BAN PEERING FRAMEWORK

The recent research in Body Area Networks (BANs) is focused on better utilizing system resources and making its communication more reliable, energy-efficient, and secure. This chapter proposes a novel BAN architectural framework for indoor hospital environments, with the intention that the proposed framework will help to improve BAN reliability and reduce network traffic load and energy consumption. This chapter of the thesis is organized as follows. Section 4.1 presents the proposed Zahoor Khan BAN (ZK-BAN) peering framework. Section 4.2 discusses the mathematical analysis of the framework. Sections 4.3 to 4.7 discuss the different possible communication scenarios of the proposed ZK-BAN peering framework. Sections 4.3, 4.4, 4.5, 4.6, and 4.7 explain the point-to-point communication, point-to-multipoint communication, peer unreachable, peer or communication type update, and NSC unreachable, respectively. Section 4.8 provides the chapter summary.

4.1. MOTIVATION

The monitoring of physiological and biochemical parameters in the human body using Wireless Sensor Networks (WSNs) is a challenging problem. As outlined in Chapter 2, challenges [70] include the high level of data reliability required for critical information, small size of implantable nodes, access to nodes due to difficult sensor replacement, context awareness due to the sensitivity of body physiology to EM radiation, power supply to implanted sensors, and patient mobility. These challenges are addressed in the new sub-field of WSNs known as Body Area Networks (BANs).

As outlined in Chapter 2, the IEEE 802.15 Task Group 6 is working to develop a low power and low frequency short range communication standard protocol for BANs. The goal is to optimize BAN operations related to the devices inside and outside of the human body but also to be compatible with other medical and consumer electronic devices [71]. As outlined in Chapter 3, several projects such as SMART [35], CareNet [68], AID-N [36], and ALARM-NET [72] have been proposed to monitor patients' data. The goal of these projects is to collect and analyze BAN data. The general BAN architecture used in these projects provides the mechanism to send the data to the central database for monitoring. However, these projects have not addressed the display in real-time of BAN data in an indoor hospital environment. Traffic congestion and database server or link failure can cause delays or stop displaying the patient's data, which can affect the patient's treatment. The mobility of the patient in the hospital may require a change of the dedicated display unit used to display patient data. In order to resolve these problems, a new BAN network architecture and four associated routing protocols for energy and QoS aware routing are proposed in this thesis.

The proposed ZK-BAN peering framework is presented in this chapter. The associated routing protocols are designed to display in real-time BAN data, avoid a fully centralized system, and discover the dedicated BAN data display unit dynamically. The proposed routing protocols are presented in Chapters 5, 6, 7, and 8 of this thesis. Two commonly used communication schemes are the centralized and distributed schemes. In the centralized scheme, all nodes are connected wirelessly with the central computer and the data sent from one node to another node must go through a central computer. Also, all information of the nodes is stored in the central computer. The centralized approach helps to ensure data privacy and increase control of node communication. However, a major disadvantage of the central approach is that of increased energy consumption and increasing network traffic, as even nodes that are closer to each other need to send their data through the central computer. The distributed approach resolves the problem of energy consumption as all the nodes can communicate directly with each other without sending data to the central computer. Every node contains the information on all other nodes; however, this compromises data privacy. In the distributed approach, the network traffic load is reduced due to not sending data to the central computer. In this thesis, both centralized and distributed approaches are used in the proposed hybrid communication scheme in order to make use of the best features of each approach. In the proposed hybrid communication architecture, only the central computer holds the information on BANs and display units, thereby improving data privacy, and helps better control BAN communication. However, the BAN data is displayed on the display unit in a distributed manner, thereby reducing network traffic load and more importantly helping to improve patient mobility. The display of BAN data does not need to go through the central computer. The BAN simply needs to get the peer information from a central computer and then sends the data directly to the associated display unit. If the patient moves to another room in the hospital, which is far from the central computer, the patient's BAN can still sends the data to the display unit without having to contact the central computer.

4.2. Proposed ZK-BAN Peering Framework

A general BAN communication framework is shown in Figure 4-1. It is a hierarchical model with three communication tiers [4]. In tier 1, the implanted and wearable sensors send data to the BAN Coordinator (BANC). The BANC is similar to a cluster-head in WSNs. The possible next hop of a BANC can be any device shown in tier 2. The communication devices with the exception of the BANC in tier 2 forward the BAN data to tier 3 communication devices.

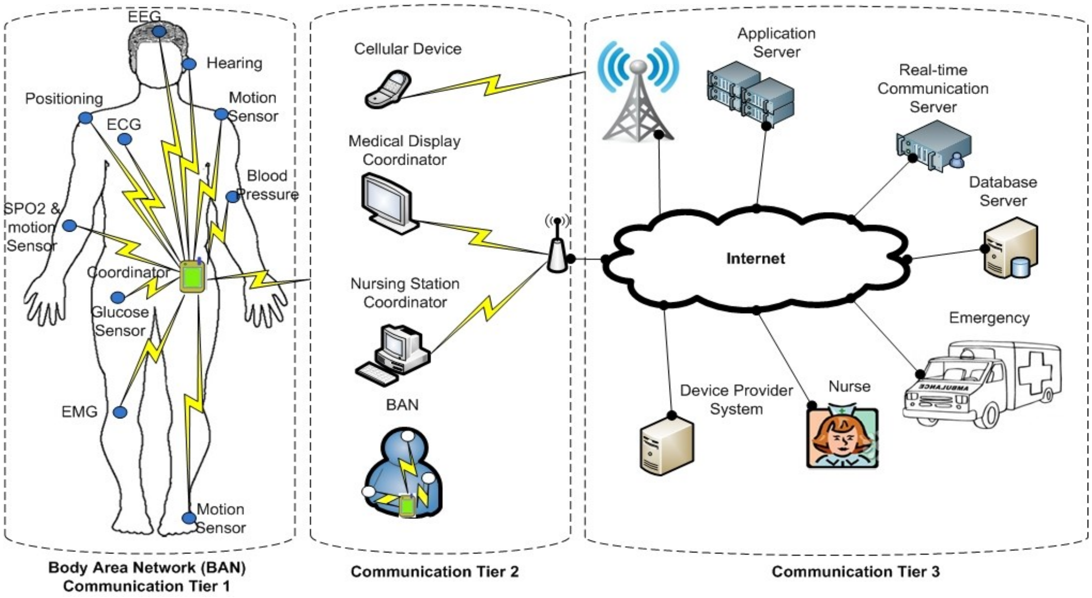


Figure 4-1: General BAN communication system

The two possible BAN communication scenarios are indoor and outdoor. The BAN in the hospital and at home is considered to be an indoor scenario. There are two kinds of communication types, point-to-point and point-to-multipoint. Point-to-point (p-p) means the BAN coordinator sends data packets to the next hop for a single destination. Point-to-multipoint (p-mp) is when the BANC sends data packets to the next hops for multiple destinations.

In hospital, it is possible to see the patient's data on more than one display unit. For example, the healthcare professional wants to see the data on his/her display device in addition to the patient's room display coordinator. So, both communication types, p-p and p-mp, are required in the indoor-hospital BAN scenario. On the other hand, only p-p communication is needed in the case of outdoor and indoor-home scenarios. Figure 4-2 explains the BAN communication scenarios and types.

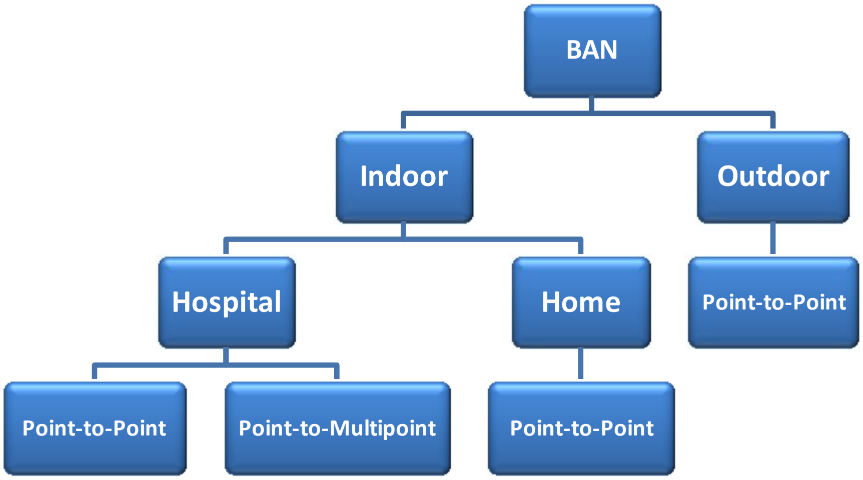


Figure 4-2: BAN communication scenarios and types

The indoor hospital BAN communication devices are categorized into three classes with respect to their available energy sources. Class 1 devices like Nursing Station Coordinators (NSCs) are directly connected to a power source. Class 2 devices like the Medical Display Coordinators (MDCs) use consumable batteries. BAN Coordinators (BANCs) with limited energy availability are considered as Class 3 devices. Two communication standards are used by Class 1 and 2 devices. IEEE 802.15.4 is used to communicate with the BANC and IEEE 802.11 for Wi-Fi. Table 4-1 shows the summary of the classes.

Table 4-1: Classification of devices in hospital environment

Class	Device name	Power Source	Channels	MAC protocol	Mobility
1	NSC	Directly Connected	2	IEEE 802.15.4 IEEE 802.11	No
2	MDC	Replaceable batteries	2	IEEE 802.15.4 IEEE 802.11	Yes
3	BANC	Limited energy available	1	IEEE 802.15.4	Yes

The NSC database contains information on all BANCs and MDCs in the ZK-BAN peering framework. Initially, BANCs search and then connect to the NSC. Each BANC receives the information about its respective peer from the NSC and then starts sending real-time BAN data to its respective peer MDC for display.

The requirements of BAN communication in an indoor-hospital environment are different from the outdoor or indoor-home BAN communication. In the hospital environment, typically, every patient's BAN needs a MDC for displaying the patient's data. Normally due to the size of the patient's room this device is placed within 3 meters of the BAN coordinator and, as already outlined in Chapter 2, this is the recommended communication distance in keeping with the proposed IEEE 802.15.6 BAN standard. For example, when a patient comes to the hospital's Emergency Room (ER) the BAN data is displayed on the MDC of the ER. Thereafter the patient may be transferred to the Operation Room (OR), Patient Room (PR), or Intensive Care Unit (ICU) for further treatment. The BAN data is then required to be displayed on the new MDC. As there are many MDCs in the hospital we need a mechanism to display in real-time BAN data on the MDC dedicated to the patient. For this, we propose a hybrid peering method. In this method the BAN will be peered with a display device (MDC). The BAN communication has two modes: centralized and distributed.

4.2.1. Centralized Mode

In the centralized mode, the BANC connects to the Nursing Station Coordinator (NSC) to obtain peering information. A logical diagram of the centralized mode is shown in Figure 4-3. Typically, when a patient is admitted to a hospital for care, the nurse enters patient information at the registration desk. It is assumed that in the ZK-BAN framework, the nurses will enter the information required by the NSC table. Such information in the proposed ZK-BAN framework includes BAN ID, communication type, and peer(s) ID as shown in Table 4-2. In the Nova Scotia healthcare system, MSI, the health card number of each patient is unique. This number can be used as the BAN ID. The nurse can simply check if the patient data needs to be displayed on one or more than one display units.

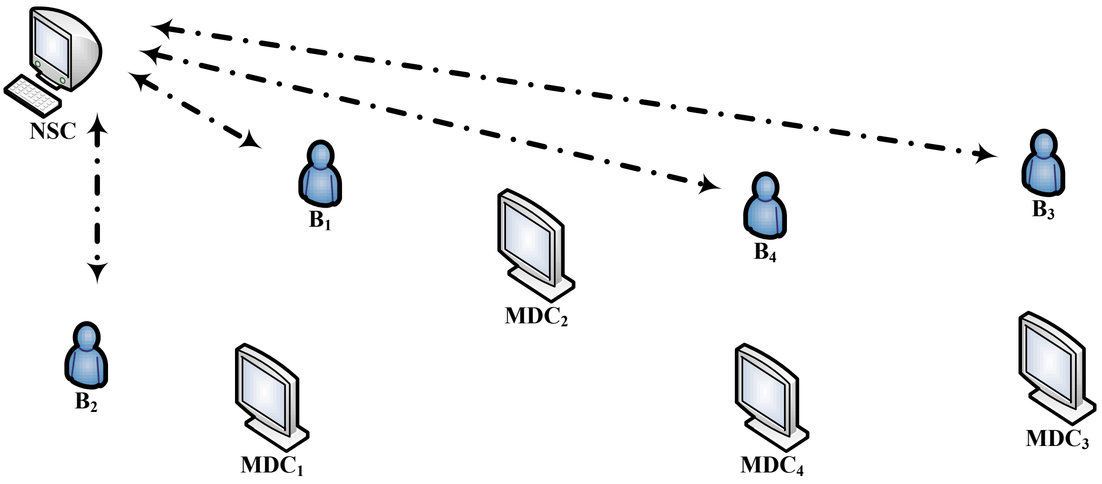


Figure 4-3: Centralized mode (logical diagram)

For communication type, p-p and p-mp are selected for single and multiple display units respectively. Peer IDs are the unique identification numbers of the MDCs. The nurse must enter the peer(s) ID, as per the selection of communication type. Both examples with p-mp and p-p are given in Table 4-2. The communication type for B_1-ID is p-mp. The peers ID are MDC_1-ID and MDC_2-ID. In the second record, the communication type and peer ID are p-p and MDC_3-ID respectively.

Table 4-2: Nursing Station Computer (NSC) table

BAN ID	Communication Type	Peer(s) ID
B_1-ID	**p-mp**	**MDC_1-ID, MDC_2-ID**
B_2-ID	**p-p**	**MDC_3-ID**
………	………….	………….

The information of all BANCs and MDCs are stored in the NSC. The use of the centralized mode helps to ensure information privacy and provide better controls of the BANs and MDCs. However, the additional energy consumption of the nodes, which are far away from the NSC, is a drawback of the centralized mode.

4.2.2. Distributed Mode

In the distributed mode, the BANCs discover and send data to their respective peers. Lower energy consumption and ease of patient mobility (and hence of the BAN devices) are the features of the distributed mode. Figure 4-4 shows that the BANCs B_1 and B_2 display the data on MDC_1 and MDC_2 respectively. The BANC B_4 is connected to two display coordinators (MDC_3 and MDC_4) using point-to-multipoint communication.

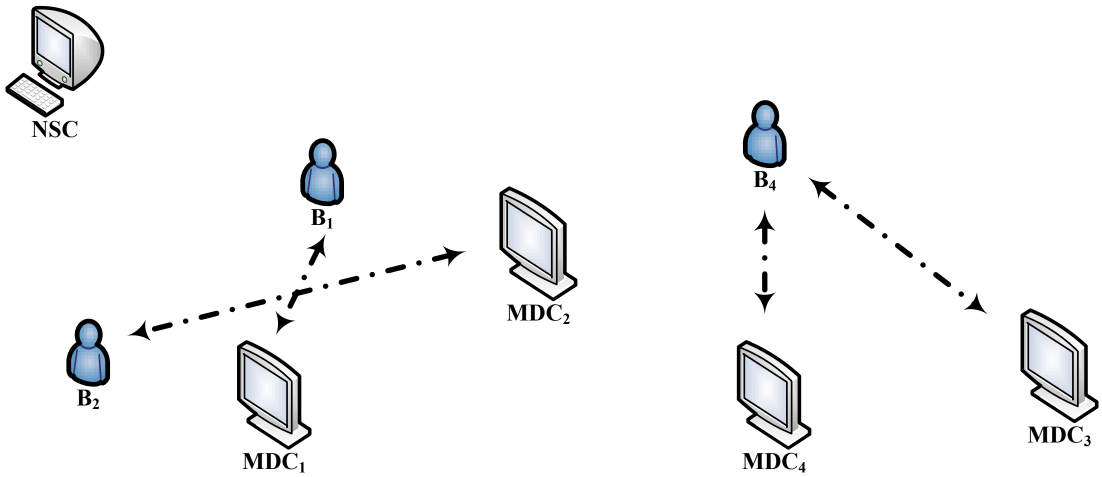


Figure 4-4: Distributed mode (logical diagram)

4.3. Mathematical Analysis

In this section, a mathematical analysis of the proposed ZK-BAN peering framework is given. Graph theory concepts are used to convert the hospital sensor network into an adjacency matrix (A). The power of the adjacency matrix of a graph provides one of the important features as stated in the below theorem.

"*If A is the adjacency matrix of a graph G (with vertices v_1,…, v_n), the (i, j)-entry of A^r represents the number of distinct r-walks from vertex v_i to vertex v_j in the graph.*"

The ZK-BAN peering framework can be modeled by a graph *G* as given below:

$$G = (V,\ E) \tag{4-1}$$

The vertices (V) is the set of all *n* (v_1, v_2, v_3,….., v_n) nodes.

$$V = \{v_1, v_2, v_3, \dots\dots, v_n\} \tag{4-2}$$

The order of the graph is given below:

$$|V| = n \tag{4-3}$$

In the proposed ZK-BAN framework, all the nodes in the hospital are divided into three types (T_1, T_2, and T_3). Type 1 devices are the nodes connected directly with the power source. The Nursing Station Coordinator (NSC) is a type 1 device. Type 2 devices are the Medical Display Coordinators (MDCs) in which the batteries are replaceable. The BAN devices are considered type 3 devices, which consist of limited energy availability.

$$\mathrm{V} = \mathrm{T}_1 \cup \mathrm{T}_2 \cup \mathrm{T}_3 \tag{4-4}$$

where

$$\mathrm{T}_1 = \{\mathrm{NSC}\}$$

$$\mathrm{T}_2 = \{\mathrm{MDC}_1, \mathrm{MDC}_2, \mathrm{MDC}_3, \dots \dots, \mathrm{MDC}_\mathrm{m} \}$$

$$\mathrm{T}_3 = \left\{\mathrm{B}_1, \mathrm{B}_2, \mathrm{B}_3, \dots \dots, \mathrm{B}_\mathrm{p} \right\}$$

m and p represent the total number of type 2 and type 3 nodes in the hospital environment used for the ZK-BAN peering framework.

E is the set of all bidirectional wireless links between the nodes.

E= {e_1, e_2, …., e_s} ; e is an edge between the two nodes.

e= (v_i, v_j) ЄE; v_i and v_j represent the two nodes *i* and *j* respectively.

The size of the graph is |E|=s; s is the total number of edges.

The use of matrices helps to explain the ZK-BAN peering framework in a better way, which helps in route determination. The wireless link between the two nodes *i* and *j* can be represented by an adjacency matrix a_{ij}:

$$a_{ij} = \begin{cases} 1, & if\ there\ is\ a\ wireless\ link\ from\ v_i\ to\ v_j \\ 0, & otherwise \end{cases} \tag{4-5}$$

The graph *G* as given in Equation 4-1 has *n* nodes. The adjacency matrix of graph *G* is defined as:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ a_{41} & a_{42} & a_{43} & \cdots & a_{4n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix} \tag{4-6}$$

The notation a_{nn} represents the edge from the node n to itself. A wireless link of a node to itself is not practical, so:

$$a_{11} = a_{22} = a_{33} \ldots\ldots = a_{nn} = 0 \tag{4-7}$$

By using the values from Equation 4-7 in Equation 4-6 we get

$$A = \begin{bmatrix} 0 & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & 0 & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & 0 & \cdots & a_{3n} \\ a_{41} & a_{42} & a_{43} & \cdots & a_{4n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & 0 \end{bmatrix} \tag{4-8}$$

The wireless links between the two nodes are always bidirectional, so the graph G in Equation 4-1 is an undirected graph. The adjacency matrix (A) of the undirected graph is equal to the transpose (A^t) of the graph.

$$A = A^t \tag{4-9}$$

The above analysis can be explained by considering a simple example of nine nodes in the hospital environment. The devices used in this example are four BANs, four MDCs, and one NSC, as shown in Figure 4-5. The nodes shown in Figure 4.5 are supposed to be 3 meters apart vertically. For example, nodes NSC, B_1, B_2, and MDC_1 are in the range of 3 meters and can have different communication between each other.The area considered is 12 by 12 square meters. The standard room size per patient in a hospital is 3 by 3 square meters.

$$\mathrm{T_1} = \{\mathrm{NSC}\}$$

$$\mathrm{T_2} = \{\mathrm{MDC_1}, \mathrm{MDC_2}, \mathrm{MDC_3}, \mathrm{MDC_4}\}$$

$$\mathrm{T_3} = \{\mathrm{B_1}, \mathrm{B_2}, \mathrm{B_3}, \mathrm{B_4}\}$$

By using the values of T_1, T_2, and T_3 in Equation 4-4, we get:

$$\mathrm{V} = \{\mathrm{NSC}\} \cup \{\mathrm{MDC_1}, \mathrm{MDC_2}, \mathrm{MDC_3}, \mathrm{MDC_4}\} \cup \{\mathrm{B_1}, \mathrm{B_2}, \mathrm{B_3}, \mathrm{B_4}\}$$

$$\mathrm{V} = \{\mathrm{NSC}, \mathrm{MDC_1}, \mathrm{MDC_2}, \mathrm{MDC_3}, \mathrm{MDC_4}, \mathrm{B_1}, \mathrm{B_2}, \mathrm{B_3}, \mathrm{B_4}\}$$

The order of the graph is:

$$|V| = 9$$

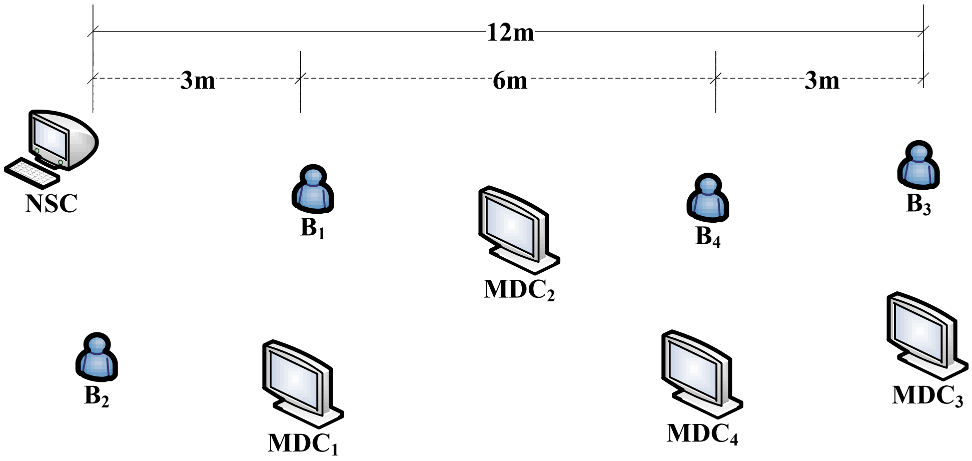


Figure 4-5: ZK-BAN peering framework example

The adjacency matrix given in Equation 4-8 can be re-written with respect to the scenario given in Figure 4-5, we get Equation 4-10

$$A = \begin{bmatrix} 0 & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} & a_{17} & a_{18} & a_{19} \\ a_{21} & 0 & a_{23} & a_{24} & a_{25} & a_{26} & a_{27} & a_{28} & a_{29} \\ a_{31} & a_{32} & 0 & a_{34} & a_{35} & a_{36} & a_{37} & a_{38} & a_{39} \\ a_{41} & a_{42} & a_{43} & 0 & a_{45} & a_{46} & a_{47} & a_{48} & a_{49} \\ a_{51} & a_{52} & a_{53} & a_{54} & 0 & a_{56} & a_{57} & a_{58} & a_{59} \\ a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & 0 & a_{67} & a_{68} & a_{69} \\ a_{71} & a_{72} & a_{73} & a_{74} & a_{75} & a_{76} & 0 & a_{78} & a_{79} \\ a_{81} & a_{82} & a_{83} & a_{84} & a_{85} & a_{86} & a_{87} & 0 & a_{89} \\ a_{91} & a_{92} & a_{93} & a_{94} & a_{95} & a_{96} & a_{97} & a_{98} & 0 \end{bmatrix} \quad (4\text{-}10)$$

where

1 = NSC; 2 = MDC$_1$; 3 = MDC$_2$; 4 = MDC$_3$; 5 = MDC$_4$; 6 = B$_1$; 7 = B$_2$; 8 = B$_3$; 9 = B$_4$

The proposed transmission range of BAN communication according to the IEEE 802.15.6 standard is 3 meters. The size of a typical patient room is 3 * 3 square meters in a hospital environment. We assume that each patient room consists of a type 2 device (MDC) and that the MDCs in the rooms are placed in a way where the distance between any two MDCs is $\leq$ 3 meters. The typical hospital scenarios are discussed later in Figure 7-9 and Figure 7-10 of Chapter 7. The wireless links between the nodes are possible if the nodes are within the range of 3 meters. As defined in Equation 4-5, the value ‘1’ is used if there is a wireless link; otherwise ‘0’ is used. The adjacency matrix (A) in Equation 4-10 is updated by placing the values of wireless links between the nodes, as shown in Figure 4-5. Equation 4-11 shows the adjacency matrix (A) of the example. Each row in the matrix A represents the wireless link availability of a node to other nodes. For example, the first row shows that the device NSC has a direct link with the devices MDC$_2$, B$_1$, and B$_2$.

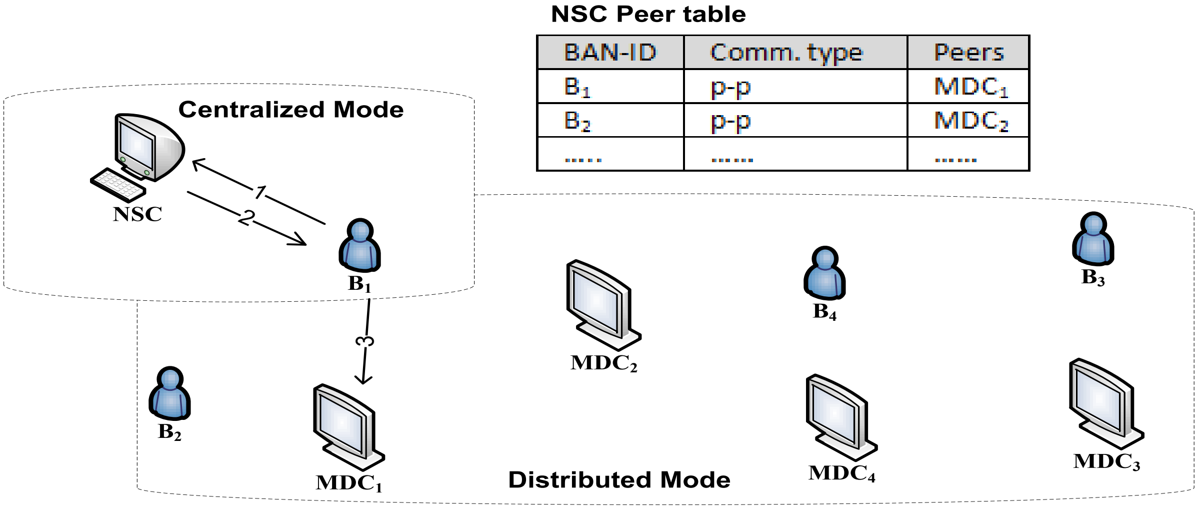
The adjacency matrix given in Equation 4-11 is used to find the number of different paths using two edges between the vertices of the graph G.

$$\begin{array}{cccccccccc}
 & \mathbf{NSC} & \mathbf{MDC_1} & \mathbf{MDC_2} & \mathbf{MDC_3} & \mathbf{MDC_4} & \mathbf{B_1} & \mathbf{B_2} & \mathbf{B_3} & \mathbf{B_4} \\
\mathbf{NSC} & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
\mathbf{MDC_1} & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
\mathbf{MDC_2} & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
\mathbf{MDC_3} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
\mathbf{MDC_4} & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\
\mathbf{B_1} & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
\mathbf{B_2} & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
\mathbf{B_3} & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\
\mathbf{B_4} & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0
\end{array} \tag{4-11}$$

4.4. Point-to-point Communication in the ZK-BAN

In the hospital, initially the BAN communication is in centralized mode, and no data is displayed on any MDC. The BAN coordinator will connect to the Nursing Station Coordinator (NSC). The purpose of this connection is to obtain information about its peer (MDC) and communication type (p-p or p-mp).

The NSC is a centralized system that holds the peering and communication information in its NSC peer table for all BANs in the hospital. By storing this information on the NSC, the privacy of patient data is ensured. The nurse/operator is responsible for entering the peering (MDC) and communication type (p-p or p-mp) information of BAN on the NSC. After getting the peering information from the NSC, the BAN coordinator will immediately switch to a distributed mode and will start searching for its peer. The detailed discussion of peer discovery is given in Chapter 5. After discovering its peer MDC, the data will be displayed on the MDC. Each MDC is also connected with a wireless access point that can transfer patient data to tier 3 communication devices. As the communication type is p-p, the BAN coordinator sends data packets to its respective peer. Three steps shown in Figure 4-6 are given below.



NSC Peer table

BAN-ID	Comm. type	Peers
B_1	p-p	MDC_1
B_2	p-p	MDC_2
.....	......	......

Figure 4-6: ZK-BAN point-to-point communications

1. B_1 → NSC: Check communication type (p-p or p-mp) and peer(s)?
2. NSC → B_1: Look at NSC peer table and send info to B_1
3. B_1 → MDC_1: Look at B_1 routing table and send data to MDC_1

BANC B_1 in steps 1 and 2 gets the information from NSC about its peer (i.e. MDC_1) and communication type (i.e. p-p). In step 3, the BAN coordinator will discover MDC_1 and display the data on it. The data from B_1 will always be displayed on MDC_1 even when B_1 moves away from MDC_1. The sequence diagram of this process is shown in Figure 4-7.

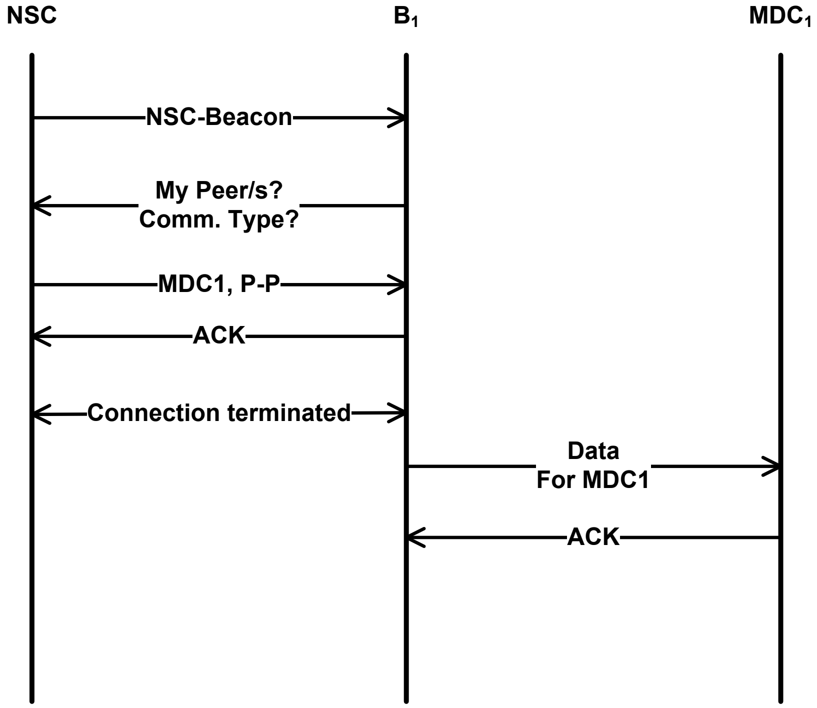
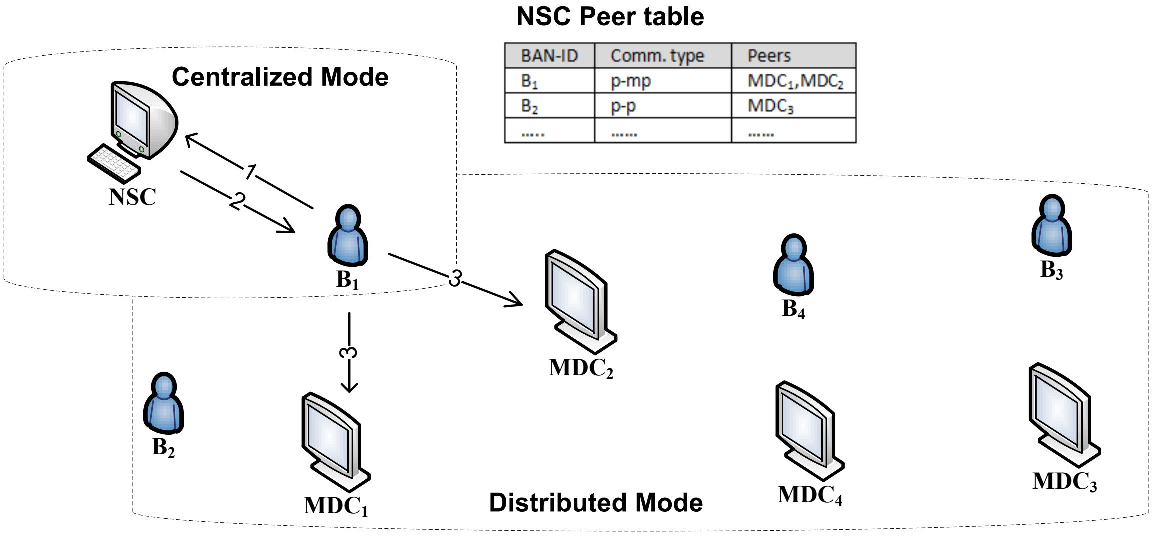


Figure 4-7: Sequence diagram of point-to-point communication in ZK-BAN

4.5. Point-to-multipoint Communication in the ZK-BAN

In some cases, we need to display the BAN data on more than one display unit. Such a situation may arise, e.g., when a doctor wants to see the patient's data on his/her office MDC. In such situations, what is needed is a p-mp as communication type (e.g., MDC_1 & MDC_2 are a BAN peers). The operator should enter these changes into the NSC Table. The B_1 will send two copies of data packets, one for MDC_1 and the other for MDC_2. The scenario below explains this p-mp situation clearly. B_1 will first contact the NSC and get information about its peers and communication type. It will then send two copies of data packets for both MDC_1 and MDC_2 after searching for these peers. This process can be summarized in three steps as shown in Figure 4-8.

1. B_1 → NSC: Check communication type (p-p or p-mp) and peer(s)?
2. NSC → B_1: Look at NSC peer table and send info to B_1
3. B_1 → MDC_1: Look B_1 routing table (construction and update of routing table is given in Chapter 5) and send data to MDC_1 and MDC_2



BAN-ID	Comm. type	Peers
B_1	p-mp	MDC_1,MDC_2
B_2	p-p	MDC_3
…..	…..	…..

Figure 4-8: ZK-BAN point-to-multipoint communications

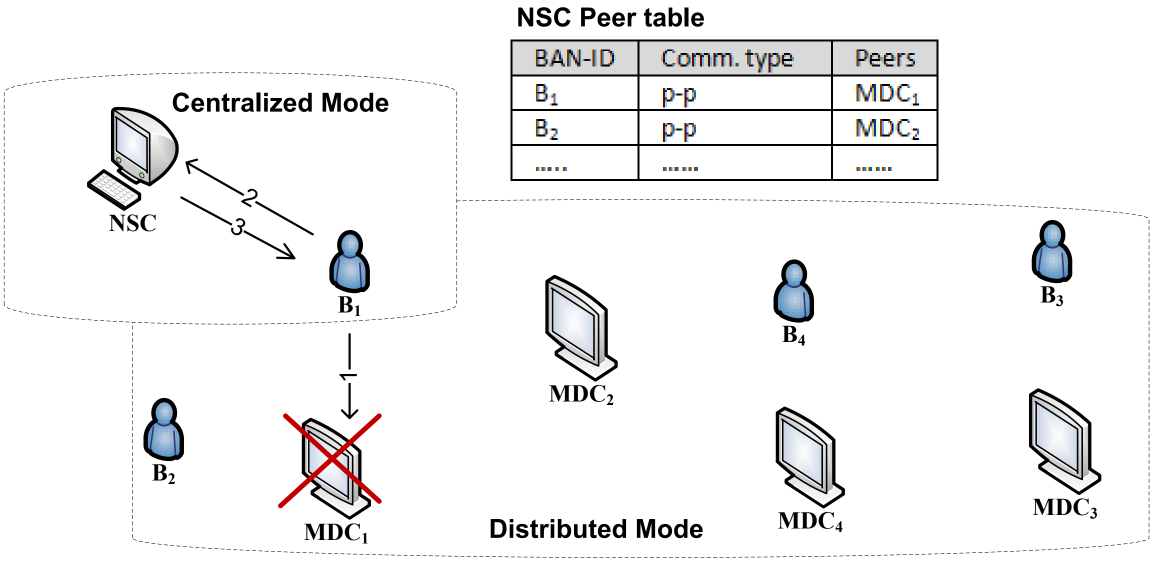
4.6. Peer Unreachable in the ZK-BAN

When B_1 is displaying its data on its peer MDC_1 and suddenly MDC_1 is unreachable due to a link or an MDC_1 failure, B_1 will change its communication mode to centralized from distributed. It will immediately stop sending the data to MDC_1 and contact the NSC for its new peer information (which is stored in the NSC table). After getting connection from the NSC, the BANC will send a peer unreachable message and again ask for peering information, as shown in Figure 4-9 steps 2 and 3. B_1 will wait for new peering and communication information from NSC and will continue the process, which is explained in Sections 4.3 and 4.4. Figure 4-9 shows the three steps required.

1. Look up B_1's routing table and send data for MDC_1 if MDC_1 is not responding, then switch to centralized mode.

2. B_1 → NSC: Peer is unreachable, check communication type (p-p or p-mp) and my Peer/s??

3. NSC → B_1: Look for new peer in the NSC table and send info to B_1.



NSC Peer table

BAN-ID	Comm. type	Peers
B_1	p-p	MDC_1
B_2	p-p	MDC_2
.....	......	......

Figure 4-9: ZK-BAN peer unreachable

4.7. Peer/communication type update in the ZK-BAN

Another important case is when there is any change to the NSC peer table about B_1's peering information; in such cases, the NSC sends a "peer update" message. After receiving this message, B_1 will immediately stop sending data to its peer(s) and change its mode to centralized. It will ask NSC about the change. After getting information from NSC, B_1 will terminate its connection to NSC and continue the process of displaying data on its new peer. This process is summarized below in five steps and is shown in Figure 4-10.

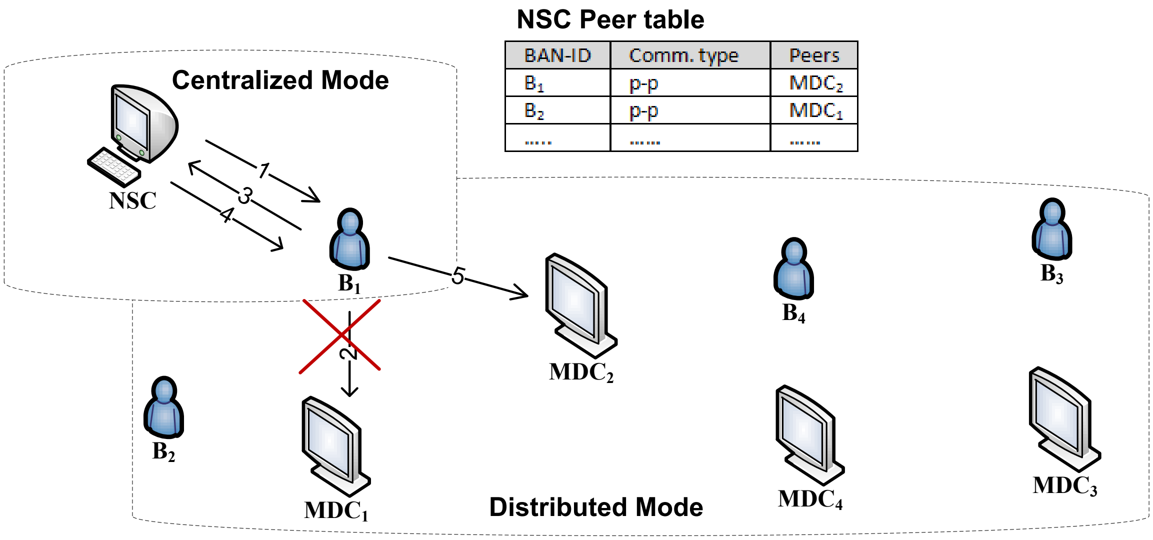
1. NSC → B_1: NSC sends "peer update" message to B_1.

2. B_1 stops communication with MDC_1 and will switch to centralized mode.

3. B_1 → NSC: Check communication type (p-p or p-mp) and peer(s)?

4. NSC → B_1: Look NSC peer table and send info to B_1.

5. B_1 sends data to new peer (MDC_2).



NSC Peer table

BAN-ID	Comm. type	Peers
B_1	p-p	MDC_2
B_2	p-p	MDC_1
……	……	……

Figure 4-10: ZK-BAN peer/communication type update

4.8. NSC Unreachable in the ZK-BAN

In centralized mode, the BAN connects with the NSC. If the NSC is unreachable then the BANC will search for an alternate path to the geographically closest MDC. All MDCs and NSC are connected via Wi-Fi, as shown in Figure 4-1. The BAN sends the NSC unreachable message to the central server.

The two steps shown in Figure 4-11 are explained below.

1. B_1 sends data to the NSC. B_1 does not receive any response in case NSC is unreachable.
2. Look at B_1 routing table and send data for NSC via closest MDC.

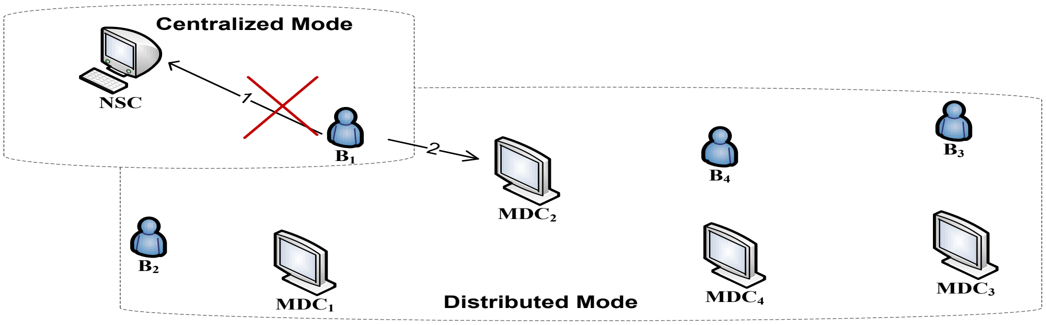


Figure 4-11: NSC unreachable

4.9. SUMMARY

In this chapter, a novel ZK-BAN peering frame for hospital scenarios is proposed. In the proposed framework, all the communication devices in the hospital are divided into three types (BANs, MDCs, and NSC). This classification of the devices is based on their energy levels. Both centralized and distributed modes of communication are used in this mechanism. The use of the centralized mode ensures data privacy and provides better control on the devices, while the distributed mode helps to reduce the overall energy consumption and increases the ease of node mobility. The mathematical analysis of the ZK-BAN framework and the associated different communication scenarios are also discussed in this chapter.

CHAPTER 5

EPR: ENERGY-AWARE PEERING ROUTING PROTOCOL

A novel BAN architecture for use in an indoor hospital environment was introduced in Chapter 4. This chapter proposes a novel mechanism of peer discovery together with a novel routing table construction method with the goal of reducing network traffic load, reducing energy consumption, and improving BAN reliability. Two scenarios with fixed and variable numbers of packets sent by source nodes are considered to better illustrate the feasibility of the proposed peer discovery and routing table construction mechanisms. Extensive simulations in the OMNeT++ based Castalia-3.2 simulation environment have been performed to demonstrate that the proposed protocol has better performance in terms of reduced BAN traffic load, increased successful packet transmission rate (throughput), reduced number of packets forwarded by intermediate nodes, less packets dropped due to MAC buffer overflow, overall lower energy consumption, and lower end-to-end delay (latency) in both stationary and movable patient scenarios when compared with similar protocols.

The remaining part of this chapter is organized as follows. Section 5.1 discusses the motivation of this work. Section 5.2 describes details of the proposed Energy-aware Peering Routing protocol (EPR). Section 5.3 presents performance evaluation of the proposed energy-aware routing protocol. Sections 5.4, 5.5, and 5.6 give the simulation results of three scenarios. Section 5.7 provides the conclusion of the chapter.

5.1. MOTIVATION

As explained previously in Chapter 4, in BAN the body implant and wearable sensors send their data to a central device known as the coordinator. The coordinator is a computationally more powerful device and behaves as a router in BAN networks. BAN communication factors include reliability, short range transmission, low data rate, lower energy consumption, and non-interference with other medical devices. The current Personal Area Network (PAN) standards do not support BAN communication [22]. However, the IEEE 802.15 task group 6 is working to develop a standard for BAN, which should be compatible with a low transmission range of 3 meters (as explained in detail in Section 3.1 of Chapter 3), data rates of up to 10Kbps, and support for QoS [3].

To address the challenges related to the management of patients' medical information, an automated monitoring of BAN data in hospital environments is required [4]. The projects [35, 68] use two tiered communication in order to send data from body sensors to the Web server or database server. Only outdoor BAN communication is considered in [36], which uses a GPS module. ALARM-NET [72] introduces an automatic monitoring system by using WSN. In [73], the store and display idea is used to send the BAN data to the database, and then from the database, the healthcare devices can be used to display the data. The network architectures used in existing projects [35, 36, 68, 72, 73] consider only centralized approaches for monitoring the patients' data. However, as mentioned previously in Chapter 3 no mechanism is provided for displaying the BAN data when there is no connectivity of the healthcare system with the central database.

A routing protocol is required to implement the ZK-BAN peering framework proposed in Chapter 4. In [33], a routing protocol is proposed in which different packet classes are handled differently depending on their QoS requirements. The research in [33] considers BAN communication in which the next hops in the network are only BAN coordinators. The BAN environment in a hospital has different requirements including different device types as next hops. In [74], the suggested BAN network architecture explains the mechanism of combining or splitting a BAN in inter-BAN communication. In [71], a reasonable idea for internetworking of BANs is presented; however, it does not consider the real-time display of BAN data in a hospital environment. There are other ideas [32, 41, 42, 43, 44, 45, 46, 47] for efficient routing in WSN, but these do not consider the requirements of BAN communication in a hospital scenario.

5.2. Energy-aware Peering Routing Protocol (EPR)

In this chapter, the proposed EPR routing protocol is intended to be employed in the indoor hospital environment for BAN communication with the objective of reducing the BAN energy consumed. In previous related work described in [33] the researchers of DMQoS propose a data-centric multiobjective QoS-aware routing protocol that is used to select the next hop node and forwards data packets by taking into consideration the QoS requirements of the data. The higher residual energy and geographic position were the two important factors used for choosing the downstream hop. Network traffic is differentiated into different classes including Ordinary Packets (OPs), Critical Packets (CPs), Reliability-driven Packets (RPs), and Delay-driven Packets (DPs) according to their generated data types. The reliability and delay control modules introduced in [33] result in better performance than several state-of-the-art approaches [41, 32, 42, 43, 44, 45, 46, 47] in terms of lower bit error rates, traffic load, and operation energy overload. However, a disadvantage of [33] is that the method used for sending the Hello packets and creating the routing table causes increased network traffic, thereby increasing BAN energy consumption. In [33], the next hops considered in the BAN communication are only BAN coordinators, and every node broadcasts its Hello packets after a specific period of time. In reality, the BAN communication in a hospital environment has different requirements including different device types (i.e. NSC, MDC, BAN) as next hops. In this chapter, we addressed these shortcomings of the DMQoS protocol with the consideration of all possible devices (NSC, MDCs and BANs) in the hospital environment by controlling the broadcasts of the Hello packets. Also, a proposed novel peering mechanism provides the details of to whom and when the Hello packets are broadcasted, which results in a greatly reduced number of Hello packets broadcasted. Unlike [33], only NSC and MDCs (which have considerably more energy than BAN nodes) broadcast Hello packets periodically and the BAN broadcasts it's Hello packet only at the reception of other nodes' Hello packets containing the NSC or MDC information. The interval of Hello packets broadcasted by NSC or MDCs depends upon the probability of new additions to the BANs. For example, when new patients are admitted into the emergency room, it is required that a minimum time is expended between the broadcasts of the Hello packets from NSC and MDCs.

In our experiments, NSC and MDCs broadcast their Hello packets every 30s (i.e., every 0.5 minutes). The time period of 30 seconds is similar to the timings employed by Routing Information Protocol (RIP) [75], and was chosen so as to reduce the energy consumed in network management activities in a large network. The 30s was determined to be sufficient time to provide the other network nodes enough time to update their routing tables before sending their Hello packet. The proposed methodology consists of three parts: 1) the new Hello protocol, 2) a novel neighbor table construction method, and 3) an innovative routing table creation algorithm based on the geographic and energy information in the neighbor table. In the ZK-BAN peering framework as discussed in Chapter 4, a BAN coordinator needs to have a connection with the NSC for obtaining its peering information, and a connection with the MDC as peer for displaying its data. An indirectly connected BAN coordinator must use another BAN as its next hop only if the other BAN can help its transmission to reach the MDC or NSC. A BAN that does not have a direct connection to a NSC or MDC will not broadcast its Hello packets, and any neighboring nodes will not consider such a BAN coordinator as their next hop. In the proposed Hello protocol, initially nodes do not broadcast any Hello packets. First, the MDCs and NSC will broadcast their Hello packets to their neighboring nodes. It is assumed that a node *i* that receives MDCs or NSC information in the Hello packet will create its neighbor table and routing table, and then start to broadcast its own Hello packets. Node *i* will stop broadcasting Hello packets if it fails to receive a Hello packet at any time, and remove all the entries from its neighbor and routing tables.

When considering energy levels of BAN devices, the devices used in the ZK-BAN network model can be divided into three types. The NSC is considered to be a type 1 device, which is connected directly to the power source. The MDC is considered to be a type 2 device, which requires the replacement of its batteries periodically. The BAN coordinator is a type 3 device because of its limited energy availability. The device type, distance from neighbor to the node, and neighbor residual energy are all important factors in building the routing table. The proposed energy-aware peering routing protocol considers the neighbor with shorter distance, lower device type, and higher residual energy in selecting the next hop node. The benefit of considering these factors is to reduce the overall network traffic load and BAN energy consumption within the network. The proposed energy-aware peering routing protocol is explained below.

5.2.1. Hello Protocol

It is assumed that each type 1 and type 2 device (NSC or MDCs) send Hello packets periodically. The Hello packet fields of node *j* are shown in Figure 5-1. The destination (*Dst*) can be a NSC or any MDC, or BANC. The Hello packet contains information about the destination device ID ($\mathrm{ID_{Dst}}$), destination location ($\mathrm{L_{Dst}}$), sender's ID ($\mathrm{ID_j}$), distance from sender node *j* to the destination ($\mathrm{D_{(j,Dst)}}$), residual energy ($\mathrm{E_j}$), and device type ($\mathrm{T_j}$).

$\mathrm{ID_{Dst}}$	**$\mathrm{L_{Dst}}$**	**$\mathrm{ID_j}$**	**$\mathrm{L_j}$**	**$\mathrm{D_{(j,Dst)}}$**	**$\mathrm{E_j}$**	**$\mathrm{T_j}$**

Figure 5-1: EPR - Hello packet format

The residual energy ($\mathrm{E_j}$) is the residual or remaining node *j* energy. The $\mathrm{D_{(j,Dst)}}$ is calculated by using Equation 5-1. Upon reception of the Hello packets from the node *j*, the receiver node *i* will store the information in its neighbor table for further processing. Moreover, the node *i* adds its own information to the received Hello packet before broadcasting its Hello packets. If the next Hello packet from the same sender is not received within a certain time period (i.e. max. 30s), this means the sender has moved away or has broken down. All the entries in the neighbor table associated with that specific sender will be deleted and the routing table will be updated.

$$\mathrm{D_{(j,Dst)}} = \sqrt{(\mathrm{X_j} - \mathrm{X_{Dst}})^2 + (\mathrm{Y_j} - \mathrm{Y_{Dst}})^2} \qquad (5\text{-}1)$$

5.2.2. Neighbor Table Constructor Algorithm

Let node *j* be the neighbor of node *i* and be located in between node *i* and destination node *Dst*. The neighbor table structure of node *i* is shown in Figure 5-2. It contains the information about the destination device ID ($\mathrm{ID_{Dst}}$), destination location ($\mathrm{L_{Dst}}$), neighbor ID ($\mathrm{ID_j}$), neighbor location ($\mathrm{L_j}$), distance from neighbor to the destination ($\mathrm{D_{(j,Dst)}}$), distance from neighbor ($\mathrm{D_{(i,j)}}$), neighbor residual energy ($\mathrm{E_j}$), neighbor device type ($\mathrm{T_j}$), and communication cost ($\mathrm{C_j}$).

$\mathrm{ID_{Dst}}$	**$\mathrm{L_{Dst}}$**	**$\mathrm{ID_j}$**	**$\mathrm{L_j}$**	**$\mathrm{D_{(j,Dst)}}$**	**$\mathrm{D_{(i,j)}}$**	**$\mathrm{E_j}$**	**$\mathrm{T_j}$**	**$\mathrm{C_j}$**

Figure 5-2: EPR - Neighbor table structure of node *i*

After receiving a Hello packet, the node *i*'s neighbor table constructor algorithm will compare the distance from neighbor to the destination ($D_{(j,Dst)}$(hp)) with the direct distance of node *i* to the destination $D_{(i,Dst)}$. It will add a new record for *Dst* if $D_{(j,Dst)}$ from the Hello packet is less than the distance between the node *i* to the destination i.e. $D_{(j,Dst)}(hp) < D_{(i,Dst)}$. For illustration purposes, an example is shown in Figure 5-3. The nodes NSC, MDC, and BAN are considered as destination *Dst*, neighbor node *j*, and source node *i*, respectively. The two benefits of doing this are as follows:

1. It ensures that the proper neighbor node is selected. The location of neighbor node is towards the destination *Dst* and has less distance to the destination than the source node itself.

2. It helps to prevent the loops.

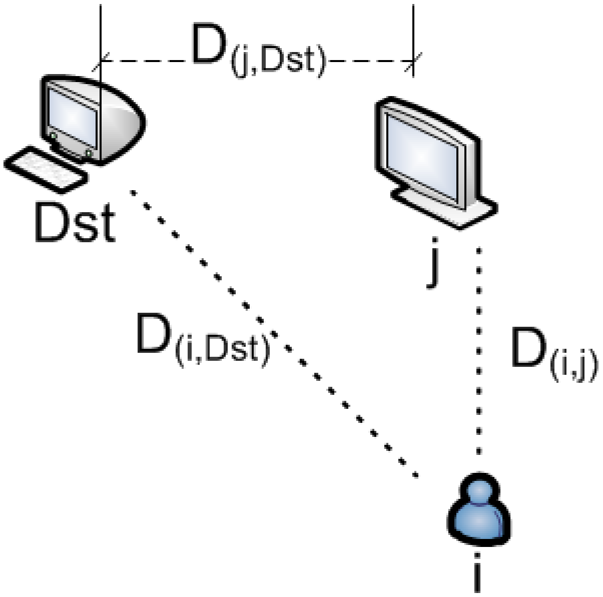


Figure 5-3: EPR – prevent looping

Equations 5-2 and 5-3 are used to calculate the distance from source node *i* to neighbor node *j* and the communication cost C_j.

$$D_{(i,j)} = \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2} \tag{5-2}$$

$$C_j = \frac{\left(T_j * D^2_{(i,j)}\right)}{E_j} \tag{5-3}$$

Algorithm 5 − 1 Neighbor table constructor algorithm for EPR, at each node *i*.

```
INPUT: Hello Packet
$D_{(i,Dst)} = \sqrt{(X_i - X_{Dst})^2 + (Y_i - Y_{Dst})^2}$
    if $\left(D_{(j,Dst)}(hp) < D_{(i,Dst)}\right)$ then
        (add a new record for the Dst's information in the neighbor table)
        $ID_{Dst}(nt) \leftarrow ID_{Dst}(hp)$
        $ID_j(nt) \leftarrow ID_j(hp)$
        $L_j(nt) \leftarrow L_j(hp)$
        $D_{(j,Dst)}(nt) \leftarrow D_{(j,Dst)}(hp)$
        $E_j(nt) \leftarrow E_j(hp)$
        $T_j(nt) \leftarrow T_j(hp)$
        $D_{(i,j)}(nt) = \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}$
        $C_j(nt) = \frac{\left(T_j\ (nt) * D^2_{(i,j)}(nt)\right)}{E_j\,(nt)}$
    end if
    (add a new record for the neighbor node j's information in the neighbor table)
        $ID_{Dst}(nt) \leftarrow ID_{(Dst)}(hp)$
        $ID_j(nt) \leftarrow ID_j(hp)$
        $L_j(nt) \leftarrow L_j(hp)$
        $D_{(j,Dst)}(nt) = 0$
        $E_j(nt) \leftarrow E_j(hp)$
        $T_j(nt) \leftarrow T_j(hp)$
        $D_{(i,j)}(nt) = \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}$
        $C_j(nt) = \frac{\left(T_j\ (nt) * D^2_{(i,j)}(nt)\right)}{E_j\,(nt)}$
```

The algorithm for neighbor table constructor for node *i* is shown in Algorithm 5-1. It is assumed that node *i* receives a Hello packet from neighbor node *j*. The hp and nt used in this algorithm stand for Hello packet and neighbor table respectively. X_i, Y_i represent the X, Y coordinates of node *i*. X_{DST}, Y_{DST} stand for the X, Y coordinates of the destination. It is assumed that the locations of the NSC and MDCs are known. The values of X_i and Y_i of node *i* are calculated by the Received Signal Strength Indication (RSSI) localization technique given in [76]. The other fields of the neighbor table have the same meanings as in the Hello packet. $D_{(i,j)}$ and C_j are calculated by using Equations 5-2 and 5-3 respectively. The values of T_j, $D_{(i,j)}$, and E_j are used to find the communication cost (C_j). As seen from Equation 5-3, a shorter distance ($D_{(i,j)}$), lower device type (T_j), and higher residual energy (E_j) will result in a lower communication cost (C_j). The node *j* with lowest value of C_j is the best choice for the next hop. The neighbor table constructor calculates the communication cost and updates the neighbor table periodically after receiving every new Hello packet.

5.2.3. Routing Table Constructor Algorithm

It can be seen that there are many records in the neighbor table for the same destination. Therefore, the routing table constructor algorithm filters the neighbor table, and only chooses the entry with the lowest communication cost. The routing table structure of node *i* is shown in Figure 5-4. It contains destination ID (ID_{Dst}), destination location (L_{Dst}), and next hop (NH). As shown in Algorithm 5-2, a new record is added in the routing table for each destination *Dst* ∈ {MDC, NSC, BAN}.

Algorithm 5 − 2: Routing table constructor algorithm

```
INPUT: Neighbor table, i's neighbor table records $NH_{(i,Dst)}$, $\forall$ Dst $\in$ {MDC, NSC, BAN}
for each destination Dst $\in$ {NSC, MDC, BAN} do
    if ($ID_j(nt) == ID_{Dst}(nt)$) then
        (add a new record for the Dst's information in the routing table)
        $ID_{Dst} \leftarrow ID_{Dst}(nt)$
        $L_{Dst} \leftarrow L_{Dst}(nt)$
        $NH \leftarrow ID_{Dst}(nt)$
    else
        if ($C_j == \min_{k \in NH_{(i,Dst)}} C_k$) then
            (add a new record for the Dst's information in the routing table)
            $ID_{Dst} \leftarrow ID_{Dst}(nt)$
            $L_{Dst} \leftarrow L_j(nt)$
            $NH \leftarrow ID_j(nt)$
        end if
    end if
end for
```

If the destination (*Dst*) and node *i* are directly connected with each other, the next hop (NH) will be the destination ID (ID_{Dst}). Otherwise neighbor node *j* with the lowest communication cost (C_j) will be selected as next hop (NH).

ID_{Dst}	L_{Dst}	NH

Figure 5-4: EPR - Routing table structure

An example is considered to understand the step by step process for the broadcast of Hello packets, construction and updating the neighbor table, and construction and updating the routing tables of the devices. The topology used in this example is shown in Figure 5-5.

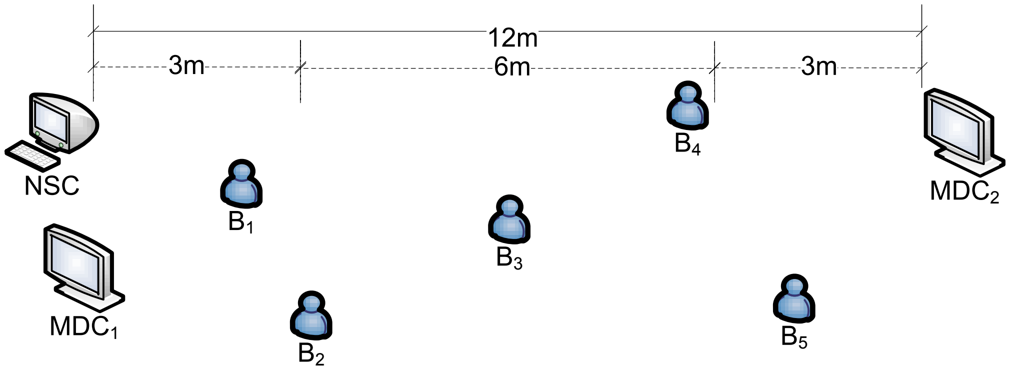


Figure 5-5: Stage 1 – Initialization: Devices before starting communication.

Let the classification of the devices with their location coordinates be as given below:

Class 1 devices: NSC (0, 5)

Class 2 devices: MDC_1 (0, 3), MDC_2 (8, 5)

Class 3 devices: B_1 (2, 4), B_2 (3, 2), B_3 (4, 4), B_4 (6, 6), B_5 (7, 3)

Stage 1 is the initialization stage, each device has just powered on. There are no Hello packets broadcasted, the neighbor tables and routing tables of all the nodes are empty at this stage. The next hops for all destinations at this stage are shown in Table 5-1 below.

Table 5-1: Stage 1 – Routing table of the nodes are same at this stage.

Source Node	Destinations							
	NSC	**MDC_1**	**MDC_2**	**B_1**	**B_2**	**B_3**	**B_4**	**B_5**
NSC	X	-	-	-	-	-	-	-
MDC_1	-	X	-	-	-	-	-	-
MDC_2	-	-	X	-	-	-	-	-
B_1	-	-	-	X	-	-	-	-
B_2	-	-	-	-	X	-	-	-
B_3	-	-	-	-	-	X	-	-
B_4	-	-	-	-	-	-	X	-
B_5	-	-	-	-	-	-	-	X

In Stage 2, each class 1 and 2 devices (i.e. NSC and MDCs) broadcasts Hello packet as shown in Figure 5-6.

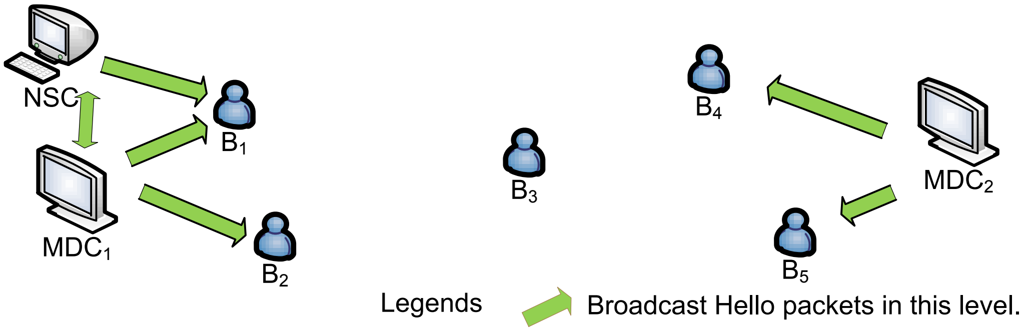


Figure 5-6: Stage 2 – Only NSC and MDCs first broadcast the Hello packets.

The devices MDC$_1$ and B$_1$ receive the Hello packets from NSC. At the same time, MDC$_1$ sends its Hello packets to its neighbor nodes NSC, B$_1$, and B$_2$. Upon reception of Hello packets from NSC and MDC$_1$, B$_1$ adds two records to its neighbor table using Algorithm 5-1. The neighbor table of device B$_1$ is given in Table 5-2.

Table 5-2: Neighbor table of B$_1$ after Stage 2

Record #	ID_{Dst}	ID_j	L_j	$D_{(i, j)}$	$D_{(j, Dst)}$	E_j	T_j	C_j
1	NSC	NSC	(0,5)	2.236	0	0.99	1	5.05
2	MDC$_1$	MDC$_1$	(0,3)	2.236	0	0.99	2	10.10

The routing table of each node is updated after receiving the Hello packets and filtering the neighbor tables by using the proposed protocol at every stage. The next hop for each destination at Stage 2 is shown in Table 5-3.

Table 5-3: Stage 2 – Next hop for each destination at this stage

Source Node	Destinations							
	NSC	**MDC$_1$**	**MDC$_2$**	**B$_1$**	**B$_2$**	**B$_3$**	**B$_4$**	**B$_5$**
NSC	X	MDC$_1$	-	-	-	-	-	-
MDC$_1$	NSC	X	-	-	-	-	-	-
MDC$_2$	-	-	X	-	-	-	-	-
B$_1$	NSC	MDC$_1$	-	X	-	-	-	-
B$_2$	-	MDC$_1$	-	-	X	-	-	-
B$_3$	-	-	-	-	-	X	-	-
B$_4$	-	-	MDC$_2$	-	-	-	X	-
B$_5$	-	-	MDC$_2$	-	-	-	-	X

According to the proposed algorithm, Class 3 devices broadcast their Hello packets only when they have information of a Class 1 or 2 device. During Stage 2, the BAN devices shown in Figure 5-7 receive the information from NSC and MDCs.

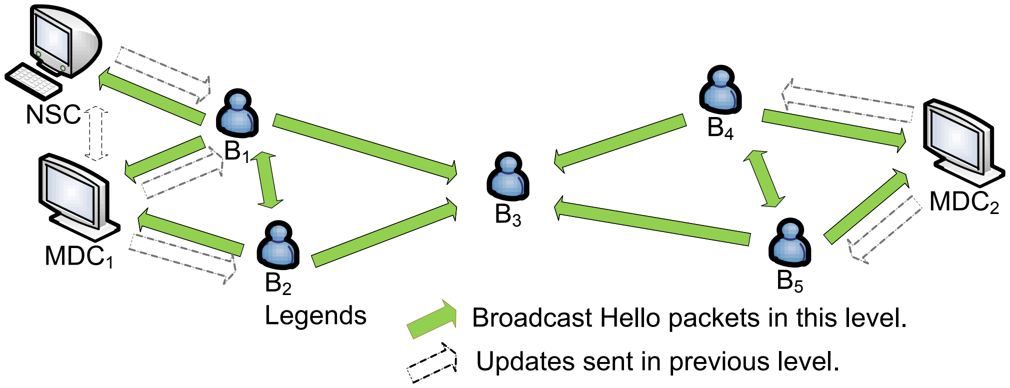

NSC
B_1
MDC_1
B_2
B_3
B_4
B_5
MDC_2
Legends
Broadcast Hello packets in this level.
Updates sent in previous level.

Figure 5-7: Stage 3 – NSC/MDC Hello packet receiver nodes broadcast their Hello packets

In Stage 3, the BAN devices (i.e. B_1, B_2, B_3, B_4, and B_5) broadcast the Hello packets to inform their neighbors that they can reach the class 1 and 2 devices through them. The entries of the Hello packet broadcasted by B_1 are shown in Table 5-4.

Table 5-4: Stage 3: Hello packet entries broadcasted by node B_1

Record #	$\mathbf{ID_{Dst}}$	$\mathbf{L_{Dst}}$	$\mathbf{ID_j}$	$\mathbf{L_j}$	$\mathbf{D_{(j,\,Dst)}}$	$\mathbf{E_j}$	$\mathbf{T_j}$
1	NSC	(0,5)	B_1	(2,4)	2.236	0.99	3
2	MDC_1	(0,3)	B_1	(2,4)	2.236	0.99	3

In accordance with line 2 of the neighbor table constructor algorithm described in Section 5.2.2, B_1 ignores the Hello packet from B_2 in which the value of ID_{Dst} equals $MDC_{1.}$ This is due to the direct distance of MDC_1 (D_{Dst}) being less than the distance from B_1 to MDC_1 via B_2 ($D_{(i,Dst)}$). One record is added in B_1's neighbor table. B_3 adds nine records into the neighbor table upon reception of the Hello packets from all neighbor nodes. The neighbor tables of B_1 and B_3 nodes are given in Table 5-5 and Table 5-6 respectively.

Table 5-5: Neighbor table of B_1 after Stage 3

Record #	$\mathbf{ID_{Dst}}$	$\mathbf{ID_j}$	$\mathbf{L_j}$	$\mathbf{D_{(i,\,j)}}$	$\mathbf{D_{(j,\,Dst)}}$	$\mathbf{E_j}$	$\mathbf{T_j}$	$\mathbf{C_j}$
1	NSC	NSC	(0,5)	2.236	0	0.99	1	5.05
2	MDC_1	MDC_1	(0,3)	2.236	0	0.99	2	10.10
3	B_2	B_2	(3,2)	2.236	0	0.99	3	15.15

The communication cost (C_j) is calculated by using Equation 5-3. The entry with the lowest value of C_j is selected as a next hop in case of multiple entries for the same destination node. For example, there are two next hop options B_1 and B_2 for MDC_1 to send the data towards the destination MDC_1, as shown by entries 2 and 3 in Table 5-5. The routing table constructor algorithm discussed in Section 5.2.3 chooses B_1 as the next hop for destination MDC_1 due to the lower value of C_j in entry 2. In a similar fashion, B_2 is selected as a next hop for reaching the destination MDC_2. The next hops from sources to destinations for the network after Stage 3 are given in Table 5-7.

Table 5-6: Neighbor table of B_3 after Stage 3

Record #	$\mathbf{ID_{Dst}}$	$\mathbf{ID_j}$	$\mathbf{L_j}$	$\mathbf{D_{(i, j)}}$	$\mathbf{D_{(j, Dst)}}$	$\mathbf{E_j}$	$\mathbf{T_j}$	$\mathbf{C_j}$
1	NSC	B_1	(2,4)	2	2.236	0.99	3	12.12
2	MDC_1	B_1	(2,4)	2	2.236	0.99	3	12.12
3	MDC_1	B_2	(3,2)	2.236	3.162	0.99	3	15.15
4	MDC_2	B_4	(6,6)	2.828	2.236	0.99	3	24.24
5	MDC_2	B_5	(7,3)	3.162	2.236	0.99	3	30.30
6	B_1	B_1	(2,4)	2	0	0.99	3	12.12
7	B_2	B_2	(3,2)	2.236	0	0.99	3	15.15
8	B_4	B_4	(6,6)	2.828	0	0.99	3	24.24
9	B_5	B_5	(7,3)	3.162	0	0.99	3	30.30

Table 5-7: Stage 3 – Next hop for each destination

Source Node	Destinations							
	NSC	$\mathbf{MDC_1}$	$\mathbf{MDC_2}$	$\mathbf{B_1}$	$\mathbf{B_2}$	$\mathbf{B_3}$	$\mathbf{B_4}$	$\mathbf{B_5}$
NSC	X	MDC_1	-	B_1	-	-	-	-
$\mathbf{MDC_1}$	NSC	X	-	B_1	B_2	-	-	-
$\mathbf{MDC_2}$	-	-	X	-	-	-	-	-
$\mathbf{B_1}$	NSC	MDC_1	-	X	B_2	-	-	-
$\mathbf{B_2}$	MDC_1	MDC_1	-	B_1	X	-	-	-
$\mathbf{B_3}$	B_1	B_2	B_4	B_1	B_2	X	B_4	B_5
$\mathbf{B_4}$	-	-	MDC_2	-	-	B_3	X	B_5
$\mathbf{B_5}$	-	-	MDC_2	-	-	B_3	B_4	X

In Stage 4, after receiving the Hello packets with the information of NSC/MDCs from neighbor nodes, B_3 now broadcasts its Hello packet, as shown in Figure 5-8.

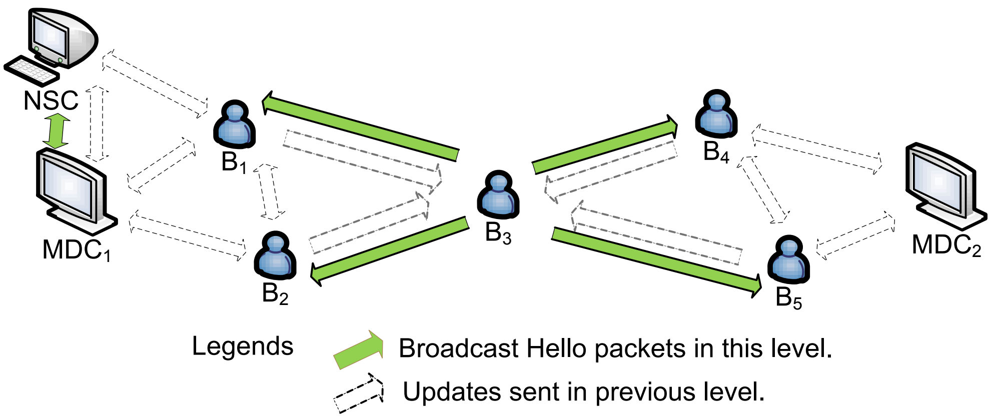

NSC
B1
B4
MDC2
MDC1
B3
B2
B5
Legends
Broadcast Hello packets in this level.
Updates sent in previous level.

Figure 5-8: Stage 4 – B_3 broadcasts Hello packet with the information of NSC, MDC_1, and MDC_2

B_3 broadcasts Hello packet with NSC, MDC_1, and MDC_2 information. The entries of the B_3 Hello packet are shown in Table 5-8.

Table 5-8: Stage 3 - Hello packet entries broadcasted by node B_3

Record #	**ID_{Dst}**	**L_{Dst}**	**ID_j**	**L_j**	**$D_{(j, Dst)}$**	**E_j**	**T_j**
1	NSC	(0,5)	B_3	(4,4)	4.123	0.99	3
2	MDC_1	(0,3)	B_3	(4,4)	4.123	0.99	3
3	MDC_2	(8,5)	B_3	(4,4)	4.123	0.99	3
4	B_1	(2,4)	B_3	(4,4)	2	0.99	3
5	B_2	(3,2)	B_3	(4,4)	2.236	0.99	3
6	B_4	(6,6)	B_3	(4,4)	2.828	0.99	3
7	B_5	(7,3)	B_3	(4,4)	3.162	0.99	3

After receiving the Hello packet from node B_3, B_1 adds these records in its neighbor table. The neighbor table of B_1 now contains the information of B_4, B_5, and MDC_2 as shown in Table 5-9.

Table 5-9: Stage 4 - Neighbor table of B_1 at this stage

Record #	**ID_{Dst}**	**ID_j**	**L_j**	**$D_{(i, j)}$**	**$D_{(j, Dst)}$**	**E_j**	**T_j**	**C_j**
1	NSC	NSC	(0,5)	2.236	0	0.99	1	5.05
2	MDC_1	MDC_1	(0,3)	2.236	0	0.99	2	10.10
3	MDC_2	B_3	(4,4)	2	2.828	0.99	3	12.12
4	B_2	B_2	(3,2)	2.236	0	0.99	3	15.15
5	B_4	B_3	(4,4)	2	3.162	0.99	3	12.12
6	B_5	B_3	(4,4)	2	4.123	0.99	3	12.12

The neighbor table of the node B_3 remains the same as in Table 5-6 because it did not receive any Hello packet from its neighbors at this stage.

Table 5-10 shows the next hops for all source nodes used in this network.

Table 5-10: Stage 4 - Next hop for each destination at this stage

Source Node	Destinations							
	NSC	**MDC_1**	**MDC_2**	**B_1**	**B_2**	**B_3**	**B_4**	**B_5**
NSC	X	MDC_1	-	B_1	MDC_1	-	-	-
MDC_1	NSC	X	-	B_1	B_2	-	-	-
MDC_2	-	-	X	-	-	-	-	-
B_1	NSC	MDC_1	B_3	X	B_2	B_3	B_3	B_3
B_2	MDC_1	MDC_1	B_3	B_1	X	B_3	B_3	B_3
B_3	B_1	B_2	B_4	B_1	B_2	X	B_4	B_5
B_4	B_3	B_3	MDC_2	B_3	B_3	B_3	X	B_5
B_5	B_3	B_3	MDC_2	B_3	B_3	B_3	B_4	X

In Stage 5, nodes B_1, B_2, B_4, and B_5 broadcast the Hello packets, as shown in Figure 5-9. B_3 receives the Hello packets with destinations of B_4, B_5, and MDC_2 from B_1 and B_2. Because B_3 has a shorter distance than the distance advertised in these received Hello packets, B_3 just discards these Hello packets. For the same reason, B_1 discards the Hello packet with destination of B_3, B_4, B_5, and MDC_2 received from the NSC, the MDC_1 and B_2. The neighbor tables of B_1 and B_3 remain the same as in Stage 4.

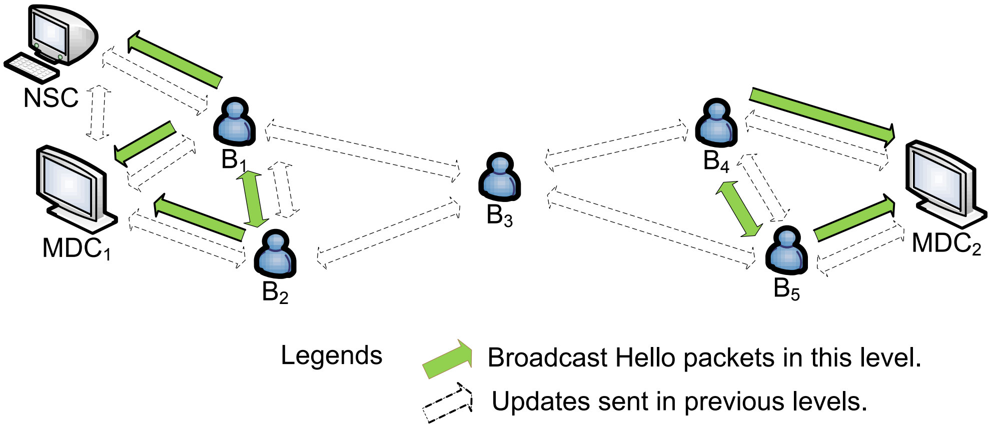

NSC
B_1
MDC_1
B_2
B_3
B_4
MDC_2
B_5
Legends
Broadcast Hello packets in this level.
Updates sent in previous levels.

Figure 5-9: Stage 5 – Every node receives the information about all nodes.

The process of routing table creation for all the destinations is completed at this stage. Table 5-11 below shows the next hop information from source node to destination after receiving new updates from each stage.

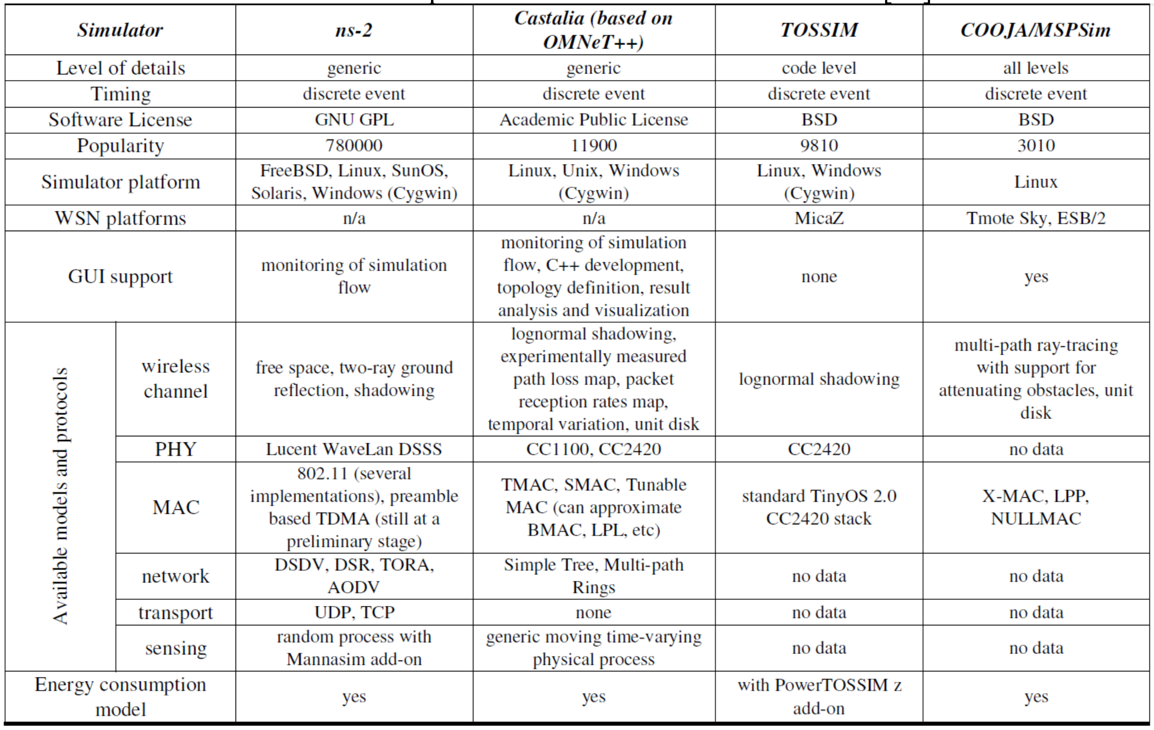
Table 5-11: Stage 5 - Complete list of next hops for all destinations

Source Node	Destinations							
	NSC	**MDC$_1$**	**MDC$_2$**	**B$_1$**	**B$_2$**	**B$_3$**	**B$_4$**	**B$_5$**
NSC	X	MDC$_1$	B$_1$	B$_1$	MDC$_1$	B$_1$	B$_1$	B$_1$
MDC$_1$	NSC	X	B$_1$	B$_1$	B$_2$	B$_1$	B$_1$	B$_1$
MDC$_2$	B$_4$	B$_4$	X	B$_4$	B$_4$	B$_4$	B$_4$	B$_5$
B$_1$	NSC	MDC$_1$	B$_3$	X	B$_2$	B$_3$	B$_3$	B$_3$
B$_2$	MDC$_1$	MDC$_1$	B$_3$	B$_1$	X	B$_3$	B$_3$	B$_3$
B$_3$	B$_1$	B$_2$	B$_4$	B$_1$	B$_2$	X	B$_4$	B$_5$
B$_4$	B$_3$	B$_3$	MDC$_2$	B$_3$	B$_3$	B$_3$	X	B$_5$
B$_5$	B$_3$	B$_3$	MDC$_2$	B$_3$	B$_3$	B$_3$	B$_4$	X

5.3. Performance Evaluation

Authors in [77] provide a detailed survey of WSN simulators. Table 5-12 shows the comparison of the most commonly used WSN simulators. NS-2 is considered the most commonly used simulator for WSN; however, OMNeT++ based Castalia simulator is found the best simulator for WSN, BAN, and generally networks of low-power embedded devices [78]. Castalia provides more available models and protocols in addition to the better GUI support when compared with NS-2.

Table 5-12: Comparisons of simulators used for WSNs [77]



Simulator		***ns-2***	***Castalia (based on OMNeT++)***	***TOSSIM***	***COOJA/MSPSim***
Level of details		generic	generic	code level	all levels
Timing		discrete event	discrete event	discrete event	discrete event
Software License		GNU GPL	Academic Public License	BSD	BSD
Popularity		780000	11900	9810	3010
Simulator platform		FreeBSD, Linux, SunOS, Solaris, Windows (Cygwin)	Linux, Unix, Windows (Cygwin)	Linux, Windows (Cygwin)	Linux
WSN platforms		n/a	n/a	MicaZ	Tmote Sky, ESB/2
GUI support		monitoring of simulation flow	monitoring of simulation flow, C++ development, topology definition, result analysis and visualization	none	yes
Available models and protocols	wireless channel	free space, two-ray ground reflection, shadowing	lognormal shadowing, experimentally measured path loss map, packet reception rates map, temporal variation, unit disk	lognormal shadowing	multi-path ray-tracing with support for attenuating obstacles, unit disk
	PHY	Lucent WaveLan DSSS	CC1100, CC2420	CC2420	no data
	MAC	802.11 (several implementations), preamble based TDMA (still at a preliminary stage)	TMAC, SMAC, Tunable MAC (can approximate BMAC, LPL, etc)	standard TinyOS 2.0 CC2420 stack	X-MAC, LPP, NULLMAC
	network	DSDV, DSR, TORA, AODV	Simple Tree, Multi-path Rings	no data	no data
	transport	UDP, TCP	none	no data	no data
	sensing	random process with Mannasim add-on	generic moving time-varying physical process	no data	no data
Energy consumption model		yes	yes	with PowerTOSSIM z add-on	yes

The main features of Castalia are given below [78]:

- Specially designed for BAN and networks of low-power embedded devices
- Based on event driven OMNeT++ platform
- Modularity, reliability, and speed of Castalia is partly enabled by OMNeT++
- Used realistic node behavior, realistic wireless channel, and radio models
- Mobility of the nodes is fully supported
- Interference is handled as received signal strength, not as separate feature
- Extended sensing modeling provides the highly flexible physical process model
- Sensing device noise, bias, and power consumption
- Availability of MAC and routing protocols
- Designed for adaptation and expansion

The performance of the proposed routing protocol is compared with the DMQoS routing protocol [33] using simulations performed in the OMNeT++ based Castalia-3.2 simulator [78]. We also compared the performance of energy-aware based routing "EPR" with no energy-aware based routing "noRouting". In noRouting, the data packets are forwarded to random next hop devices instead of the algorithm's next hop based on energy-aware routes. The comparison of EPR with noRouting is used to verify whether sending the packets to a random next hop device results in a more successful transmission rate than the proposed energy-aware routing protocol. The total area used in DMQoS [33] is 2000m X 2000m = 4,000,000 m^2, and each coordinator is placed in 63.3m X 63.3m = 4000 m^2, which is not feasible for the indoor-hospital environment considered in this thesis. Typically, an MDC is placed within 3 meters of the patient's bed. A typical hospital scenario is considered to be one in which NSC, MDCs, and BAN coordinators are used within an area of 9m X 9m = 81 m^2. The overall energy consumption during construction and update of the routing tables is shown in Table 5-13.

Table 5-13: Overall energy consumption during construction and update of routing tables

Transmit power (dBm)	EPR (mJ)	DMQoS (mJ)
-25	10930	10928
-15	11016	11013
-10	11033	11043

Different values of transmit power i.e. -10dBm, -15dBm, and -25dBm are employed in the simulations. The network parameters used in the simulations are summarized in Table 5-14.

Table 5-14: Parameters information

Deployment	Area	9 m X 9 m
	Deployment type	Case 1 and 2: All nodes are static Case 3: NSC and MDCs are fixed but BANs are movable
	Number of nodes	4 BANs, 3 MDCs, 1 NSC
	Initial nodes locations	NSC(0,1), MDC_1 (0,5), MDC_2 (0,3), MDC_3 (1,3), B_1(2,3), B_2 (3,5), B_3(3,0), B_4(6,3)
	Initial node energy	18720 J (= 2 AA batteries)
	Buffer size	32 packets
	Link layer trans. Rate	250 Kbps
	Transmit power	Different transmission powers (−10dBm, −15dBm, −25dBm)
	Reception power	7dBm
Task	Application type	Event − driven
	Max. packet size	32 Bytes
	Traffic type	CBR (Constant Bit Rate)
MAC	IEEE 802.15.4	Default values
Simulation	Time	2003 Seconds including 3s for nodes initialization (Simulation results are the average of three rotations)

Three scenarios are considered: in Scenario 1 a fixed number of packets are sent and all nodes are static, in Scenario 2 a variable number of packets are sent and all nodes are static, and in Scenario 3 a variable number of packets are sent and the BANs are movable (this is to model patient mobility). The results are then observed and compared. The simulator Castalia 3.2 used for the performance testing of proposed protocols has 11 distinct random number streams that effect different parts of the simulation [79]. Each simulation run uses one set of random seeds. The results shown in this chapter are the average of three repetitions and the results are quite different compared to the ones when only one simulation is executed. Another thing to notice is that the results are "smoother"; there is less extreme variation. Higher number of repetitions provides smoother results, but too many simulation runs take longer to complete. Castalia provides the tool to calculate the confidence intervals of the results over the repetitions it executed [79]. To achieve a 97% confidence interval for the illustrative results, three runs are simulated in every experiment, which may introduce a maximum error of $3x10^{-3}$, based on the error calculation done by Castalia simulator [79]. Performance parameters measured include successful transmission rate, MAC buffer overflow, packets forwarded by intermediate nodes, network traffic, and overall energy consumption for all the three scenarios in the 4K to 80K range. The results of these scenarios are discussed below.

5.4. Scenario 1

In this case, each BAN coordinator sends 1000 packets to the corresponding MDC or NSC. The deployment of the nodes is shown in Figure 5-10. From all BANs, B_1 is the closest BAN node to the NSC and MDCs. In DMQoS [33], B_1 is responsible for forwarding the data packets from other nodes to NSC or MDCs. This results in more energy consumption for B_1 and increased congestion experienced by B_1. EPR resolves these problems by choosing the most appropriate next hop based on the lowest value of communication cost. In the proposed EPR scheme, the BAN coordinator does not send data to another BAN coordinator unless it is necessary.

Figure 5-11 shows the number of packets forwarded by the intermediate nodes. It is seen from Figure 5-11 that 332 data packets go through the intermediate nodes before reaching to the destinations in EPR when the transmit power is -25dBm. For transmit power of -15dBm and -10dBm, there is no packet which goes through intermediate nodes in EPR because the destinations are in range due to the high transmit powers.

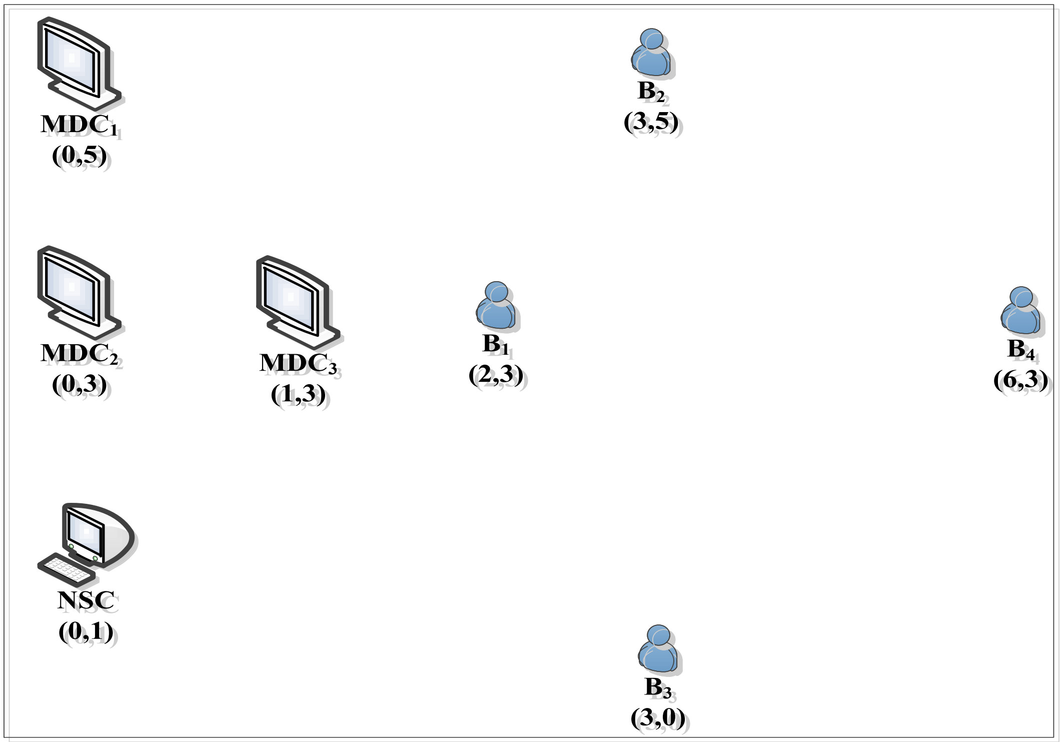


Figure 5-10: EPR - Deployment of nodes

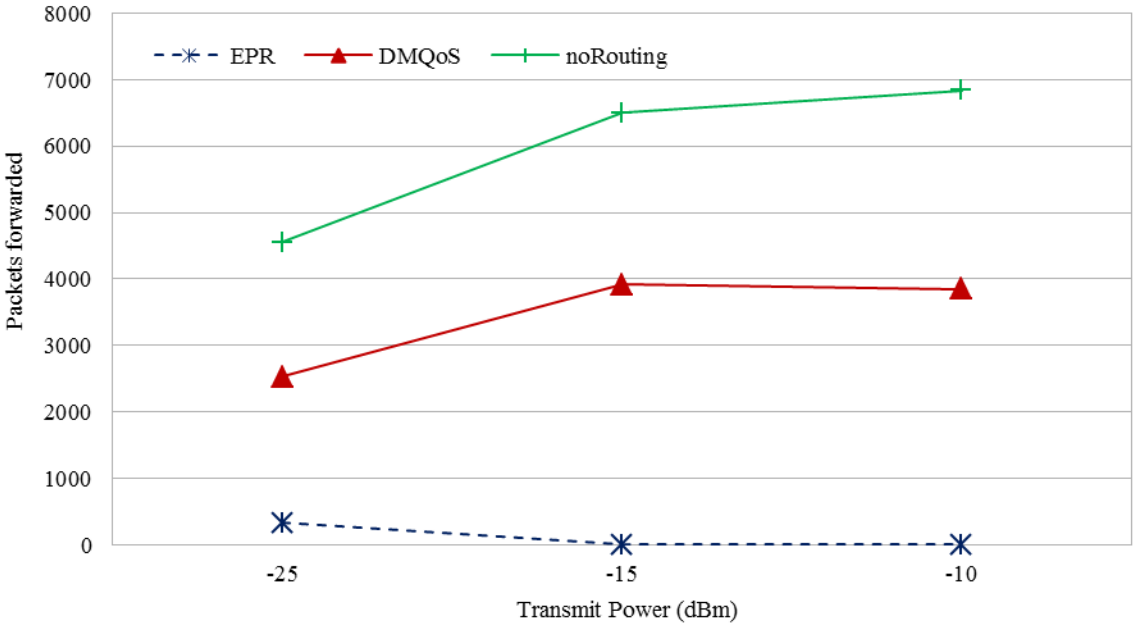


Figure 5-11: Packets forwarded by intermediate nodes

In comparison, there are 2526, 3922, and 3849 packets forwarded by intermediate nodes in DMQoS for the transmit powers of -25dBm, -15dBm, and -10dBm respectively. The packets forwarded by intermediate nodes in noRouting are 4553, 6497, and 6838 for transmit powers of -25dBm, -15dBm, and -10dBm respectively. Due to the reduced numbers of broadcast Hello packets and fewer data packets forwarded by intermediate nodes, EPR results in reduced network traffic load and overall energy consumption as shown in Figure 5-12 and Figure 5-13 respectively.

Figure 5-12 shows that the traffic load reduction in EPR as compared to DMQoS is 64%, 91%, and 87% when the transmit power is -25dBm, -15dBm, and -10dBm respectively. The network traffic load in noRouting is almost double that of the network traffic load in EPR for all three transmit powers. The energy consumption of EPR is 9381 mJ, 9335 mJ, and 9372 mJ when the transmit power is -25dBm, -15dBm, and -10dBm respectively as shown in Figure 5-13. For the same transmit powers, DMQoS consumes 9474 mJ, 9536 mJ, and 9588 mJ energies. The protocol noRouting needs higher energies i.e. 9570 mJ, 9626 mJ, and 9692 mJ energies when transmit power is -25dBm, -15dBm, and -10dBm respectively.

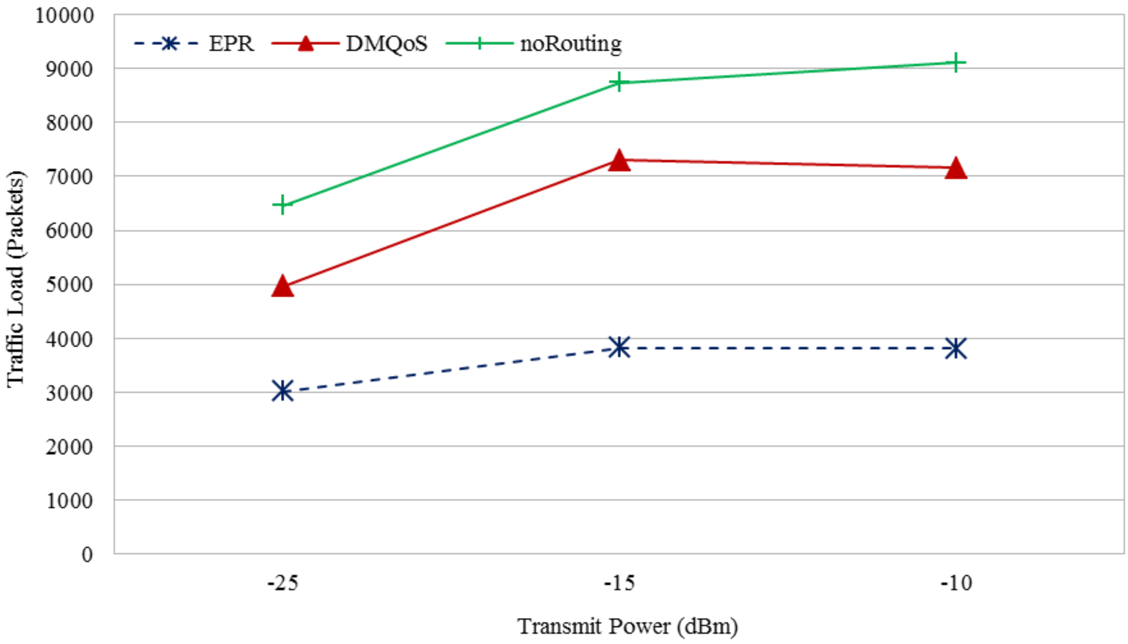


Figure 5-12: Network traffic load

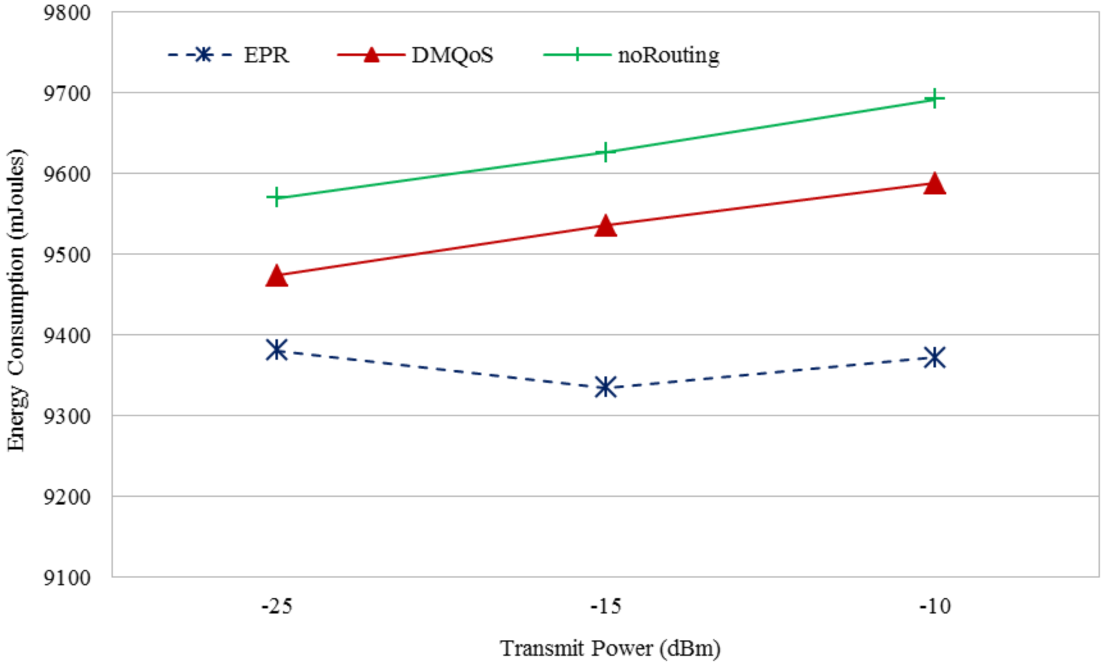


Figure 5-13: Overall energy consumption

The energy saved by all nodes in EPR is 93mJ, 201mJ, and 216mJ for the transmit power of -25dBm, -15dBm, and -10dBm respectively, as shown in Figure 5-14.

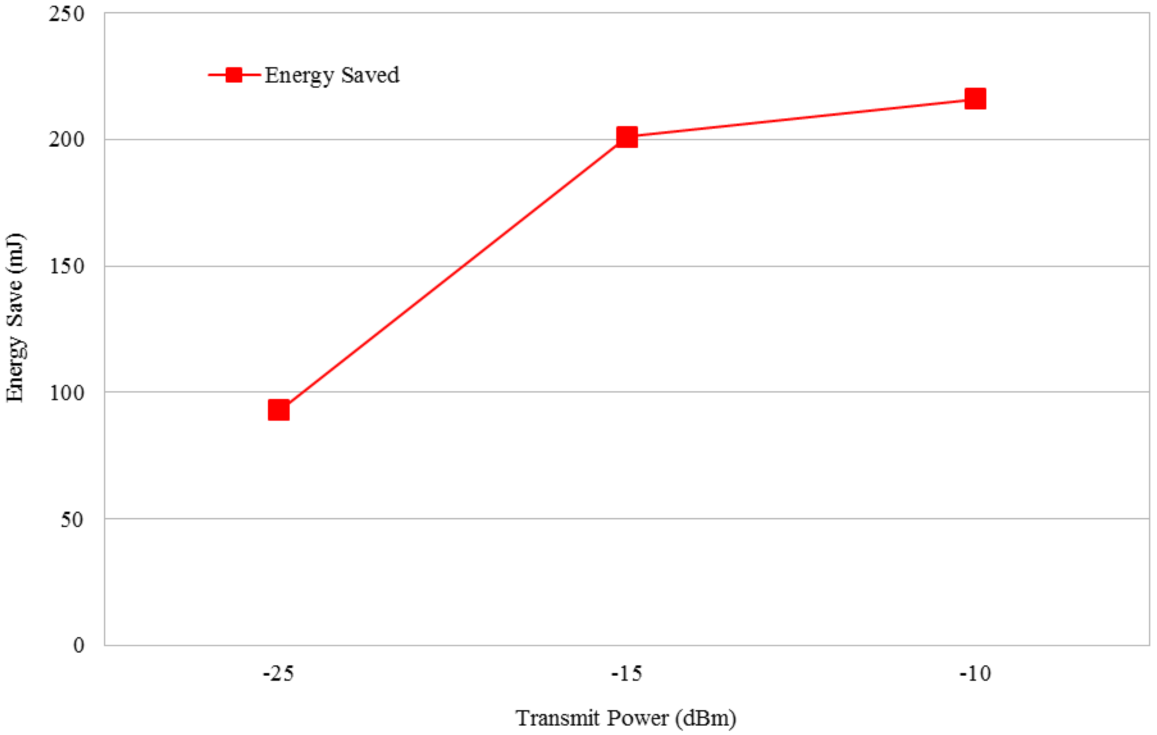


Figure 5-14: Energy save in EPR vs DMQoS

The packets dropped due to the MAC buffer overflow is negligible in EPR and DMQoS when transmit power is -25dBm as shown in Figure 5-15. EPR drops 675 packets as compared to 952 and 1767 packets dropped by DMQoS and noRouting respectively for the transmit power of -15dBm. EPR, DMQoS, and noRouting drop 748, 995, and 1800 packets respectively when transmit power is -10dBm. It is seen from Figure 5-15 that EPR drops fewer number of packets compared to the other two methods for all transmit power levels.

The consequent reduction in overall reduced BAN traffic increases the probability of successful data transmission. The successful transmission rate (throughput) is shown in Figure 5-16. The successful transmission rate in EPR as compared to DMQoS has increased by 6%, 11%, and 13% when the transmit power is -25dBm, -15dBm, and -10dBm respectively. EPR delivers 19%, 40%, and 39% more packets than noRouting for transmit power of -25dBm, -15dBm, and -10dBm respectively. It is observed that EPR delivered more packets successfully than DMQoS and noRouting at all transmit power levels.

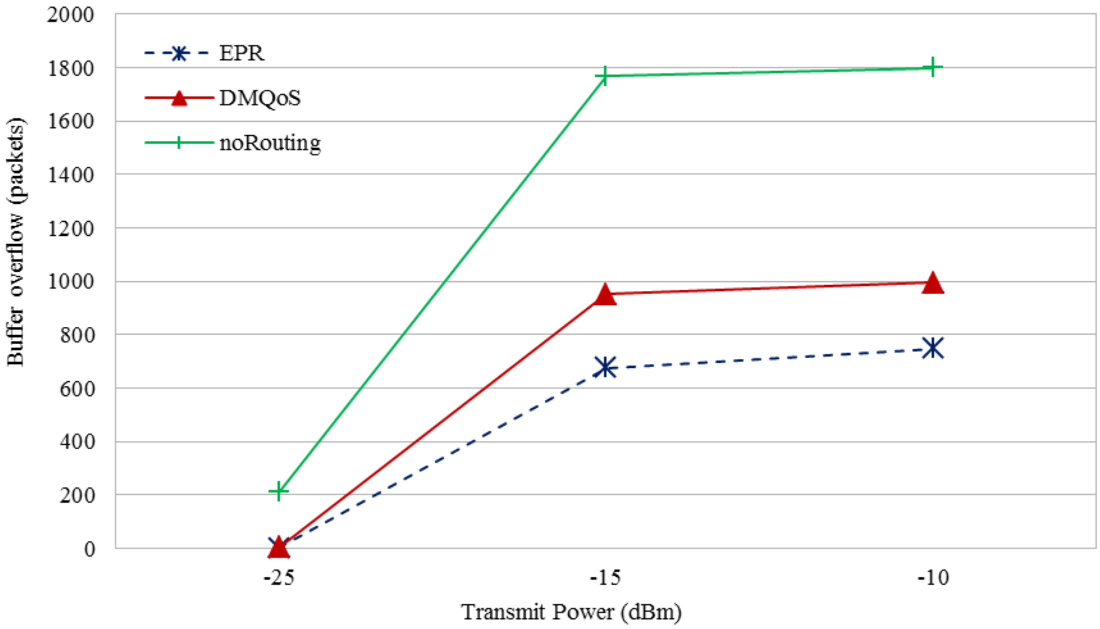


Figure 5-15: No. of packets dropped due to MAC buffer overflow

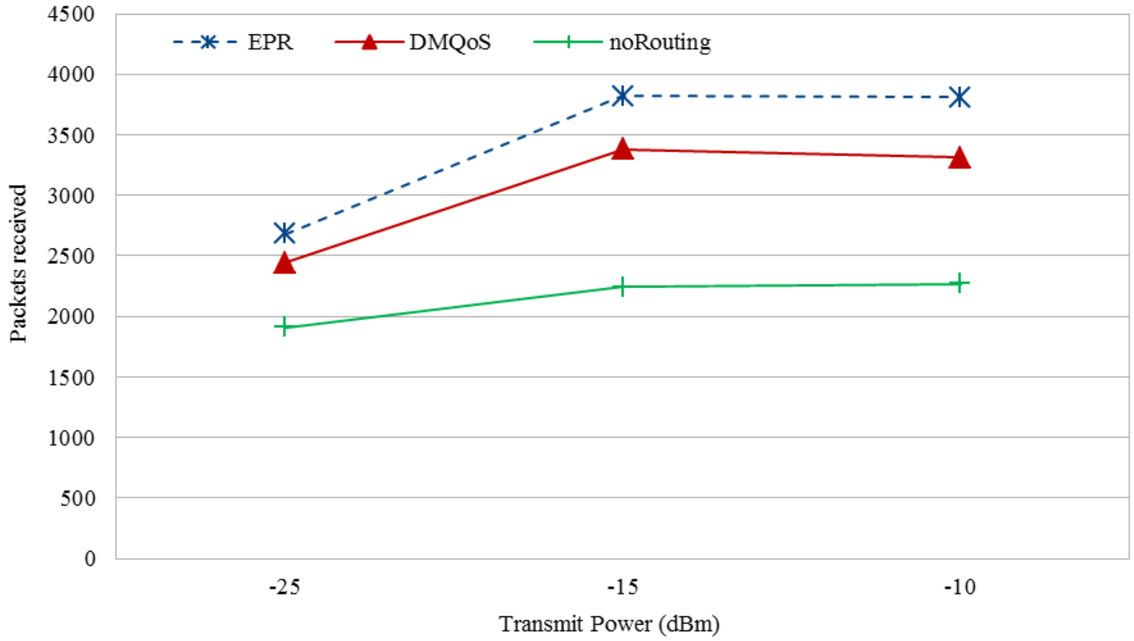


Figure 5-16: Throughput vs Transmit powers

5.5. Scenario 2

The devices (BANs) B_1, B_2, B_3, and B_4 are considered as source nodes and devices (NSC and MDCs) are the destination nodes. B_1 sends packets to MDC_1, B_2 sends packets to MDC_2, B_3 sends packets to MDC_3, and B_4 sends packets to NSC. The data of B_4 has to go through the other devices to reach NSC.

5.5.1. Performance Comparison of Scenario 2

The source nodes send a total of 80K packets. The throughput, packets forwarded by intermediate nodes, network traffic load, MAC buffer overflow, and end-to-end delay (latency) are observed and recorded after every 4000 packets until 28K are transmitted and thereafter when 40K, 60K, and 80K packets are sent by all BANs.

5.5.1.1. Throughput

From Figure 5-17 it is observed that for low transmit power of -25dBm, EPR maintains its throughput from 67% to 76%; whereas, DMQoS provides throughput from 61% to 67% and noRouting has the smallest throughput of 47% to 51%.

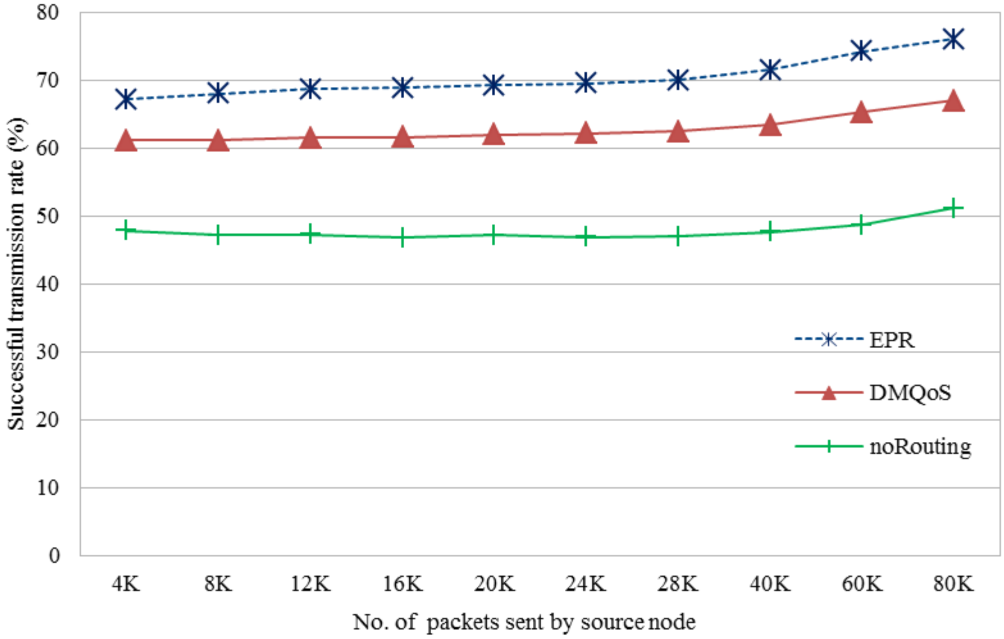


Figure 5-17: Throughput vs offered load when transmit power = -25 dBm

When transmit power is -15dBm and -10dBm, as shown in Figure 5-18 and Figure 5-19 respectively, EPR provides consistently throughput of 95%. Whereas, DMQoS has a lower throughput ranging from 83% to 88% and 82% to 88% for the transmit power of -15dBm and -10dBm, respectively. The throughput of noRouting is the smallest and ranges from 56% to 66% for both transmit powers of -15dBm and -10dBm.

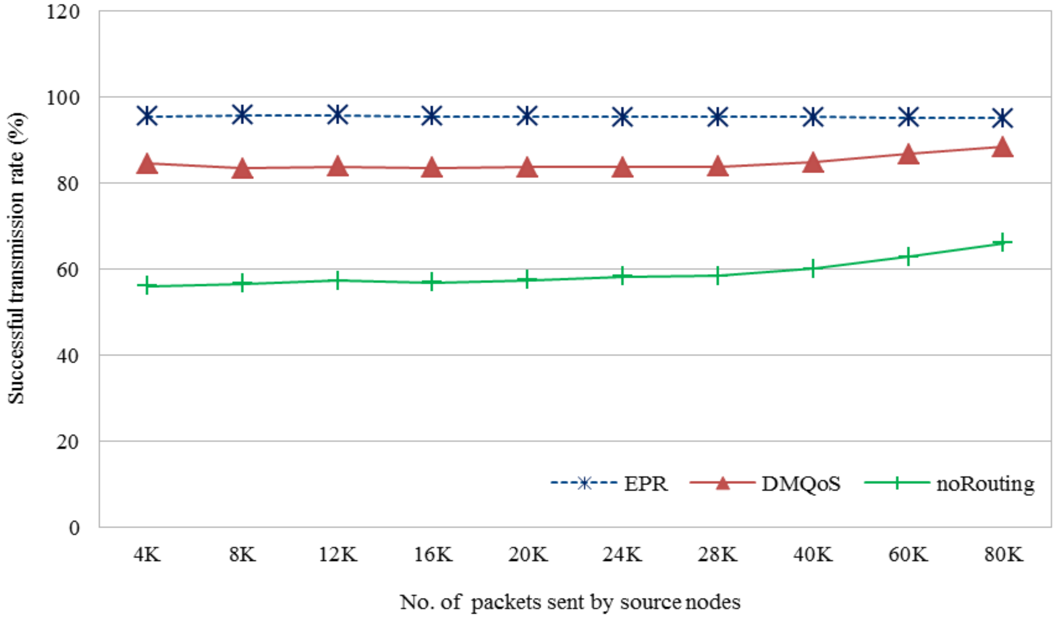


Figure 5-18: Throughput vs Offered load when transmit power is -15 dBm

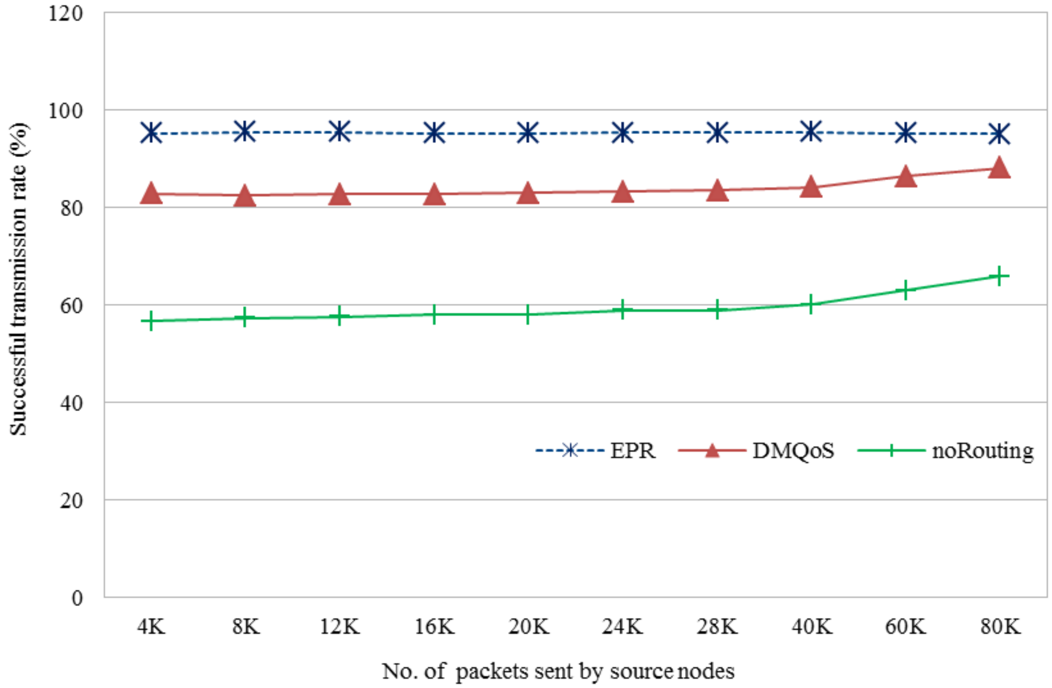


Figure 5-19: Throughput vs Offered load when transmit power is -10 dBm

5.5.1.2. Packets Forwarded by Intermediate Nodes

Figure 5-20 shows the number of packets forwarded by intermediate nodes. In the EPR protocol, when the transmit power is -25dBm and for 4K to 80K packets sent from source nodes, 332 to 7843 packets are forwarded by the intermediate nodes. In comparison, the intermediate nodes in DMQoS and noRouting forward 2.5K to 55.5K packets and 4.5K to 97.7K packets respectively as shown in Figure 5-20. The improved performance values can be attributed to the fact that unlike DMQoS [33] which sends the data to the closest neighbor node, EPR chooses the most appropriate next hop node based on the lowest energy communication cost. The BAN coordinator in the proposed EPR sends data to another BAN coordinator only if it is necessary. The BAN coordinators send the data packets directly to the destinations when the transmit power is -15dBm or higher.

Figure 5-21 and Figure 5-22 show the number of packets forwarded by intermediate nodes when the transmit power is -15dBm and -10dBm. It is seen from the above figures that EPR does not forward any packet as compared to DMQoS and noRouting in which the intermediate nodes forward 4K to 87K and 6.5K to 160K packets respectively, for the same 4K to 80K packets sent by the source nodes.

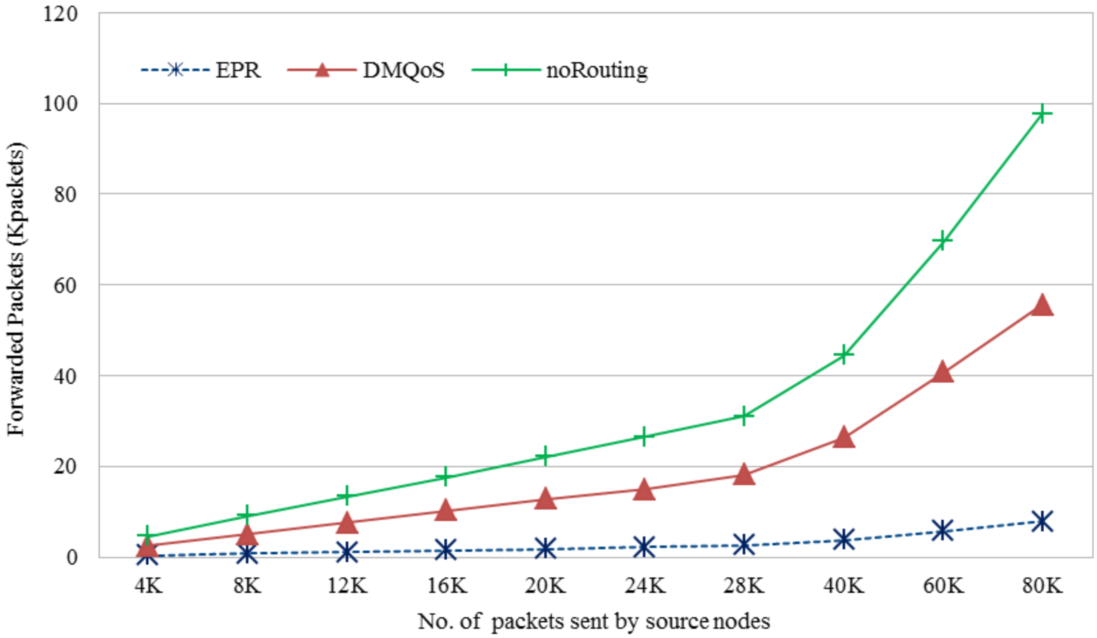


Figure 5-20: Packets forwarded by intermediate nodes vs Offered load when transmit power is -25 dBm

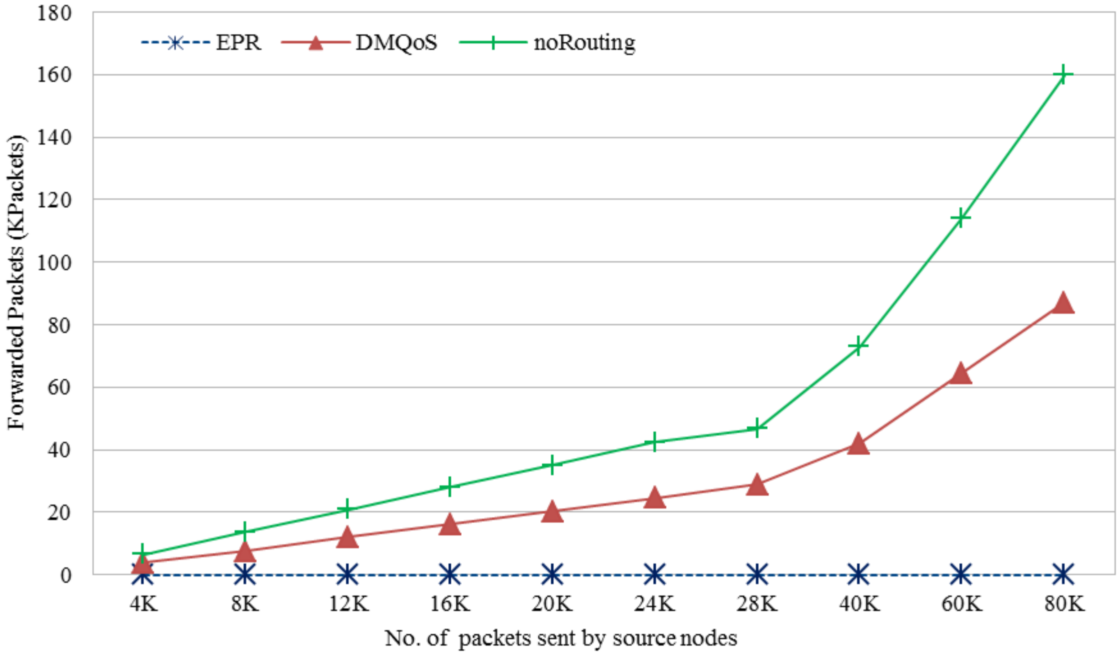


Figure 5-21: Packets forwarded by intermediate nodes vs Offered load when transmit power is -15 dBm

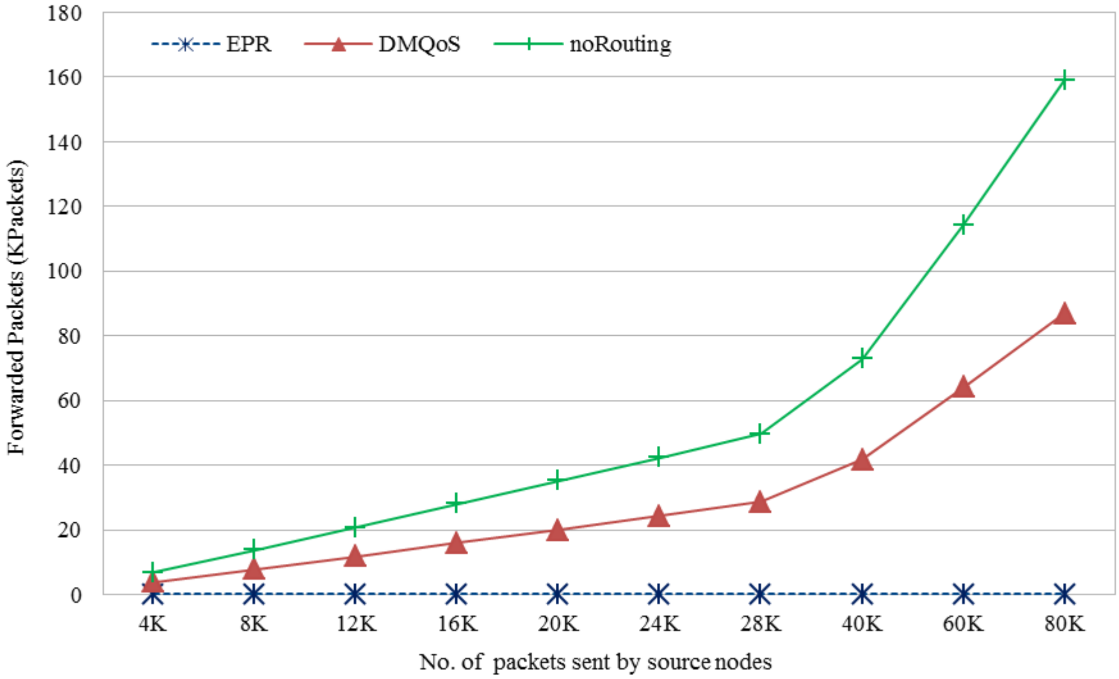


Figure 5-22: Packets forwarded by intermediate nodes vs Offered load when transmit power is -10 dBm

The source node in EPR sends the data directly to the destination if destination is in range. On the other hand DMQoS forwards the packets to the nearest upstream neighbor nodes. Figure 5-20 shows that due to the low transmit power of -25dBm; EPR sends data packets to the next hop intermediate nodes because destination nodes are not in range. In summary, some of the BANs in EPR must send the data packets to the destination through an intermediate node when the transmit power is less than -15dBm.

5.5.1.3. Network Traffic

The exchange of Hello packets is used to update the routing tables of the nodes. The number of Hello packets and the number of packets forwarded by the intermediate nodes affect the total network traffic which results in higher energy consumption. The improved Hello packet mechanism employed in EPR reduces the number of Hello packets and the data packets forwarded by intermediate nodes which results in lower overall network traffic than DMQoS. For 80K packets, EPR generates about 37%, 52%, and 52% less network traffic when transmit powers of -25dBm, -15dBm, and -10dBm are used respectively, compared to DMQoS as shown in Figure 5-23, Figure 5-24, and Figure 5-

25. Figure 5-23 compares the performance of EPR with DMQoS and noRouting in terms of network traffic. It is seen that EPR has a 0% to 27% lower network traffic for the low to high offered traffic load when the transmit power is -25dBm.

From Figure 5-23, Figure 5-24, and Figure 5-25, it is observed that the network traffic generated by noRouting is more than double that of EPR generated network traffic for all transmit powers. Due to the lower transmit power; more data packets need to go through the intermediate nodes before reaching the destination node. Figure 5-24 and Figure 5-25 show that EPR consistently reduces the network traffic by 52% for all offered traffic when the transmit powers are -15dBm and -10dBm.

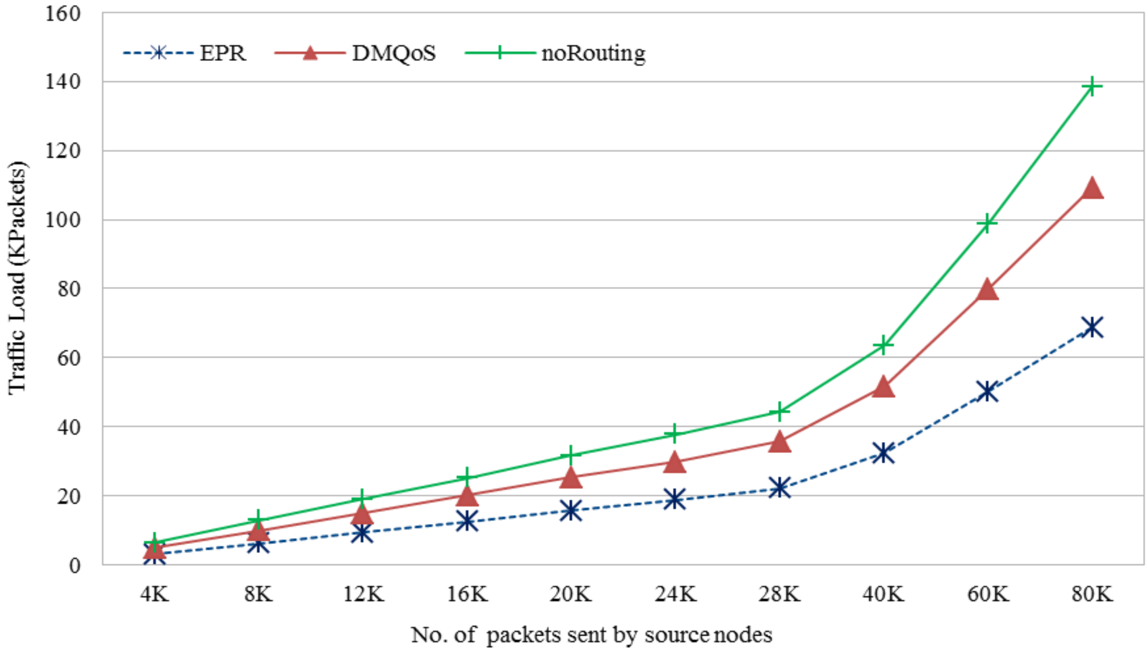


Figure 5-23: Network traffic vs Offered load when transmit power is -25 dBm

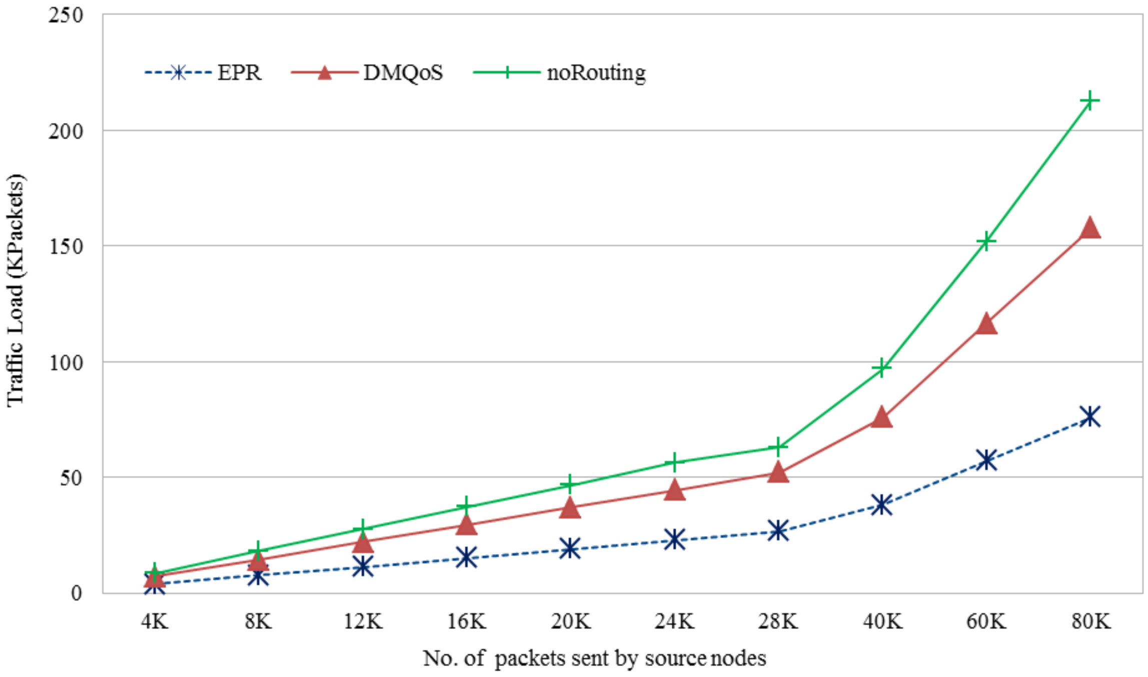


Figure 5-24: Network traffic load vs Offered load when transmit power is -15 dBm

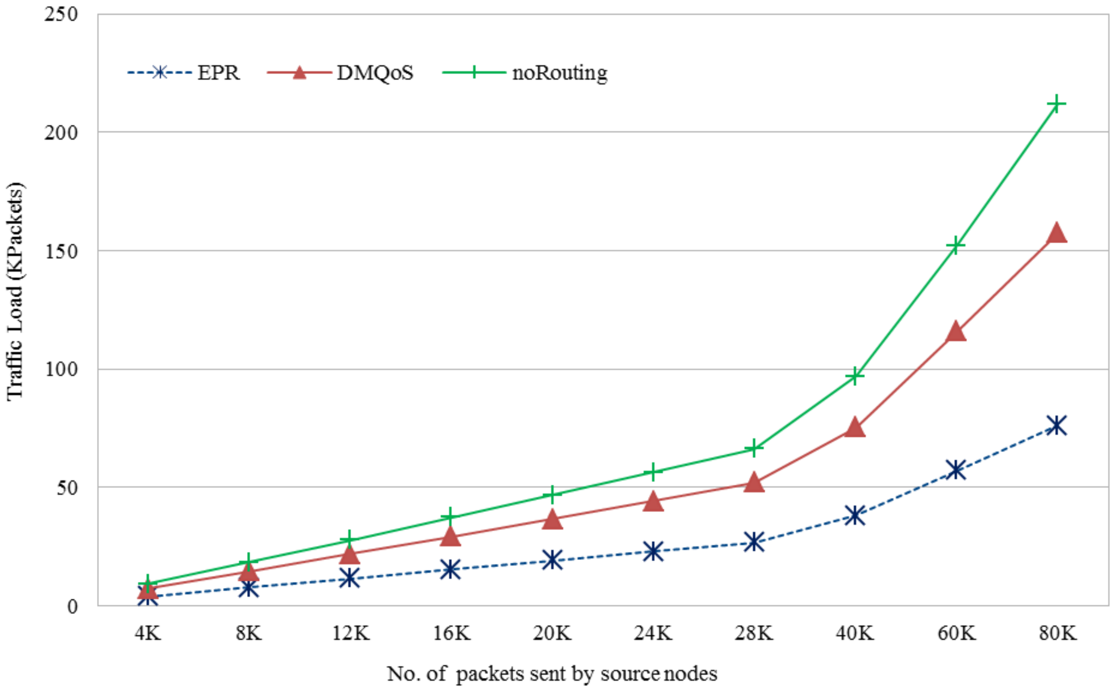


Figure 5-25: Network traffic load vs Offered load when transmit power is -10 dBm

5.5.1.4. MAC Buffer Overflow

The buffer overflow as a function of offered traffic is shown in Figures 5-26, 5-27, and 5-28. Figure 5-26 shows that there is no buffer overflows in EPR for transmit power -25dBm. DMQoS performs well for transmit power of -25dBm by not having any buffer overflow; whereas, 0.2K to 2.4K packets are dropped due to buffer overflow in noRouting when 4K to 80K packets were sent by source nodes.

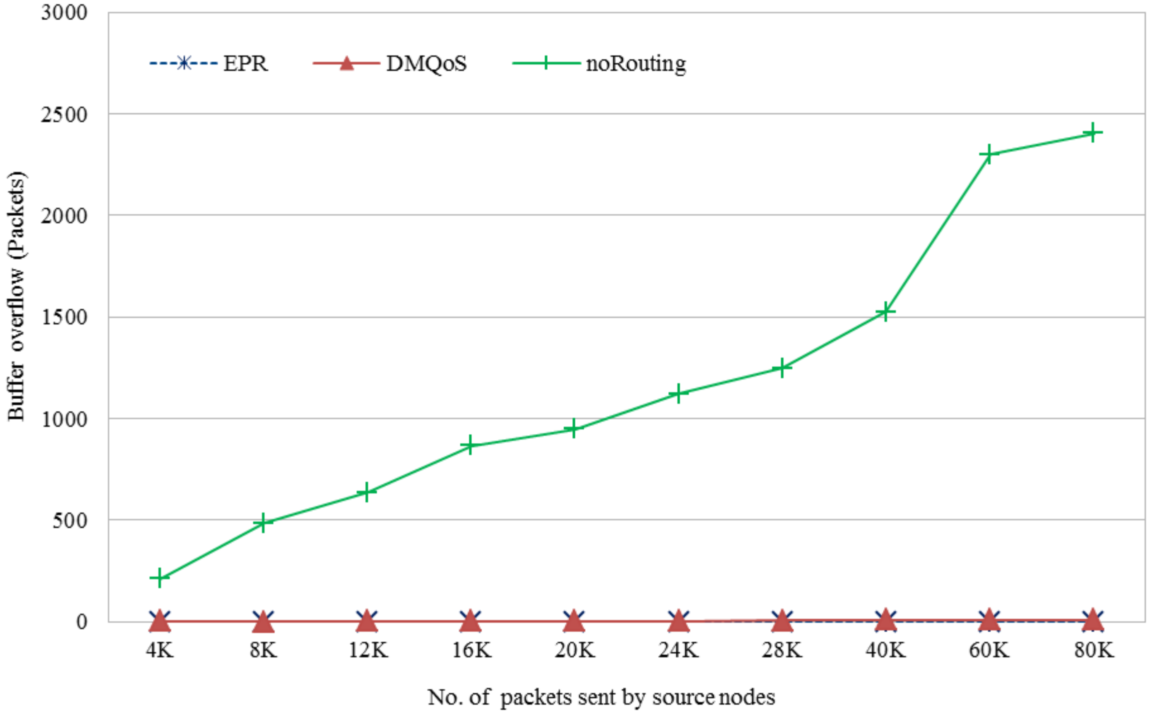


Figure 5-26: MAC buffer overflow vs Offered traffic when transmit power is -25 dBm

However, for high transmit powers of -15dBm and -10dBm, due to the traffic congestion, MAC buffer overflow causes 0.7K to 22K packets to be dropped in EPR and 0.9K to 15K packets dropped in DMQoS as shown in Figure 5-27 and Figure 5-28. EPR drops 16% to 38% less packets than DMQoS for low to medium-high traffic load. The packets dropped in noRouting are very high for all transmit powers. EPR performs well in terms of packets dropped for low transmit power and is in keeping with the recommended transmit power of BAN nodes in hospital environment of -25dBm.

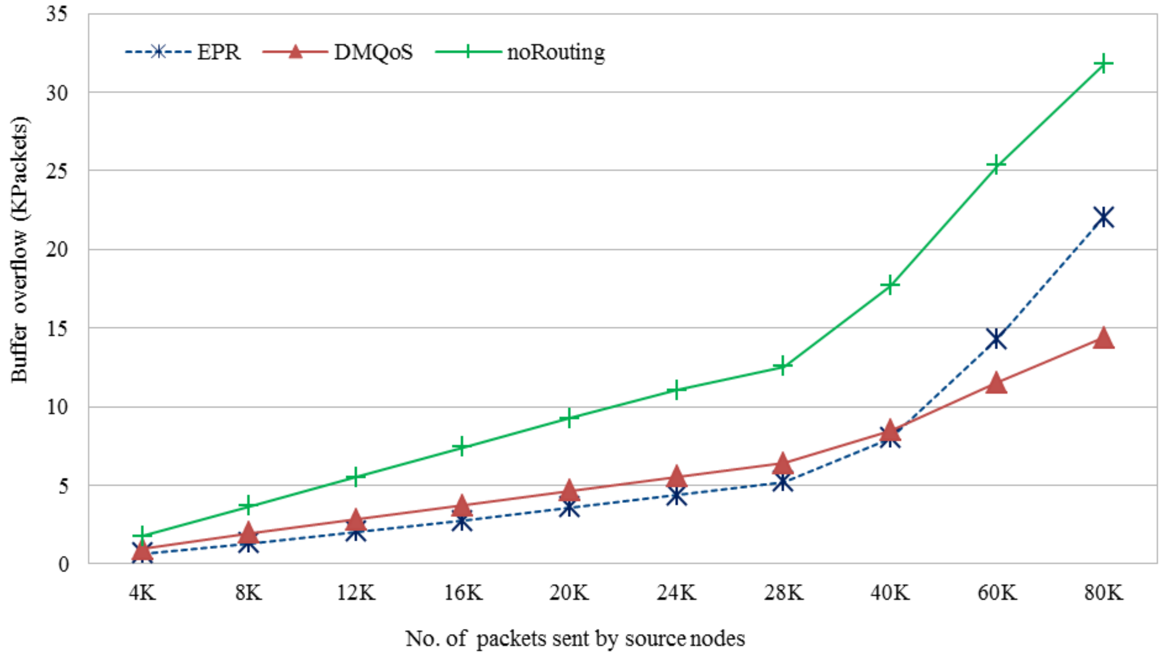


Figure 5-27: MAC buffer vs Offered traffic when transmit power is -15 dBm

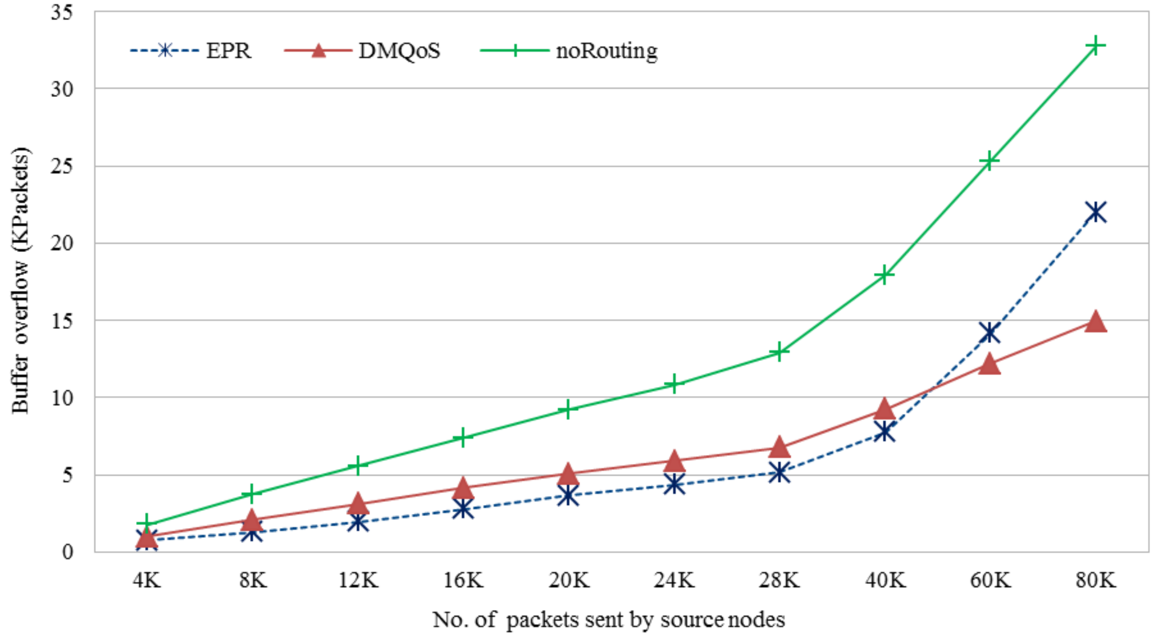


Figure 5-28: MAC buffer vs Offered traffic when transmit power is -10 dBm

In summary, since the recommended transmit power for BAN in hospital environment is -25dBm, we observed that the results of EPR in terms of transmission rate, forwarded packets by intermediate nodes, network traffic load, and packets dropped due to buffer overflow (as discussed above) are better than that for DMQoS and noRouting when transmit power is -25dBm. Hence, it proved that EPR is an ideal candidate for use in such environment as compared to DMQoS and noRouting.

5.5.1.5. End-to-End Delay (Latency)

The amount of time taken by the packets from source to destination is the latency. Latency is an important metric to measure the performance of a network. Extensive simulations were performed to measure the efficiency of the proposed routing protocol. The source nodes send a total of 80K packets in Scenario 2. The latency is calculated after every 20 ms for the transmission of every 4K packets until 28K and then at 40K, 60K, and 80K packets. In this section, the latencies of lower traffic loads (i.e. 4K and 8K), medium traffic loads (i.e. 28K and 40K) and high traffic loads (i.e. 60K and 80K) are observed for each transmit power of -25dBm, -15dBm and -10dBm. Figure 5-28 and 5-29 show the latency values for static source nodes when transmit power is -25dBm and lower traffic is sent. It is observed that EPR delivers more packets with lower delays than other similar protocols for all the time intervals.

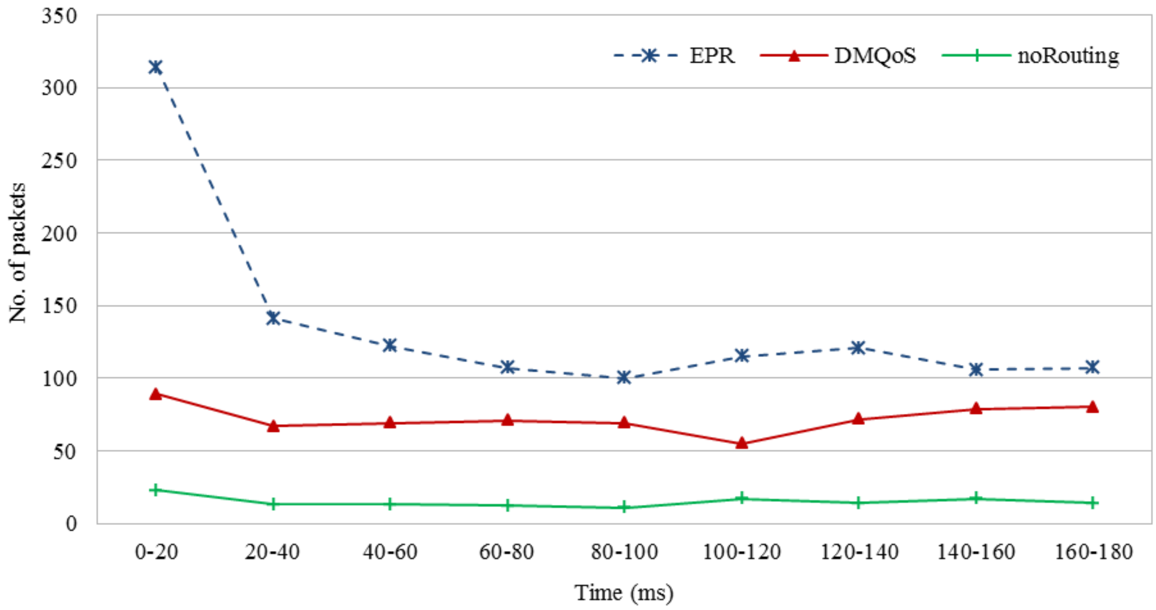


Figure 5-29: Latency for 4K packets sent by source nodes when transmit power is -25 dBm

The number of packets successfully received by the destinations after every 20ms for lower traffic loads are shown in Figure 5-29 to Figure 5-30. Figure 5-29 shows that EPR delivers 314 packets in comparison to 89 and 23 packets delivery in DMQoS and noRouting, respectively for the first interval of time 0-20ms. The energy communication cost based routing mechanism used in EPR helps to deliver the packets in a shorter time interval. The number of packets delivered by EPR for both 4K and 8K traffic is higher for all time intervals such as 20-40ms, 40-60ms, 60-80ms, 80-100ms, 120-140ms, 140-160ms, 160-180ms, and 180-200ms.

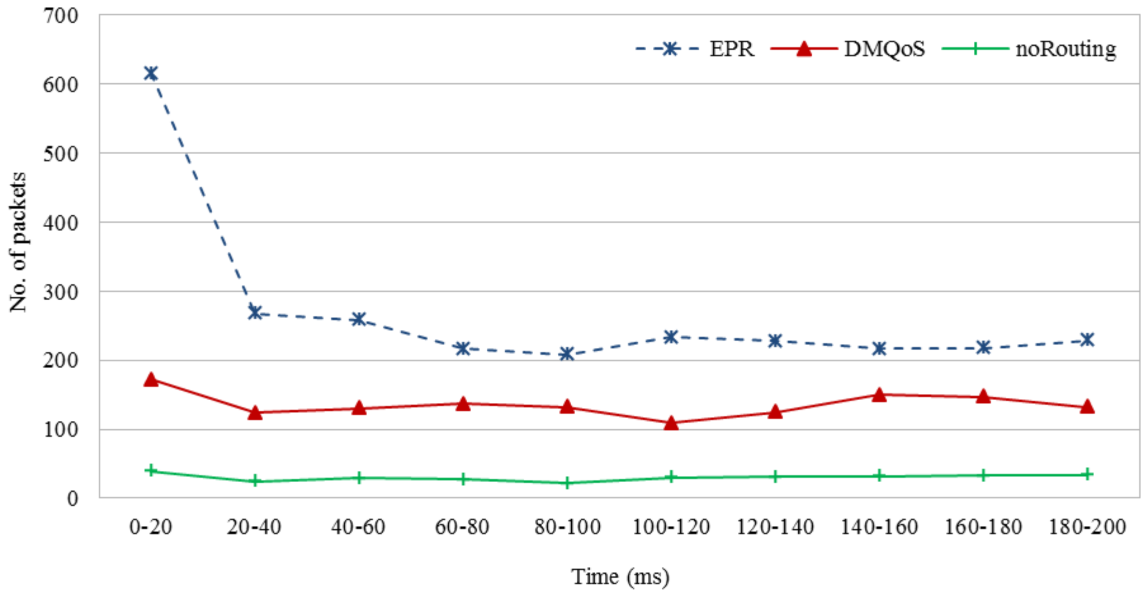


Figure 5-30: Latency for 8K packets sent by source nodes when transmit power is -25 dBm

Figure 5-31 and Figure 5-32 show the number of packets delivered by all three protocols for the 20ms intervals of time when medium traffic load of 28K and 40K is sent. EPR delivers 1915 packets in first 20 ms interval, on the other hand for the same time interval, DMQoS and noRouting deliver successfully 598 and 152 data packets to the destinations, respectively as shown in Figure 5-31.

To summarize, EPR delivers more packets with much lower per packet latency which is average 0.013ms per packet latency compared to 0.027ms and 0.088ms per packet for DMQoS and noRouting respectively for all time intervals when 40K packets are sent by source nodes as shown in Figure 5-32.

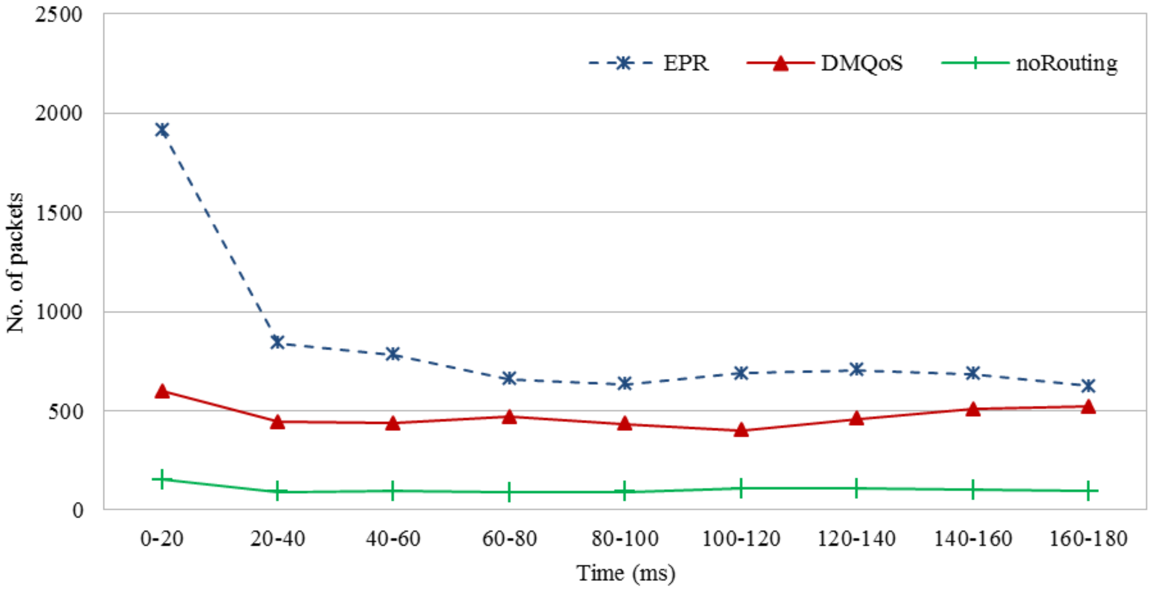


Figure 5-31: Latency for 28K packets sent by source nodes when transmit power is -25 dBm

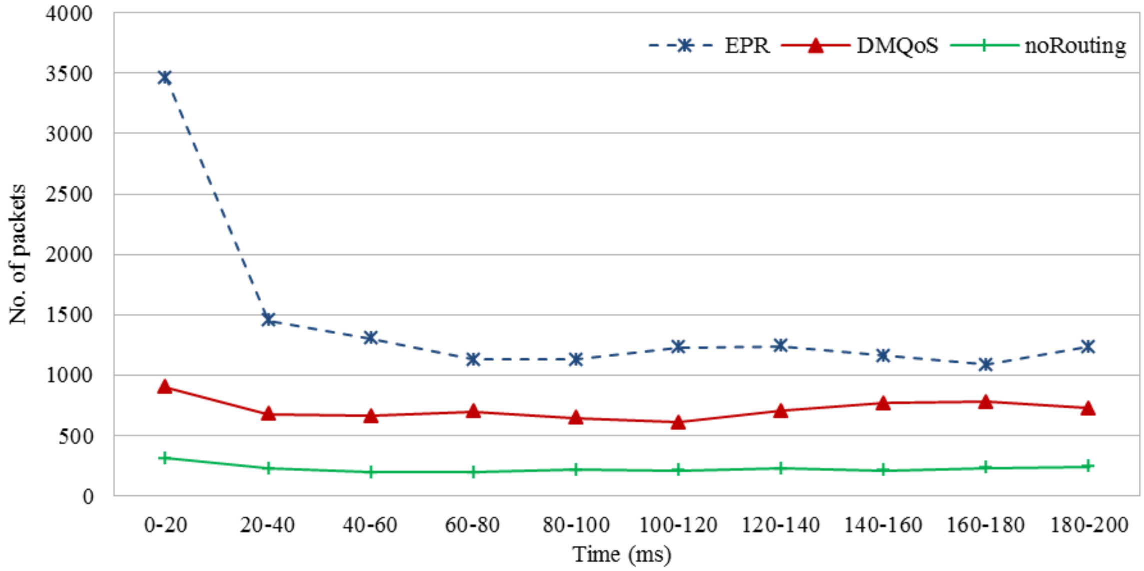


Figure 5-32: Latency for 40K packets sent by source nodes when transmit power is -25 dBm

The successful data packets delivery for higher traffic load sent by source nodes (i.e. 60K and 80K) is shown in Figure 5-33 and Figure 5-34. It is observed that EPR, DMQoS, and noRouting deliver 5.8K, 1.5K, and 0.6K packets respectively, in first 20ms time interval when the packets sent by source nodes are 60K as shown in Figure 5-33.

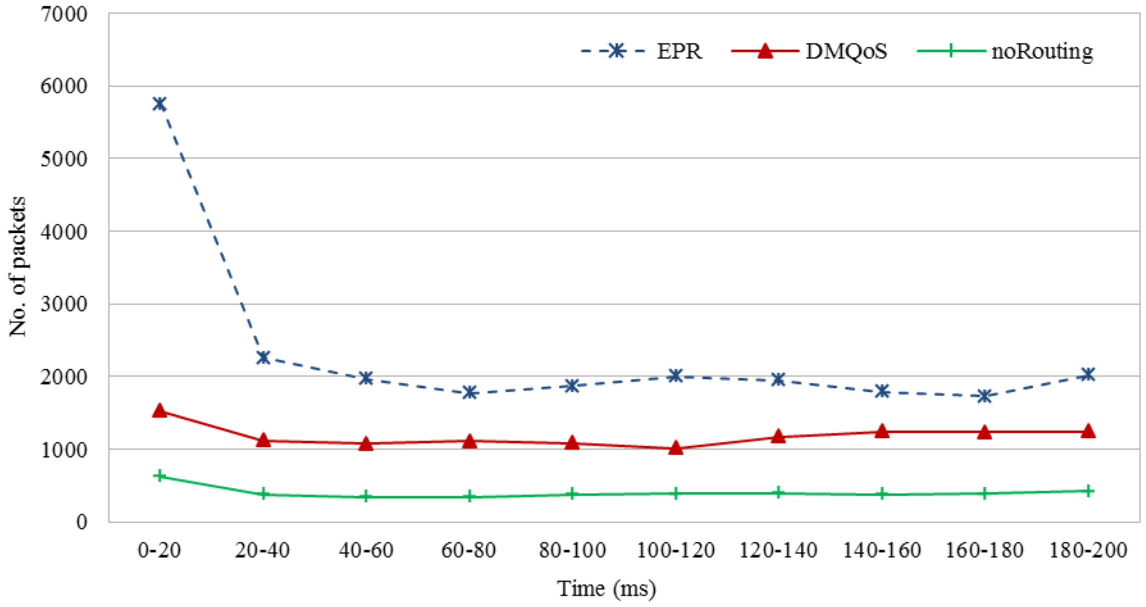


Figure 5-33: Latency for 60K packets sent by source nodes when transmit power is -25 dBm

Figure 5-34 shows that EPR out performs the other two routing protocols for all time intervals. EPR delivers 8.25K packets for the first time interval (i.e. 20ms) which is much higher than the packet delivery 2.2K and 1K packets by DMQoS and noRouting respectively.

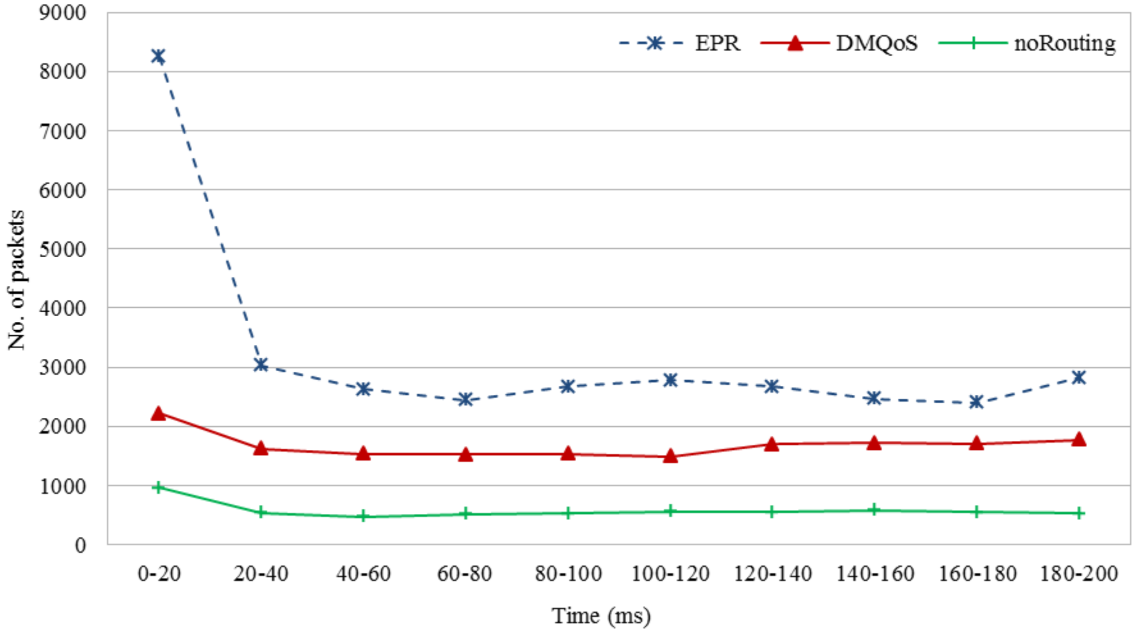


Figure 5-34: Latency for 80K packets sent by source nodes when transmit power is -25 dBm

The number of packets delivered to the destination nodes are counted after every 20ms for the transmit power of -15dBm when packets (low load 4K, medium load 40K and high traffic load 80K) are sent by source nodes. The Figure 5-35 shows that EPR delivers 347 packets as compare to 17 and 13 packets delivered by DMQoS and noRouting protocols respectively, for first time interval (i.e. 0-20ms). EPR outperforms by delivering on average 10 times more packets than DMQoS, and 67 times more packets than noRouting protocol, in all the other time intervals.

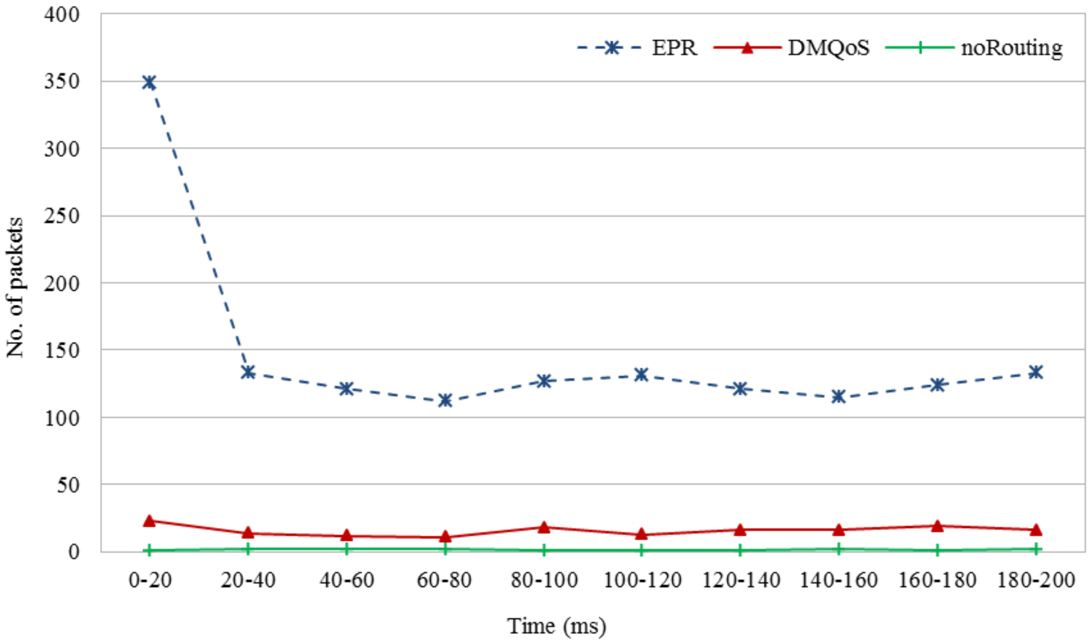


Figure 5-35: Latency for 4K packets sent by source nodes when transmit power is -15 dBm

It is seen from Figure 5-36 to Figure 5-37 that EPR outerperforms DMQoS and noRouting for medium and high load traffic sent by source nodes when the transmit power is -15dBm. Figure 5-36 shows that EPR delivers 57 and 112 times more packets than DMQoS and noRouting for first 20ms. After the first interval, the packets received by destinations per interval in EPR are on average 14 times and 70 times more than the DMQoS and noRouting protocols respectively.

Figure 5-37 shows the packets received by destinations when high traffic loads (i.e. 80K) is generated from source nodes and transmit power is -15dBm. EPR delivers 21 times and 33 times more packets than DMQoS and noRouting in the first 20ms. After the first interval, the packets received by destinations per interval in EPR are on average 6 times and 21 times more than the DMQoS and noRouting protocols, respectively.

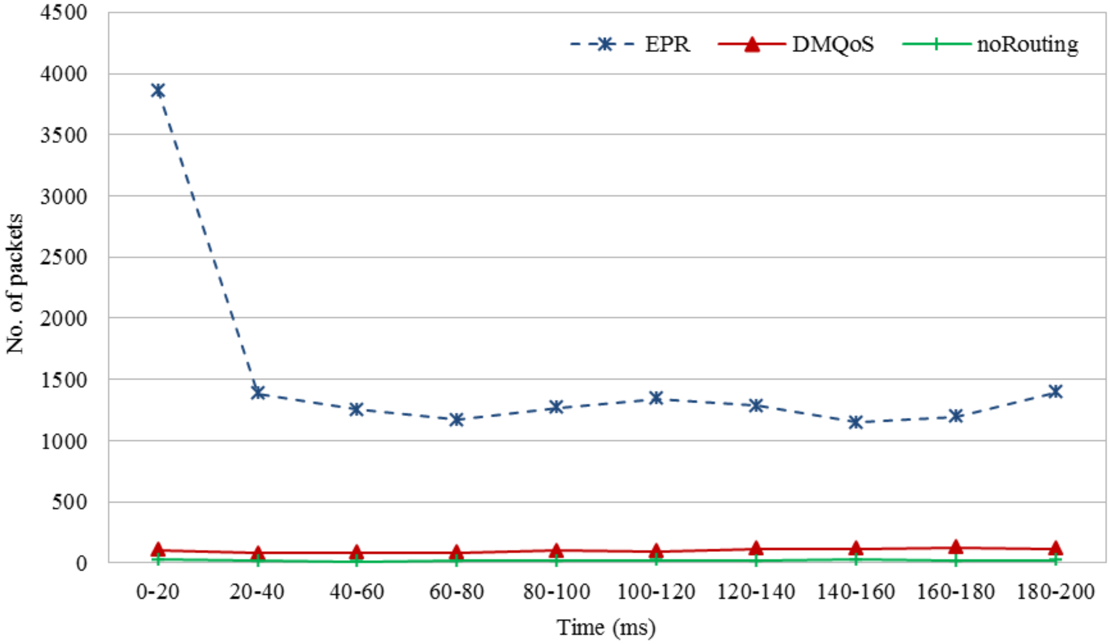


Figure 5-36: Latency for 40K packets sent by source nodes when transmit power is -15 dBm

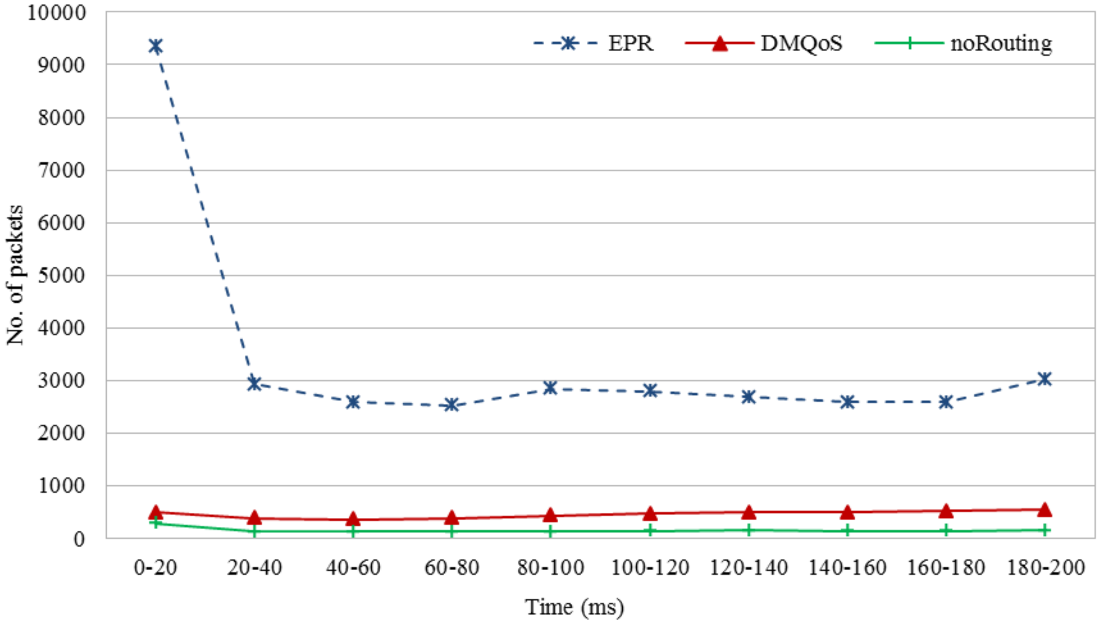


Figure 5-37: Latency for 80K packets sent by source nodes when transmit power is -15 dBm

Figure 5-38 to Figure 5-40 show the number of packets delivered by EPR, DMQoS and noRouting protocols after every 20ms interval of time when low, medium and high traffic loads are sent by source nodes and the transmit power is -10dBm. Figure 5-38 shows that

EPR delivers 15 times and 345 times more packets than DMQoS and noRouting in the first 20ms. After the first interval, the packets received by dentinations in EPR are on average 9 times and 90 times more than the DMQoS and noRouting protocols respectively.

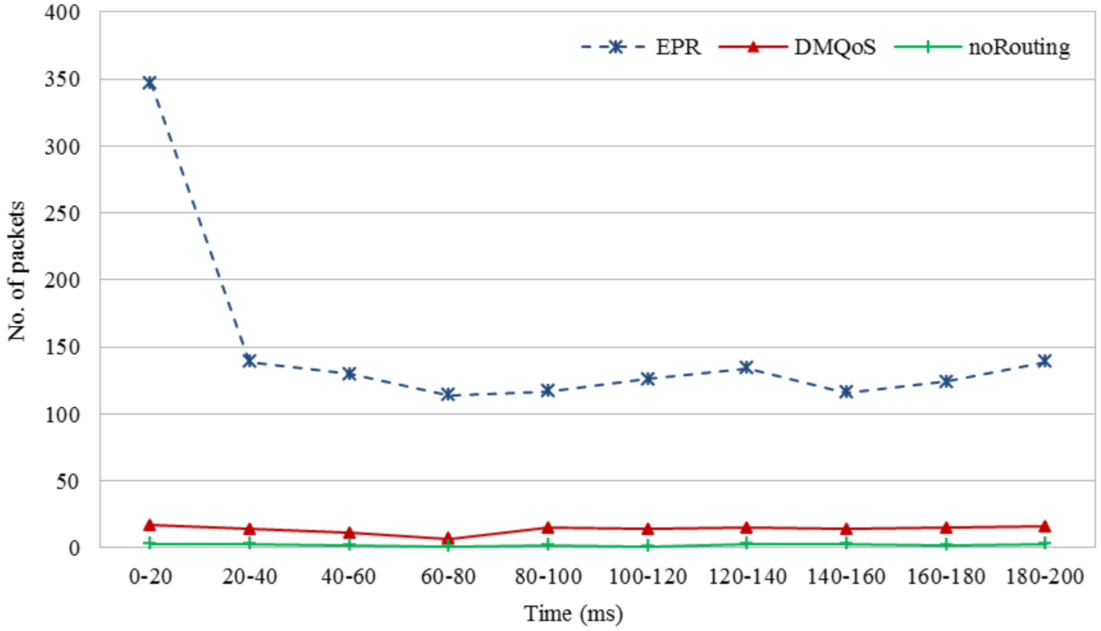


Figure 5-38: Latency for 4K packets sent by source nodes when transmit power is -10 dBm

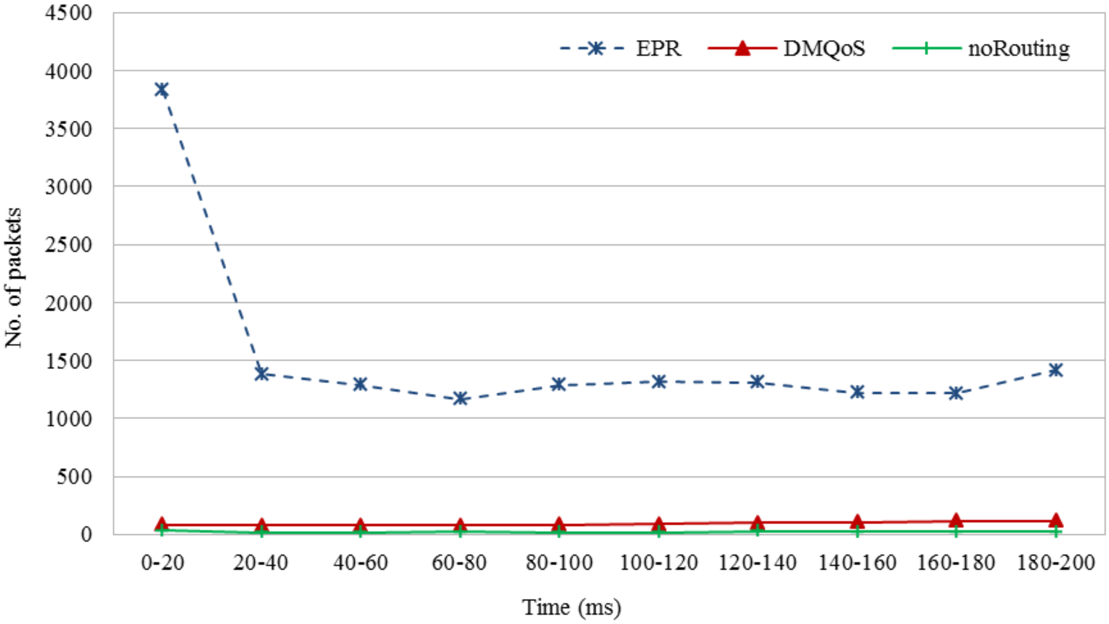


Figure 5-39: Latency for 40K packets sent by source nodes when transmit power is -10 dBm

Figure 5-39 shows the numbers of packets received by all three protocols when medium traffic load is sent (or offered) by source nodes. EPR delivers 37 times and 137 times more packets than DMQoS and noRouting respectively, in the first time interval of 20ms. After the first interval, the packets received by dentinations per interval in EPR are on average 12 times and 69 times more than the DMQoS and noRouting protocols respectively.

Figure 5-40 shows the packets received by destinations when high traffic load (i.e. 80K packets) is (offered) sent by the source nodes and the transmit power is -10dBm. The packets delivered by EPR in the first 20ms are 19 times and 33 times more than DMQoS and noRouting respectively. After the first interval, EPR delivers on average 6 times and 20 times more packets than the DMQoS and noRouting protocols respectively.

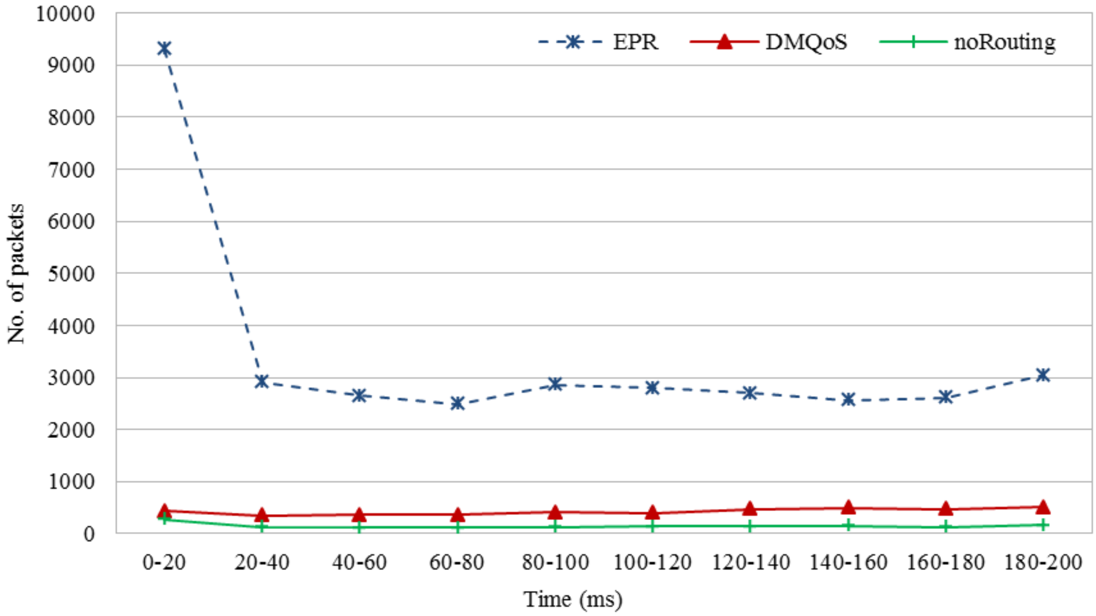


Figure 5-40: Latency for 80K packets sent by source nodes when transmit power is -10 dBm

5.6. Scenario 3

In this scenario, the source node B_4 is moving at the speed of 1 meter per second vertically. It is assumed that the speed of a patient walking briskly is 1 meter per second. The throughput, packets forwarded by intermediate nodes, network traffic, MAC buffer overflow, and end-to-end delay are observed during the simulations of this scenario. The transmit power used for this scenario is -25dBm. Once again, it is observed that EPR provides better results than DMQoS and noRouting in case of mobile source node. Figure

5-41 shows that EPR has 70% to 82% throughput as compared to 63% to 66% of DMQoS and 47% to 51% of noRouting.

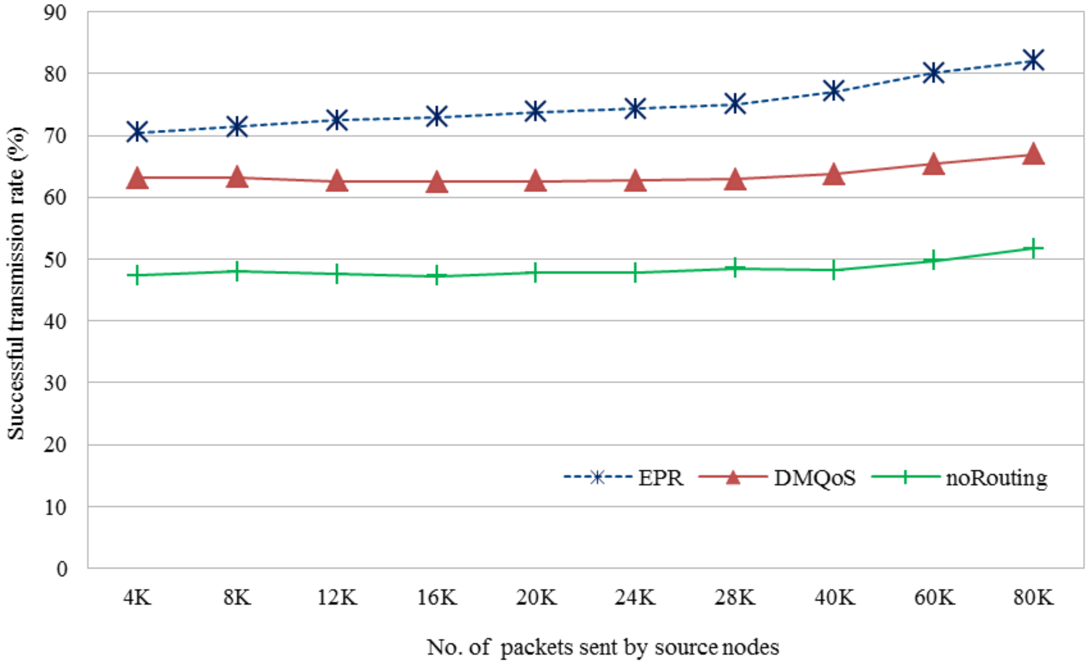


Figure 5-41: Throughput vs Offered load

In EPR, the intermediate nodes forwards 5 times and 8 times fewer packets than DMQoS and noRouting protocols respectively, as shown in Figure 5-42. The lower number of forwarded packets helps to reduce the overall network traffic load.

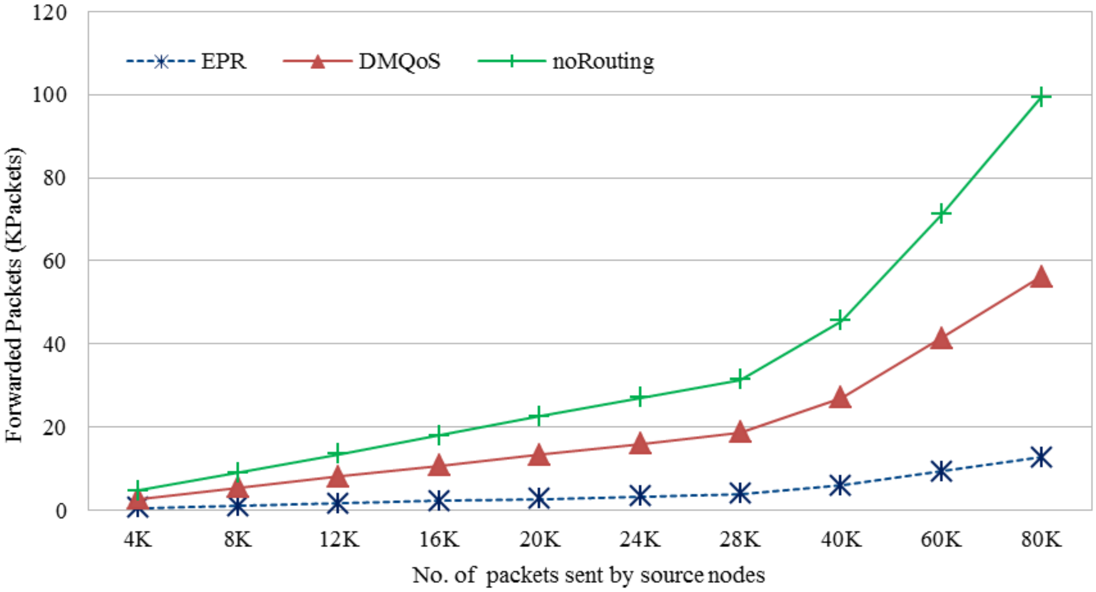


Figure 5-42: Packets forwarded by intermediate nodes vs Offered traffic

Figure 5-43 shows that the network traffic for EPR is on average 1.7 and 2.1 times less than the traffic loads in DMQoS and noRouting protocols, respectively. Figure 5-44 shows that there is no packet dropped by EPR due to the MAC buffer overflow. Only a few packets are dropped in DMQoS; whereas, 2.15K packets are dropped in noRouting.

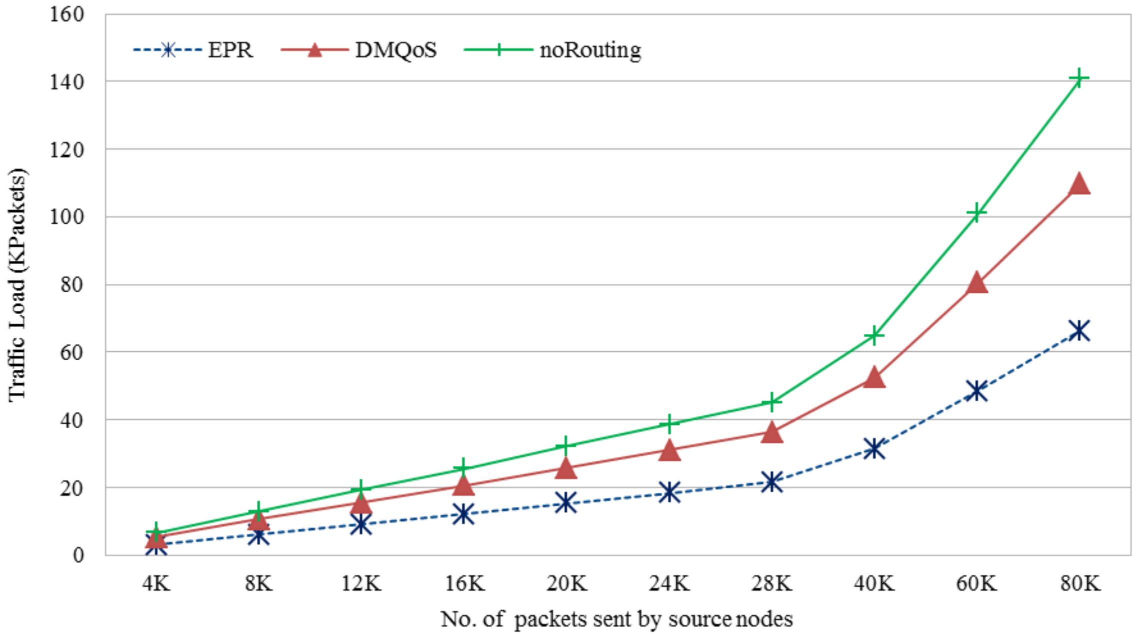


Figure 5-43: Network traffic load vs Offered traffic

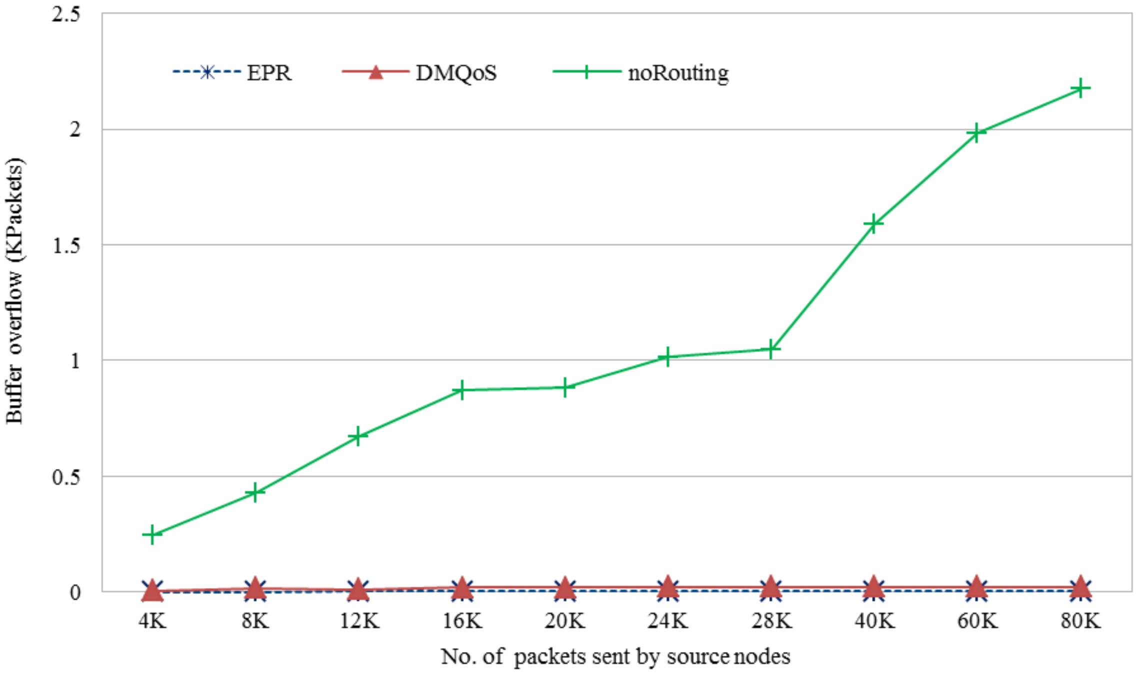


Figure 5-44: MAC buffer overflow vs Offered traffic

The number of packets successfully received by destination nodes is measured when the source node is mobile and the transmit power is -25dBm. Figure 5-45, Figure 5-46 and Figure 5-47 show that EPR provides better results in terms of end-to-end packet delay for all load traffic (low load - 4K, medium load - 40K and high load - 80K) sent by the source nodes. For low traffic load (i.e. 4K packets) as shown in Figure 5-45, EPR results in more numbers of packets deliveries to the destination when compared with DMQoS and noRouting. During first 20ms, EPR delivers 346 packets as compared to 112 and 19 packets delivered by DMQoS and noRoutiong respectively, as shown in Figure 5-47. EPR delivers on average 2 times and 10 times more packets than DMQoS and noRouting protocols respectively after the first interval.

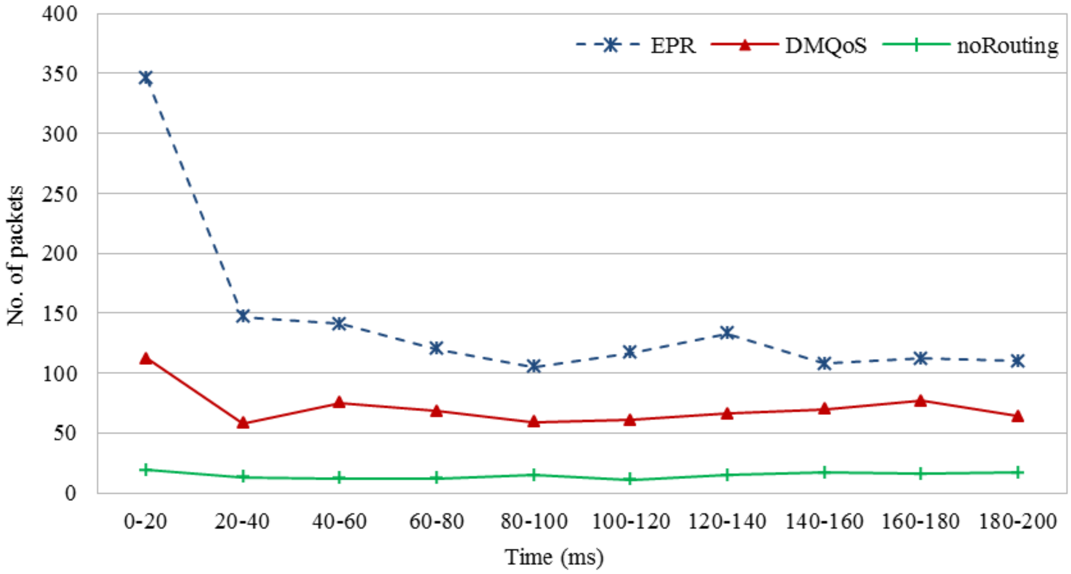


Figure 5-45: Latency for 4K packets sent by source nodes when source node is mobile and transmit power is -25 dBm

Figure 5-46 to Figure 5-47 shows that EPR outperforms the other two routing protocols for both medium and high traffic loads. Figure 5-46 shows that EPR delivers 3.6K packets when DMQoS and noRouting deliver 1.1K and 0.3K packets respectively during the first interval of 20ms. The packets delivered by EPR are on average 2 times and 4.7 times more than DMQoS and noRouting protocols respectively, after the first interval. EPR provides on average 1.8 times and 5 times better performance than DMQoS and noRouting protocols, as shown in Figure 5-47.

In summary, EPR outperforms the DMQoS and noRouting even when the source node is mobile.

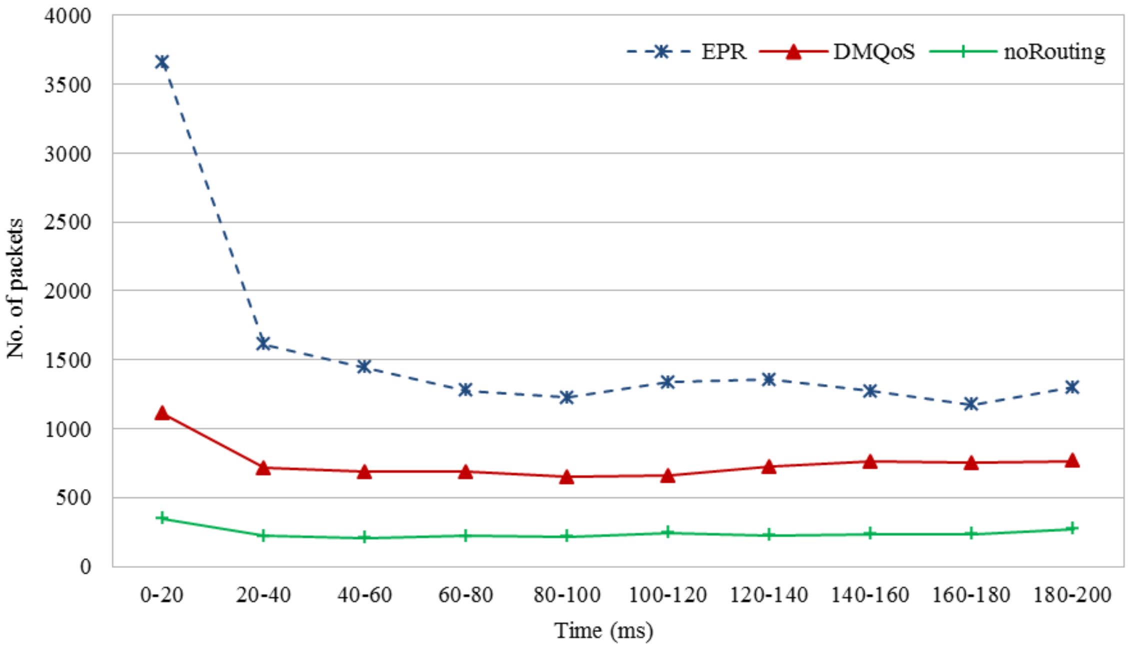


Figure 5-46: Latency for 40K packets sent by source nodes when source node is mobile and transmit power is -25 dBm

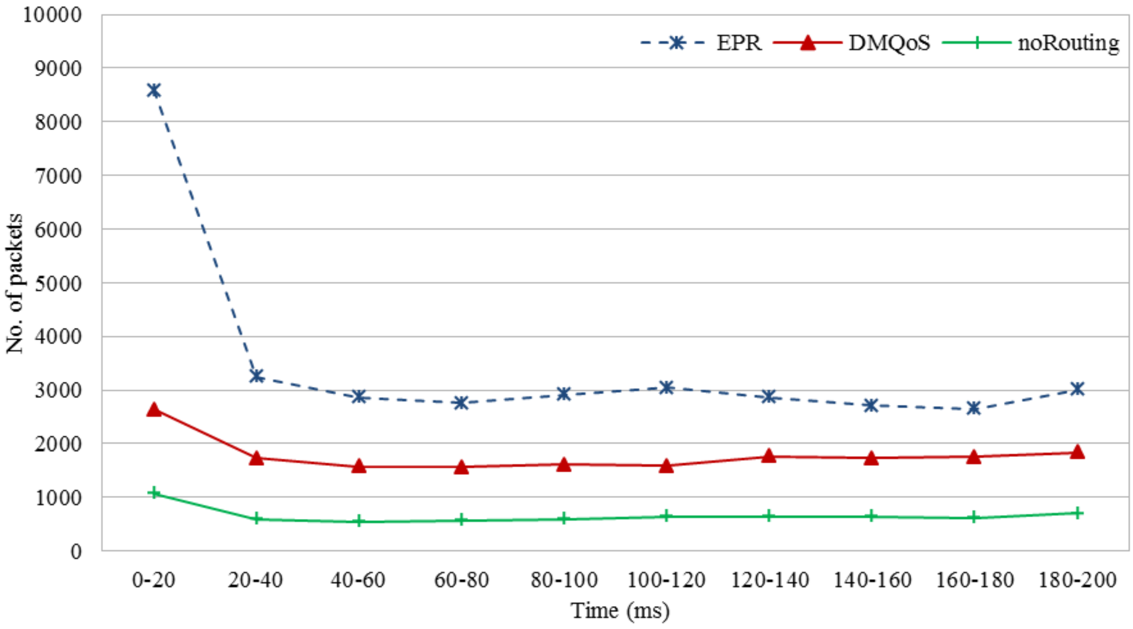


Figure 5-47: Latency for 80K packets sent by source nodes when source node is mobile and transmit power is -25 dBm

5.7. SUMMARY

This chapter proposed a new Energy-aware Peering Routing protocol (EPR) which includes three parts: 1) the new Hello protocol, 2) the neighbor table constructor algorithm, and 3) the routing table constructor algorithm. The new Hello protocol and the technique are used to choose the next hop that considers both residual energy and the geographic information of the neighbor nodes, thereby helping to reduce network traffic and energy consumption while simultaneously increasing the number of packets successfully received by the destinations for the offered low, medium, and high traffic loads. Extensive simulations were performed in the OMNeT++ based Castalia 3.2 simulator for three scenarios, with fixed and variable numbers of packets, and help to demonstrate the feasibility and better performance of the proposed protocol within the ZK-BAN framework. Both static and mobile patient cases were considered. The results showed that, for different transmit powers, the EPR reduced average traffic load by 44%, and the number of packets received successfully by the destinations has increased on average by 20% for transmit powers of -15dBm and -10dBm. The energy saved in EPR was on average 93mJ, 201mJ, and 216mJ for the transmit power of -25dBm, -15dBm, and -10dBm respectively, over 120 seconds. No buffers overflows were observed at the intermediate nodes in EPR for very low transmit power of -25dBm in both the static and mobile BAN scenarios. EPR consistently reduces the network traffic by 53% for all offered traffic loads when the transmit power is -15dBm and -10dBm. These results signify that the energy communication cost employed and the proposed EPR protocol yield better performance characteristics compared to similar protocols.

CHAPTER 6

QPRD: QoS-aware Peering Routing Protocol for Delay-Sensitive Data

Consistent performance, energy efficiency, and reliable transfer of data are critical factors for real-time monitoring of a patient's data, especially in a hospital environment. In this chapter, a routing protocol is proposed by considering the QoS requirements of the Body Area Network (BAN) data packets. A mechanism for handling delay-sensitive packets is provided by this protocol. Extensive simulations using the OMNeT++ based simulator Castalia 3.2 illustrate that the proposed algorithm provides better performance than other QoS-aware routing protocols in terms of higher successful transmission rates (throughputs), lower overall network traffic, and fewer number of packet timeouts in both the mobile and static patient scenarios.

This chapter is organized as follows: Section 6.1 provides the introduction; Section 6.2 discusses related work; Section 6.3 formulates the proposed QoS-aware peering routing protocol for delay-sensitive data (QPRD); Section 6.4 describes the performance evaluation of the proposed QoS-aware peering routing protocol; and Section 6.5 presents the conclusions.

6.1. Motivation

The real-time monitoring of patients requires the transmission of delay-sensitive data such as video imaging, motion sensing, and ElectroMyoGraphy (EMG) using BAN. Some projects like SMART [35], CareNet [68], AID-N [36], and ALARM-NET [34] provide different methods to monitor the patient data. In these methods, the transmission of BAN data from body sensors to the central database is considered and then BAN data is downloaded and monitored from the central database. However, these techniques do not monitor or display in real-time BAN data in hospital environment. The advantages of using a centralized system are to have better control and maintain the data privacy of the patient. However, traffic congestion, server failure or link failure can cause considerable delays in monitoring the patient data which can badly effect treatment. On the other hand, distributed data approaches help to reduce the traffic load and can better accomodate patient mobility. The ZK-BAN peering framework proposed in Chapter 4 suggests a semi-centralized system for reliably monitoring BAN data. The hybrid ZK-BAN uses both centralized and distributed techniques.

The routing protocol EPR, proposed and discussed in Chapter 5, resolves the problem of handling ordinary data packets. The requirement of real-time display for delay-sensitive packets is different from those of ordinary packets. Hence, a new QoS-aware routing protocol is required to handle delay-sensitive packets. A novel routing protocol that addresses the issue of handling delay-sensitive data and displaying in real-time delay-sensitive BAN data is proposed in this chapter. The proposed QoS-aware Peering Routing protocol for Delay-sensitive packets (QPRD) is designed for the ZK-BAN peering framework discussed in Chapter 4. QPRD provides an innovative approach to the reliable transmission of Ordinary Packets (OPs) and Delay-Sensitive Packets (DSPs).

6.2. RELATED WORK

A smart monitoring system of BAN data in hospital environment can resolve the challenges related to the management of patients' medical information [43]. The Scalable Medical Alert and Response Technology (SMART) [35] is designed to monitor the patient's data in hospital emergency area. The data from sensors is transferred to the PDA and then the PDA sends it to the next tier by using wireless standard 802.11b. CareNet [68] provides an integrated wireless sensor based solution to monitor the patient's data from remote hospitals. The two-tier wireless communication is used in the projects [35, 68]. A GPS system is used in [36] to monitor the patient's data only in outdoor BAN communication. A wireless sensor network for assisted-living and residential monitoring system with a query based protocol is provided in ALARM-NET [34]. A three-tier communication approach is used in [73] to store the BAN data on the server and then make this data available for the physician to analyze the patient's data. The projects [34, 35, 36, 68, 73] used a centralized approach to monitor the patient's data. However, the real-time display of data by considering the delay requirements of delay-sensitive packets is not considered. To access the data from a centralized server may cause delay and even a simple link failure can completely disconnect the healthcare system from the central server.

In Chapter 5, an Energy-aware Peering Routing protocol (EPR) was presented which considers the energy level and geographic information of the neighbor nodes for choosing the best next hop. The EPR only considers ordinary packets. It was shown that EPR has an overall lower energy consumption than comparable protocols [33, 43, 44, 45, 46], and provides better results in terms of reduced overall network traffic, reduced number of packets forwarded by intermediate nodes, and higher successful data transmission rates. However, EPR does not provide a mechanism for dealing with Delay-Sensitive Packets (DSPs). In this chapter, delay-sensitive packets are considered by the proposed QoS-aware Peering Routing protocol for Delay-sensitive data (QPRD) and their performance is investigated by comparing it to the existing DMQoS protocol [33]. In [33] , DMQoS categorizes the data packets into four types: Ordinary Packets (OPs), Critical Packets (CPs), Reliability-driven Packets (RPs), and Delay-driven Packets (DPs). The DMQoS [33] provides better results for delay-driven packets than several previously investigated methods [43, 44, 45, 46] in terms of end-to-end path delay. However, DMQoS employs a *hop-by-hop* approach to determine the next hop. DMQoS considers the neighbor device with the lowest delay, and the next hop then determines the best next upstream hop with least delay. The disadvantage of this *hop-by-hop* delay-driven approach employed in DMQoS is that only neighboring nodes delay information is considered by source node. The source node forwards the packet to a particular neighbor node which has lower node delay than the required delay. The neighbor node sends the acknowledgement of the successfully received packet to the source node. Now, the packet receiving neighbor node determines its best upstream node in terms of delay requirement and forwards the packet to the upstream node if the node delay of upstream node is less than the required delay. In case, the neighbor node doesn't find any upstream node with node delay less than required delay then the packet is dropped. In this case, the packet doesn't reach to the destination, but the source node assumes that the packet has been successfully received by the destination. Furthermore, the *hop-by-hop* approach used in DMQoS causes an increase in overall network traffic, and the required end-to-end latency may not be guaranteed. In this thesis, the proposed QPRD addresses these shortcomings by selecting and choosing the next hop device based on the lowest end-to-end path delay from the source node to the destination.

6.3. QoS-aware Peering Routing Protocol for Delay-sensitive Data (QPRD)

The proposed QoS-aware routing protocol is used in an indoor hospital ZK-BAN peering framework discussed in Chapter 4. The proposed QPRD provides a mechanism to 1) calculates the node delays and path delays of all possible paths from the source node to the destination, 2) determines the best path, and 3) chooses the best next hop NH_D based on the delay requirements of the packet. For each destination, the routing table contains information about the next hop device connected to the path with the least end-to-end latency. For any DSP, if the path delay ($DL_{path(i,Dst)}$) is less than or equal to the delay requirement, the source node sends the DSP through that path.

The architecture of proposed QPRD is shown in Figure 6-1. It consists of seven modules: MAC receiver, Delay Module (DM), Packet Classifier (PC), Hello Protocol Module (HPM), Routing Services Module (RSM), QoS-aware Queuing Module (QQM), and MAC transmitter. The modules are discussed below.

6.3.1. MAC Receiver

The MAC receiver receives the data or Hello packets from other nodes (BAN, MDC, or NSC). It checks the MAC address of the packet. It only forwards the broadcast packets or the packets which have the same node’s MAC address as destination address to the network layer.

6.3.2. Delay Module (DM)

The delay module monitors the time required to capture the channel ($DL_{channel(i)}$), MAC layer queuing delay ($DL_{MAC\_queue(i)}$), and transmission time ($DL_{trans(i)}$) of a packet. The delay module sends this information to the network layer. The network layer uses this information to calculate the node delay ($DL_{node(i)}$).

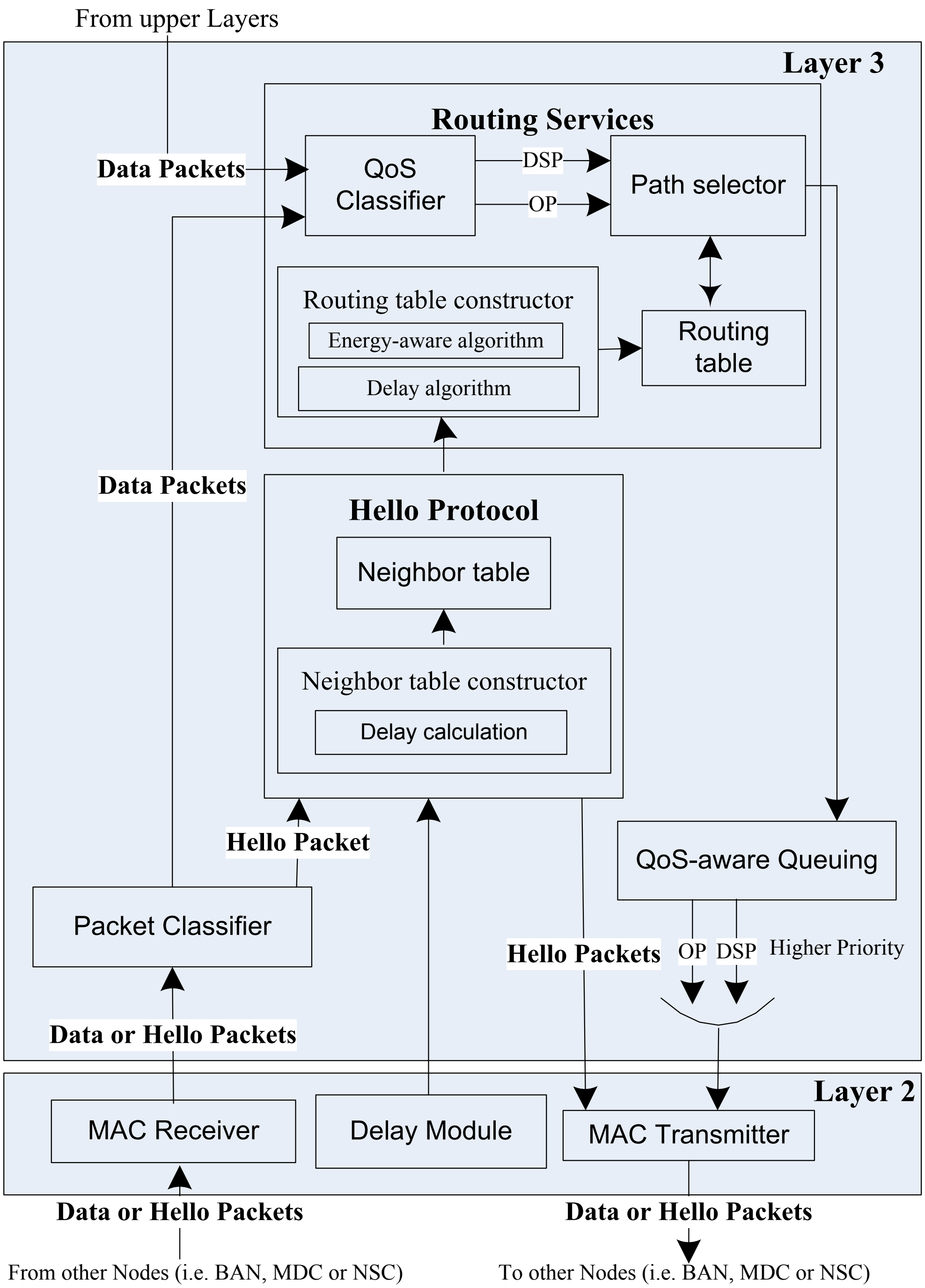


Figure 6-1: Protocol architecture

6.3.3. Packet Classifier (PC)

The Packet Classifier (PC) receives all the packets from the MAC receiver. The data packets and Hello packets are differentiated by the PC. The PC forwards the data and

Hello packets to the routing services module and Hello protocol module respectively.

6.3.4. Hello Protocol Module (HPM)

The neighbor table constructor and the neighbor table are the two sub-modules of Hello protocol module. The information received from the delay module of the MAC layer, and the Hello packets is used by the neighbor table constructor to construct the neighbor table. Initially, Hello packets are broadcasted by each type 1 (NSC) and type 2 (MDC) devices. The node *i* receives the Hello packet. The neighbor table constructor of node *i* calculates its own $DL_{path(i,Dst)}$ based on the information in the Hello packets. The Hello packet is updated and forwarded by node *i* to the other nodes. The Hello packet fields of node *j* are shown in Figure 6-2.

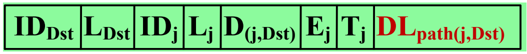


Figure 6-2: Hello packet structure

The notations used in this chapter and their descriptions are summarized in Table 6-1.

Table 6-1: Notations for the proposed algorithm

Field ID	Description
Node *i*	Source node
Node *j*	Neighbor node of source node
Node *Dst*	Destination node (i.e. NSC, MDC, BAN)
$\mathbf{ID_{Dst}}$	Destination ID
$\mathbf{L_{Dst}}$	Destination Location
$\mathbf{ID_j}$	Neighbor node *j* ID
$\mathbf{L_j}$	Neighbor node *j* location
$\mathbf{D_{(j,Dst)}}$	Distance between neighbor node *j* and destination *Dst*
$\mathbf{E_j}$	Residual energy of node *j*
$\mathbf{T_j}$	Device type of node *j*
$\mathbf{D_{(i,j)}}$	Distance between node *i* to neighbor node *j*
$\mathbf{NH_{(i,Dst)}}$	Next Hop between node *i* and destination *Dst*
$\mathbf{NH_E}$	Energy-aware next hop
$\mathbf{NH_D}$	Next hop for delay-sensitive packets
$\mathbf{DL_{path(i,Dst)}}$	Path delay from node *i* to destination *Dst*
$\mathbf{DL_{node(i)}}$	Time delay within the node *i*
$\mathbf{DL_{req}}$	Required path delay for delay-sensitive packets

The neighbor table contains fields for both hop-by-hop delay ($DL_{node(i)}$), and end-to-end path delay ($DL_{path(i,Dst)}$). The neighbor table constructor updates the neighbor table periodically after receiving every new Hello packet. The neighbor table structure of node *i* is shown in Figure 6-3.

$\mathbf{ID_{Dst}}$	$\mathbf{L_{Dst}}$	$\mathbf{ID_j}$	$\mathbf{L_j}$	$\mathbf{D_{(j,Dst)}}$	$\mathbf{D_{(i,j)}}$	$\mathbf{C_j}$	$\mathbf{T_j}$	$\mathbf{DL_{node(i)}}$	$\mathbf{DL_{path(i,Dst)}}$

Figure 6-3: Neighbor table structure

The node delay ($\mathrm{DL_{node(i)}}$) can be found by adding the packet delays due to transmission, queuing, processing, and channel capturing. This is given in Equation 6-1.

$$\mathrm{DL_{node(i)}} = \mathrm{DL_{trans(i)}} + \mathrm{DL_{queues+channel}} + \mathrm{DL_{proc}} \qquad (6\text{-}1)$$

The node updates its Hello packets periodically, 4 seconds are used in QPRD for simulation purposes. The time interval 4 seconds are used because the delay module sends the delays of MAC queue and channel capture after every 4 seconds. The average transmission delay ($\mathrm{DL_{trans}}$) before sending the Hello packets is calculated by using the Equation 6-2.

$$\mathrm{DL_{trans}} = \frac{1}{\mathrm{R_{bit}}} \frac{\sum_{z=1}^{n} \mathrm{N_{bit\,(z)}}}{\mathrm{n}} \qquad (6\text{-}2)$$

where

$\mathrm{R_{bit}}$ is the data rate, as per BAN requirement 250 kbps is used in the simulations.

$\mathrm{N_{bit}}$ is the total number of bits in each packet.

n is the number of packets transmitted in 4 seconds.

The delay due to the MAC & network layers' Queues and capturing the channel can be calculated by using the Exponentially Weighted Moving Average (EWMA) formula and is given in Equation 6-3.

$$\mathrm{DL_{queues+channel}} = (1-\rho) * \mathrm{DL_{queues+channel}} + \rho * \mathrm{DL_{queues+channel}} \qquad (6\text{-}3)$$

where

queues are the both network and MAC layers' queues.

Initial values of $\mathrm{DL_{queues+channel}}$ are the delay of the first packet send by the node. ρ is the average weighting factor that satisfies $0 < \rho \leq 1$. The selection of ρ value is heuristic and was chosen based on simulations experience. The recommended values are $0.2 \leq \rho \leq 0.3$. The best suited value of ρ found for QPRD simulations is 0.2.

The path delay between node *i* and destination node *Dst* ($\mathrm{DL}_{\mathrm{path(i,Dst)}}$) is calculated by using the Equation 6-4.

$$\mathrm{DL}_{\mathrm{path(i,Dst)}} = \mathrm{DL}_{\mathrm{node(i)}} + \mathrm{DL}_{\mathrm{path(j,Dst)}} \tag{6-4}$$

where initial value of $\mathrm{DL}_{\mathrm{path(j,Dst)}}$ is zero when *j*=Dst.

An example of finding the path delay from node *i* (B_3) to *Dst* (NSC) is shown in Figure 6-4. The delay calculation of two paths B_3-B_1-MDC_2-NSC (path1) and B_3-MDC_3-B_2-MDC_1-NSC (path2) is given for illustrative purposes. The typical assumed values are chosen for illustrated purposes. The individual node delays used in this example are given below.

$$\mathrm{DL}_{\mathrm{node(NSC)}} = 20 \text{ ms} \tag{6-5}$$

$$\mathrm{DL}_{\mathrm{node(MDC2)}} = 40 \text{ ms} \tag{6-6}$$

$$\mathrm{DL}_{\mathrm{node(B1)}} = 30 \text{ ms} \tag{6-7}$$

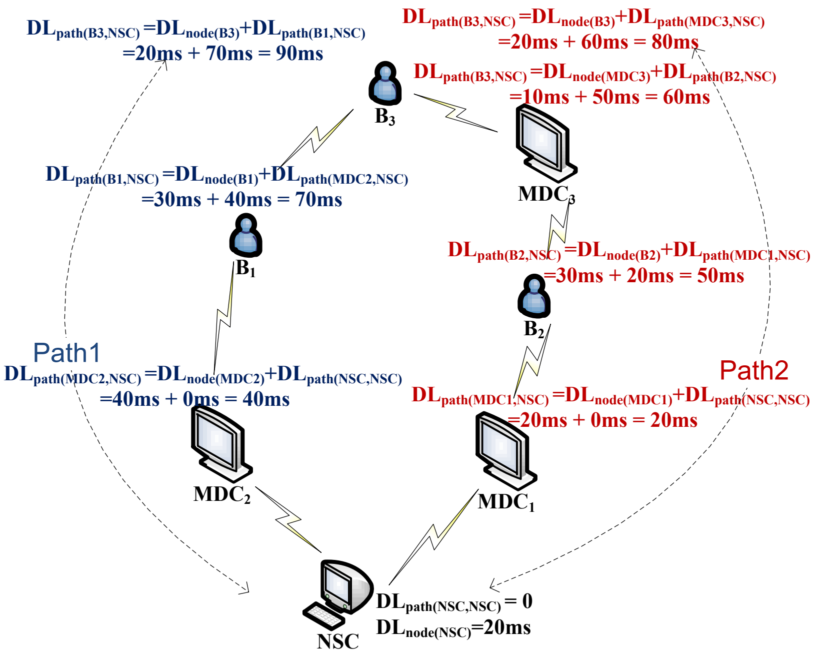

$\mathrm{DL}_{\mathrm{path(B3,NSC)}} = \mathrm{DL}_{\mathrm{node(B3)}} + \mathrm{DL}_{\mathrm{path(B1,NSC)}}$
=20ms + 70ms = 90ms
$\mathrm{DL}_{\mathrm{path(B3,NSC)}} = \mathrm{DL}_{\mathrm{node(B3)}} + \mathrm{DL}_{\mathrm{path(MDC3,NSC)}}$
=20ms + 60ms = 80ms
$\mathrm{DL}_{\mathrm{path(B3,NSC)}} = \mathrm{DL}_{\mathrm{node(MDC3)}} + \mathrm{DL}_{\mathrm{path(B2,NSC)}}$
=10ms + 50ms = 60ms
B_3
$\mathrm{DL}_{\mathrm{path(B1,NSC)}} = \mathrm{DL}_{\mathrm{node(B1)}} + \mathrm{DL}_{\mathrm{path(MDC2,NSC)}}$
=30ms + 40ms = 70ms
MDC_3
B_1
$\mathrm{DL}_{\mathrm{path(B2,NSC)}} = \mathrm{DL}_{\mathrm{node(B2)}} + \mathrm{DL}_{\mathrm{path(MDC1,NSC)}}$
=30ms + 20ms = 50ms
B_2
Path1
$\mathrm{DL}_{\mathrm{path(MDC2,NSC)}} = \mathrm{DL}_{\mathrm{node(MDC2)}} + \mathrm{DL}_{\mathrm{path(NSC,NSC)}}$
=40ms + 0ms = 40ms
Path2
$\mathrm{DL}_{\mathrm{path(MDC1,NSC)}} = \mathrm{DL}_{\mathrm{node(MDC1)}} + \mathrm{DL}_{\mathrm{path(NSC,NSC)}}$
=20ms + 0ms = 20ms
MDC_2
MDC_1
$\mathrm{DL}_{\mathrm{path(NSC,NSC)}} = 0$
$\mathrm{DL}_{\mathrm{node(NSC)}}$=20ms
NSC

Figure 6-4: Example of finding the path delay

$$\mathrm{DL}_{\mathrm{node(B3)}} = 20 \text{ ms} \tag{6-8}$$

$$\mathrm{DL}_{\mathrm{node(MDC1)}} = 20 \text{ ms} \tag{6-9}$$

$$\mathrm{DL}_{\mathrm{node(B2)}} = 30 \text{ ms} \tag{6-10}$$

$$\mathrm{DL}_{\mathrm{node(MDC3)}} = 10 \text{ ms} \tag{6-11}$$

The path delay of destination ($\mathrm{DL}_{\mathrm{path(Dst,Dst)}}$) is approximately zero, because the time required to receive the packet from MAC to network layer is negligible. So, in this example initial path delay is given below.

$$\mathrm{DL}_{\mathrm{path(NSC,NSC)}} = 0 \text{ ms} \tag{6-12}$$

Each node calculates the path delay from itself to the NSC. First, the calculations of the path delay for path1 (B_3-B_1-MDC_2-NSC) are considered.

The path delay of MDC_2 ($\mathrm{DL}_{\mathrm{path(MDC2,NSC)}}$) is calculated by using Equation 6-4.

$$\mathrm{DL}_{\mathrm{path(MDC2,NSC)}} = \mathrm{DL}_{\mathrm{node(MDC2)}} + \mathrm{DL}_{\mathrm{path(NSC,NSC)}}$$

Using the values from Equations 6-5 and 6-12 in the above Equation, we get

$$\mathrm{DL}_{\mathrm{path(MDC2,NSC)}} = 40 + 0 = 40\text{ms}$$

The path delay of BAN B_1 is calculated below

$$\mathrm{DL}_{\mathrm{path(B1,NSC)}} = \mathrm{DL}_{\mathrm{node(B1)}} + \mathrm{D}_{\mathrm{path(MDC2,NSC)}}$$

$$\mathrm{DL}_{\mathrm{path(B1,NSC)}} = 30 + 40 = 70\text{ms} \tag{6-13}$$

The node B_3 determines the path delay by using the values from Equations 6-8 and 6-13.

$$\mathrm{DL}_{\mathrm{path(B3,NSC)}} = \mathrm{DL}_{\mathrm{node(B3)}} + \mathrm{D}_{\mathrm{path(B1,NSC)}}$$

$$\mathrm{DL}_{\mathrm{path(B3,NSC)}} = 20 + 70 = 90\text{ms} \tag{6-14}$$

In the same manner, the path delay of path2 (B_3-MDC_3-B_2-MDC_1-NSC) can be calculated as follows:

$$\mathrm{DL}_{\mathrm{path(MDC1,NSC)}} = 20 + 0 = 20\mathrm{ms}$$

$$\mathrm{DL}_{\mathrm{path(B2,NSC)}} = 30 + 20 = 50\mathrm{ms}$$

$$\mathrm{DL}_{\mathrm{path(MDC3,NSC)}} = 10 + 50 = 60\mathrm{ms}$$

$$\mathrm{DL}_{\mathrm{path(B3,NSC)}} = 20 + 60 = 80\mathrm{ms} \tag{6-15}$$

Equations 6-14 and 6-15 show that the path delays of path1 and path2 are 90ms and 80ms respectively. It is quite possible that the path with less delay is longer (has more hops) than the other paths. As it is observed from the above example, path2 includes five devices and path1 has four devices. However, the path delay of path2 is lower than the path delay of path1.

6.3.5. Routing Services Module (RSM)

The routing services module is responsible for constructing the routing table, categorizing the data packets into Delay-Sensitive Packets (DSPs) and Ordinary Packets (OPs). It also chooses the best path(s) for each category (DSPs or OPs) of traffic. QoS classifier, routing table constructor, path selector, and routing table are the sub-modules of routing services module. The routing table structure for node *i* is shown in Figure 6-5.

$\mathbf{ID_{Dst}}$	$\mathbf{L_{Dst}}$	$\mathbf{NH_E}$	$\mathbf{NH_D}$	$\mathbf{DL_{path(i,Dst)}}$

Figure 6-5: Routing table structure for QPRD

The notations and their descriptions are listed in Table 6-1. Two next hop entries $\mathrm{NH_E}$ and $\mathrm{NH_D}$ are given for each destination *Dst* in routing table. The routing table constructor contains the energy-aware and delay algorithms. The energy-aware algorithm discussed in Chapter 5 is used to find next hop $\mathrm{NH_E}$ for OPs. Residual energy and geographic location of the neighbor nodes are considered for choosing $\mathrm{NH_E}$. For DSPs, the new proposed algorithm finds the best possible path to ensure the minimum required path delay. The routing table is constructed by using the neighbor table entries. Neighbor table contains multiple records for each destination. For example, Figure 6-4 shows that there are many paths from $\mathrm{B_3}$ to NSC. Some of these paths are $\mathrm{B_3}$-$\mathrm{B_1}$-$\mathrm{MDC_2}$-NSC, $\mathrm{B_3}$-$\mathrm{MDC_3}$-

B_2-MDC_1-NSC, etc. For each destination, the routing table constructor stores the next hop (NH_D) which has the lowest latency.

Algorithm 6-1 Routing table construction algorithm for delay-sensitive packets

```
INPUT: Neighbor table, $i$'s neighbor table records $\mathrm{NH}_{(\mathrm{i,Dst})}, \forall\, Dst \in \{\mathrm{MDC, NSC, BAN}\}$
1. for each destination $\mathrm{Dst} \in \{\mathrm{NSC, MDC, BAN}\}$ do
2.   $\mathrm{NH} = \{\text{All neighbor nodes } \mathrm{j} \in \mathrm{NH}_{(\mathrm{i,Dst})}\}$
3.    if $(|\mathrm{NH}| == 1)$ then
4.       $\mathrm{NH}_D \leftarrow \mathrm{NH}$
5.       else if $(|\mathrm{NH}| > 1)$ then
6.        Sort NH in ascending order of $\mathrm{DL}_{\mathrm{path(i,Dst)}}$
7.        $\mathrm{NH}_\mathrm{D} = \text{first neighbor node } \mathrm{j} \in \mathrm{NH};$
8.      end if
9. end for
```

Algorithm 6-1 shows that node i identifies the next hop candidates by searching the records which have the same ID_{Dst} in the neighbor table. The path delay has been calculated by using the neighbor table constructor and stored in neighbor table for each next hop candidate, using Equation 6-4. The node stores the neighbor nodes' IDs in the variable NH (line 2). If NH has only one entry, this means there is only one path available. The node stores this entry to NH_D (line 4). Otherwise, the node sorts the NH entries in ascending order of delay, and then stores the first entry which has the lowest path delay in NH_D (lines 6, 7). The next hop candidate NH_D is then stored with its path delay value ($DL_{path(i,Dst)}$) in the routing table.

The data packets from both upper layers and packet classifier are received by QoS classifier. The QoS classifier classifies the packets into DSP and OP data. For each data packet, the Path Selector (PS) checks the QoS requirement and chooses the most appropriate next hop(s) by using Algorithm 6-2. The Path Selector compares the delay requirement (DL_{req}) with the path delay ($DL_{path(i,Dst)}$) of NH_D which is stored in the routing table. If the path delay ($DL_{path(i,Dst)}$) is lower than required delay (DL_{req}), the packet is sent to NH_D (line 3-4). Otherwise, the packet is dropped (line 6).

For ordinary packets, the PS returns the next hop NH_E which is discussed by the EPR (lines 8-9) else the packet is dropped.

Algorithm 6-2 Path selector algorithm for delay-sensitive packets

```
INPUT: Routing table, i's routing table records $NH_{(i,Dst)}, \forall Dst \in \{MDC, NSC, BAN\}$
1.  for each data packet do
2.    if data packet is delay − sensitive packet (DSP)
3.      if ($DL_{path(i,Dst)} <= DL_{req}$) then
4.        send to $NH_D$
5.      else
6.        Drop the packet immediately
7.      end if
8.    else if data packet is Ordinary Packet (OP)
9.        send to $NH_E$
10.     else
11.       drop the packet immediately
12.   end if
13. end for
```

6.3.6. QoS-aware Queuing Module (QQM)

The routing services module passes the data packets to the QoS-aware Queuing Module (QQM) after choosing the appropriate next hop(s). The QQM receives the data packets and separates these packets in two classes (DSP and OP). An individual queue is used for each class of packets. QQM functions are the same as discussed in [33]. The priority of the DSP queue is higher than that of the OP queue. By default, the DSP queue with higher priority sends the packets first. The packets from lower priority OP queue will be sent only when the DSP queue is empty. However, for fair treatment of OP data, a timeout is used by all the queues. A queue sends the packets to the MAC layer within the period specified by the timeout for that queue. QQM changes the control from higher priority queue to lower priority queue after the queue timeout occurs or when the higher priority queue is empty whichever is earlier.

6.3.7. MAC Transmitter

The MAC transmitter receives the data and Hello packets from the network layer and stores it in the queue. The queue works in a First-In-First-Out (FIFO) fashion. It transmits the packets after capturing the channel by using CSMA/CA algorithm.

6.4. Performance Evaluation

Simulations are performed on OMNeT++ based simulator Castalia-3.2 [78]. In this section, the proposed QPRD algorithm is compared with the DMQoS [33] and noRouting protocols. In noRouting, the delay-sensitive data packets are forwarded to random next hop devices instead of algorithm's next hop based on end-to-end path delay routes. The network parameters used in simulations are shown in Table 6-2.

Table 6-2: Parameters information

Deployment	Area	Scenario 1 and 2: 9m by 9m
	Deployment type	Scenario 1: All nodes are static Scenario 2: Movable source node B4
	Number of nodes	Scenario 1 and 2: 8 nodes (4 BANs, 3 MDCs, 1 NSC)
	Initial nodes locations	Scenario 1 and 2: NSC(0,3), $MDC_1(2,5)$, $MDC_2(2,1)$, $MDC_3(3,3)$ $B_1(5,5)$, $B_2(5,1)$, $B_3(6,3)$, $B_4(9,3)$
	Initial node energy	18720 J (= 2 AA batteries)
	Buffer size	32 packets
	Link layer trans. Rate	250 Kbps
	Transmit power	−25dBm
Task	Application type	Event − driven
	Max. packet size	32 Bytes
	Traffic type	CBR (Constant Bit Rate)
MAC	IEEE 802.15.4	Default values
Simulation	Time	2003 Seconds (3 seconds are setup time. Simulation results are the average of three rotations.)

Figure 6-6 shows the deployment of the experimental network. The transmit power used in the simulations is -25dBm. The type 1 devices (BANCs: B_1, B_2, B_3, and B_4) are considered as source nodes, and type 2 devices (NSC and MDCs) are the destination nodes. B_1 sends packets to MDC_1, B_2 sends packets to MDC_2, B_3 sends packets to MDC_3, and B_4 sends packets to NSC. The data of B_4 has to go through the other devices to reach NSC. The source nodes send a total of 20K delay-sensitive packets. The successful transmission rate, overall network traffic, and the number of timeout packets are calculated after every 1000 packets until 4K and then every 4000 packets sent by all BANCs.

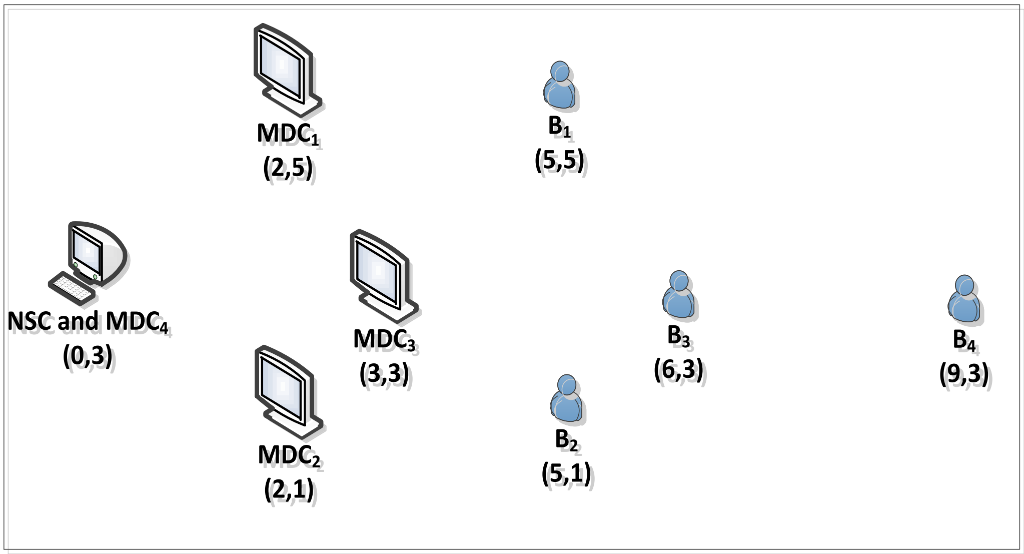


Figure 6-6: Node deployment for scenario 1

Two scenarios are considered for simulation. All the nodes used in scenario 1 are static; whereas, the source node B_4 is moving in scenario 2. The performance of the QPRD is measured by calculating the throughput, number of packets forwarded by the intermediate nodes, overall network traffic, packets timeout due to not fulfilling the required delay condition, and packets dropped due to the buffer overflow. The simulation results will show that the end-to-end path delay mechanism used in QPRD helps to reduce the packets forwarded by intermediate nodes, and the packets dropped due to the buffer overflow, which results in higher throughput and lower overall network traffic. To achieve a 97% confidence interval for the illustrative results, the average of three runs are simulated in every experiment which may introduce a maximum error of $3x10^{-3}$, based on the error calculation done by Castalia 3.2 simulator [79]. The results obtained for both scenarios are discussed below.

6.4.1. Scenario 1 – Static Nodes

All the nodes are static in this scenario, as shown in Figure 6-6. Figure 6-7 shows the throughput of the packets. From Figure 6-7, it is seen that QPRD consistently provides throughput of 94% or more. In comparison, noRouting provides an average of 74% transmission rate; whereas, DMQoS has a throughput ranging from 49% to 57%. For low offered data loads of 1K, DMQoS has a throughput of 57% that continues to decrease especially for high offered data loads of 20K, when the throughput is 49%. The low throughput in DMQoS may be explained by the way it selects the next hop using the Energy Aware Geographic Forwarding scheme. Because the best next hop doesn't guarantee that it has the smallest latency connection to the destination, the packet may timeout when it is sent using the 'best' next hop. Moreover, the energy aware geographic forwarding scheme used in DMQoS prefers the nearest next hop candidate in terms of hop count and ignores next hop nodes having a lower delay. As a result, the network traffic is increased and the packets are dropped due to timeout before reaching the destination. QPRD resolves these issues by using the end-to-end path delay.

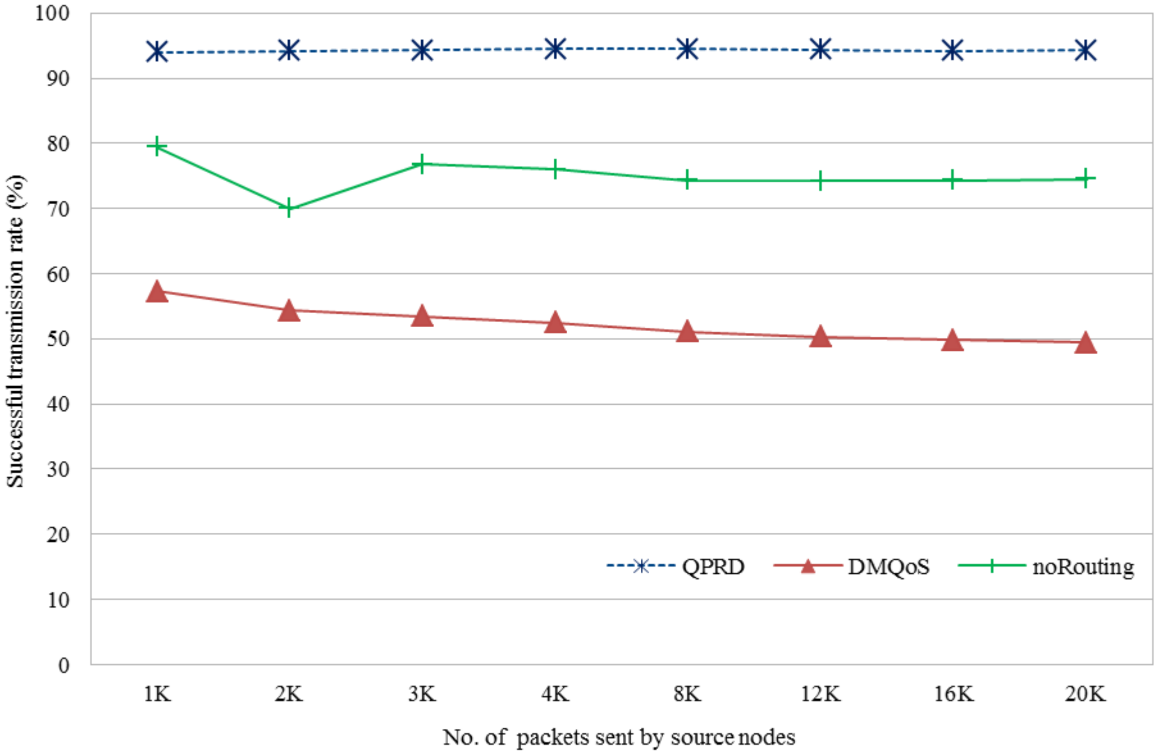


Figure 6-7: Throughput as a function of offered load when source nodes are static

B_2 is the closest node to the desitination nodes (i.e. NSC or MDCs) as shown in Figure 6-6. In DMQoS [33], B_2 is responsible for forwarding the data packets from other nodes to NSC or MDCs. This results in more energy consumption for B_2 and increased traffic congestion experienced by B_2. EPR resolves these problems by choosing the most appropriate next hop. In the proposed QPRD scheme, the BAN coordinator does not send data to another BAN coordinator unless it is absolutely necessary. Figure 6-8 shows the number of packets forwarded by the intermediate nodes. It is seen from Figure 6-8 that number of data packets forwarded by intermediate nodes before reaching the destinations in QPRD are on average 0.5 times and 3 times lower than DMQoS and noRouting respectively.

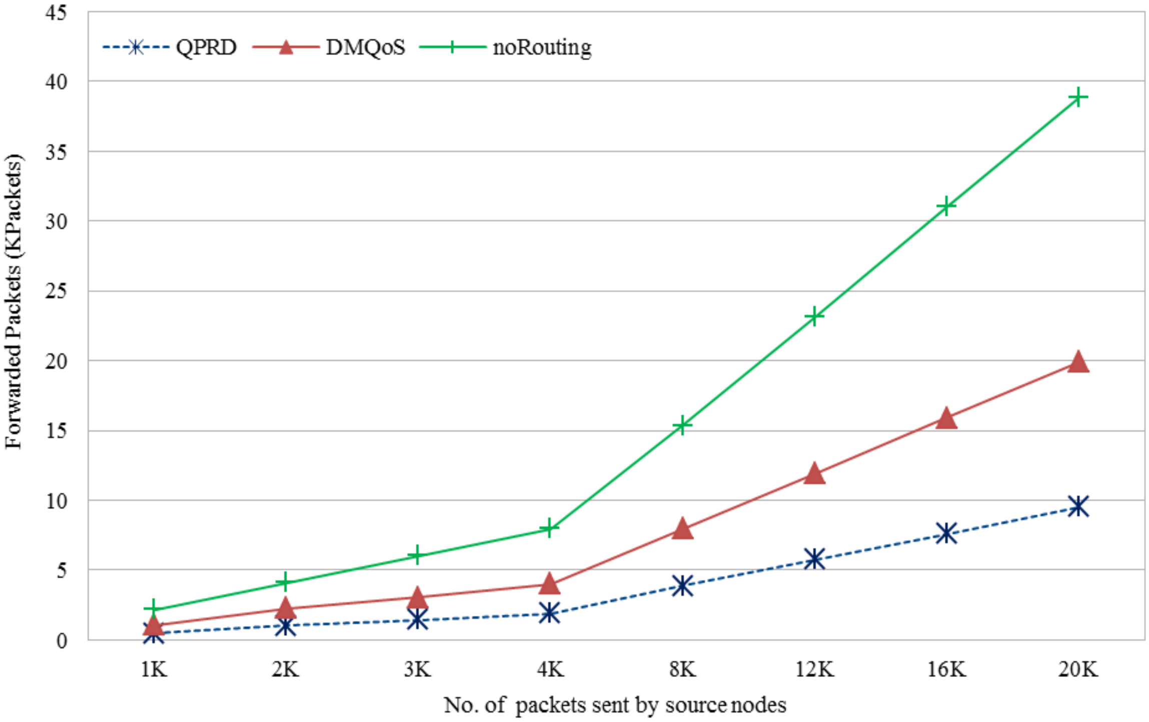


Figure 6-8: Packets forwarded by intermediate nodes vs offered load when source node is static

The lower number of forwarded packets by inermediate nodes helps to reduce the overall network traffic. Figure 6-9 shows the total network traffic generated by QPRD, DMQoS, and noRouting as a function of the offered traffic load. From this Figure, it is seen that QPRD generates about an average of 26% and 99% less traffic in the network compared to DMQoS and noRouting respectively. The path calculation in QPRD considers the delay of all the nodes and uses the best path delay information to select the next hop to send the data from source to destination.

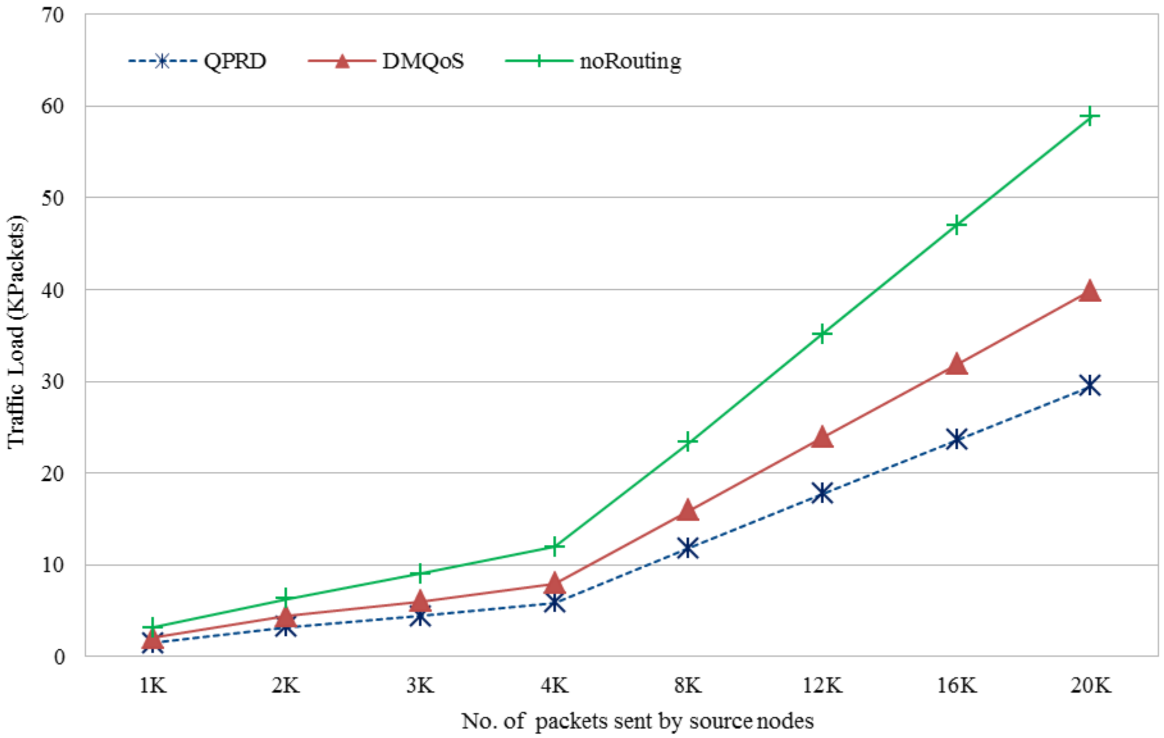


Figure 6-9: Overall network traffic as a function of offered load when source nodes are static

In contrast, to the method used in DMQoS which decides on the immediate next hop based merely on next hop delay instead of overall path delay. Each upstream hop in DMQoS sends the packet to its next hop and resultant path in DMQoS may not be the most optimal.

From Figure 6-10 it is observed that QPRD and noRouting have no packets that were timed out for all offered traffic loads (number of data packets sent by source node range from 0K to 20K). QPRD has better performance in terms of reduced overall network traffic and fewer numbers of dropped packets due to timeout, because the clear end-to-end path delay information helps the packet to reach the destination within the requested delay requirement. Moreover, the path calculation in QPRD considers the delay of all the nodes in the network and chooses only those paths which can guarantee delivering the packet to the destination before it times out.

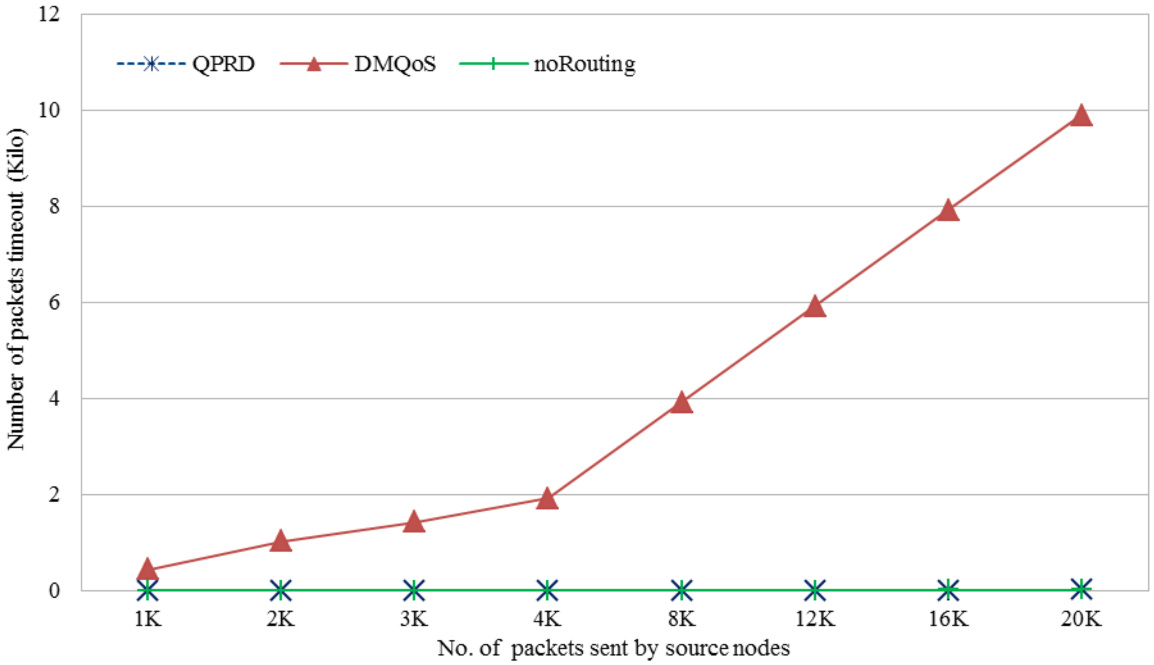


Figure 6-10: Number of packets timeout when source nodes are static

Figure 6-11 shows that there is no packet dropped due to the MAC buffer overflow in QPRD protocol. Only few packets are drops in DMQoS; whereas, 7.5K packets are dropped in noRouting. In summary, QPRD outer performs DMQoS and noRouting when the source node is static.

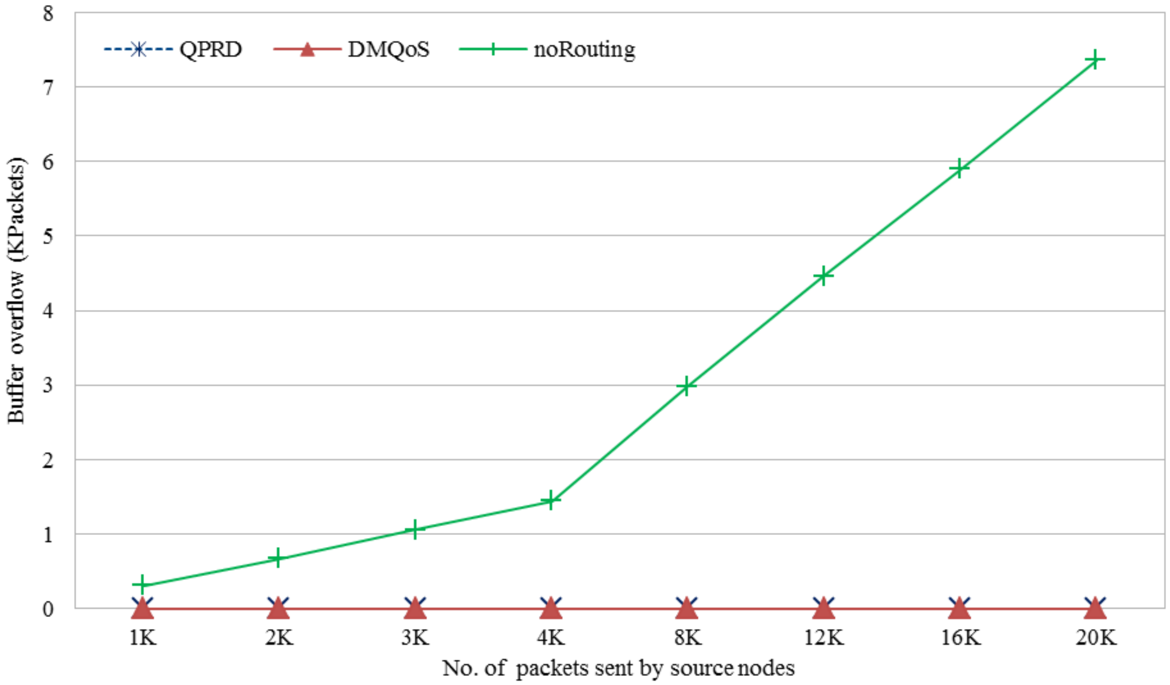


Figure 6-11: Packets dropped due to MAC buffer overflow

It is seen from the Figure 6-12 that the end-to-end path delay mechanism used in QPRD does not effect the overall energy consumption when compared with DMQoS. QPRD and DMQoS consume the same 18 Joules to 275 Joules of energy when 1K to 20K packets are sent by source nodes. On the other hand, the energy consumption of noRouting protocol is 2.6 Joules to 47.7 Joules when 1K to 20K packets are sent by source nodes. The data packets in noRouting are randomnly forwarded to three neighbor nodes without considering the delay requirements. The additional computations for delay in QPRD consume on average 6 times more energy than noRouting. However, it must be noted that noRouting results on average a 99% higher overall network traffic. This may be attributed to the 3 times more packets forwarded by intermediate nodes in noRouting resulting in a 20% lower throughput as compared to QPRD.

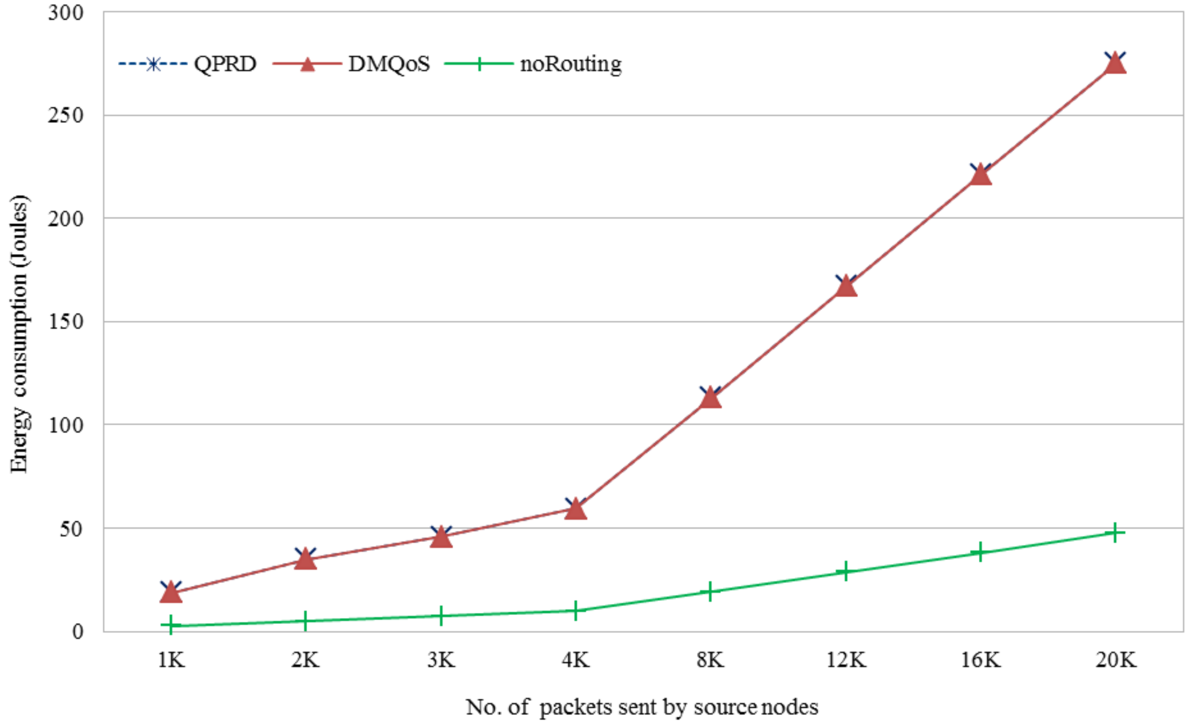


Figure 6-12: Overall energy consumption

6.4.2. Scenario 2 – Mobile Source Node

In the second scenario, the source node B_4 is moving at the speed of 1 meter per second vertically as shown in Figure 6-13. It is assumed that the speed of a fast walking patient is 1 meter per second. Once again, it is observed that QPRD provides better results than DMQoS and noRouting in case of mobile source node scenario.

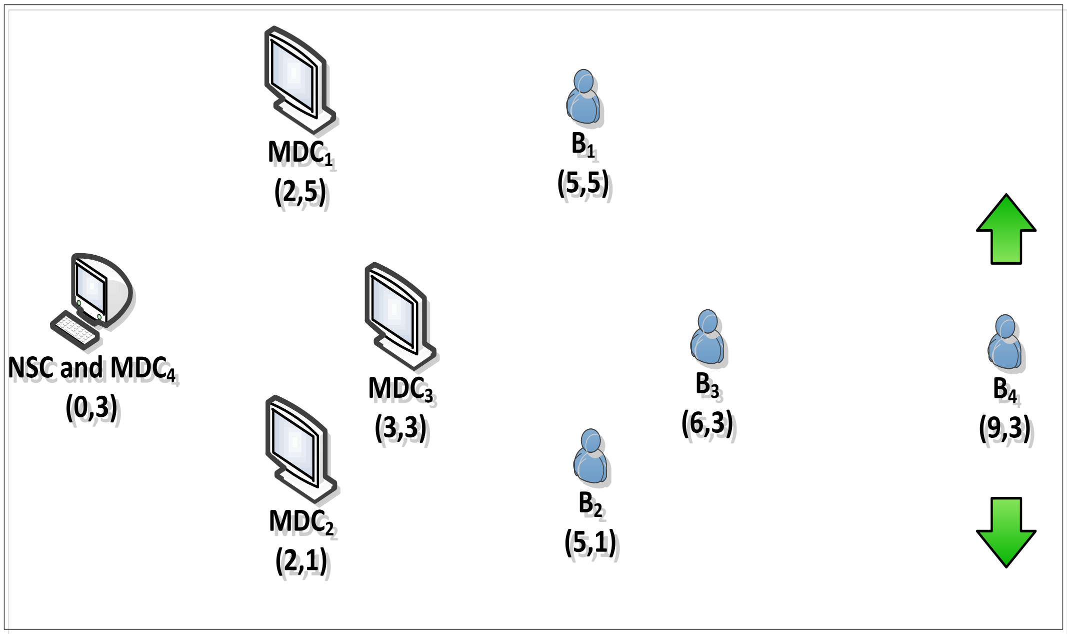


Figure 6-13: Node deployment for scenario 2

Figure 6-14 shows that the throughput is in excess of 80% in QPRD for offered data packet rates less than 8K. The throughput reduces slightly at higher offered data packet rates of 8K and more, and reduces to 71% when total offered packets sent by the source is 20K. In contrast with DMQoS, it is observed that when the offered data packet load is increased, DMQoS suffers from a much lower successful data transmission rate that reduces from 50% to 32% with resultant low throughput. Due to node mobility, the source node moves away from its neighbor nodes resulting in a connection lost which results in more packets being lost. QPRD handles this situation much more gracefully than DMQoS. In QPRD, the mobile nodes resume the connection more rapidly once the nodes come back into the range of neighbor node. The overall lower throughput in this scenario is due to the packet lost when the mobile node is out of range. The noRouting provides the lower throughput with an average of 64%.

Figure 6-15 shows that the number of packets forwarded by the intermediate nodes in QPRD is on average 0.75 times and 9 times lower when compared to the number of packets forwarded by intermediate nodes in DMQoS and noRouting protocols, respectively. The routing mechnism used in the QPRD protocol helps to send the data directly to the destination without transfering the packets to the intermediate nodes in case where the destination is in range. The perfromance of noRouting for this parameter is worst as it forwards upto 26K packets which increases the overall network traffic.

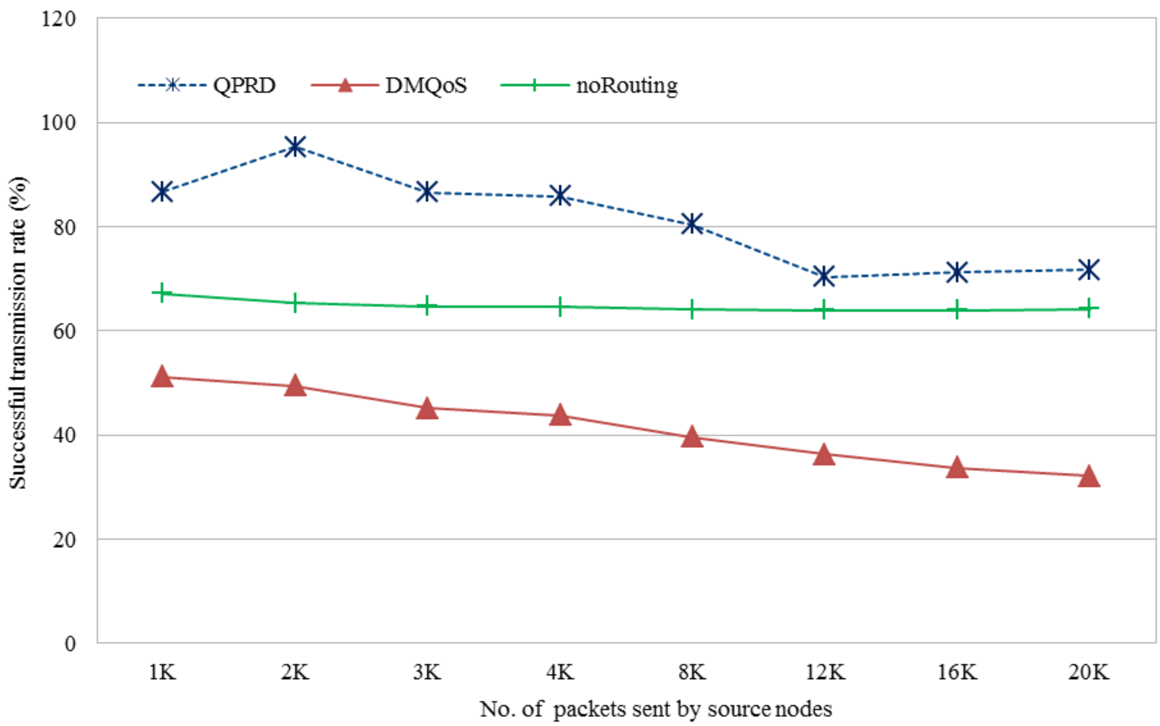


Figure 6-14: Throughput vs Offered load when source node is mobile

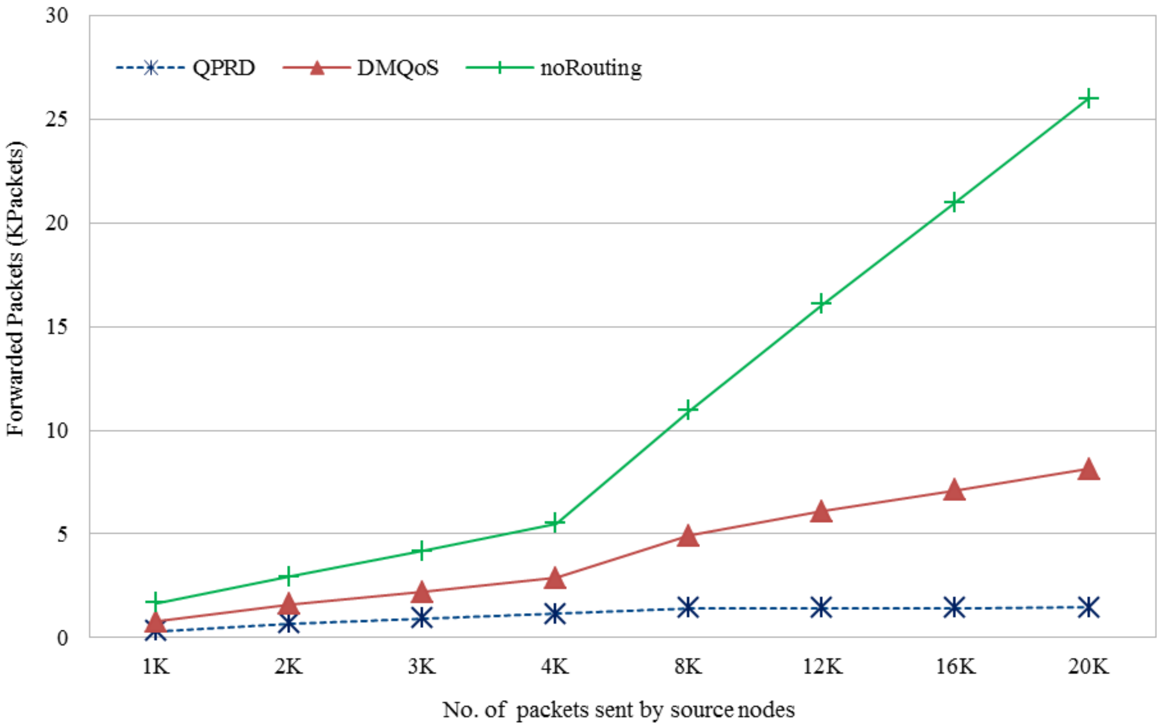


Figure 6-15: Packets forwarded by intermediate nodes

It is observed from Figure 6-16 that the overall network traffic in QPRD is about 25% and 50% less than DMQoS and noRouting protocols respectively, for all offered network data loads considered. This is due to the end-to-end path calculation mechanism used in QPRD. The delay of all the nodes are considered and QPRD algorithm selects the best next hop, on the basis of ene-to-end path delay information, to send the data from source to destination.

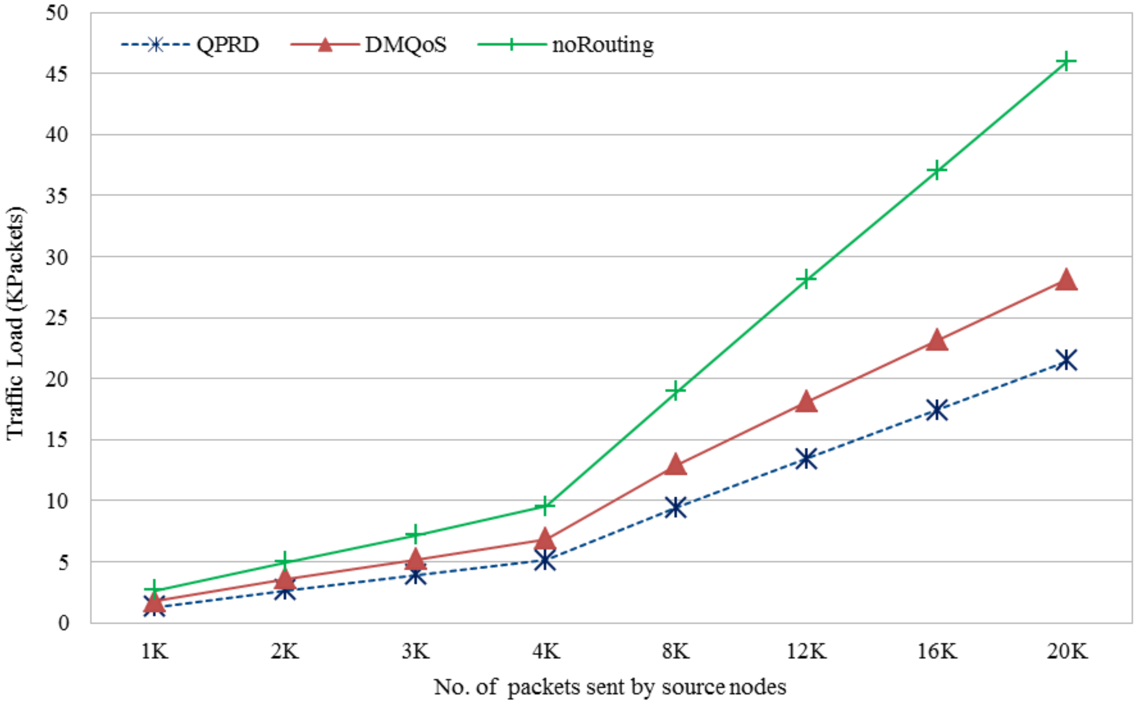


Figure 6-16: Overall Network traffic as a function of offered load - when source node is mobile

From Figure 6-17, it is seen that QPRD has no packets that were timed out for data packet transmissions at 8K or less. For high data packets (above 8K), the source node moves out of the neighbors' radio range which causes more packets to timeout. On the other hand, DMQoS has more timeout packets than QPRD. Initially for low offered data packet rates below 4K, about 40% of data packets were timed out, and for higher offered data packets (above 4K) the 40% of data packet timeouts increases to 50% (approx.). This is because the packets travel through different nodes by using hop-by-hop delay calculation as discussed in detail in Scenario 1. The source node mobility makes the packet timeout worse than the Scenario 1 of Figure 6-10.

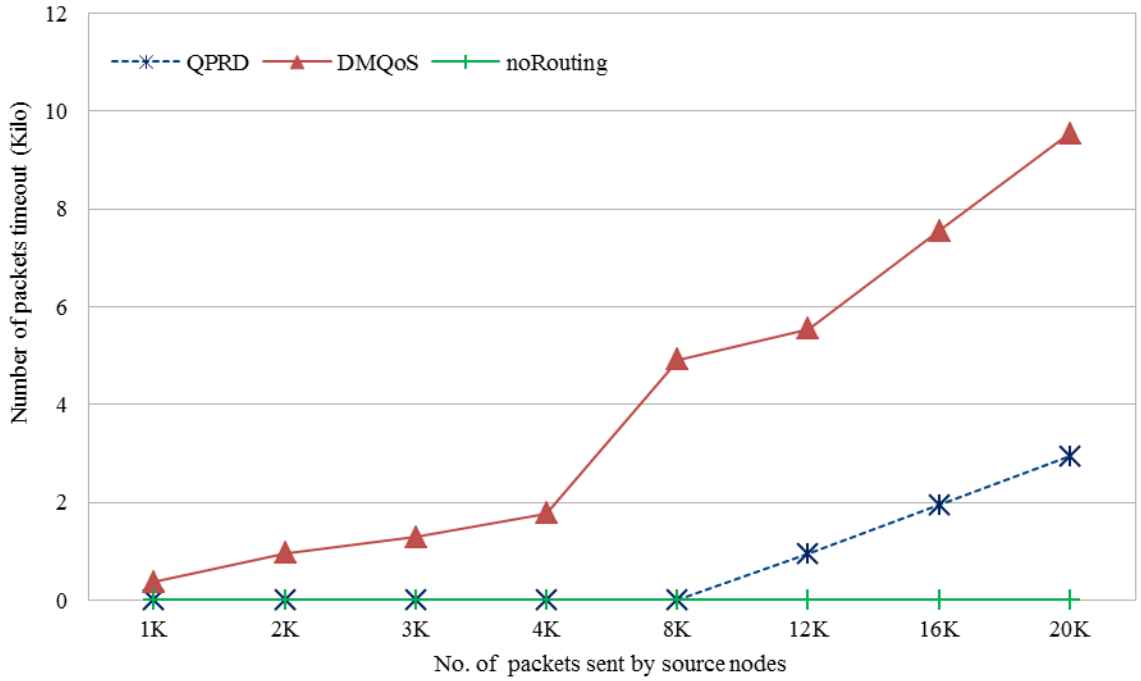


Figure 6-17: Packets timeout when source node is mobile

Figure 6-18 shows that there is no packet drops due to MAC buffer overflow in QPRD and DMQoS protocols; whereas, 9K packets are dropped in noRouting. The performance of DMQoS is similar to QPRD in terms of MAC buffer overflow; however, DMQoS has on average 39% lower throughput and an average of 25% higher overall network traffic.

From Figure 6-19, it is observed that the overall energy consumptions of QPRD and DMQoS are 18.9 Joules to 275.7 Joules when 1K to 20K packets are sent by source nodes. The noRouting consumes 2.6 Joules to 47 Joules when 1K to 20K packets are sent by source nodes. The computations for delay in QPRD are almost similar to the DMQoS but QPRD provides on average 25% lower overall network traffic, 73% fewer packets forwarded by intermediate nodes, and more importantly, a 40% higher successful data transmission rate (throughput) as compared to DMQoS.

In summary, the overall performance of QPRD is better than DMQoS and noRouting when the source node is mobile.

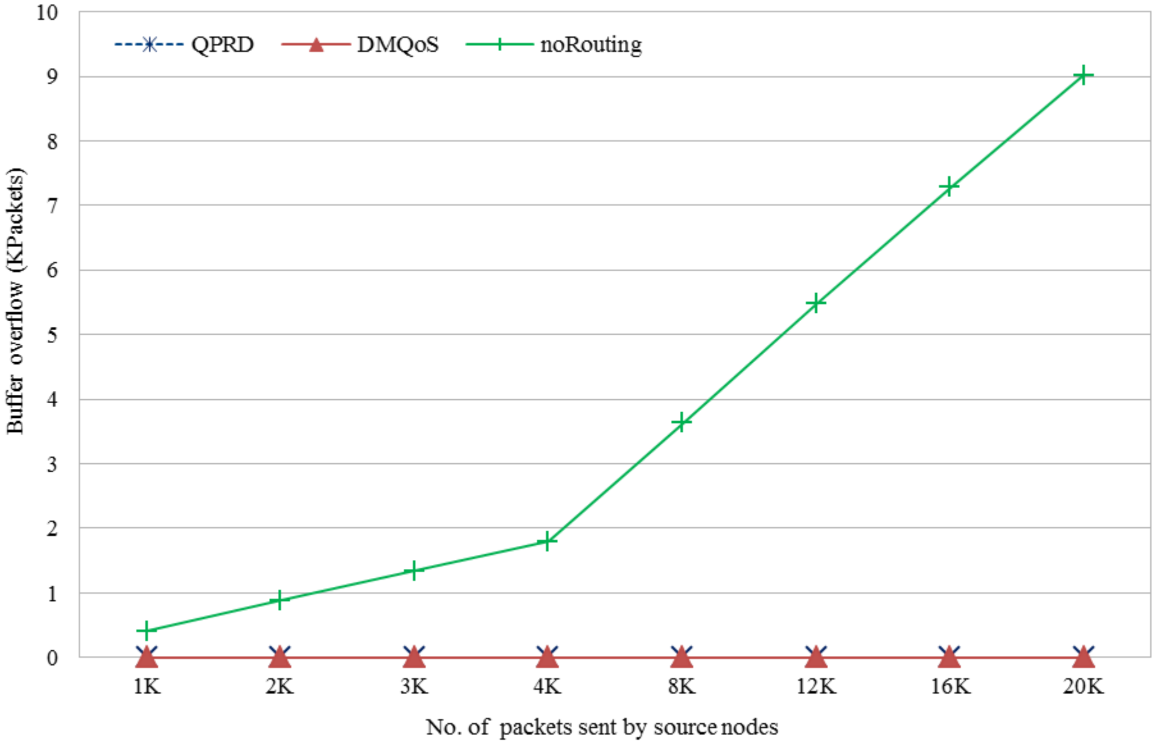


Figure 6-18: Packets dropped due to MAC buffer overflow

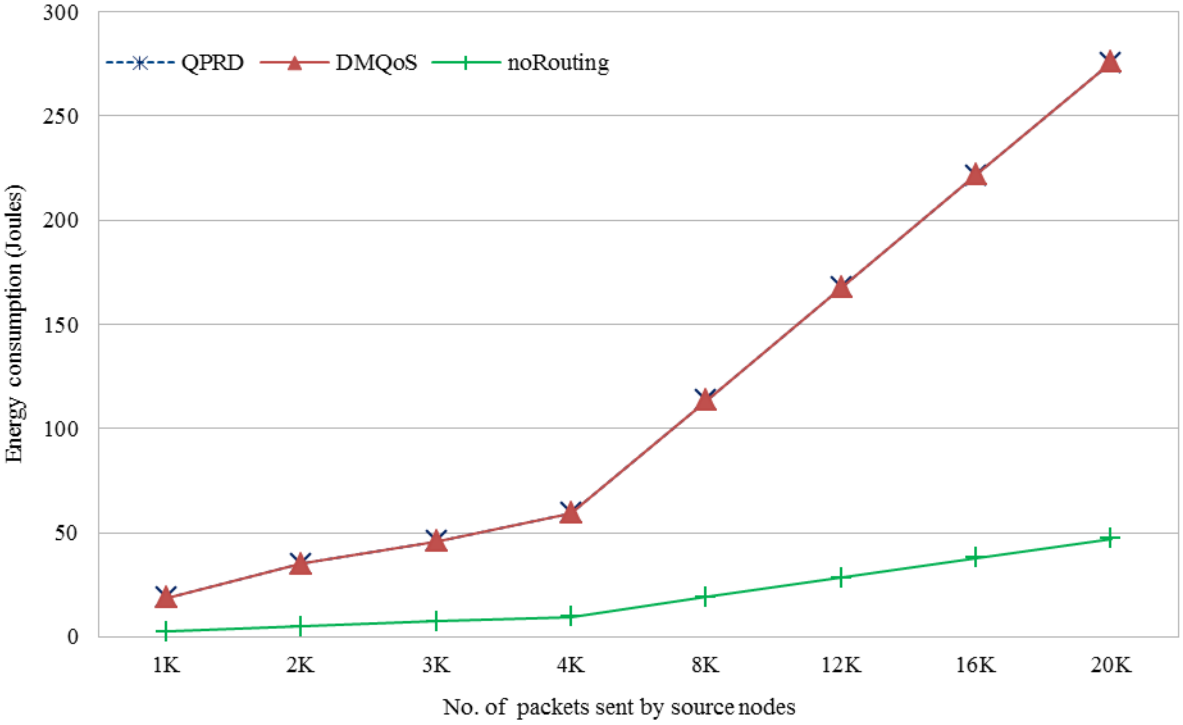


Figure 6-19: Overall energy consumption

6.5. SUMMARY

A new novel modular QoS-aware routing protocol for hospital BAN communication is proposed in this chapter. The architecture of the new protocol consists of seven modules: the MAC receiver, the Delay Module (DM), the Packet Classifier (PC), the Hello Protocol Module (HPM), the Routing Services Module (RSM), the QoS-aware Queuing Module (QQM), and the MAC transmitter.

The proposed routing protocol provides a mechanism for the end-to-end path delay calculation of all possible paths from a source to destination and then decides the best possible path by considering the path delay requirements of the delay-sensitive packets.

OMNeT++ based simulator Castalia 3.2 [78] is used to test the performance of the proposed protocol (QPRD) and compare it with DMQoS [33] and noRouting. The simulations are performed for both the movable source and stationary scenarios. The results show that the QPRD offers over 94% successful data transmission rates for delay-sensitive packets in a stationary patient scenario. QPRD provides about 35% better results in terms of successful transmission rate than DMQoS in the movable patient scenario. The simulation results show that the QPRD improves the reliability of body area networks by 40% on average for each scenario by decreasing the number of packet timeouts with zero and averaging 729 packets for the static and mobile patient scenarios, respectively. In addition, QPRD results in an average of 25% lower overall network traffic for each mobile and static patient scenarios as compared to similar protocols.

CHAPTER 7

QPRR: QoS-aware Peering Routing Protocol for Reliability-Sensitive Data

The reliability, energy efficiency, and real-time display of patient's data are important factors for Body Area Network (BAN) communication in the hospital environment. This chapter proposes a novel routing protocol by considering the reliability requirements of QoS sensitive BAN data. The proposed algorithm improves on the reliable delivery of critical BAN data at the destination. Extensive simulations in the OMNeT++ based simulator Castalia 3.2 have been performed to show the better performance of the proposed extensions to the QoS based routing protocol to increase the reliable delivery of sensitive data. Enhanced reliability is demonstrated in terms of increased successful transmission rate, lower network routing traffic (Hello packets), and lower end-to-end delay (latency) in both stationary and movable patient scenarios. This chapter is organized as follows: Sections 7.1 and 7.2 discuss the motivation and related work, respectively. Section 7.3 provides the proposed QoS-aware Peering Routing protocol for Reliability-sensitive data (QPRR). Section 7.4 presents performance evaluation of the proposed QoS-aware peering routing protocol. Section 7.5 gives a summary of the chapter.

7.1. Motivation

Body Area Network (BAN) is an emerging field which is used to monitor a patient's vital signs as well as other contextual information. The challenges and characteristics of BAN are different than the conventional Wireless Sensor Network (WSN) due to the specific requirements of its architecture, density, data rate, latency, and mobility requirements. These challenges [33] include the high level of data reliability required in the communication of critical information such as blood pressure (BP), electrocardiography (ECG), and electroencephalography (EEG) readings. Other challenges [11] include the small size of implanted body sensors, access to sensors due to their implanting in the human body, the very low power supply to implanted sensors, and mobility of the sensors when a patient is moving. Reliable communication of patient data in real-time and efficient routing are some of the challenges that need to be addressed. The projects like SMART [35], CareNet [68], AID-N [36], and ALARM-NET [34] provide different methods to monitor patient data. In these mechanisms, the implanted or wearable sensors transmit the BAN data to the central database server and the central database allows the users to monitor BAN data. However, techniques to increase the reliable monitoring of real-time display of BAN data in hospital environment have not been considered by these methods.

The mobility of the patient in the hospital may require a change to the dedicated display unit which is used to display patient data. In order to resolve these problems, a new QoS based routing protocol for reliable BAN communication is required in order to display real-time patient data in such environment. The proposed BAN routing protocol is designed to reliably communicate and display real-time BAN data, and dynamically discover the dedicated medical display device even when the patient is transferred from one room to another within an indoor hospital environment. The proposed QPRR is deployed in the ZK-BAN peering framework discussed in Chapter 4, and uses both centralized and distributed approaches. In the centralized approach, the information of BANs and display units are stored in a central computer which helps to improve privacy and better control on BAN communication. On the other hand, since the BAN data is displayed on the display unit in a distributed manner, this reduces overall network traffic and helps to improve the reliable monitoring of vital signs even when the patient is moving.

7.2. Related Work

Typically, in BAN communication, the body implant and wearable sensors send their data to a central and computationally more powerful device known as the coordinator. The coordinator also behaves like a router in BANs. BAN communication factors include high reliability, short range transmission, variable data rate, low energy consumption of devices, and non-interference with other wireless devices. Support for BAN communication is not provided in current Personal Area Network (PAN) standards [22].

However, the IEEE 802.15 task group 6 is assigned the task of developing a standard for BAN which should be compatible with a transmission range of 3 meters, data rates of up to 10Kbps and support for QoS [3]. The goal is to optimize BAN operations not only related to inside or outside of the human body, but also to be compatible with other applications like consumer medical and personal entertainment.

The challenges related to the management of patients' medical information in hospital environment can be resolved by using a smart monitoring system for BAN data [43]. The Scalable Medical Alert and Response Technology (SMART) [35] provides a solution to monitor the patient's data in hospital emergency areas. The body sensors transmit the data to the PDA and then PDA forwards it to the next tier by using wireless standard IEEE 802.11b. An integrated wireless sensor based solution is provided by CareNet [68] for monitoring the patient's data from remote hospitals. Both projects [35, 68] use two-tier wireless communication. The mechanism in [36] suggests a GPS based system to monitor the patient's data in outdoor BAN communication. ALARM-NET [34] provides a wireless sensor network for assisted-living and residential monitoring system with a query based protocol. In [73], a three-tier communication approach is suggested to store the BAN data on the central database and then the patient's data is available for monitoring and analysis. The projects [34, 35, 36, 68, 73] consider the centralized approach for the monitoring of the patient's data. However, a real-time display of data by considering the reliability-sensitive packets is not addressed. To monitor the patient's data from the central server increases delay and is exacerbated when the central server is disconnected from the healthcare system due to link failure.

Typically, BAN communication is based on a hierarchical model with three communication tiers [80]. In Chapter 4, a general BAN communication framework ZK-BAN is presented. In tier 1, implanted and wearable sensors send data to the BAN coordinator. In tier 2, the next hop of a BAN coordinator can be any wireless device including smart phones, a nursing station computer or medical display devices. The communication devices forward BAN data to tier 3 communication devices such as a wireless access point or a broadband connection to the Internet.

BAN communication can be in indoor and outdoor environment. The indoor scenario is when the BAN communication is in hospital and home. The indoor-hospital BAN communication has different requirements than indoor-home BAN communication. Typically, every patient's BAN needs a Medical Display Coordinator (MDC) for displaying the patient's data in the indoor hospital BAN communication scenario. Normally, the placement of MDC is within 3 meters of BAN coordinator. For example, the BAN data is displayed on the MDC of the hospital Emergency Room (ER) when a patient is in the ER. For further treatment the patient can be moved to the Operation Room (OR), Patient Room (PR), or Intensive Care Unit (ICU). In such cases, the patient's BAN data needs to be displayed on the MDC at the new location.

The two modes of BAN communication are the centralized and distributed modes. The Nursing Station Coordinator (NSC) is the central computer which contains the information about all BAN coordinators and MDCs. The operator/nurse updates the peering information of the BANs in NSC. Each BAN coordinator will first get the peering information from NSC in centralized communication mode, and then it will discover and send data to its peer in the distributed communication mode. As there are many MDCs in the hospital, a mechanism is required to seamlessly route and reliably store or display real-time BAN data on the MDC dedicated to the patient. For this, we suggest a hybrid peering framework (ZK-BAN) in Chapter 4. In this framework the BAN coordinator is peered with a medical display device. The Energy-aware Peering Routing protocol (EPR) given in Chapter 5 chooses the next hop device by considering the energy level and geographic location of the devices. BAN data packets can be divided into two classes Ordinary Packets (OPs) and Reliability-Sensitive Packets (RSPs). OP contains information such as glucose level, SPO2, and body temperature. RSPs contain critical information such as PH, blood pressure (BP), electrocardiography (ECG), and electroencephalography (EEG) readings. QPRD [7] divides the BAN data into two types, OPs and Delay-Sensitive Packets (DSPs), and provides the mechanism to find the best route for both data types by considering the QoS requirements. EPR finds the best energy-efficient path for sending OPs. However, EPR and QPRD [7] do not provide any mechanisms for reliably transferring RSPs from the source to the destination. A routing protocol suggested in [33] provides different algorithms for handling the different classes of data packets depending on their QoS requirements. DMQoS [33] considers only BAN coordinators as a next hop in BAN communication. However BAN environment in a hospital has different requirements including different device types like NSC, BANs or MDCs as next hops. [74] provides a mechanism for combining or splitting multiple BANs in inter-BAN communication. Authors in [74] present a reasonable approach for internetworking of BANs; however, it does not consider QoS requirements and reliable real-time display of BAN data. There are other ideas [43, 44, 45, 46] for efficient routing of wireless sensor network but these do not consider the requirements of reliable BAN communication in a hospital scenario.

7.3. QoS-aware Peering Routing Protocol for Reliability-sensitive Data (QPRR)

The proposed QoS-aware routing protocol is intended to be employed with the indoor hospital ZK-BAN peering framework presented in Chapter 4. The devices used in indoor hospital BAN communication are divided into three types by considering their energy levels. The Nursing Station Coordinator (NSC) is type 1 device which is connected directly with the power source. The Medical Display Coordinators (MDCs) are considered to be type 2 devices which use replaceable batteries. The BAN coordinators have limited energy availability and are considered to be type 3 devices. In the ZK-BAN peering framework, the information of BAN coordinators and their respective peer MDCs are stored at the NSC. The BAN coordinator needs to first connect with NSC for getting peer information and then it starts displaying the real-time data on the peer MDC. An Energy-aware Peering Routing protocol (EPR) is provided in Chapter 5 for choosing the best next hop for Ordinary Packets (OPs) by considering the energy availability and geographic information of the devices.

The EPR results in better performance than other protocols [33, 43, 44, 45, 46] in terms of reduced network traffic, successful data transmission rate, reduced number of packets forwarded by intermediate nodes, and overall lower energy consumption. However the mechanism for sending Reliability-Sensitive Packets (RSPs) has not been considered. The DMQoS [33] classifies the data into four types: Ordinary Packets (OPs), Critical Packets (CPs), Reliability-driven Packets (RPs), and Delay-driven Packets (DPs). The performance of DMQoS [33] for reliability-driven packets is better than several state-of-the-art approaches [43, 44, 45, 46] in terms of successful transmission rate, traffic load, and operation energy overload. DMQoS determines the next hop by considering the highest reliability of the device, and then determines the other most reliable next hop towards destination. The disadvantage of this hop-by-hop reliability proposed in DMQoS is that the source node depends only on the neighbor node's reliability information. It is possible that the source node sends the packets to the neighbor node with highest reliability, but the neighbor node doesn't find the required reliability among its neighbor nodes, resulting in dropped data packets. In this case the source is getting acknowledgements from neighbor node that the packets are successfully transmitted but in reality the packets are dropped by upstream nodes instead of being forwarded to the destination.

Moreover, by using the hop-by-hop reliability, the network traffic is increased and end-to-end reliability is not ensured by DMQoS. Also, the high transmit power of the devices and the stationary natures of the nodes are other shortcomings of the DMQoS approach. The proposed protocol QPRR addresses these shortcomings by considering a low transmit power of -25dBm, and both stationary and movable nodes, and more importantly, by choosing the next hop device based on the most reliable end-to-end path(s) from the source node to the destination.

The notations used in this chapter and their descriptions are summarized in Table 7-1. In the proposed QPRR, a mechanism is introduced to 1) calculate the path reliabilities of all possible paths from the source node to the destination and 2) determine the degree of duplication for sending reliability-sensitive packets. For each destination, the routing table contains information about the next hop devices connected to the three most reliable paths and the path reliabilities values of first path ($R_{option1(i,Dst)}$), aggregate reliability of two paths ($R_{option2(i,Dst)}$), and aggregate reliability of three paths ($R_{option3(i,Dst)}$). For any RSP, if the single path ($R_{option1(i,Dst)}$) can achieve the reliability requirement, the source node sends the RSP through that path. If the $R_{option1(i,Dst)}$ is lower than the required data reliability (R_{req}) then QPRR compares the R_{req} with the $R_{option2(i,Dst)}$. The node sends a copy of RSP to each next hop of two paths whose aggregate reliability is better than the required reliability, if $R_{option2(i,Dst)}$ is greater than the R_{req}. Otherwise, QPRR compares the $R_{option3(i,Dst)}$ with the R_{req}. The node now sends a duplicate packet of RSP to each next hop of three paths whose aggregate reliability is better than the required reliability. The node drops the RSP if $R_{option3(i,Dst)}$ is less than R_{req}. The benefit of using redundant paths is to improve the end-to-end reliability.

Table 7-1: Notations for the proposed algorithm

Field ID	Description
Node *i*	Source node
Node *j*	Neighbor node of source node
Node *Dst*	Destination node (i.e. NSC, MDC, BAN)
$\mathbf{ID_{Dst}}$	Destination ID
$\mathbf{L_{Dst}}$	Destination location
$\mathbf{ID_{j}}$	Neighbor node *j* ID
$\mathbf{L_{j}}$	Neighbor node *j* location
$\mathbf{D_{(j,Dst)}}$	Distance between neighbor node *j* and destination *Dst*
$\mathbf{E_{j}}$	Residual energy of node *j*
$\mathbf{T_{j}}$	Device Type of node *j*
$\mathbf{R_{path(j,Dst)}}$	Path Reliability between neighbor *j* and destination
$\mathbf{D_{(i,j)}}$	Distance between node *i* to neighbor node *j*
$\mathbf{R_{link(i,j)}}$	Link reliability from node *i* to neighbor node *j*
$\mathbf{R_{path(i,Dst)}}$	Path reliability from node *i* to destination *Dst*
$\mathbf{NH_{(i,Dst)}}$	Next hop between node *i* and destination *Dst*
$\mathbf{NH_{E}}$	Energy-aware next hop
$\mathbf{NH_{R1}}$	1st reliable next hop
$\mathbf{NH_{R2}}$	2nd reliable next hop
$\mathbf{NH_{R3}}$	3rd reliable next hop
$\mathbf{R_{path1(i,Dst)}}$	1st Path Reliability from node *i* to destination *Dst*
$\mathbf{R_{path2(i,Dst)}}$	2nd Path Reliability from node *i* to destination *Dst*
$\mathbf{R_{path3(i,Dst)}}$	3rd Path Reliability from node *i* to destination *Dst*
$\mathbf{R_{req}}$	Required Reliability of reliability-sensitive packets
$\mathbf{R_{option1(i,Dst)}}$	1st option Reliability for sending reliability-sensitive packets
$\mathbf{R_{option2(i,Dst)}}$	2nd option Reliability for sending reliability-sensitive packets
$\mathbf{R_{option3(i,Dst)}}$	3rd option Reliability for sending reliability-sensitive packets

The architecture of proposed QPRR is shown in Figure 7-1. It consists of five modules: the Reliability Module (RM), the Packet Classifier (PC), the Hello Protocol Module (HPM), the Routing Services Module (RSM), and the QoS-aware Queuing Module (QQM). The QPRR modules are discussed below.

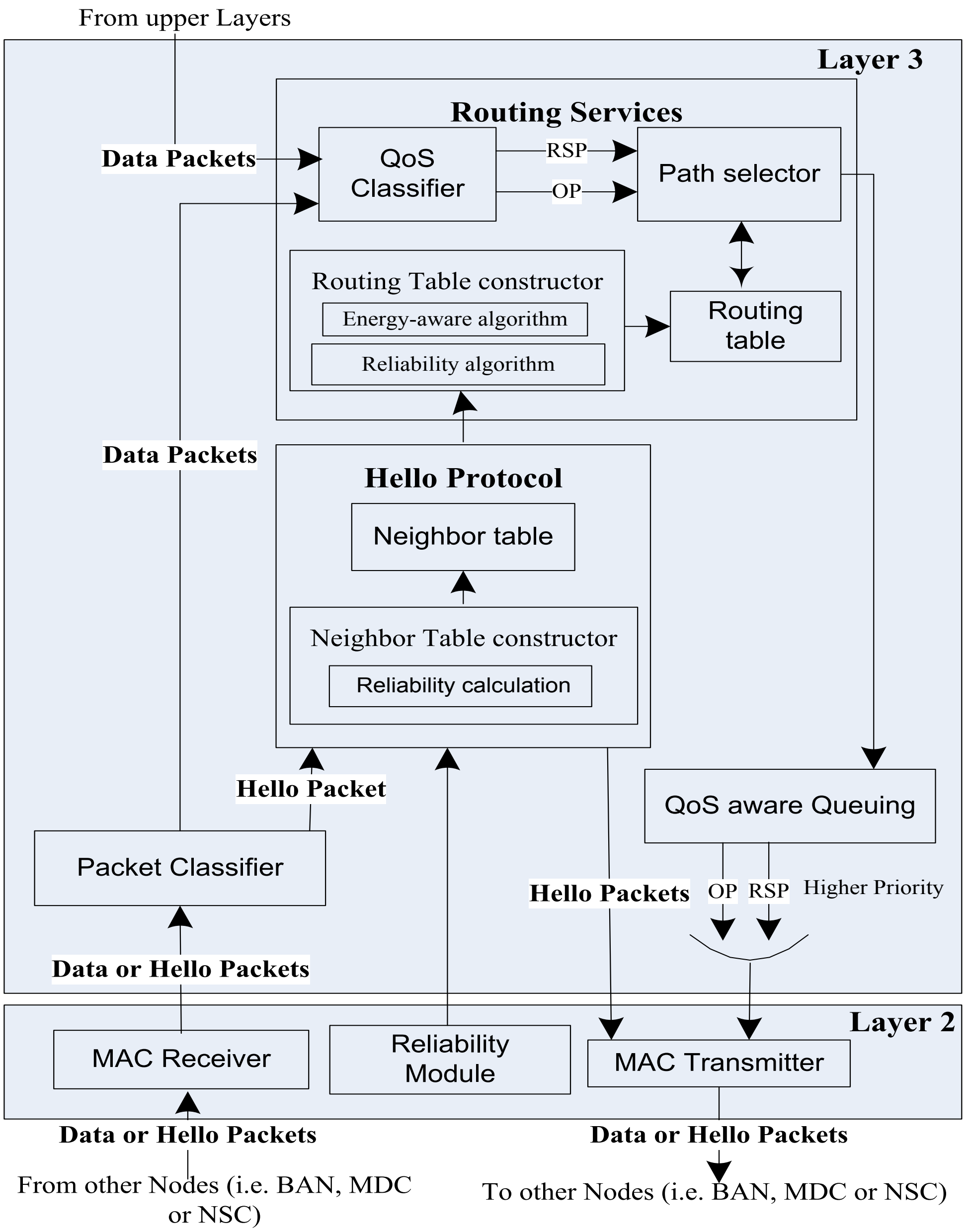


Figure 7-1: QPRR protocol architecture

7.3.1. Reliability Module (RM)

The reliability module monitors the numbers of packets sent to neighbor node *j* and the number of acknowledgements received from neighbor node *j*. Reliability module passes the information of successful data packets' transmission acknowledgements from MAC layer to the network layer. The network layer uses this information to calculate the link reliability between the node *i* to the neighbor node *j* ($R_{link(i,j)}$).

7.3.2. Packet Classifier (PC)

The packet classifier differentiates the data packets and Hello packets received from MAC receiver. The packet classifier forwards the data and Hello packets to the routing services module and Hello protocol module respectively.

7.3.3. Hello Protocol Module (HPM)

Hello protocol module consists of two sub-modules: the neighbor table constructor and the neighbor table. The function of neighbor table constructor is to build the neighbor table according to the information received from both Hello packet and the MAC layer reliability module. We assume that each type 1 (NSC) and type 2 (MDC) device first broadcasts their Hello packets. After node *i* receives the Hello packet, the neighbor table constructor updates the Hello packet with its own $R_{path(i,Dst)}$. The new Hello packet will be broadcast to the other nodes. The Hello packet fields are shown in Figure 7-2.

ID_{Dst}	L_{Dst}	ID_j	L_j	$D_{(j,Dst)}$	E_j	T_j	$R_{path(j,Dst)}$

Neighbor table contains the fields of the reliabilities for both hop-by-hop ($R_{link(i,j)}$) and end-to-end ($R_{path(i,Dst)}$). The neighbor table is periodically updated with the reception of new Hello packets. The neighbor table structure of node *i* is shown in Figure 7-3.

ID_{Dst}	L_{Dst}	ID_j	L_j	$D_{(j,Dst)}$	$D_{(i,j)}$	E_j	T_j	$R_{link(i,j)}$	$R_{path(i,Dst)}$

Figure 7-3: Neighbor table structure

The average probability of successful transmission after every 4 seconds is calculated by using Equation 7-1.

$$\bar{X}_i = \frac{N_{Acks}}{N_{Trans}} \qquad (7\text{-}1)$$

The link reliability between node *i* and neighbor node *j* ($R_{link(i,j)}$) is calculated by using the Exponentially Weighted Moving Average (EWMA), as given in Equation 7-2.

$$R_{link(i,j)} = \left(1 - \rho\right)R_{link(i,j)} + \rho \times \bar{X}_\imath \qquad (7\text{-}2)$$

where ρ is the average weighting factor that satisfies $0 < \rho \leq 1$. The best suited value of ρ for our simulations is 0.4.

The path reliability between node *i* and destination node *Dst* ($\mathrm{R_{path(i,Dst)}}$) is calculated by using the Equation 7-3.

$$R_{path(i,Dst)} = R_{link(i,j)} \times R_{path(j,Dst)} \qquad (7\text{-}3)$$

An example of finding the path reliability from node *i* (B_4) to *Dst* (NSC) is shown in Figure 7-4.

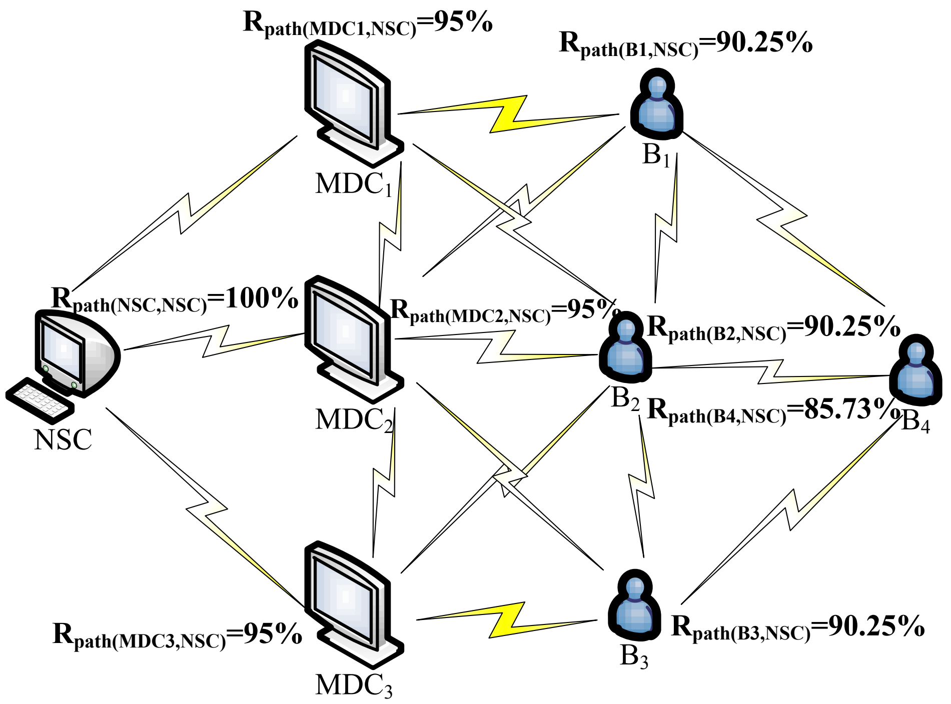


Figure 7-4: Example of finding the path reliabilities

The calculation of the reliability of a single path B_4-B_1-MDC_1-NSC is illustrated here. For illustration purposes, we assume that the $R_{path(NSC,NSC)}$ is 100% and $R_{link(i,j)}$ for every node is 95%. Each node calculates the path reliability from itself to the NSC. The path reliability of MDC_1 ($\mathrm{R_{path(MDC_1,NSC)}}$) is calculated by using Equation 7-3.

$$\mathrm{R}_{\mathrm{path}(\mathrm{MDC}_1,\mathrm{NSC})} = \mathrm{R}_{\mathrm{link}(\mathrm{MDC}_1,\mathrm{NSC})} \times \mathrm{R}_{\mathrm{path}(\mathrm{NSC},\mathrm{NSC})} = 95\% \times 100\% = 95\%$$

The path reliability of BAN B$_1$ ($\mathrm{R}_{\mathrm{path}(\mathrm{B}_1,\mathrm{NSC})}$) is calculated as shown below.

$$\begin{aligned}\mathrm{R}_{\mathrm{path}(\mathrm{B}_1,\mathrm{NSC})} &= \mathrm{R}_{\mathrm{link}(\mathrm{B}_1,\mathrm{MDC}_1)} \times \mathrm{R}_{\mathrm{path}(\mathrm{MDC}_1,\mathrm{NSC})} \\ &= 95\% \times 95\% = 90.25\%\end{aligned}$$

The node B$_4$ calculates its path reliability ($\mathrm{R}_{\mathrm{path}(\mathrm{B}_4,\mathrm{NSC})}$) as shown below.

$$\begin{aligned}\mathrm{R}_{\mathrm{path}(\mathrm{B}_4,\mathrm{NSC})} &= \mathrm{R}_{\mathrm{link}(\mathrm{B}_4,\mathrm{B}_1)} \times \mathrm{R}_{\mathrm{path}(\mathrm{B}_1,\mathrm{NSC})} \\ &= 95\% \times 90.25\% = 85.73\%\end{aligned}$$

7.3.4. Routing Services Module (RSM)

The function of the routing services module is to build the routing table, classify data packets into RSPs and OPs, and choose the best path(s) for each data class. The routing services module consists of routing table constructor, routing table, QoS classifier, and path selector. The routing table structure of node *i* is shown in Figure 7-5.

$\mathrm{ID}_{\mathrm{Dst}}$	$\mathrm{L}_{\mathrm{Dst}}$	NH_{E}	$\mathrm{NH}_{\mathrm{R1}}$	$\mathrm{NH}_{\mathrm{R2}}$	$\mathrm{NH}_{\mathrm{R3}}$	$\mathrm{R}_{\mathrm{option1(i,Dst)}}$	$\mathrm{R}_{\mathrm{option2(i,Dst)}}$	$\mathrm{R}_{\mathrm{option3(i,Dst)}}$

Figure 7-5: Routing table structure

In the routing table, four next hop entries NH$_E$, NH$_{R1}$, NH$_{R2}$, and NH$_{R3}$ are given for each destination *Dst*. The energy-aware algorithm (EPR) and reliability algorithm are used by the routing table constructor. EPR [6] is used to find next hop NH$_E$ for OP. NH$_E$ is calculated by considering the residual energy and geographic location of the neighbor nodes.

For RSP, the new proposed algorithm finds three possible paths to ensure the minimum required reliability is met. The numbers of three possible paths are experimentaly chosen. The overall network traffic increases with the increase of redundant paths. Our experiments show that the use of maximum two possible redundant paths does not provide the enough throughputs. The neighbor table entries are in turn used to build the routing table. There are many records in the neighbor table for each destination. For example, Figure 7-4 shows that there are many paths from B$_4$ to NSC. Some of these paths are B_4-B_1-MDC_1-NSC, B_4-B_2-MDC_2-NSC, B_4-B_3-MDC_3-NSC, B_4-B_2-MDC_1-NSC, B_4-B_1-MDC_2-NSC, etc. For each destination, the three paths with highest reliabilities ($R_{path1(i,Dst)}$, $R_{path2(i,Dst)}$, $R_{path3(i,Dst)}$) are chosen and their corresponding next hops (NH_{R1}, NH_{R2}, NH_{R3}) are stored in the routing table. The routing table constructor calculates and stores the three options for RSP. The first reliability path option ($R_{option1(i,Dst)}$) is the reliability of the highest path, i.e. $R_{path1(i,Dst)}$.

$$R_{option1(i,Dst)} = R_{path1(i,Dst)} \tag{7-4}$$

The error probabilities of the three paths are calculated by using Equations 7-5, 7-6, and 7-7.

$$P_{error(1)} = 1 - R_{path1(i,Dst)} \tag{7-5}$$

$$P_{error(2)} = 1 - R_{path2(i,Dst)} \tag{7-6}$$

$$P_{error(3)} = 1 - R_{path3(i,Dst)} \tag{7-7}$$

The $R_{option2(i,Dst)}$ is calculated by using the error probabilities of the two paths having the highest reliability values.

$$R_{option2(i,Dst)} = 1 - \prod_{k=1}^{2} P_{error(k)} \tag{7-8}$$

The error probabilities of all three paths are used to calculate the $R_{option3(i,Dst)}$.

$$R_{option3(i,Dst)} = 1 - \prod_{k=1}^{3} P_{error(k)} \tag{7-9}$$

The $R_{option1(i,Dst)}$, $R_{option2(i,Dst)}$, and $R_{option3(i,Dst)}$ are the reliabilities of sending the data by using one-path, two-paths, and three-paths respectively.

Algorithm 7.1 shows that the node *i* identifies the next hop candidates by searching the records which have the same ID_{Dst} in neighbor table and stores them in the variable NH_R (line 2). If NH_R is empty, this means there is no next hop stored in NH_R. The node stores NULL to NH_{R1}, NH_{R2}, NH_{R3}, $R_{option1(i,Dst)}$, $R_{option2(i,Dst)}$ and $R_{option3(i,Dst)}$. If NH_R is not empty, the next hop nodes information are stored in the routing table one after another in descending order of their path reliabilities $R_{path(i,Dst)}$.

The first neighbor node j with the highest reliability in the routing table is stored as NH_{R1} (line 7). If there are two entries in NH_R then the aggregate reliability of first and second paths ($R_{option2(i,Dst)}$) is calculated (line 10-13). In case of more than two entries in NH_R, the aggregate reliability of first, second, and third paths ($R_{option3(i,Dst)}$) is calculated (line 15-18). Another important component of RSM is Path Selector (PS). The data packets from both upper layers and packet classifier are received by QoS classifier. The QoS classifier classifies the packets into RSPs and OPs.

Algorithm 7.1 Routing table construction algorithm for reliability – sensitive data

```
     INPUT: Neighbor table, $i$'s neighbor table records $NH_{(i,Dst)}$, $\forall\ Dst \in \{MDC, NSC, BAN\}$
1.  for each destination $Dst \in \{NSC, MDC, BAN\}$ do
2.    $NH_R = \{$All neighbor nodes $j \in NH_{(i,Dst)}\}$
3.    if ($NH_R == NULL$) then
4.      Put NULL in $NH_{R1}, NH_{R2}, NH_{R3}, R_{option1(i,Dst)}, R_{option2(i,Dst)}, R_{option3(i,Dst)}$
5.      else
6.       Sort $NH_R$ in descending order of $R_{path(i,Dst)}$
7.       $NH_{R1}$ = first neighbor node $j \in NH_R$
8.       $R_{option1(i,Dst)} = R_{path(i,Dst)}$
9.       $P_{error} = 1 - R_{option1(i,Dst)}$
10.          if ($|NH_R| > 1$)
11.            $NH_{R2}$ = second neighbor node $j \in NH_R$
12.            $P_{error} = P_{error} * (1 - R_{path(i,Dst)})$
13.            $R_{option2(i,Dst)} = 1 - P_{error}$
14.          end if
15.          if ($|NH_R| > 2$)
16.            $NH_{R3}$ = third neighbor node $j \in NH_R$
17.            $P_{error} = P_{error} * (1 - R_{path(i,Dst)})$
18.            $R_{option3(i,Dst)} = 1 - P_{error}$
19.          end if
20.     end if
21.   end for
```

For each data packet, the PS checks the QoS requirement and chooses the appropriate next hop(s) by using Algorithm 7.2. If the packet is RSP, the PS sends these packets by using a single path through NH_{R1} if the reliability of that path is more than R_{req} (lines 3-4). Otherwise, two paths are used if the aggregate reliability of these paths exceeds the R_{req} (lines 5-6). If not, three paths are used as long as their aggregate reliability is more than R_{req} (lines 7-8). Otherwise the packet is dropped. If the packet is an OP, the PS returns the next hop NH_E which is calculated by the EPR (lines 12-13).

```
Algorithm 7.2 Path selector algorithm for reliability − sensitive data
INPUT: Routing table, $i$'s routing table records $\mathrm{NH}_{(i,Dst)}, \forall Dst \in \{\mathrm{MDC}, \mathrm{NSC}, \mathrm{BAN}\}$
for each data packet do
    if data packet is reliability − sensitive packet (RSP)
        if ($R_{option1(i,Dst)} > R_{req}$)
            send to $\mathrm{NH}_{R1}$
        else if ($R_{option2(i,Dst)} > R_{req}$)
            send to $\mathrm{NH}_{R1}$ and $\mathrm{NH}_{R2}$
        else if ($R_{option3(i,Dst)} > R_{req}$)
            send to $\mathrm{NH}_{R1}$, $\mathrm{NH}_{R2}$ and $\mathrm{NH}_{R3}$
        else
            drop the packet immediately
        end if
    else if data packet is Ordinary Packet (OP)
        send to $\mathrm{NH}_{E}$
    else
        drop the packet immediately
    end if
end for
```

7.3.5. QoS-aware Queuing Module (QQM)

After choosing the proper next hop(s) the RSM sends the data packets to the QoS-aware Queuing Module (QQM). QQM works the same way as given in [33]. QQM differentiates the data packets in two types and enqueues each packet in the separate queue. The priority of the RSP queue is higher than the OP queue. All the packets from higher priority RSP queue are sent first. The packets from lower priority OP queue are sent only when the RSP queue is empty. However, in order to treat both queues fairly, a specific period of time is assigned to the RSP queue for sending all of its data to the MAC layer. If the RSP queue fails to send all the data within this allotted time, the OP queue sends its data for a specific time period before returning to service the RSP queue once again.

7.4. PERFORMANCE EVALUATION

The OMNeT++ based simulator Castalia-3.2 [78] is used to perform simulations for comparing the performance of the proposed QPRR protocol. The performance of QPRR is compared with the “DMQoS” routing protocol [33] and no reliability based routing "noRouting". In the noRouting case, the packets are forwarded to random next hop devices instead of algorithm's next hop based on end-to-end reliability. The comparison with noRouting is used to verify whether forwarding the packets to a random next hop device results in a better successful transmission rate than the QPRR routing based on *end-to-end* reliability. The simulation results prove that the QPRR approach based on the path reliabilities is more effective. The simulations are done for different cases of node deployments. The details about the cases, parameters, and comparison results for the simulations are provided below.

7.4.1. Case 1 - Eight Static Nodes

Case 1 uses eight nodes with stationary BAN Coordinators (BANCs). The nodes used in this case are 4 BANCs, 3 MDCs, and 1 NSC. This scenario is similar to an emergency/ICU room or a place in the hospital where the real-time display of many patients' data is required at a time. The density of nodes in this scenario is high with respect to the deployment area i.e. 8 nodes are placed in 6 meter by 6 meter area. The location of these nodes is shown in Figure 7-6.

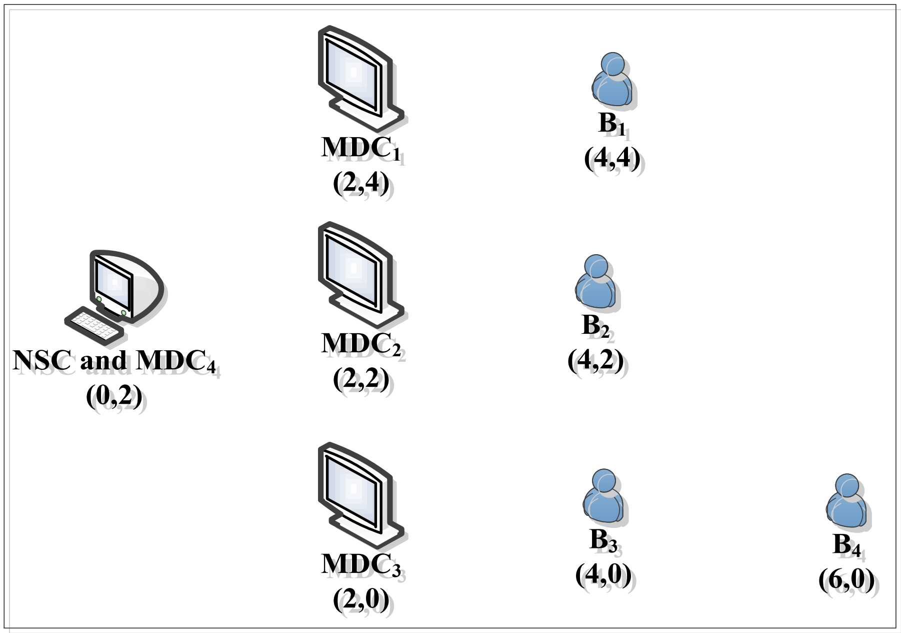


Figure 7-6: Node deployment for Case 1

The source nodes BANCs B_1, B_2, B_3, and B_4 send the data to their peer display units

MDC_1, MDC_2, MDC_3, and MDC_4 (NSC) respectively. The Nursing Station Coordinator (NSC) also works as a display unit for BANC B_4. The peer table of NSC contains the information of the respective peers of all BANCs. All BANC nodes first connect with the NSC in centralized mode. The nodes send the data to their respective peers after getting the peer information from NSC. The data from node B_4 reaches to its respective peer NSC/MDC_4 via one of the nodes B_3, B_2, B_1, MDC_3, MDC_2, and MDC_1.

7.4.2. Case 2 - Eight Nodes with Movable Source Node

Case 2 is similar to Case 1 but has B_4 as movable BAN Coordinator (BANC). The speed of movable BANC is set to 1 meter per second. The speed 1m/s is considered as a fast walking patient. The node B_4 moves vertically as shown by green arrows in Figure 7-7. The source node B_4 displays its data to MDC_4/NSC.

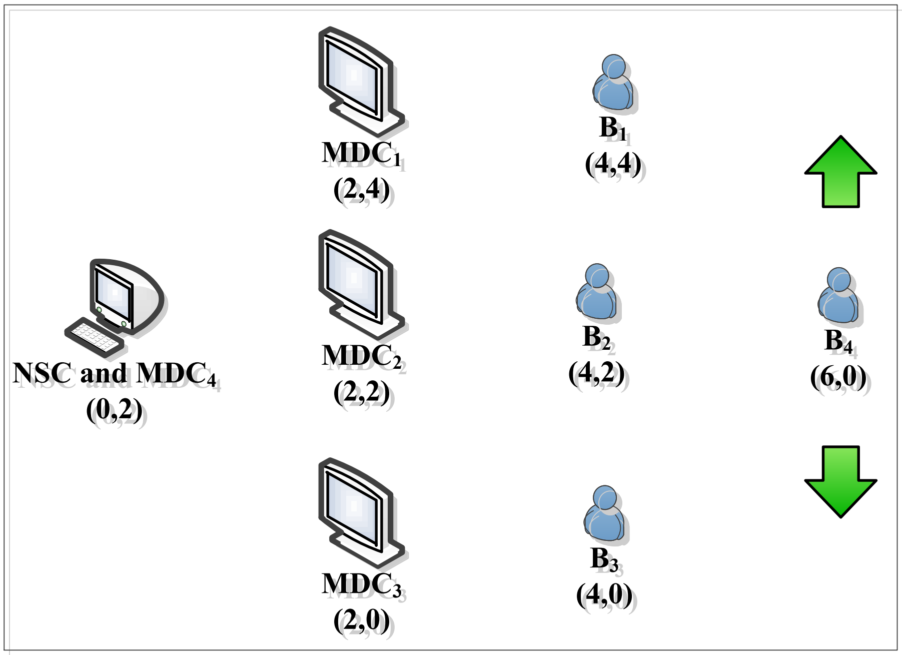


Figure 7-7: Node deployment for Case 2

7.4.3. Case 3 - 49 Nodes with NSC in the Centre

To test the scalability of the proposed QPRR protocol, in Case 3, a 24 bed hospital unit is considered where each bed has a BAN and a MDC, as shown in Figure 7-8. The NSC is in the middle of the unit. The distance between two beds is 3 meters. Each BAN transmits the data to its respective MDC. All the BANs and MDCs are sending or receiving Hello protocols to/from other nodes and the NSC. The proposed protocol QPRR is compared with DMQoS. The experimental setting given in Case 3 is similar to the deployment of nodes used in DMQoS protocol [33]. The performance of our protocols is compared with the DMQoS so the same deployment of nodes helps us to study the results of both protocols in a better way.

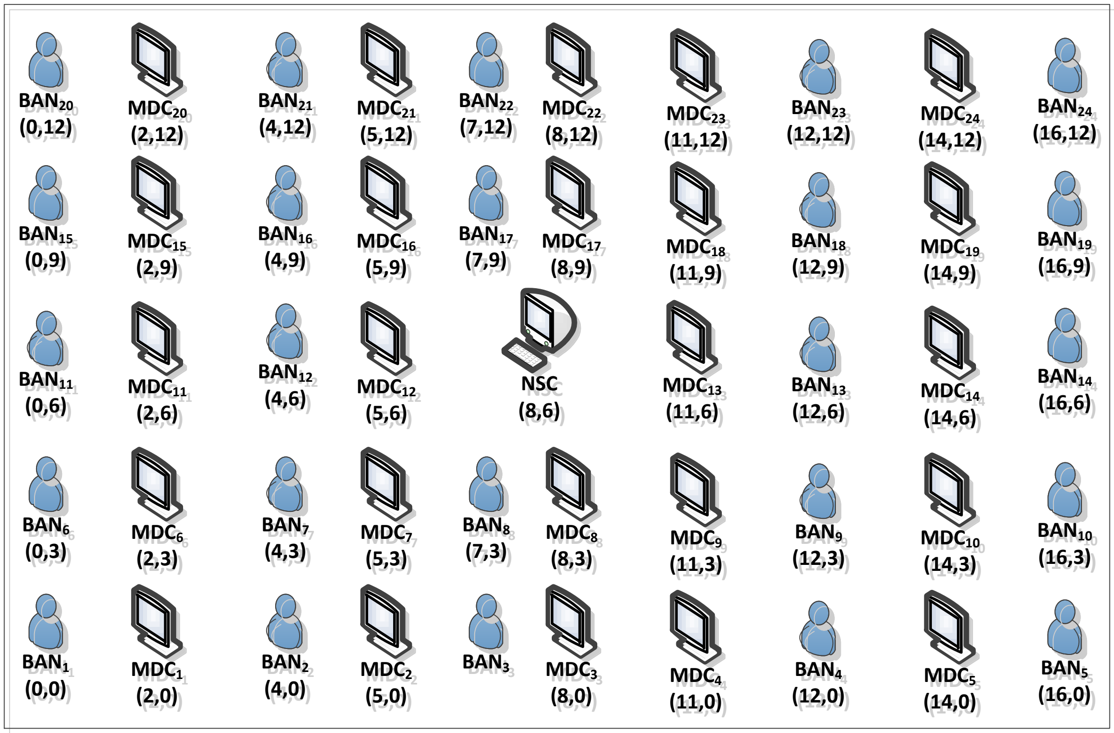


Figure 7-8: Node deployment for Case 3

7.4.4. Case 4 - 49 Nodes in Hospital Environment

In Case 4, a real hospital scenario is considered as shown in Figure 7-9. The Hematology-Oncology unit of any hospital is one of the most important units where cancer patients are treated. The approximate measurements used for this hospital environment are similar to the Hematology-Oncology unit of the Children Hospital named IWK Health Centre Halifax, NS, Canada. The approximate area covered by this unit is 16m by 21m. The total number of patient beds and the size of each bed in this scenario are similar to those in Case 3. Unlike case 3, the NSC in this case is on leftmost side of the deployment area. The patient rooms are in four rows. The room numbers 1-7, 8-12, 13-17, and 10-24 are in rows 1, 2, 3, and 4 respectively. Room number 18 and the nursing station are just in front of all these rows. The MDCs and BANs are movable but normally a MDC placed in a room moves only within that room. BANs can move freely anywhere. We assume that the MDC of one room has a connection with the MDC of the next room.

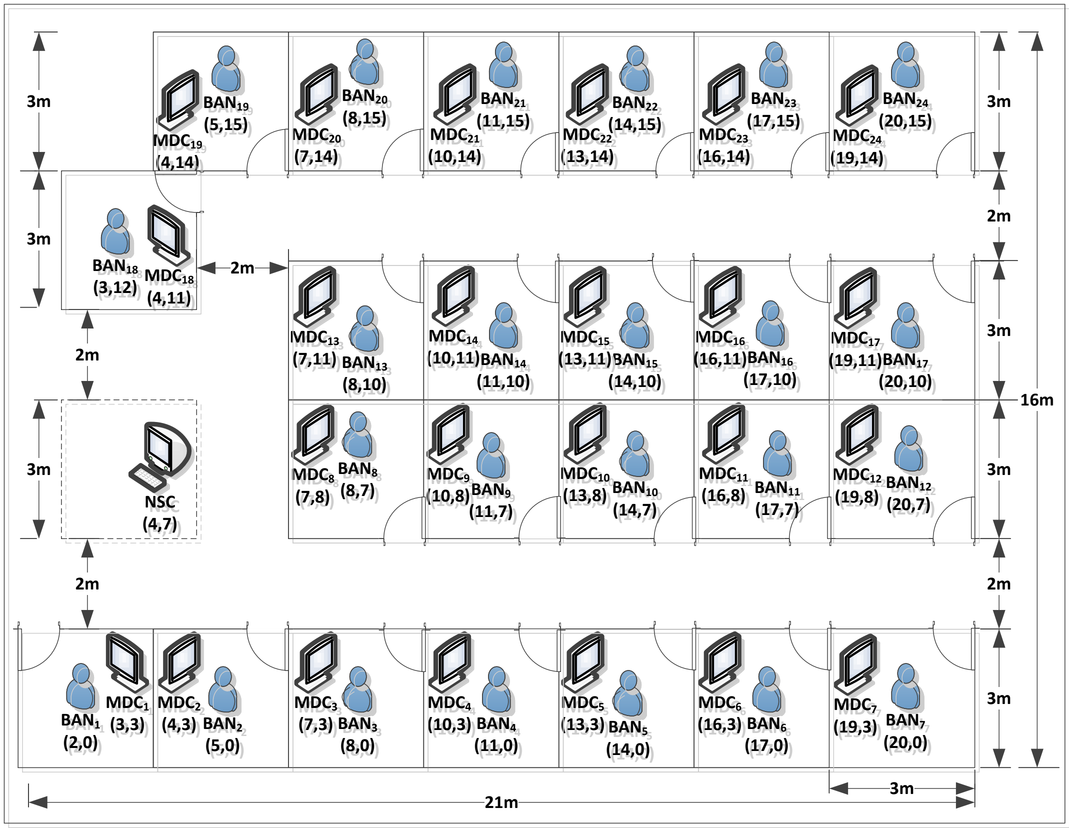


Figure 7-9: Node deployment for Case 4 – 24 patient beds in hospital environment

7.4.5. Case 5 - 93 Nodes in Hospital Environment

Case 5 employs 93 nodes that simulates a real hospital with 46 patients and each patient contains a stationary BANC. This Case illustrates the scalability of the proposed QPRR protocol. The scenario used in this Case is similar to the Pediatric Medical Unit (PMU) of the Children Hospital IWK Health Centre Halifax, NS, Canada. The total area considered is 16m by 39m as shown in Figure 7-10. The nursing station is in the centre of the unit. The size of each room is 3m by 3m. The patient rooms are in four rows. The room numbers 1-13, 14-23, 24-33, and 34-46 are in rows 1, 2, 3, and 4 respectively. The pathways are two meters wide. The placement of MDCs and BANs in each room is similar to Case 4.

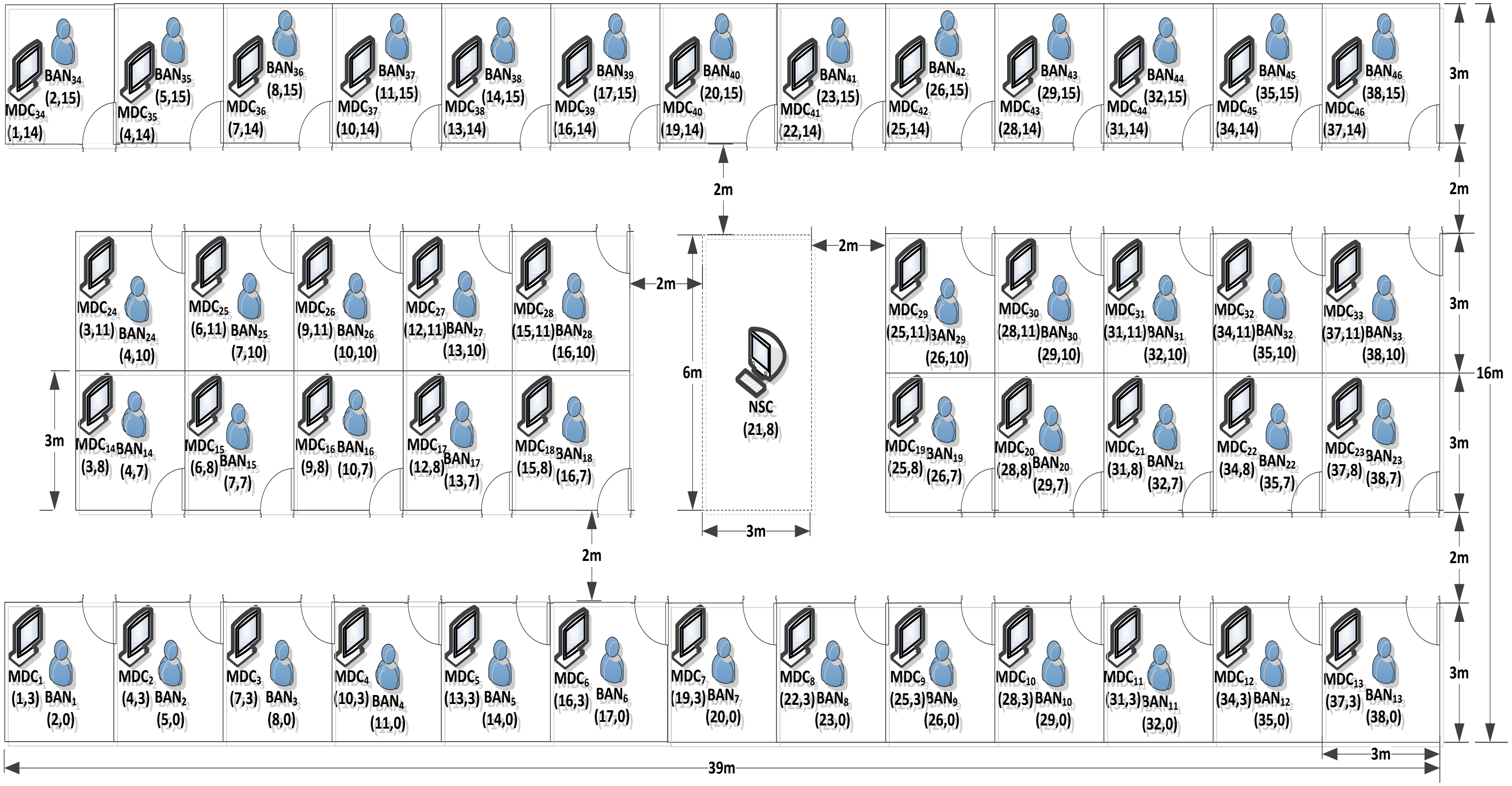


Figure 7-10: Node deployment for case 5 – 46 patient beds in hospital environment

7.4.6. Parameters Used for Simulations

The transmit power used in simulations is -25dBm for all the three cases. The transmission range of -25dBm is about 3 meters which is the recommended value for BAN communication in hospital environment. The network parameters used in our simulations are shown in Table 7-2.

Table 7-2: Parameters information

Deployment	Area	Cases 1 and 2: 6m by 6m Cases 3 and 4: 16m by 21m Case 5: 16m by 39m
	Deployment type	Cases 1, 3, 4, and 5: All nodes are static, Case 2: Movable source node B4
	Number of nodes	Cases 1 and 2: 8 nodes (4 BANs, 3 MDCs, 1 NSC) Cases 3 and 4: 49 nodes (24 BANs, 24 MDCs, 1 NSC) Case 5: 93 nodes (46 BANs, 46 MDCs, 1 NSC)
	Initial nodes locations	Cases 1 and 2: NSC(0,2), MDC$_1$(2,4), MDC$_2$(2,2), MDC$_3$(2,0) B$_1$(4,4), B$_2$(4,2), B$_3$(4,4), B$_4$(6,0) Case 3: as shown in Figure 7-8 Case 4: as shown in Figure 7-9 Case 5: as shown in Figure 7-10
	Initial node energy	18720 J (= 2 AA batteries)
	Buffer size	32 packets
	Link layer trans. Rate	250 Kbps
	Transmit power	−25dBm
Task	Application type	Event − driven
	Max. packet size	32 *Bytes*
	Traffic type	CBR (Constant Bit Rate)
MAC	IEEE 802.15.4	Default values
Simulation	Time	2003 Seconds (3 seconds are setup time. Simulation results are the average of three rotations.)

7.4.7. Performance Comparison for Different Parameters

The successful transmission rate, number of reliability packets dropped, MAC buffer overflow, network traffic, number of Hello packets, overall energy consumption, and end-to-end latency are measured for all the five cases.

The MDCs (MDC$_1$, MDC$_2$, and MDC$_3$) and NSC are considered as destination nodes of the source nodes (BANCs: B$_1$, B$_2$, B$_3$, and B$_4$) respectively. The source node B$_4$ is considered as stationary in Case 1 and movable in Case 2. The source nodes send a total of 20K reliability-sensitive packets in first two cases. The above mentioned parameters are calculated after the transmission of every 1000 packets sent by the source nodes.

In Cases 3 and 4, a 24 bed hospital unit is considered where each bed has a BAN and a MDC, as shown in Figure 7-8 and Figure 7-9. Each BAN transmits the data to its respective MDC. All the BANs and MDCs are sending or receiving Hello protocols to/from other nodes and NSC. In these cases all the BANs send a total of 57.5K reliability-sensitive packets. Different parameters are calculated after the transmission of 3.5K packets initially and then after every 6K packets sent by the source.

To test the scalability of the proposed QPRR protocol in Case 5, a larger area of 16m by 39m is considered. This area models a hospital unit that contains 46 patient rooms. All the BANs send a total of 108.5K reliability-sensitive packets. The calculation of different parameters are done first at 6K packets and then after every 11.5K packets sent by the source nodes. To achieve a 97% confidence interval for the illustrative results, the average of three runs are simulated in every experiment which may introduce a maximum error of $3x10^{-3}$, based on the error calculation done by Castalia 3.2 [79]. The performance comparison of each parameter is discussed below.

7.4.7.1. Throughput

The reliability is measured by calculating the number of packets received successfully at the destination nodes. The throughput or successful transmission rate is measured after the transmission of every 1000 packets sent by the source. Figure 7-11 and Figure 7-12 show that QPRR provides a consistent reliability which is in excess of 88% and 75% for Case 1 and Case 2 respectively. However, from Figure 7-11, it is seen that the reliabilities of DMQoS and noRouting are on average 32% and 35% respectively. For movable BANCs, Figure 7-12 shows that the reliabilities of DMQoS and noRouting decrease to 36% and 20% respectively for 20K packets. In this case the DMQoS protocol, which is using a hop-by-hop reliability, sends data to B_3 because of its higher reliability (calculation done by Castalia) than B_1 and B_2. B_3 forwards the same data to MDC_3 which has higher reliability than MDC_1 and MDC_2.

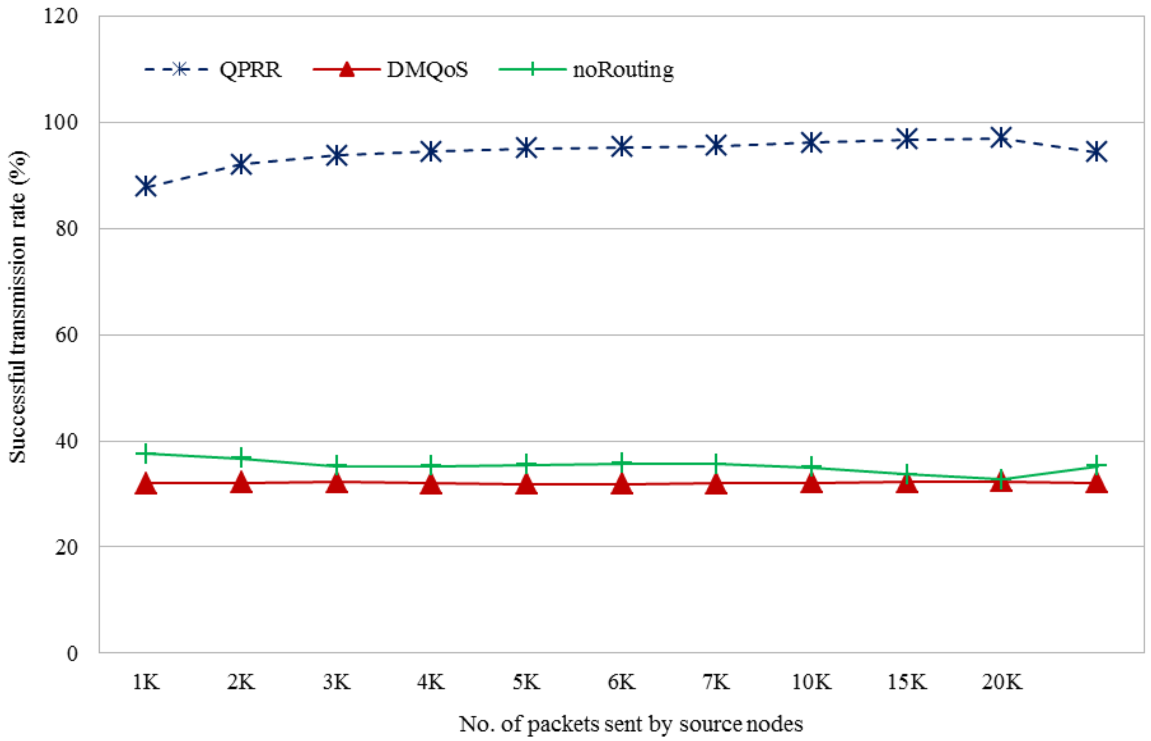


Figure 7-11: Throughput vs Offered load for Case 1

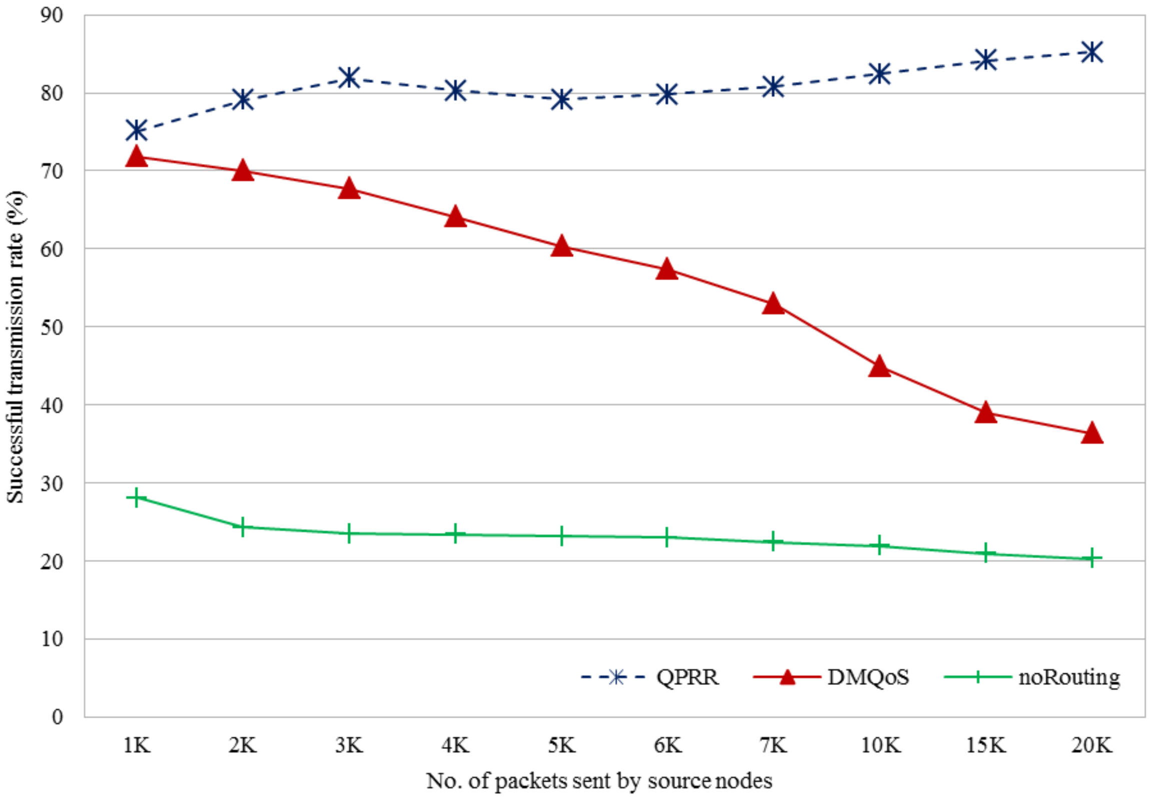


Figure 7-12: Throughput vs Offered load for Case 2

However, the link reliability of MDC_3 to its upstream neighbour nodes is much lower than the required data reliability. As a result of this MDC_3 drops the packets resulting in a low successful transmission rate. On the other hand, QPRR overcomes this issue by using the end-to-end path reliability and transmission of RSP data over redundant paths to ensure the requested reliability is met. The path selection mechanism of QPRR considers all nodes' reliabilities in the network and ensures that the requested end-to-end reliability is met, even if the data has to be transmitted over redundant paths.

From Figures 7-13, 7-14, and 7-15 it is seen that the reliabilities of QPRR are initially 74%, 63%, and 50% and then reach to 90%, 91%, and 86% with higher offered traffic loads for Case 3, 4, and 5 respectively. The same figures show that DMQoS can deliver only on average 6%, 10%, and 1% reliability-sensitive packets to the destinations for Case 3, 4, and 5 respectively. The performance of noRouting is slightly better than DMQoS but very poor when compared with QPRR. The successful transmission rates of noRouting protocol are 19%, 19%, and 15% for Case 3, 4, and 5 respectively.

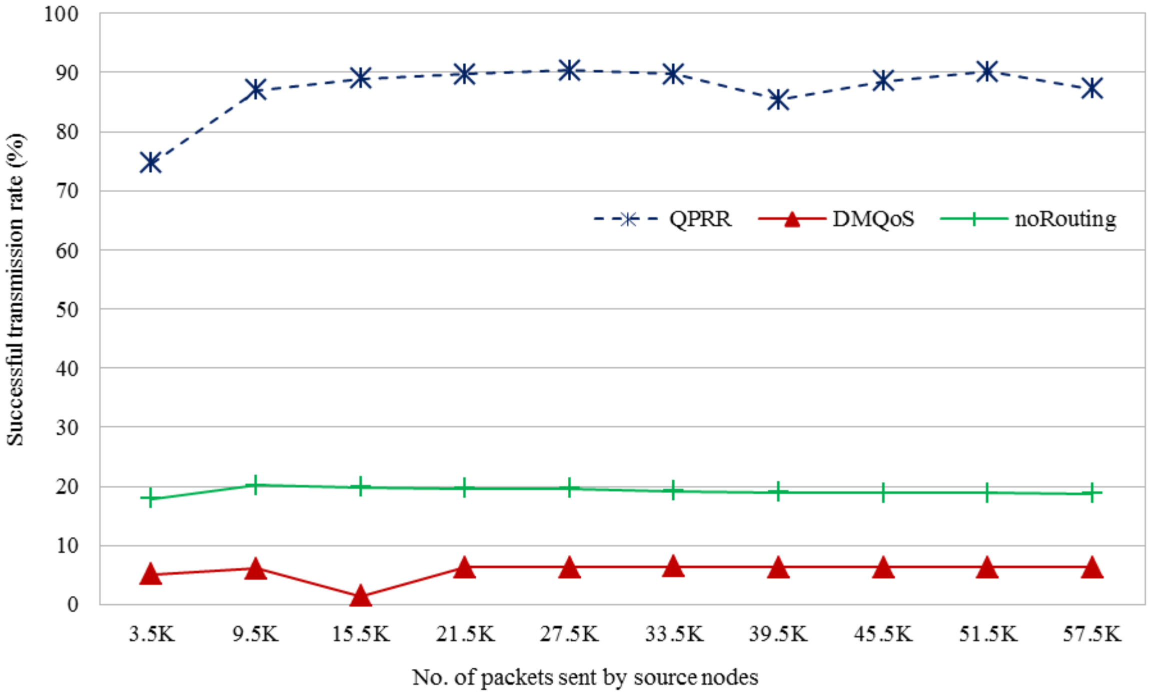


Figure 7-13: Throughput vs Offered load for Case 3

The two reasons for lower reliability of DMQoS are due to the channel variation and large number of Hello packets. The channel variation causes the link to go down sporadically. As all the nodes are sending and receiving Hello packets, the nodes in the network suffer from congestion due to the large number of these Hello packets. Either of the above reasons could decrease the reliability. In DMQoS, once the reliability is lower than the required reliability, it stops sending data packets which results in the lower overall reliability (as it stops sending data). Because of this reason, the successful transmission rate in DMQoS is lower for larger numbers of nodes in the network. On the other hand, QPRR uses redundant paths to enhance the reliability. The nodes send the packets along different paths. In case packets are dropped in a path, the data can reach via other redundant paths. As a result, QPRR provides better reliability.

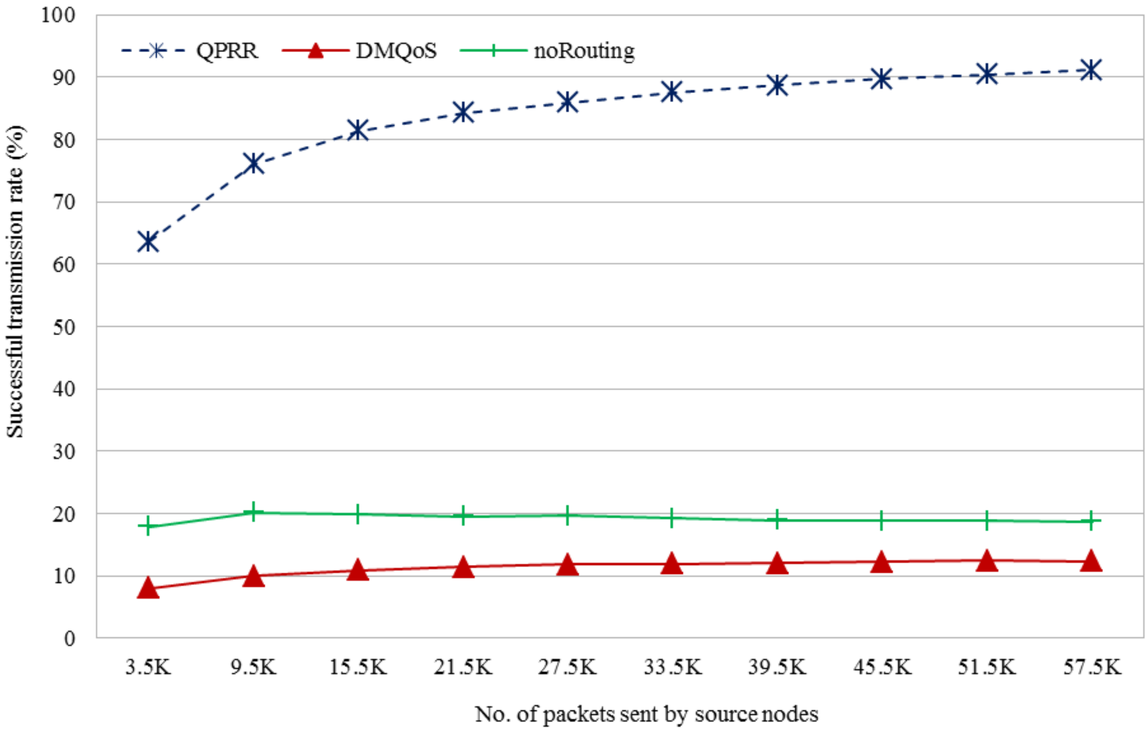


Figure 7-14: Throughput vs Offered load for Case 4

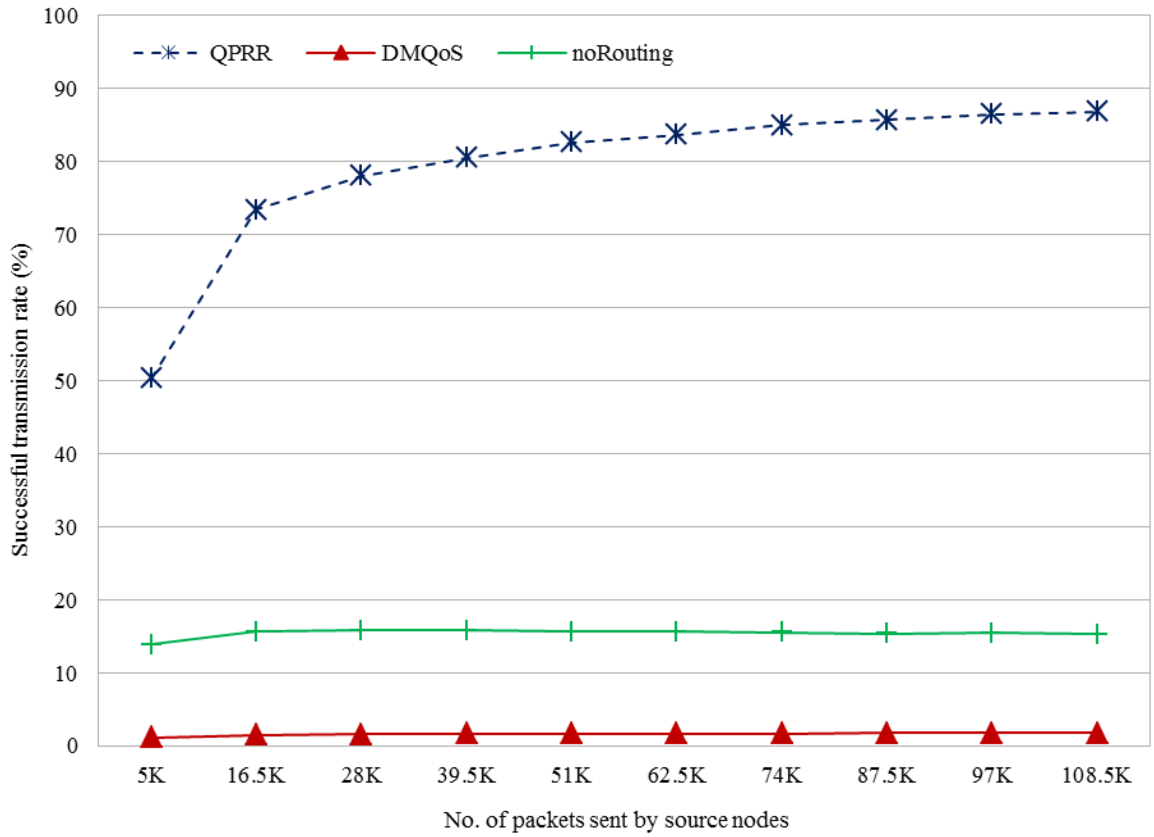


Figure 7-15: Throughput vs Offered load for Case 5

7.4.7.2. Reliability Packets Dropped

The reliability packets dropped for all five cases are discussed in this section. DMQoS drops most of the reliability packets due to the traffic congestion in all the cases. The source nodes in DMQoS calculate the hop-by-hop reliability of the next hop nodes and send the data to the best next hop which has highest reliability. The next hop then calculates the reliabilities of its upstream nodes. The pakcets are dropped in case of not having the required reliability by all neighboring upstream nodes. QPRR resolves this problem by using the end-to-end path reliabilities. Also the use of three redundant paths in QPRR ensures maximum transmission rates. QPRR does not drop any reliability-sensitive packet for Cases 1 and 2; whereas, it drops very small number of reliability packets as compared to DMQoS for Cases 3, 4, and 5. The noRouting simply forwards the packets to the random neighbor nodes without any kind of calculation. The reliability packets dropped in noRouting are all almost negligible for all cases; however, noRouting doesn't ensure the delivery of the packets to the destination nodes which results in low transmission rate as discussed previously in Section 7.4.7.1.

Figure 7-16 shows that the packets dropped by QPRR and noRouting protocols are zero in comparison to DMQoS which drops 0.5K to 13K packets when 1K to 20K packets are sent by source nodes.

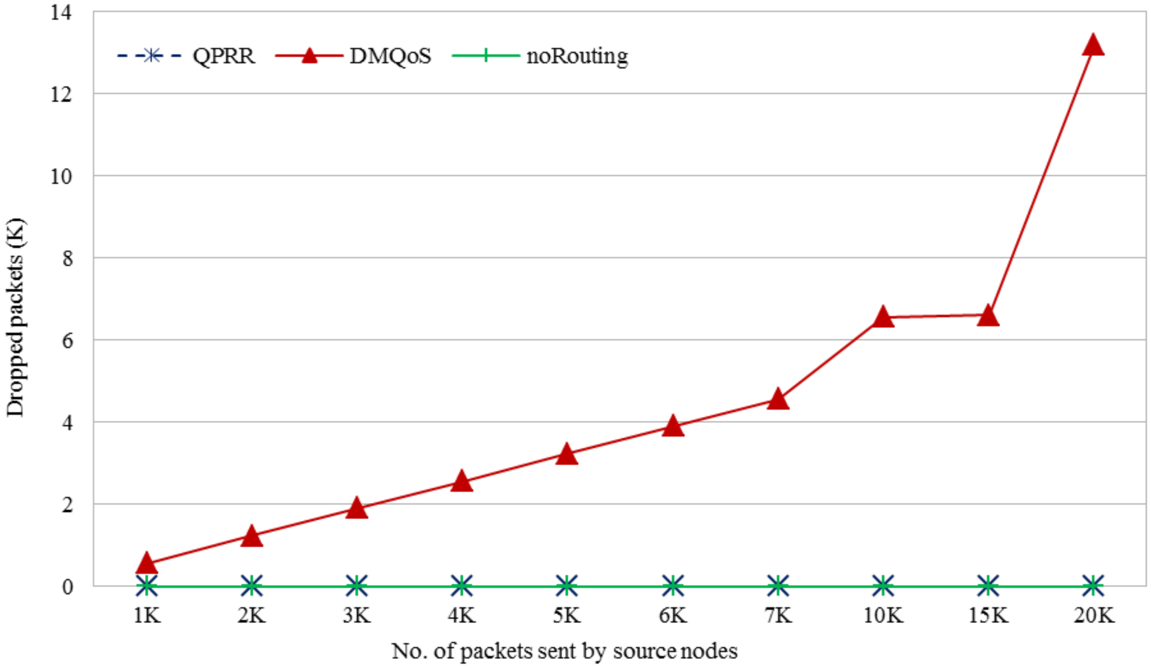


Figure 7-16: Reliability packets dropped for Case 1

In Case 2, noRouting performs better than other protocols by not dropping any packets as shown in Figure 7-17. The performance of QPRR is also good. Initially, until 3K RSP packets sent by source nodes, QPRR doesn't drop any reliability packet. After 3K, QPRR starts dropping the packets and reaches 0.16K packets when 20K packets are sent. Due to the traffic congestion, DMQoS drops 0.13K to 11.5K reliability packets when 1K to 20K packets are sent by source nodes. The protocol noRouting does not drop any packet because it forwards all the traffic to the random neighbor nodes without considering the reliability requirements. The reliability packets dropped in noRouting are all almost negligible for all cases; however, noRouting doesn't ensure the delivery of the packets to the destination nodes which results in low transmission rate as discussed previously in Section 7.4.7.1.

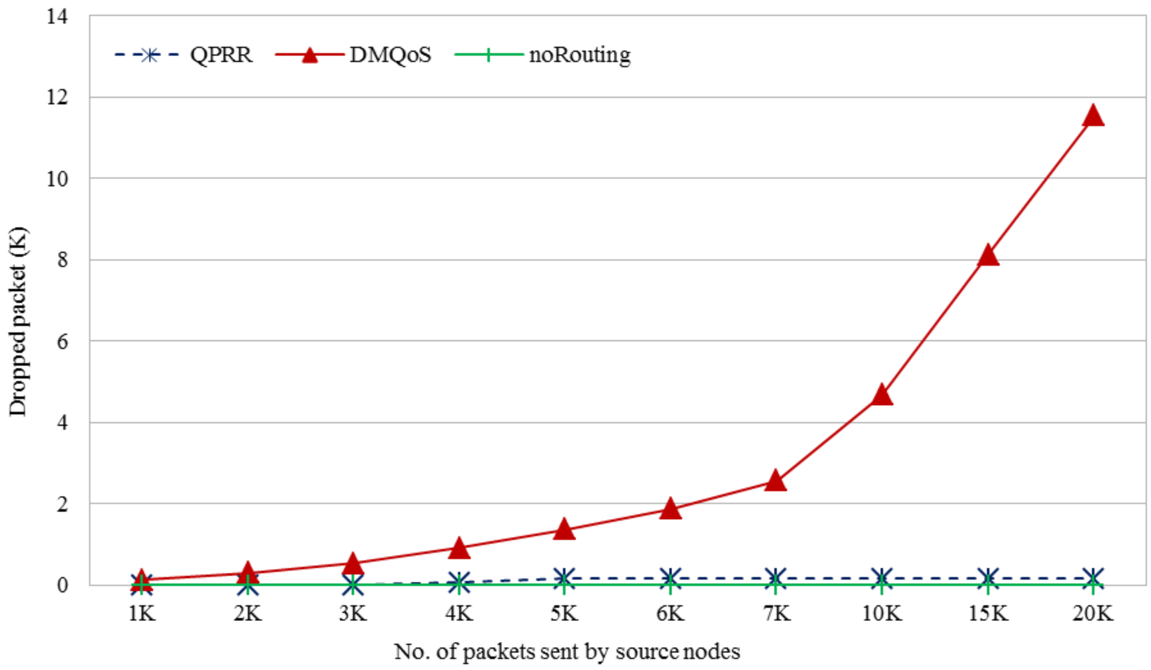


Figure 7-17: Reliability packets dropped for Case 2

The reliability packets dropped in QPRR are upto 10K and 18K for Case 3 and 4 respectively, when 57.5K packets are sent by source nodes as shown in Figure 7-18 and Figure 7-19. QPRR drops on average 5.6 times and 3.4 times less packets than DMQoS in Case 3 and 4, respectively.

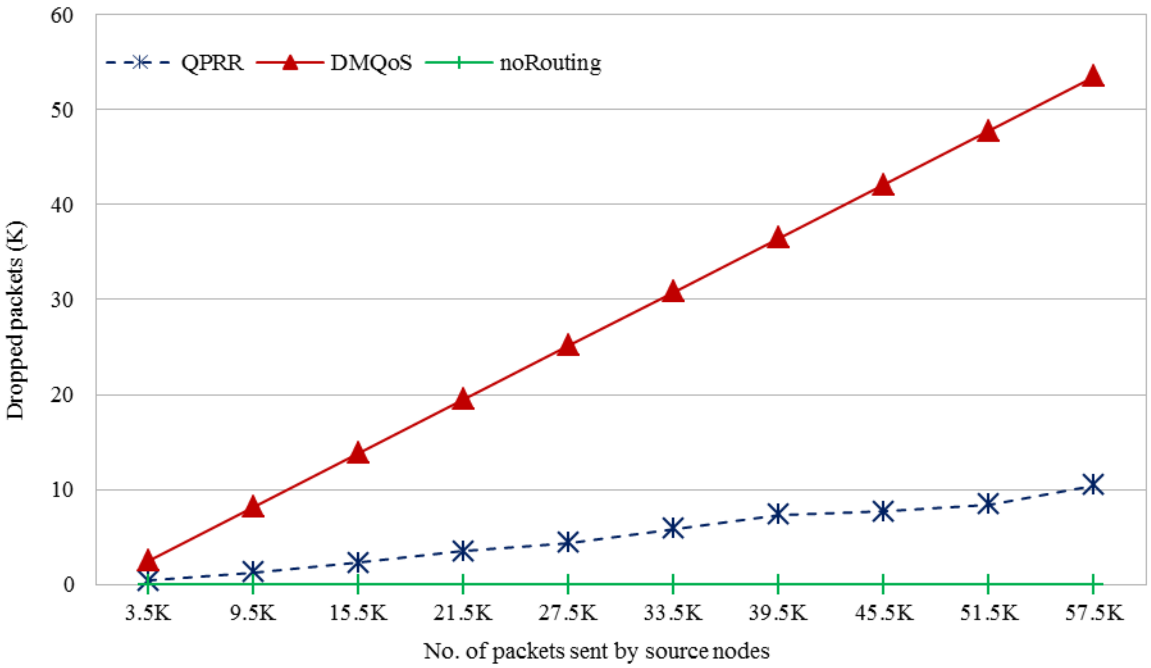


Figure 7-18: Reliability packets dropped for Case 3

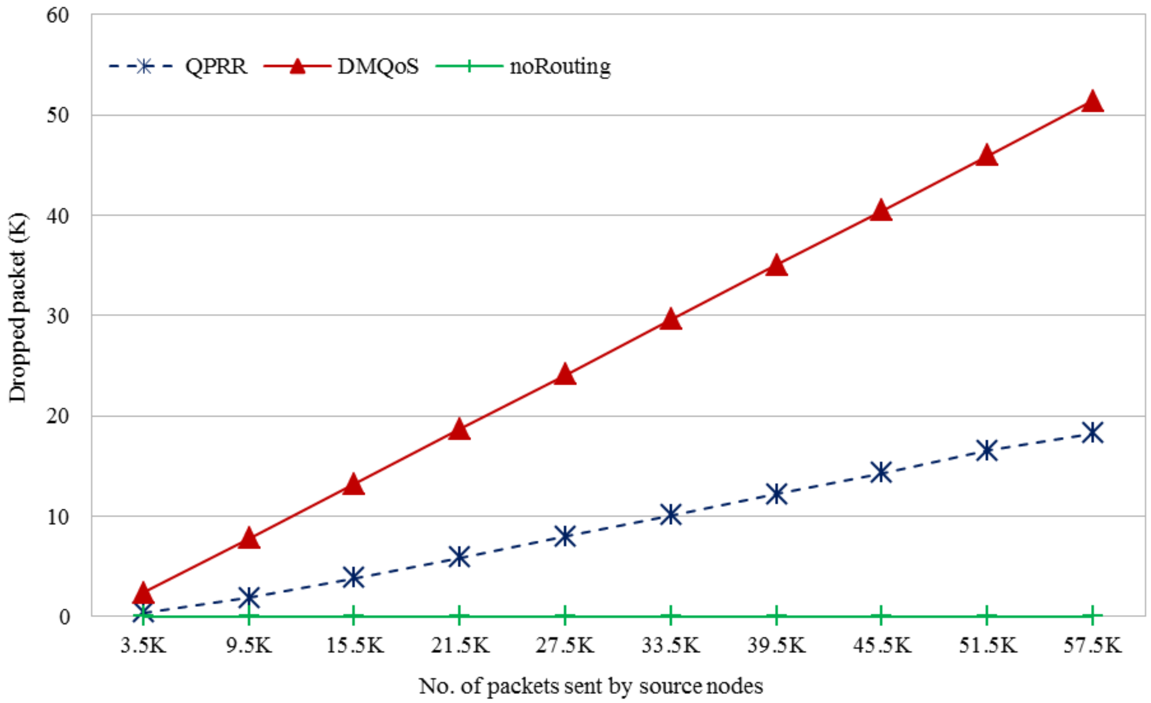


Figure 7-19: Reliability packets dropped for Case 4

Figure 7-20 shows the performance of QPRR is about 3 times better than DMQoS in Case 5.

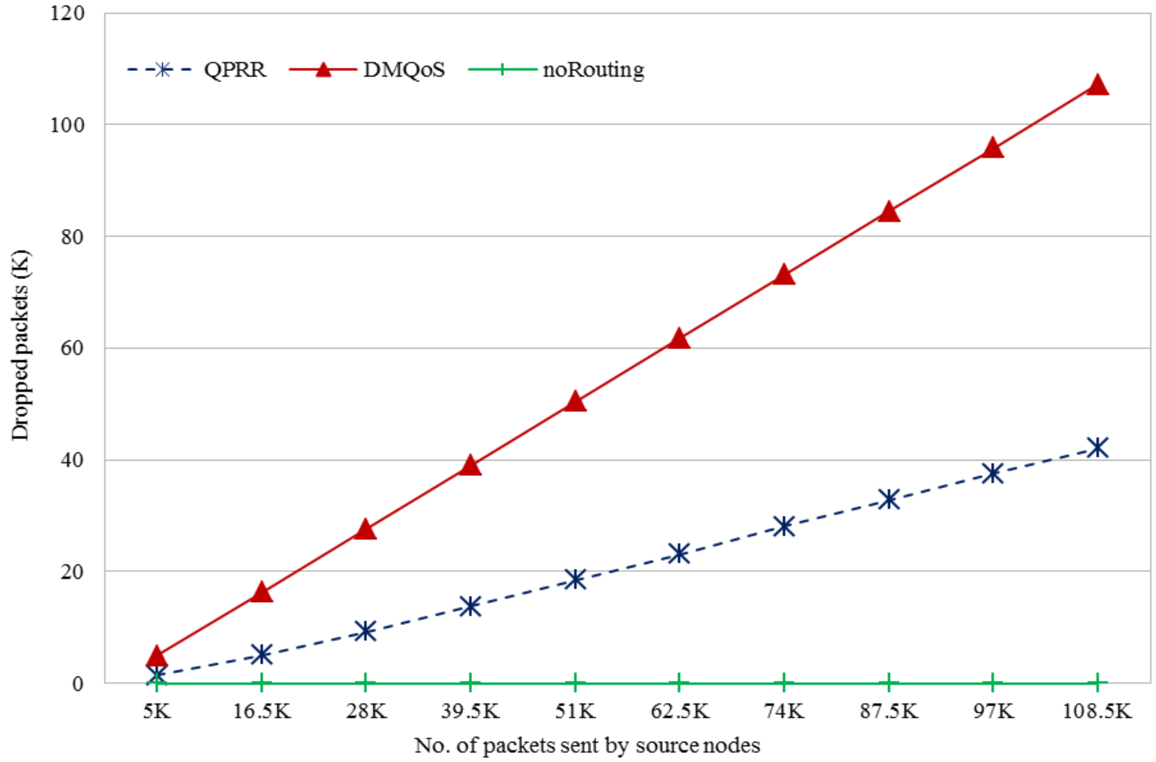


Figure 7-20: Reliability packets dropped for Case 5

A general observation regarding the performance of the noRouting protocol is that the reliability packets dropped in noRouting are all almost negligible for all cases; however, noRouting doesn't ensure the delivery of the packets to the destination nodes which results in low transmission rate as discussed previously in Section 7.4.7.1.

7.4.7.3. MAC Buffer Overflow

The MAC buffer overflows as a function of offered traffic load is shown in Figures 7-21 to 7-25. Figure 7-21 shows that, for Case 1, only 3 packets are dropped due to MAC buffer overflow in QPRR as compared to an average of 27 and 65 packets dropped in DMQoS and noRouting respectively. QPRR performs well initially but then after 4K packets sent by source nodes, it drops more packets than DMQoS and noRouting protocols in Case 2.

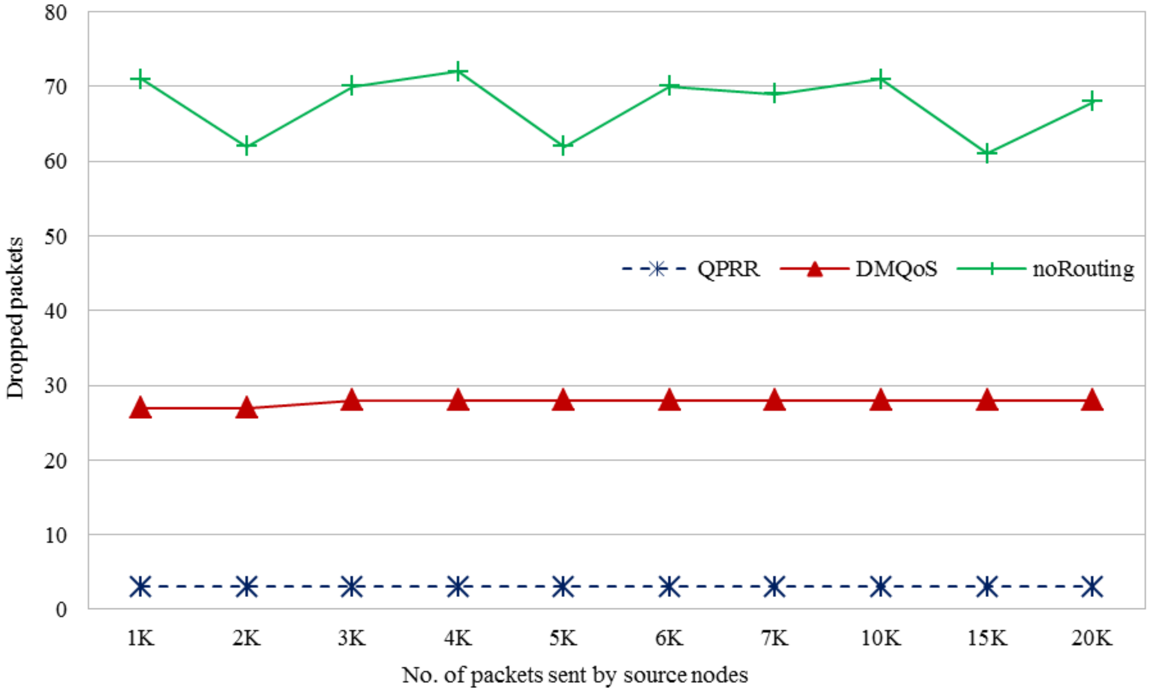


Figure 7-21: MAC buffer overflow for Case 1

The mobility of source node in Case 2 causes more packet drops in QPRR but this higher packet loss does not affect the higher transmission rate in QPRR than other similar protocols, as shown in Figure 7-22. The packets dropped due to the MAC buffer overflow in QPRR is zero until 3K packets sent by source nodes but the number of packets dropped increases and reaches to 585 when 6K packets are sent by source nodes. DMQoS and noRouting drop 22 and 60 packets, respectively. However, the DMQoS and noRouting protocol in particular doesn’t ensure the delivery of the packets to the destination nodes which results in low transmission rate as discussed previously in Section 7.4.7.1.

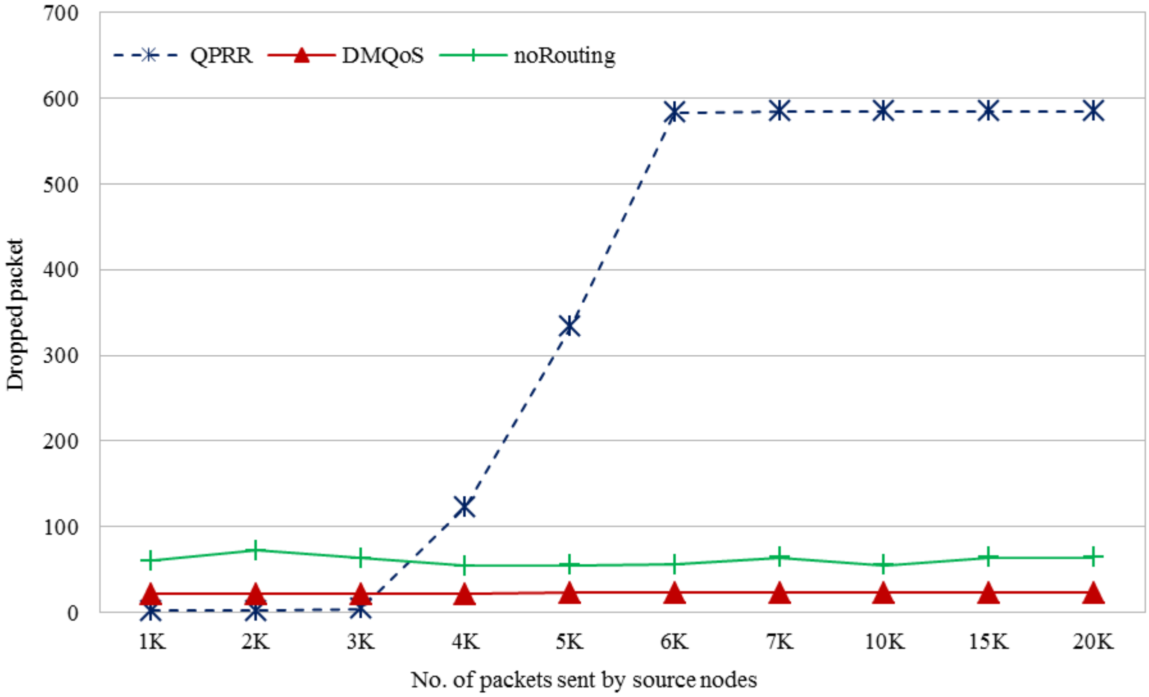


Figure 7-22: MAC buffer overflow for Case 2

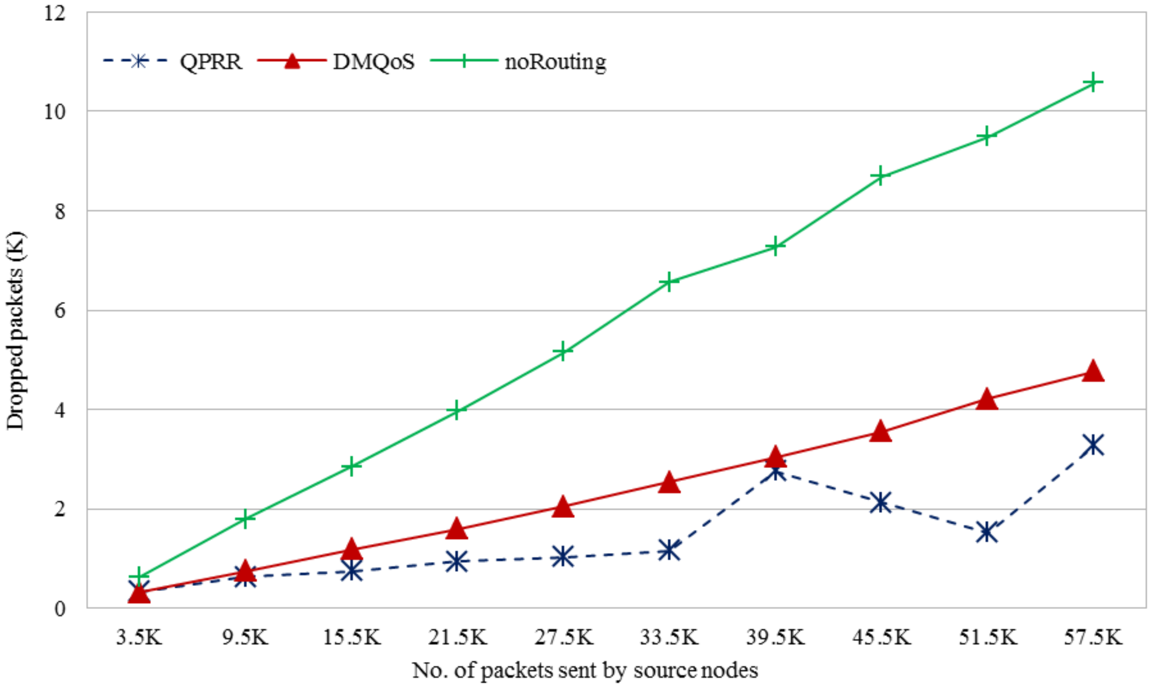


Figure 7-23: MAC buffer overflow for Case 3

The packets dropped due to the MAC buffer overflow for Case 3 are on average 1.7 and 4 times lower in QPRR than DMQoS and noRouting respectively as shown in Figure 7-23.

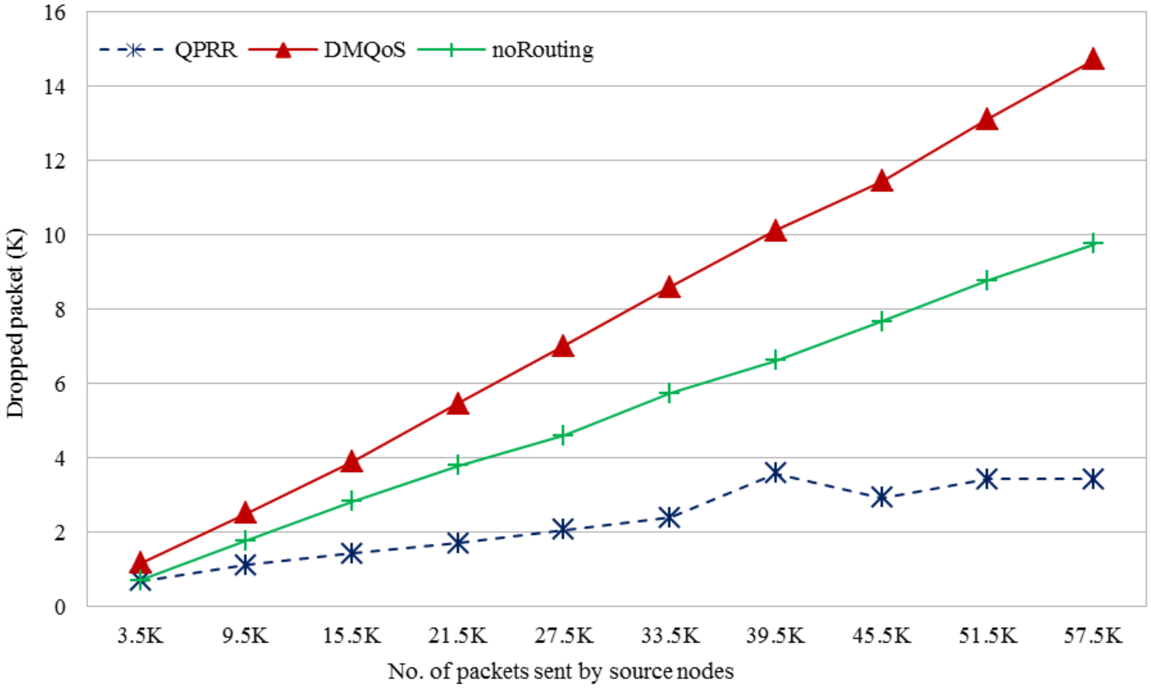


Figure 7-24: MAC buffer overflow for Case 4

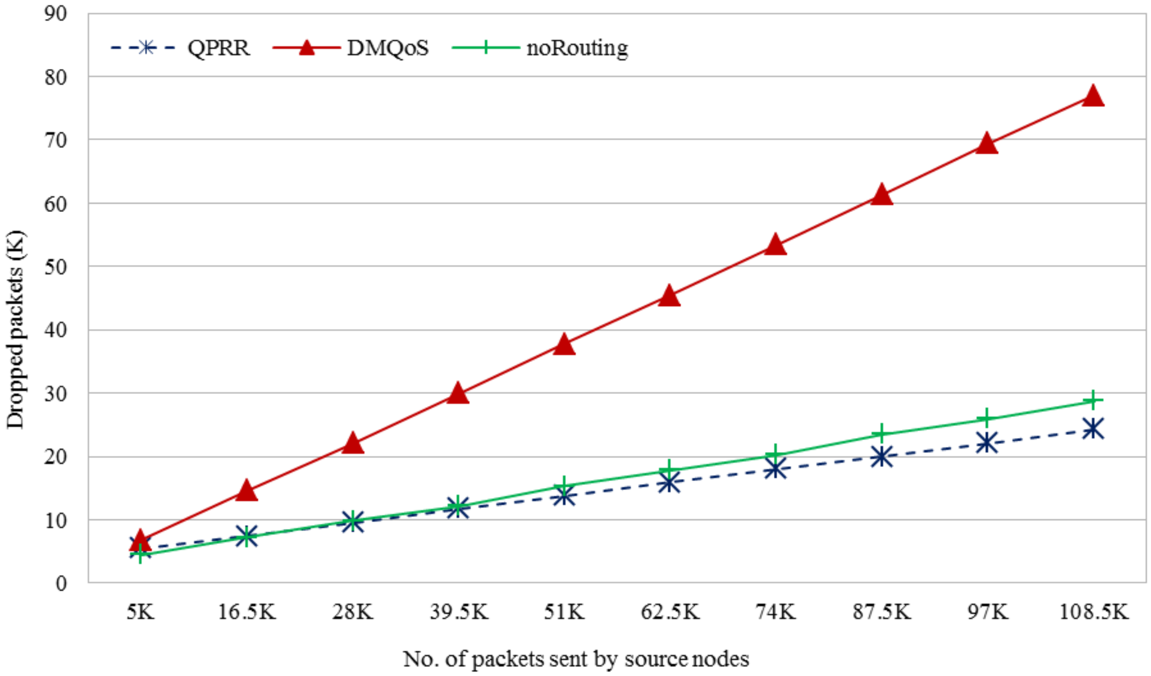


Figure 7-25: MAC buffer overflow for Case 5

From Figure 7-24 and Figure 7-25, it is observed that DMQoS and noRouting drops more packets due to traffic congestion when more number of nodes are used in the network;

whereas, the path reliability calculation mechanism used in QPRR helps to send the data through a path or over several redundant paths in order to ensure much higher rates of data delivery to the destination.

7.4.7.4. Overall Network Traffic

The QPRR ensures better reliable data delivery, but this superior performance comes with a corresponding increase in the overall network traffic. The increase in the network traffic at higher offered traffic loads can be ascribed to the use of multiple redundant data paths for the reliable transfer of data. In the first three cases, it is seen from Figures 7-26, 7-27, and 7-28 that the overall network traffic of QPRR as a function of the offered traffic load (x-axis) is higher when compared with DMQoS. The reason for lower overall network traffic in DMQoS is the higher number of packets dropped due to the MAC buffer overflow and traffic congestion. However, this results in lower successful data transmission rates in DMQoS.

Figure 7-26 shows the overall network traffic in QPRR is on average 5.5 times and 2 times more than that of DMQoS and noRouting protocols respectively.

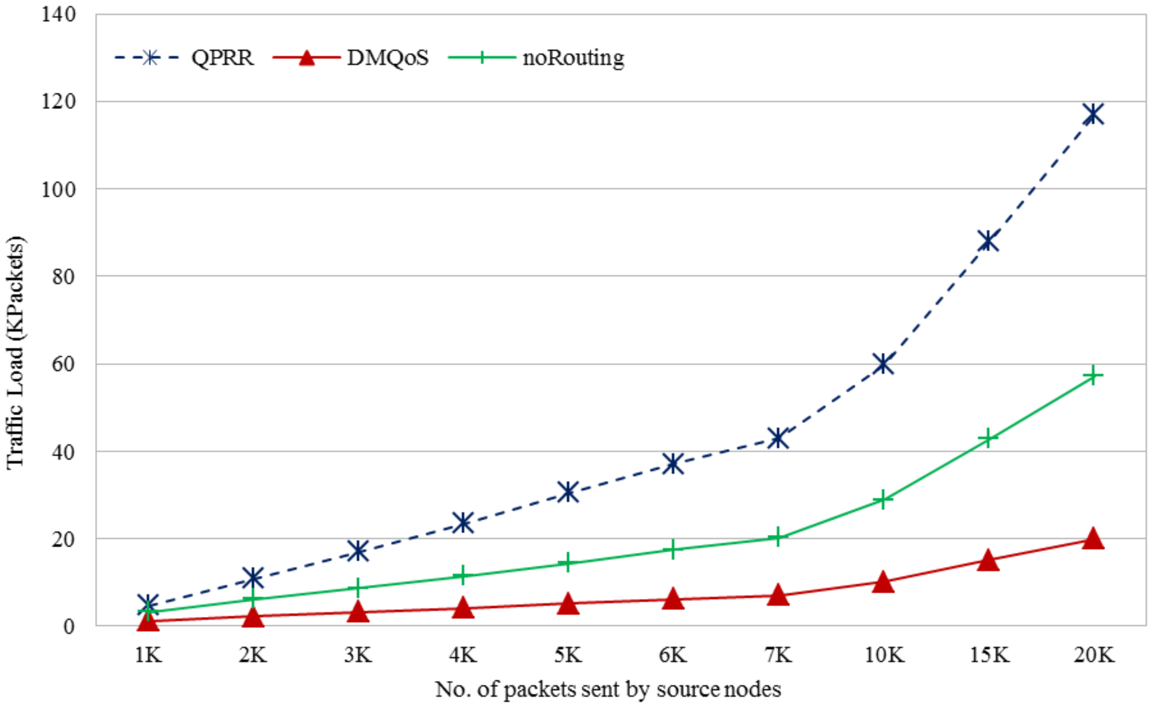


Figure 7-26: Overall network traffic for Case 1

In the case of mobile nodes, Case 2, the overall network traffic in QPRR is on average 3.5 times and 3.2 times more than the overall network traffic in DMQoS and noRouting protocols respectively.

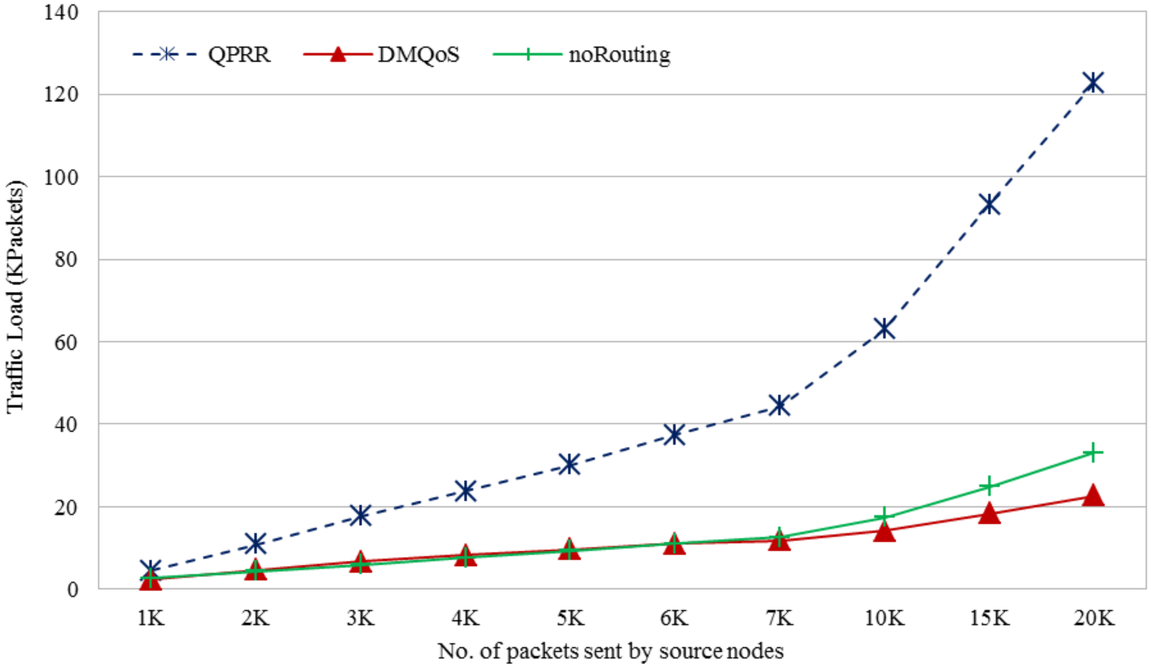


Figure 7-27: Overall network traffic for Case 2

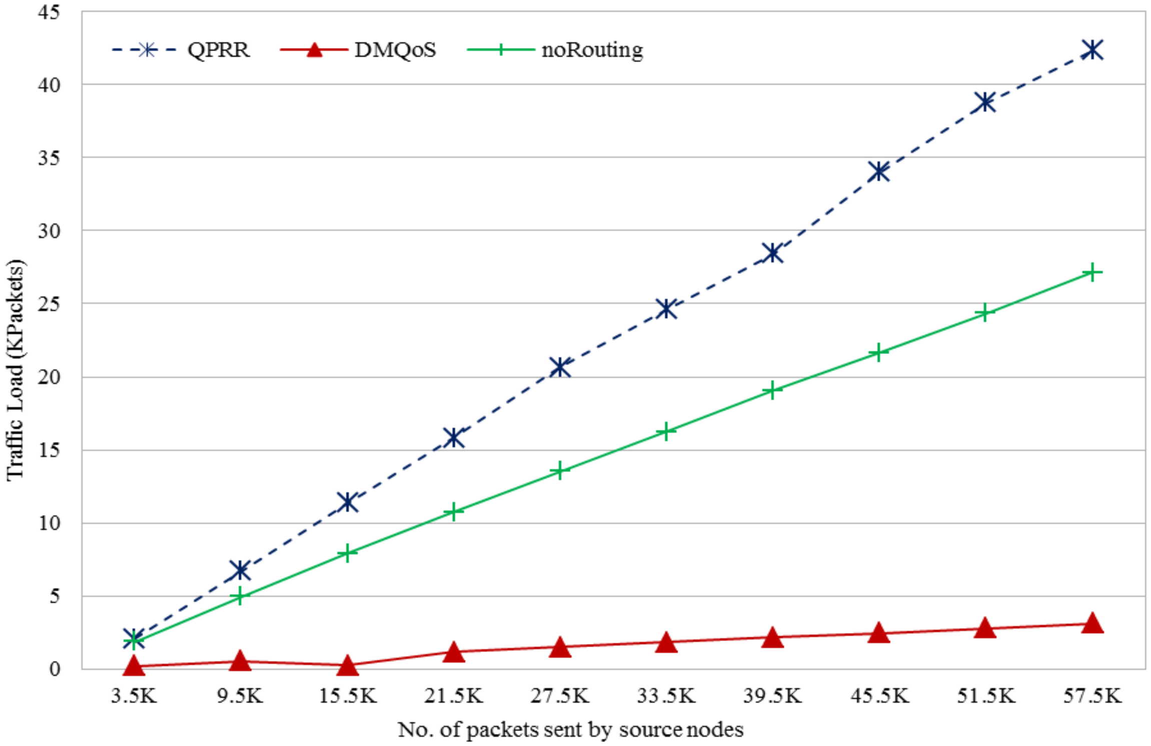


Figure 7-28: Overall network traffic for Case 3

In Case 3, when 49 nodes with NSC in the centre are used, Figure 7-28 shows that QPRR has on average 16.5 times and 1.5 times higher overall network traffic than DMQoS and noRouting respectively.

The overall network traffic for both QPRR and the noRouting is almost the same in Case 4 as shown in Figure 7-29. As discussed in Section 7.4.7.1, QPRR provides about 80% higher successful data transmission rates then DMQoS for Case 4. This higher data transmission rate in QPRR results in 7 times (on average) more overall network traffic than DMQoS as shown in Figure 7-29.

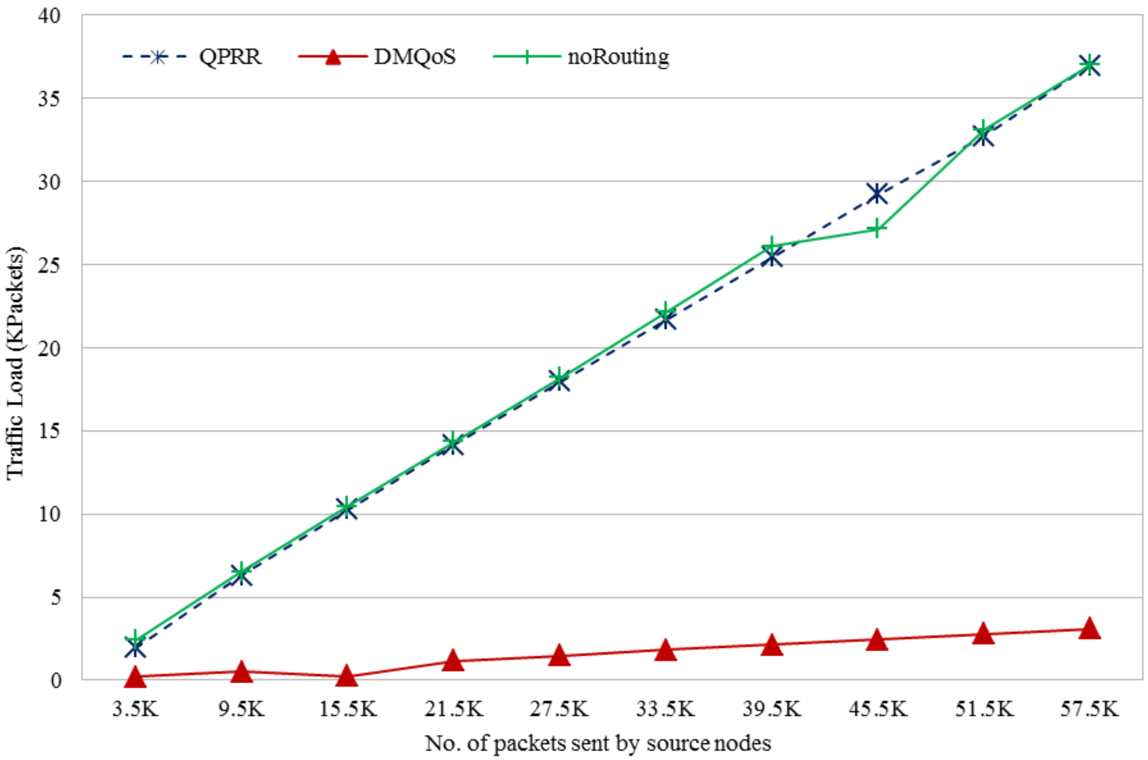


Figure 7-29: Overall network traffic for Case 4

In the Case 5, real hospital scenario with 93 nodes, QPRR provides 80% successful transmission rate in comparison of 1.6% and 15.4% successful transmission rates of DMQoS and noRouting protocols respectively, as discussed in Section 7.4.7.1. The end-to-end path reliabilities with redundant paths used in QPRR increases the throughput; however, it also results in 44.8 times more overall network traffic than DMQoS as shown in Figure 7-30. It is also seen that QPRR provides on average 17% less overall network traffic than noRouting.

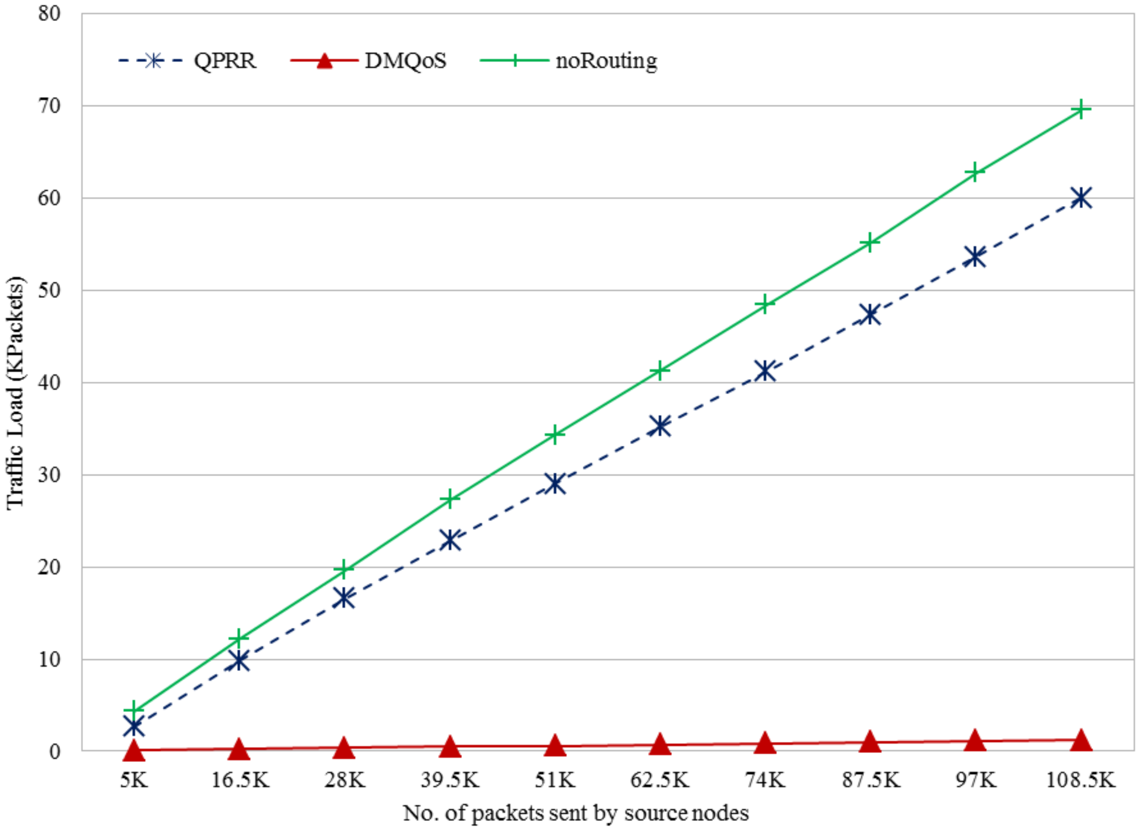


Figure 7-30: Overall network traffic for Case 5

7.4.7.5. Hello Packets

The broadcast of Hello packets are important in updating the routing tables of the nodes in the BAN network. The drawback to employing Hello packets is that they result in an increase in overall network traffic. QPRR provides a mechanism to reduce the number of Hello packets by the method employed to defines who broadcasts and when to broadcast the Hello packets. In this section the numbers of Hello packets generated by nodes in all five cases is discussed. Figure 7-31 show the total numbers of Hello packets in QPRR are on average 23% and 6% fewer than DMQoS and noRouting protocols respectively, for the stationary Case 1.

In case of mobile source node, Case 2, QPRR performs better then Case 1. The numbers of Hello packets in QPRR are on average 22% and 17% less than DMQoS and noRouting respectively as shown in Figure 7-32.

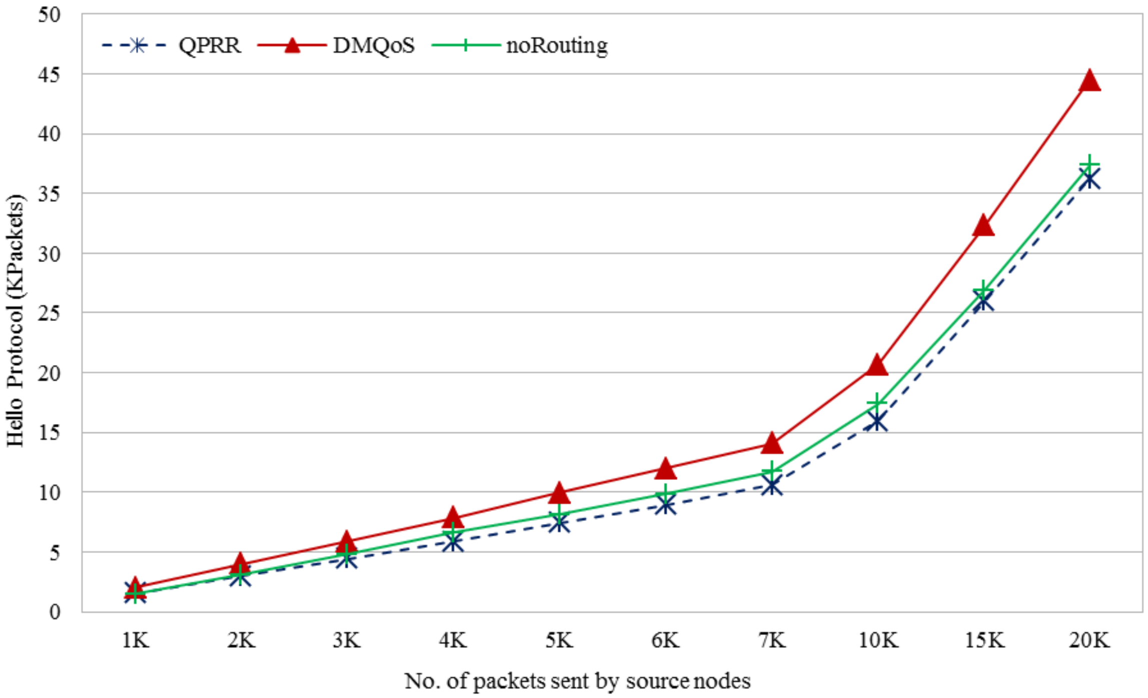


Figure 7-31: Hello protocol packets for Case 1

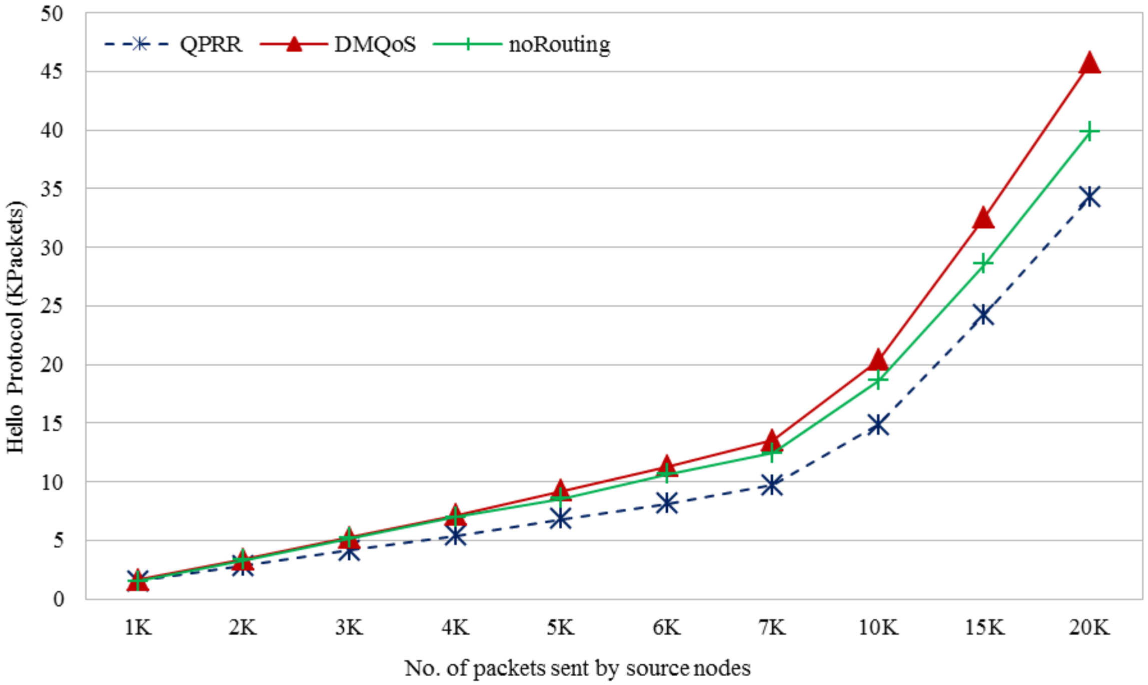


Figure 7-32: Hello protocol packets for Case 2

In the real hospital scenarios with higher number of nodes, Cases 3 and 4, the total numbers of Hello packets in QPRR are on average 43% less than DMQoS but on average 50% more than noRouting as shown in Figure 7-33 and Figure 7-34. This is due to the absence of Hello protocol updates in the noRouting protocol.

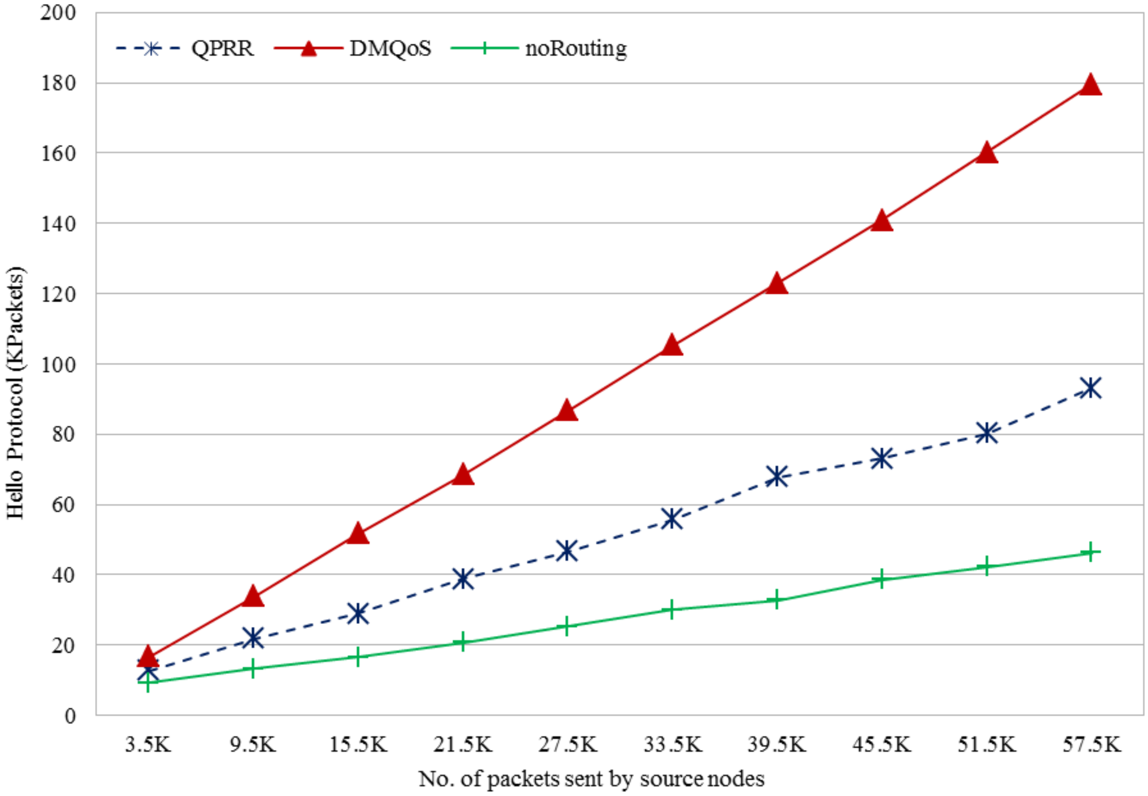


Figure 7-33: Hello protocol packets for Case 3

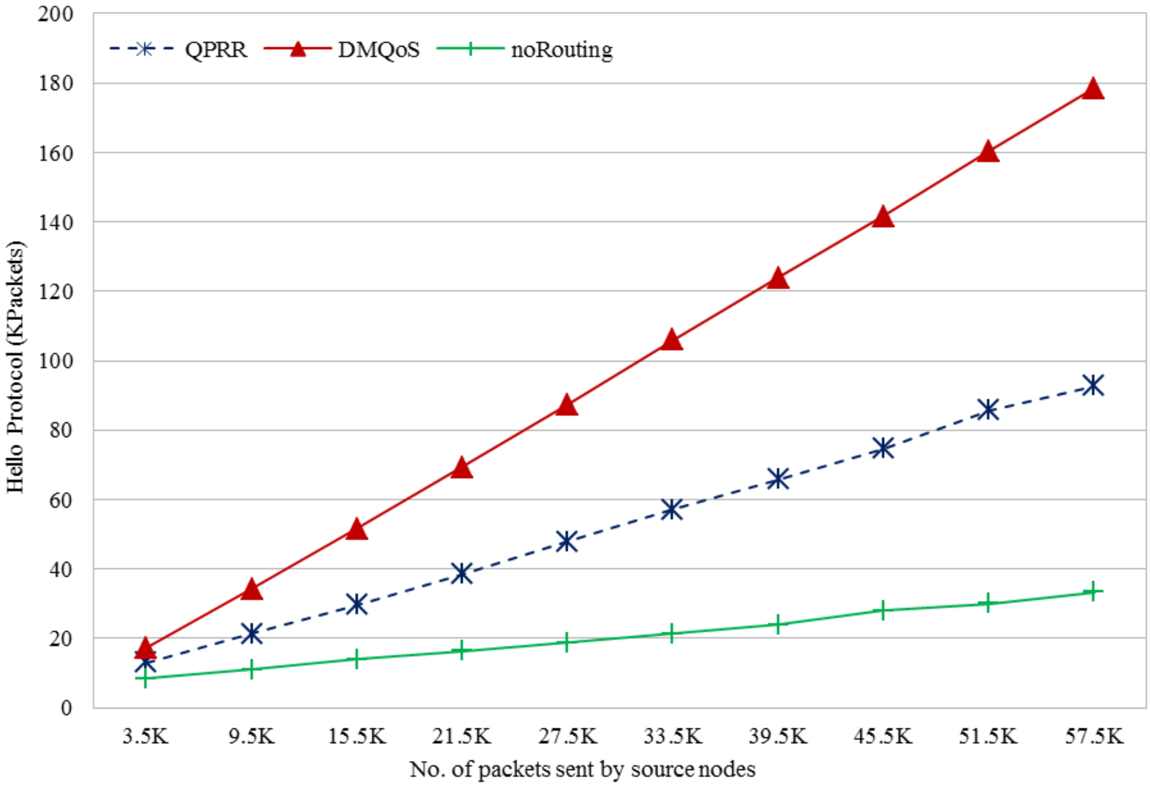


Figure 7-34: Hello protocol packets for Case 4

Case 5 discusses the real hospital scenario with 93 nodes. The mechanism used by QPRR to broadcast the Hello packets, construct and update the routing tables provides the best paths which helps improve the throughput by 80% as compared to 1.6% and 15% for the DMQoS and noRouting protocols respectively. Figure 7-35 shows that the numbers of Hello packets in QPRR are 35% fewer than DMQoS and 2.7 times more than the noRouting protocol.

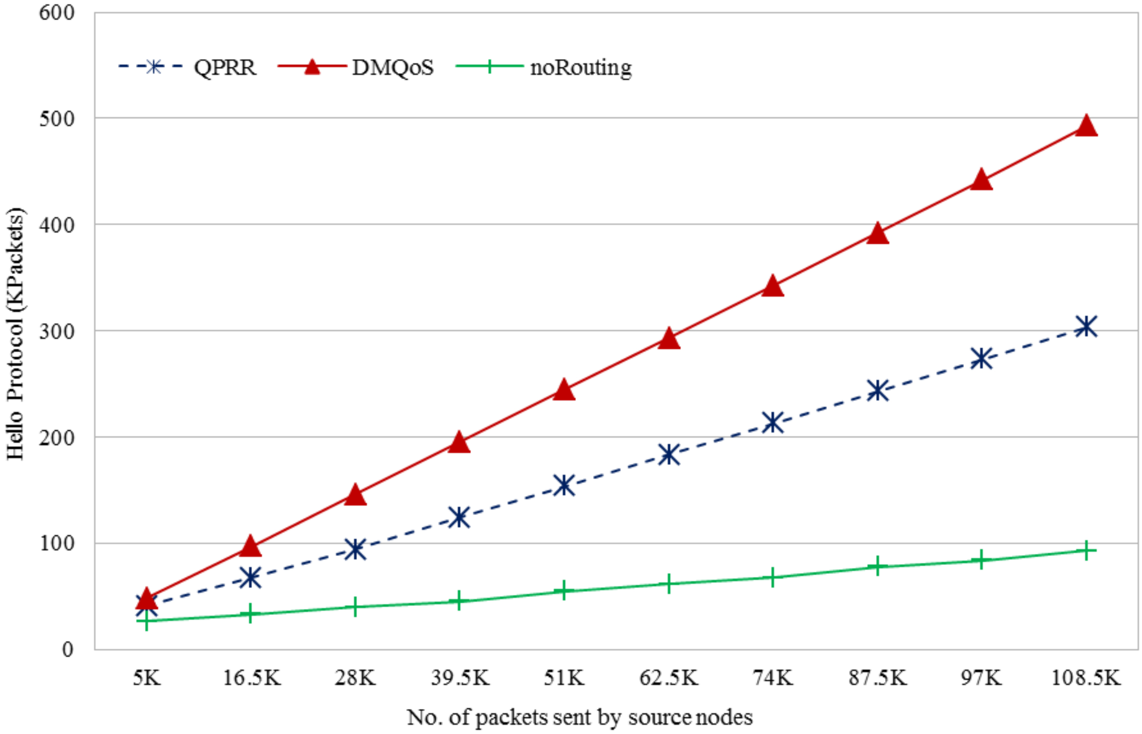


Figure 7-35: Hello protocol packets for Case 5

7.4.7.6. Overall Energy Consumption

This section discusses the overall energy consumption in all the five cases for QPRR, DMQoS, and noRouting. It shows that QPRR provides a consistent and more reliable delivery of critical packets as previously discussed in Section 7.4.7.1 while consuming the same energy as the DMQoS and noRouting protocols.

Figure 7-36 shows that QPRR consumes 9.004 to 180.461 Joules when the offered load is 1K to 20K reliability packets as sent by source nodes. For the same offered load, DMQoS needs 8.992 to 178.212 Joules; whereas, noRouting consumes 9.049 to 180.874 Joules.

In Case 2, for the mobile source node case, QPRR consumes on average 1.11% and 0.54% more energy than DMQoS and noRouting protocols respectively as shown in Figure 7-37.

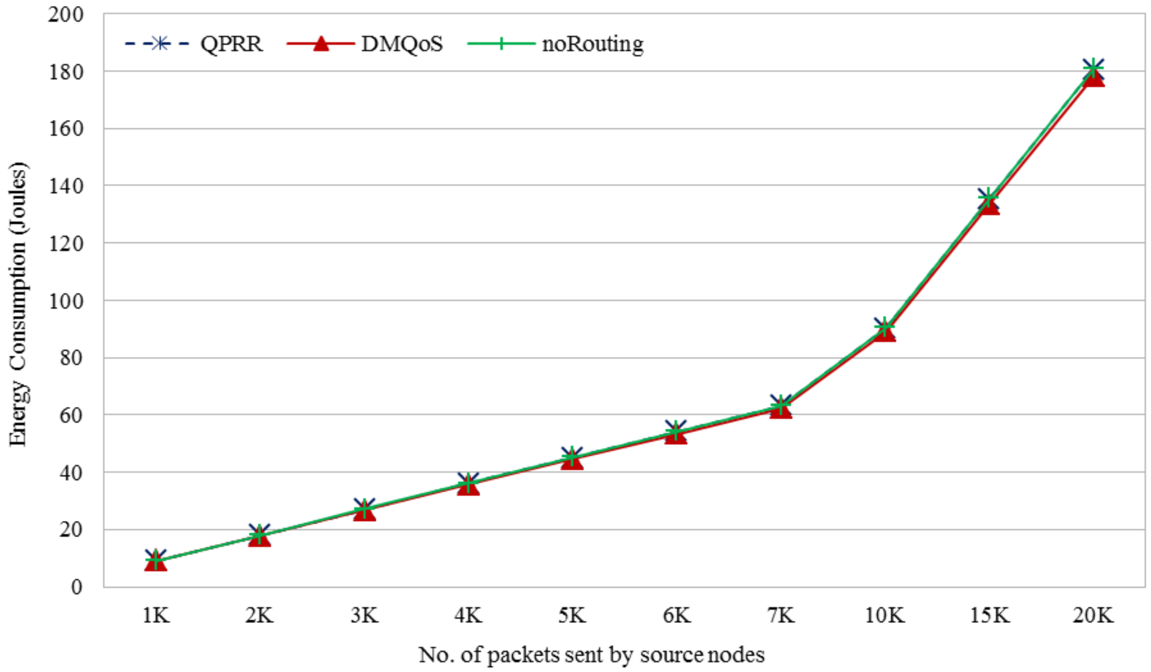


Figure 7-36: Overall energy consumption for Case 1

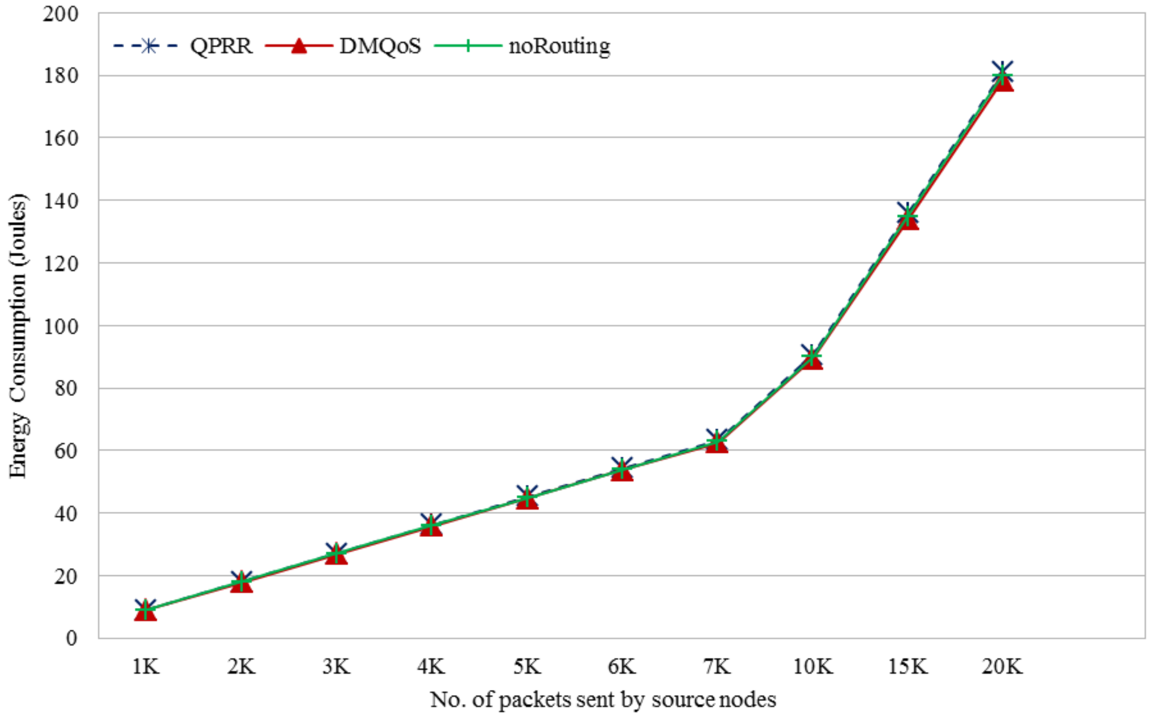


Figure 7-37: Overall energy consumption for Case 1

In Case 3, with 24 hospital bed with NSC placed in the centre, the overall energy consumption by all the three protocols is almost the same, as shown in Figure 7-38. QPRR consumes on average 0.46% more and 0.66% less energy than DMQoS and noRouting protocols respectively.

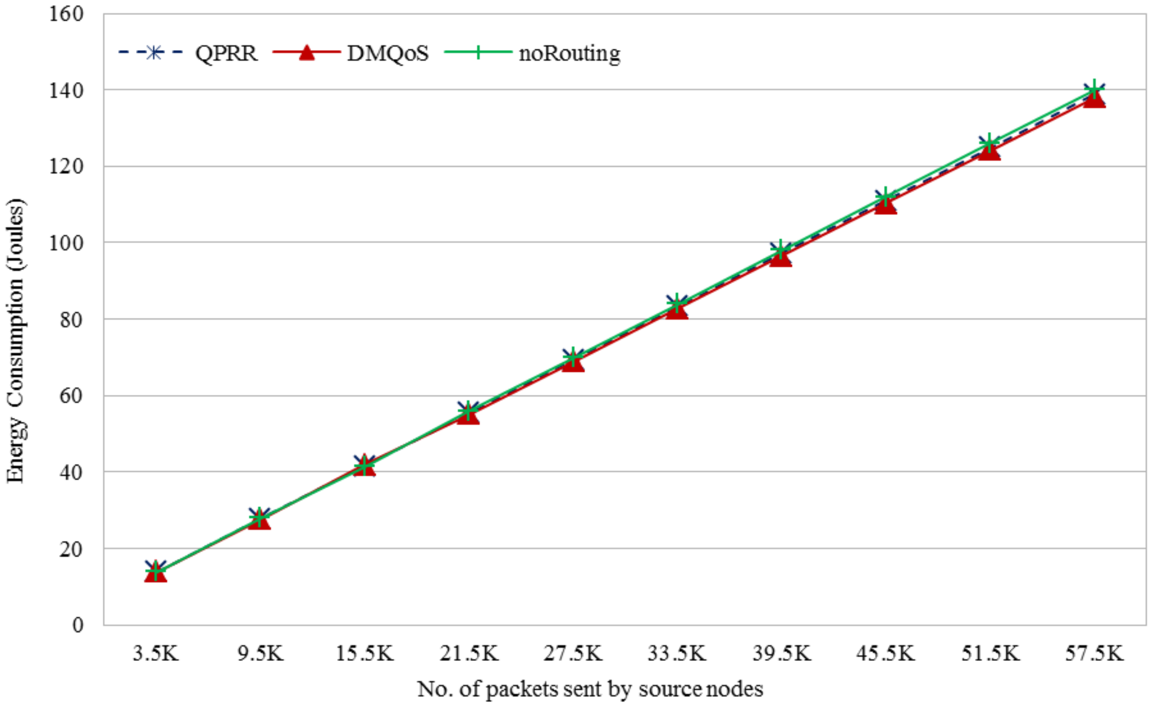


Figure 7-38: Overall energy consumption for Case 3

In case of a real hospital scenario with 49 nodes, QPRR consumes 0.46% more and 1.24% less energy than DMQoS and noRouting protocols respectively as shown in Figure 7-39.

Figure 7-40 shows the overall energy consumption by all three routing protocols in Case 5. Case 5 considers a real hospital with 93 nodes. It is observed from the Figure 7-40 that the energy consumption in QPRR is 0.37% more and 1.1% less than the DMQoS and noRouting protocols respectively.

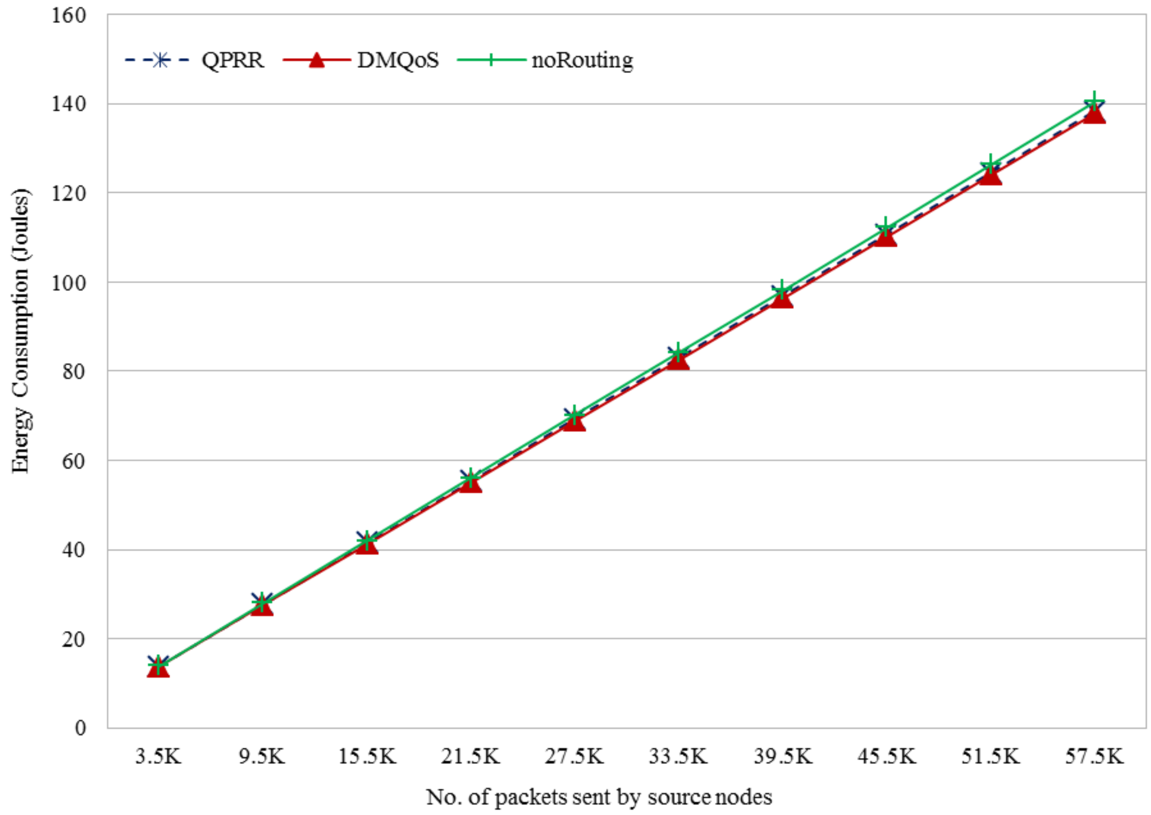


Figure 7-39: Overall energy consumption for Case 4

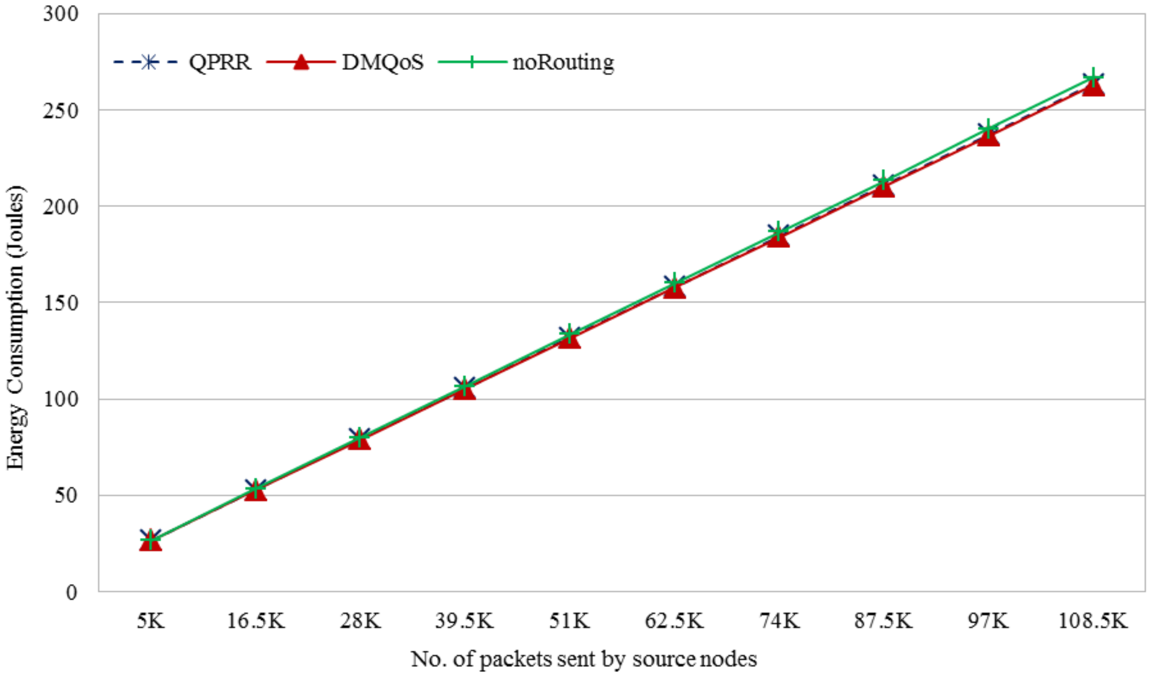


Figure 7-40: Overall energy consumption for Case 5

7.4.7.7. Latency

The application level latency is measured after every 20 ms for all the five cases. The majority of the data packets reach the destination nodes with excess delay when the number of nodes increases in a real wireless sensor networks application [81]. This is due to the fact that packets need to pass through more nodes before reaching the destination. From Figures 7-41 to 7-45 it is seen that more reliability-sensitive packets are delivered by QPRR than DMQoS and noRouting for any given delay intervals.

Figures 7-41 and 7-42 show that QPRR consistently delivers on average 2.5 times and 4 times more packets to the destinations as compared to DMQoS for all time intervals in Case 1 and 2 respectively. The end-to-end delay in noRouting protocol is much higher due to the random transmission of packets to the next hop. The number of packets delivered in any time interval is not more than 0.1K packets.

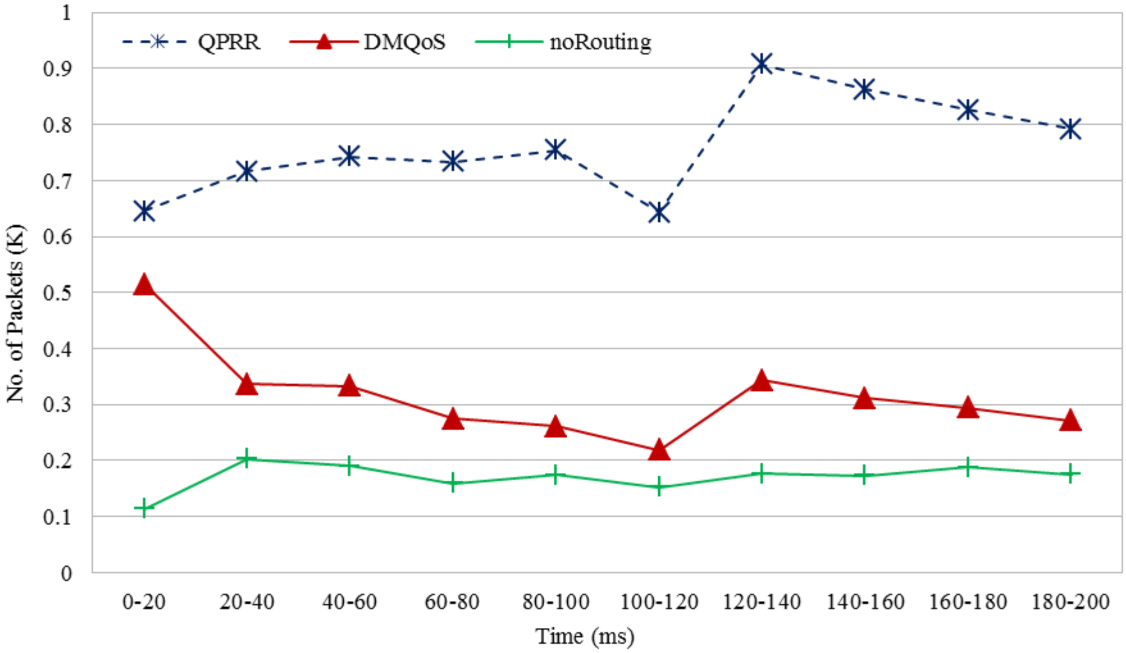


Figure 7-41: Latency for 20K RSPs for Case 1

In Cases 3 and 4, QPRR outperforms DMQoS and noRouting in terms of number of packets delivered reliably to the destinations. QPRR delivers 4K and 3.2K packets when DMQoS delivers 0.3K and 0.6K packets during the first 0-20ms interval. After the first interval, QPRR provides on average 13 times and 5 times more packets are delivered than

DMQoS and noRouting protocols as shown in Figure 7-43 and Figure 7-44.

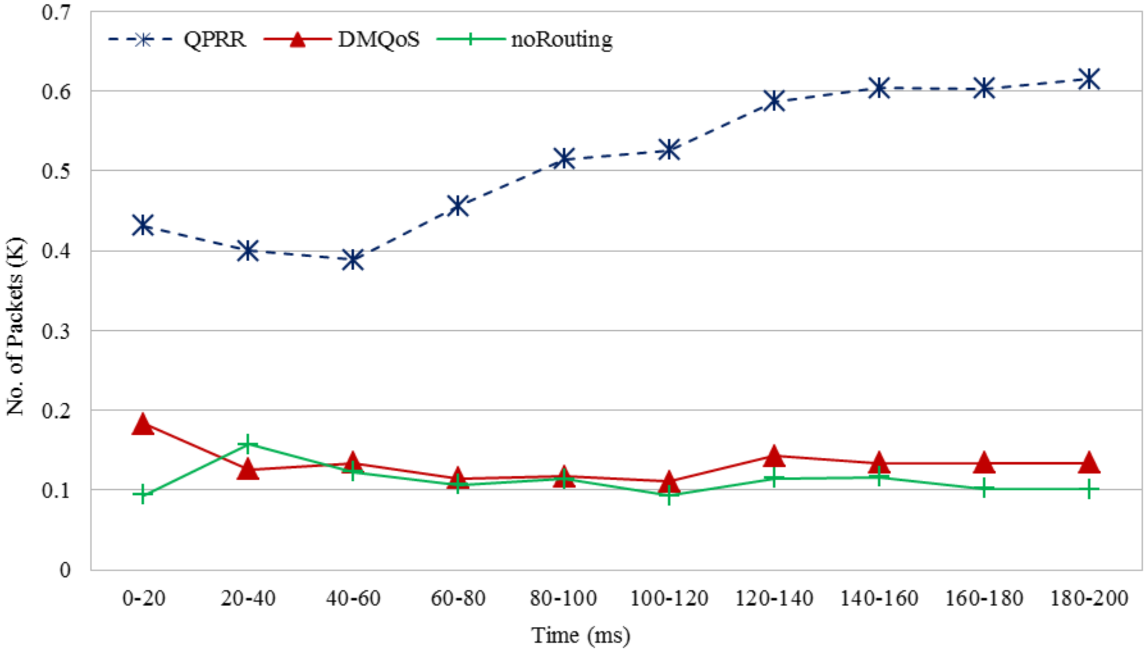


Figure 7-42: Latency for 20K RSPs for Case 2

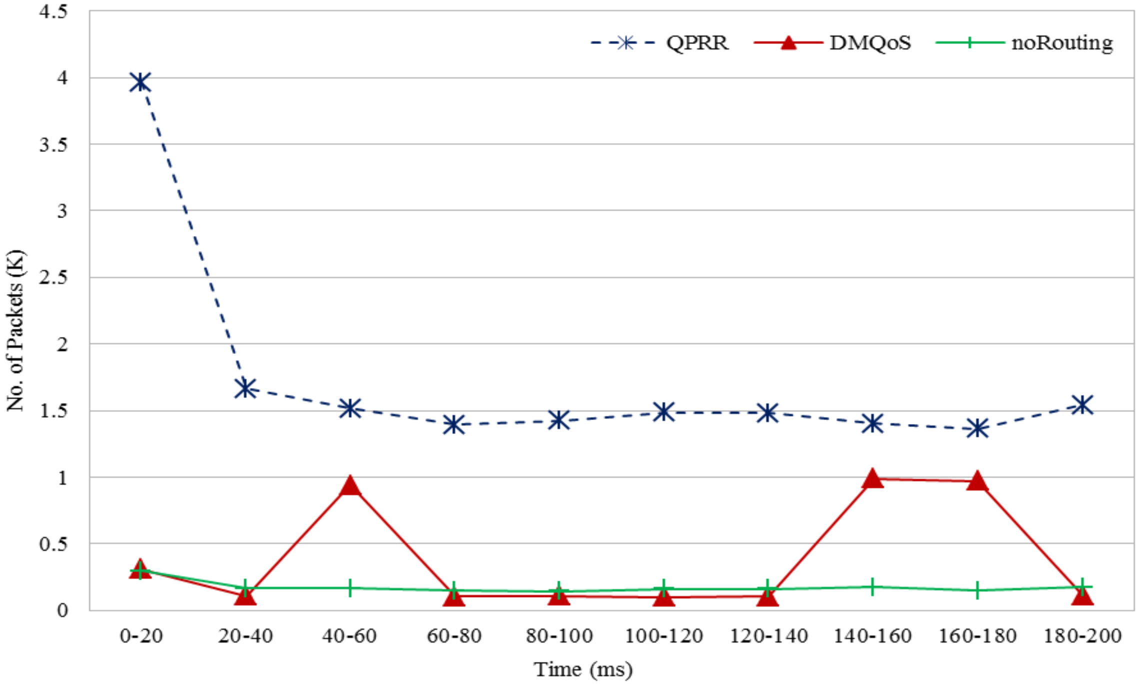


Figure 7-43: Latency for 57.5K RSPs for Case 3

In the real hospital scenario with higher number of nodes (Case 5), it is seen that QPRR continues to outperform DMQoS and noRouting in terms of number of packets delivered reliably to the destination as shown in Figure 7-45. QPRR delivers on average 10 times more packets in comparison to DMQoS and noRouting protocols during all time intervals.

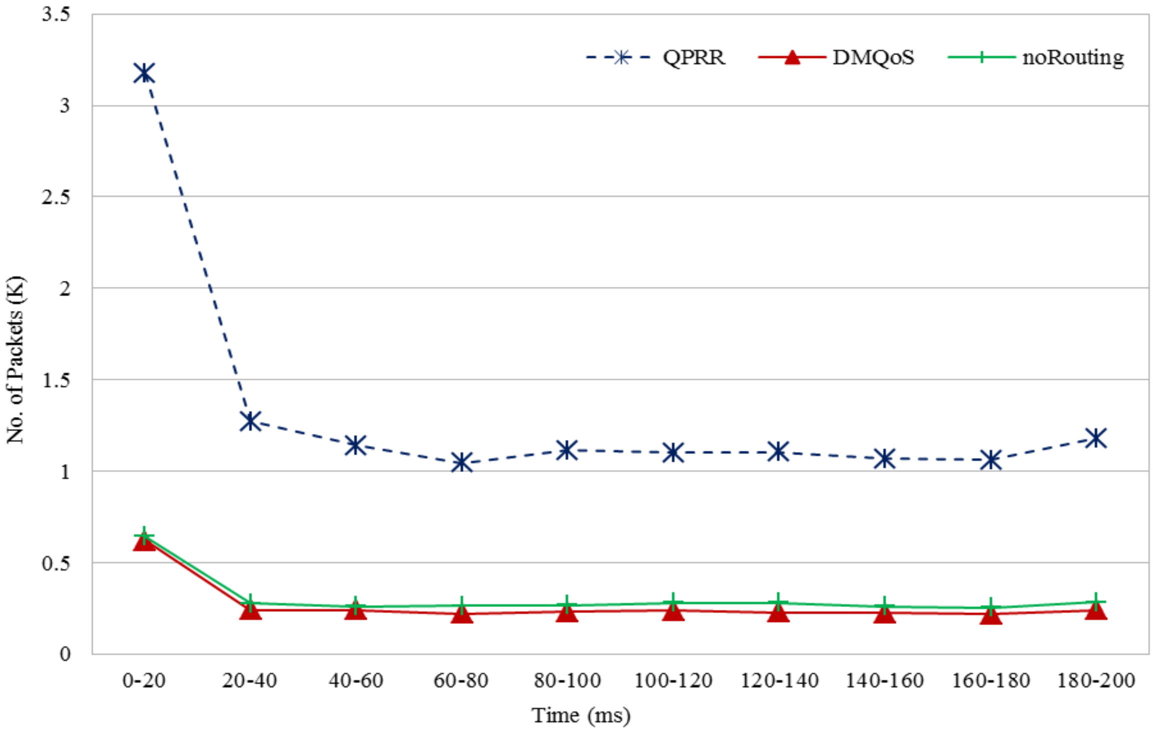


Figure 7-44: Latency for 57.5K RSPs for Case 4

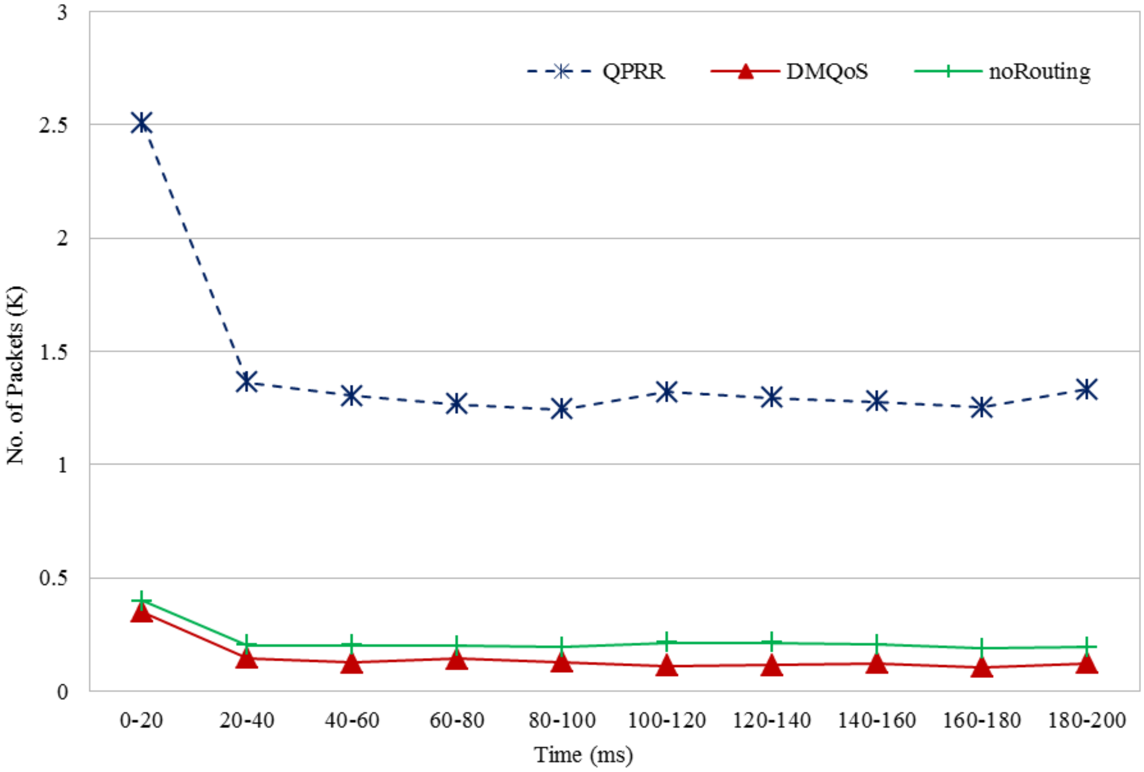


Figure 7-45: Latency for 108.5K RSPs for Case 5

7.5. SUMMARY

This chapter proposed a novel modular QoS-aware routing architecture and associated QoS-aware routing protocol for reliable delivery of critical data packets in hospital BAN communication. The modular architecture includes five modules 1) reliability module, 2) packet classifier, 3) Hello protocol module, 4) routing services module, and 5) QoS-aware queuing module. The proposed novel routing protocol QPRR uses the end-to-end path reliability and redundant paths to ensure increased reliability in BAN communication.

A mechanism is proposed to calculate the end-to-end path reliabilities of all possible paths from source to destination and then decide the degree of path redundancy required to meet the requested data reliability. The simulation results prove that QPRR provides more consistent performance for both movable and stationary patient scenarios with lower device transmit power of -25dBm. The simulations were done to model five different hospital scenarios. Simulations were performed in the OMNeT++ based Castalia 3.2 simulator to observe the successful transmission rate, number of reliability packets dropped, MAC buffer overflow, overall network traffic, number of Hello packets, overall energy consumption, and end-to-end delay (latency) of the proposed QPRR protocol for all five cases. The results show that QPRR provides consistently higher successful transmission rates for a low transmit power of -10dBm. It is shown that for low density stationary BAN nodes, QPRR reliability is in excess of 88% while that of DMQoS is 32% and noRouting is 33%. It is shown that for low density movable BAN nodes, QPRR reliability is in excess of 75% while that of DMQoS is on average 36% and noRouting is on average 23%. For a network with a large number of stationary nodes (i.e. 49 nodes), as found in a real hospital (Case 3 and 4), the reliability of QPRR is about 83% while that of DMQoS is 11% and for noRouting is 19%. In Case 5, with 93 nodes, it is observed that QPRR provides an average of 80% reliability; whereas,the average reliabilies of DMQoS and noRouting are 2% and 15%, respectively.

To summarize, even in the real hospital scenarios, simulating a hospital with 24 and 46 beds requiring the transmission of critical data packets with stringent reliability requirements, QPRR outperforms DMQoS and noRouting in terms of having a much higher successful data transmission rate, with lower network traffic overhead and lower latency, while consuming the same power as the other comparable protocols. However, the improved successful data transmission rates of reliability-sensitive packets come with higher network traffic, as is expected.

CHAPTER 8

ZEQoS: Zahoor Energy and QoS Aware Routing Protocol

This chapter proposes a new routing protocol with the considerations of energy, end-to-end latency, and reliability requirements of BAN data. All the functionalities of the routing protocols EPR, QPRD, and QPRR discussed in previous Chapters 5, 6, and 7 respectively, are integrated in Zahoor Energy and QoS aware routing protocol (ZEQoS). Extensive simulations using OMNeT++ based simulator Castalia 3.2 demonstrate that the performance of the proposed algorithm is satisfactory when tested on a real hospital scenario, and all data types including Ordinary Packets (OPs), Delay-Sensitive Packets (DSPs), and Reliability-Sensitive Packets (RSPs) are used as offered traffic.

The chapter is organized as follows: Section 8.1 provides the motivation of this protocol; Section 8.2 explains the proposed routing protocol (ZEQoS); Sections 8.3 and 8.4 provide the MAC and Network layer modules respectively; Section 8.5 discusses the performance evaluation of ZEQoS; and Section 8.6 provides the summary of this chapter.

8.1. Motivation

An Energy-aware Peering Routing protocol (EPR) discussed in Chapter 5 is used to choose the best next hop for only Ordinary Packets (OPs) by considering the energy availability and geographic information of the devices. In Chapter 6, the QPRD was extended to consider Delay-Sensitive Packets (DSPs) as well as OPs. The resulting QPRD proposed an algorithm to route DSPs in addition to OPs. The redundant paths with the help of end-to-end path reliabilities are used in QPRR, discussed in Chapter 7, to ensure the reliable transmission of Reliability-Sensitive Packets (RSPs) and OPs. These proposed routing protocols EPR, QPRD, and QPRR are not capable of handling DSPs and RSPs, RSPs, and DSPs respectively. For real-time display of patient data in the hospital environment, an energy and QoS aware routing protocol is required to handle all three data types (i.e. OPs, DSPs, and RSPs). With the integration of EPR, QPRD and QPRR, ZEQoS provides the reliable solution for the transmission of OPs, RSPs, and

DSPs and display real-time BAN data.

8.2. Proposed Zahoor Energy and QoS Aware Routing Protocol (ZEQoS)

The proposed energy and QoS aware routing protocol is intended to be associated with the indoor hospital ZK-BAN peering framework presented in Chapter 4. ZK-BAN categorizes the hospital devices into three types with the consideration of their energy levels. The device directly connected with the power source is considered as type 1 device such as NSC. Devices with replaceable batteries (e.g. MDCs) and non-replaceable batteries (e.g. BANCs) are counted in type 2 and type 3 devices respectively. According to ZK-BAN peering framework, the information of BANCs and their respective peer MDCs are stored at the NSC. In centralized mode, the BANCs get the information of its respective peer from the NSC. In distributed mode, BANCs send the data reliably to its peer MDC in order to achieve the purpose of real-time display of patient data. The detailed discussion of ZK-BAN peering framework can be found in Chapter 4.

ZEQoS calculates the best next hops for OPs, DSPs, and RSPs with the help of different modules and algorithms. The next hop for OPs (i.e. NH_E) is based on the communication cost (C_i) which is calculated with the consideration of geographic and energy information of the neighbor nodes. Hello protocol, discussed in Section 5.2.1 of Chapter 5, is used to broadcast the important information of a node to the other nodes. For DSPs, ZEQoS calculates the node delay and end-to-end path delays of all possible paths from source to destination, and then chooses the next hop (i.e. NH_D) device based on the lowest end-to-end path delay. For RSPs, ZEQoS 1) computes the end-to-end path reliabilities of all possible paths, 2) selects the three most reliable paths for each destination, 3) determines the degree of path redundancy and 4) chooses the next hop device(s) based on the most reliable end-to-end path(s) from the source node to the destination. ZEQoS improves the reliability with the help of redundant paths.

The architecture of proposed ZEQoS routing protocol is shown in Figure 8-1 and notations used in this protocol are given in Table 8-1.

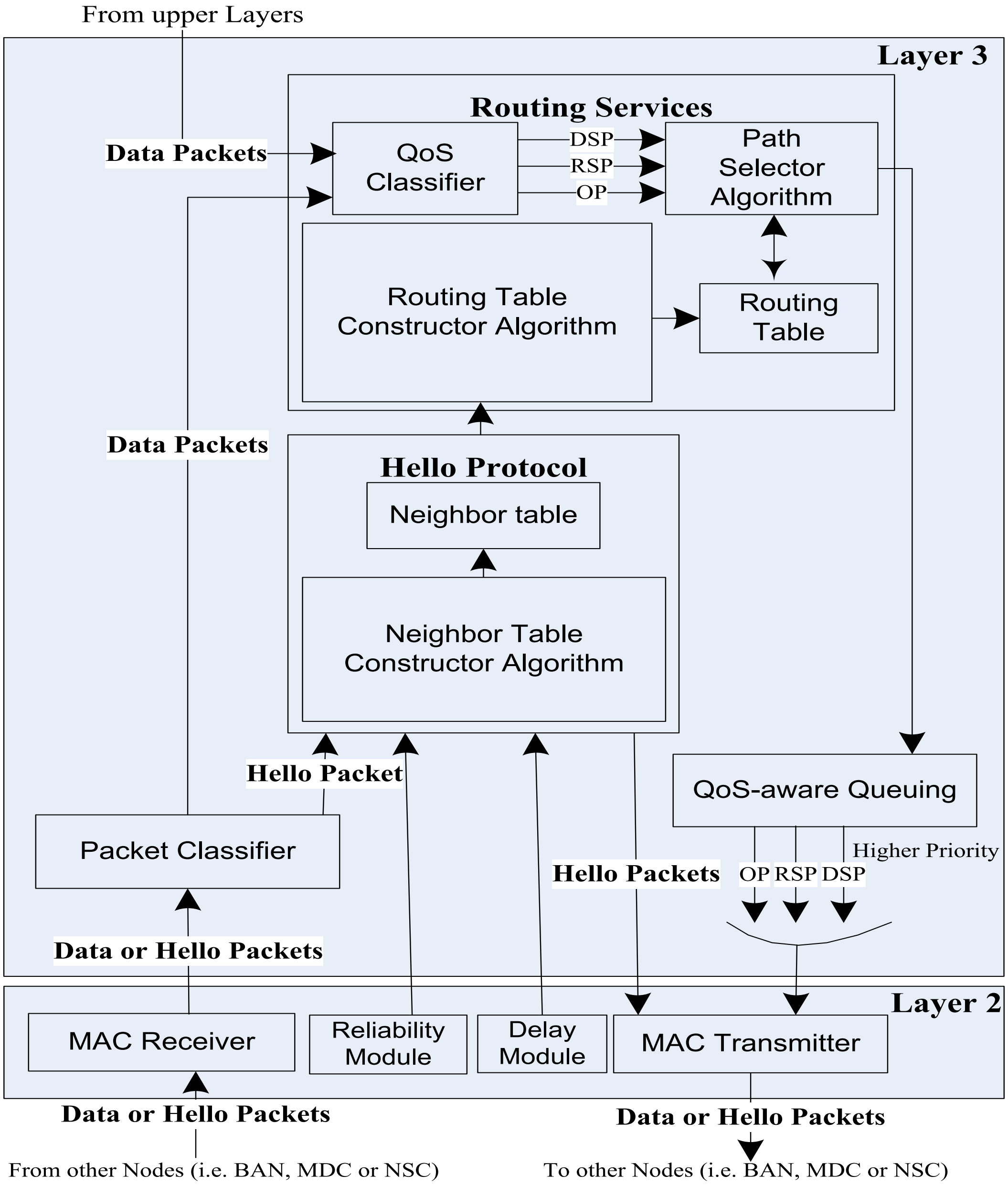


Figure 8-1: ZEQoS routing protocol architecture

Table 8-1: Notations for the proposed algorithm

Field ID	Description
Node *i*	Source node
Node *j*	Neighbor node of source node
Node Dst	Destination node (i.e. NSC, MDCs, BAN)
ID_{Dst}	Destination ID
L_{Dst}	Destination Location
ID_j	Neighbor node *j* ID
L_j	Neighbor node *j* location
$D_{(j,Dst)}$	Distance between neighbor node *j* and destination *Dst*
E_j	Residual energy of node *j*
C_j	Communication cost
T_j	Device type of node *j*
$R_{path(j,Dst)}$	Path reliability between neighbor *j* and destination
$D_{(i,j)}$	Distance between node *i* to neighbor node *j*
$R_{link(i,j)}$	Link reliability from node *i* to neighbor node *j*
$R_{path(i,Dst)}$	Path reliability from node *i* to destination *Dst*
$NH_{(i,Dst)}$	Next Hop between node *i* and destination *Dst*
NH_E	Energy-aware Next Hop
NH_{R1}	1st reliable Next Hop
NH_{R2}	2nd reliable Next Hop
NH_{R3}	3rd reliable Next Hop
NH_D	Next Hop for delay-sensitive packets
$DL_{path(i,Dst)}$	Path delay from node *i* to destination Dst
$DL_{node(i)}$	Time delay within the node *i*
DL_{req}	Required path delay for delay-sensitive packets
R_{req}	Required reliability of reliability-sensitive packets
$R_{option1(i,Dst)}$	1st option reliability for sending reliability-sensitive packets
$R_{option2(i,Dst)}$	2nd option reliability for sending reliability-sensitive packets
$R_{option3(i,Dst)}$	3rd option reliability for sending reliability-sensitive packets

The modules used in ZEQoS are spread into two layers: MAC layer and Network layer. MAC and Network layer modules are discussed below.

8.3. MAC LAYER MODULES

MAC layer contains four modules: MAC receiver, reliability module, delay module, and MAC transmitter. The data or Hello packets from other nodes (i.e. BANC, MDC, or NSC) are received by MAC receiver of the node *i*. MAC receiver checks the MAC address of the packets and only forwards the packets, which contain the broadcast address or MAC address of the node *i* as destination address, to the network layer.

The reliability module of node *i* on MAC layer calculates the numbers of packets sent to neighbor node *j* and the number of acknowledgements received from neighbor node *j*.

The delay module monitors the time required to capture the channel ($DL_{channel(i)}$), MAC layer queuing delay ($DL_{MAC\_queue(i)}$), and transmission time ($DL_{trans(i)}$) of a packet. The delay and reliability modules send their information to the Hello protocol module of the network layer. The neighbor table constructor algorithm in Hello protocols module uses these information to calculate the node delay ($DL_{node(i)}$) and the link reliability between the node *i* and the neighbor node *j* ($R_{link(i,j)}$).

The data and Hello packets from the network layer are received by the MAC transmitter sub-module which stores these packets in the MAC layer queue. The MAC layer queue works in a first-in-first-out (FIFO) fashion. MAC transmitter uses CSMA/CA algorithm to send the data when the channel is captured.

8.4. NETWORK LAYER MODULES

Network layer consists of four modules: Packet Classifier (PC), Hello Protocol Module (HPM), Routing Services Module (RSM), and QoS-aware Queuing Module (QQM). The detailed discussion of these modules is given below.

8.4.1. Packet Classifier

The packet classifier receives data and Hello packets from the MAC receiver module of the MAC Layer. The job of packet classifier is to differentiate and forward the data packets and Hello packets to the routing services module and Hello protocol module respectively.

8.4.2. Hello Protocol Module (HPM)

According to the Hello protocol, type 1 and type 2 devices (NSC or MDCs) send Hello packets periodically and the BANCs broadcast their Hello packets only at the reception of other nodes' Hello packets which contain the NSC or MDC information. The Hello packet fields of node *j* are shown in Figure 8-2. The possible destination (Dst) can be a NSC, MDC or BANC. The Hello Packet contains the information about the destination device ID (ID_{Dst}), destination location (L_{Dst}), sender's ID (ID_j), residual energy (E_j), device type (T_j), distance ($D_{(j,Dst)}$), path reliability ($R_{path(j,Dst)}$) and path delay ($DL_{path(j,Dst)}$). The subscript (j, Dst) means from sender node j to the destination.

ID_{Dst}	**L_{Dst}**	**ID_j**	**L_j**	**$D_{(j,Dst)}$**	**E_j**	**T_j**	**$R_{path(j,Dst)}$**	**$DL_{path(j,Dst)}$**

Figure 8-2: Hello packet structure

The node i receives the Hello packet. The information received from the reliability module, delay module, and Hello packets of the MAC receiver module are used by the neighbor table constructor algorithm to construct the neighbor table. The neighbor table constructor algorithm of node i calculates its own $DL_{path(i,Dst)}$ and $R_{path(i,Dst)}$ based on the information in the Hello packets. Node i updates the values of Hello packet fields and broadcasts it to the other nodes. The mechanism of Hello protocol used in ZEQoS is same as described in Section 5.2 of Chapter 5.

Neighbor table and neighbor table constructor algorithm are the two sub-modules of the Hello protocol module. In addition to Hello packet fields, the neighbor table contains fields for both hop-by-hop delay ($DL_{node(i)}$) and reliability ($R_{link(i)}$), and end-to-end delay ($DL_{path(i,Dst)}$) and reliability ($R_{path(i,Dst)}$). Neighbor table also uses communication cost (C_i) instead of residual energy (E_i). The neighbor table structure of node i is shown in Figure 8-3.

ID_{Dst}	**L_{Dst}**	**ID_j**	**L_j**	**$D_{(j,Dst)}$**	**$D_{(i,j)}$**	**C_j**	**T_j**	**$R_{link(i,j)}$**	**$R_{path(i,Dst)}$**	**$DL_{node(i)}$**	**$DL_{path(i,Dst)}$**

Figure 8-3: Neighbor table structure

8.4.2.1. Neighbor Table Constructor Algorithm

The neighbor table constructor algorithm updates the values of the neighbor table fields periodically after receiving every new Hello packet. Neighbor table constructor algorithm calculates the values of the additional field used in neighbor table such as $DL_{node(i)}$, $R_{link(i)}$, $DL_{path(i,Dst)}$, $R_{path(i,Dst)}$, C_i, and $D_{(i,j)}$. The terms rm, hp, dm, and nt used in Algorithm 8-1 stand for reliability module, Hello packet, delay module, and neighbor table respectively.

The average probability of successful transmission $\bar{X}_i$ after every 4 seconds is calculated by using Equation 8-1.

$$\bar{X}_i = \frac{N_{Acks}}{N_{Trans}} \tag{8-1}$$

where

N_{Acks} = Number of acknowledgement and

N_{Trans} = Number of transmissions.

The link reliability between node i and neighbor node j ($\mathrm{R}_{\mathrm{link(i,j)}}$) is calculated by using the Exponentially Weighted Moving Average (EWMA) Equation 8-2.

$$R_{link(i,j)} = (1 - \rho_{\mathrm{r}}) R_{link(i,j)} + \rho_{\mathrm{r}} \times \bar{X}_i \tag{8-2}$$

where ρ_{r} is the average weighting factor that satisfies $0 < \rho_{\mathrm{r}} \leq 1$. Algorithm 8-1 uses $\rho_{\mathrm{r}} = 0.4$.

The path reliability between node i and destination node *Dst* ($\mathrm{R}_{\mathrm{path(i,Dst)}}$) is calculated by using the Equation 8-3.

$$R_{path(i,Dst)} = R_{link(i,j)} \times R_{path(j,Dst)} \tag{8-3}$$

The values of $R_{link(i,j)}$ and $R_{path(j,Dst)}$ are used from Equation 8-2 and Hello packet (hp) respectively. The calculation of finding $\mathrm{R}_{\mathrm{path(i,Dst)}}$ is given in Algorithm 8-1 (lines 1-4).

The delay due to the queues of MAC & network layers and channel capture ($\mathrm{DL}_{\mathrm{queue+channel}}$) is calculated by using the Exponentially Weighted Moving Average (EWMA) formula.

$$\begin{aligned} &= (1 - \rho_{\mathrm{d}}) * (\mathrm{DL}_{\mathrm{MAC\_queue}}(dm) + \mathrm{DL}_{\mathrm{channel}}(\mathrm{dm}) + \mathrm{DL}_{\mathrm{Net\_queue}}) \\ &\qquad + \rho_{\mathrm{d}} * (\mathrm{DL}_{\mathrm{MAC\_queue}}(dm) + \mathrm{DL}_{\mathrm{channel}}(\mathrm{dm}) + \mathrm{DL}_{\mathrm{Net\_queue}}) \end{aligned} \tag{8-4}$$

where

The values of MAC queue delay and channel capture time are received from delay module (dm); whereas, the values of network queue delays are calculated on network layer. The Initial value of $\mathrm{DL}_{\mathrm{queue+channel}}$ is the delay of the first packet send by the node. The selection of ρ_{d} value is the personal choice and experience, but it should satisfy

$0 < \rho_{\mathrm{d}} \leq 1$. The recommended values are $0.2 \leq \rho_{\mathrm{d}} \leq 0.3$. Algorithm 8-1 uses $\rho_{\mathrm{d}} = 0.2$.

The value of node delay ($\mathrm{DL}_{\mathrm{node(i)}}$) is calculated with addition of the packet delays due to transmission, queuing, processing, and capturing of the channel.

$$\mathrm{DL}_{\mathrm{node(i)}} = \mathrm{DL}_{\mathrm{trans(i)}}(\mathrm{dm}) + \mathrm{DL}_{\mathrm{queue+channel}} + \mathrm{DL}_{\mathrm{proc}} \tag{8-5}$$

The path delay between node *i* and destination node *Dst* ($\mathrm{DL}_{\mathrm{path(i,Dst)}}$) is calculated by using the Equation 8-6.

$$\mathrm{DL}_{\mathrm{path(i,Dst)}} = \mathrm{DL}_{\mathrm{node(i)}} + \mathrm{DL}_{\mathrm{path(j,Dst)}}\ (\mathrm{hp}) \tag{8-6}$$

where

Initial value of $\mathrm{DL}_{\mathrm{path(j,Dst)}}$ is zero when *j=Dst*.

The values of $\mathrm{DL}_{\mathrm{node(i)}}$ is calculated in Equation 8-5 and $\mathrm{DL}_{\mathrm{path(n,Dst)}}$ is received from Hello packet (hp).

The calculation of finding $\mathrm{DL}_{\mathrm{path(i,Dst)}}$ is shown in Algorithm 8-1 from lines 5-9.

The algorithm 8-1 (lines 11-12) calculates the communication cost ($\mathrm{C_j}$) and distance from node *i* to the neighbor node *j* ($\mathrm{D_{(i,j)}}$) by using the Equations 8-7 and 8-8.

$$\mathrm{D_{(i,j)}} = \sqrt{(\mathrm{X_i} - \mathrm{X_j})^2 + (\mathrm{Y_i} - \mathrm{Y_j})^2} \tag{8-7}$$

$$\mathrm{C_j} = \frac{\left(\mathrm{T_j} * \mathrm{D}^2_{(i,j)}\right)}{\mathrm{E_j}} \tag{8-8}$$

where $\mathrm{X_i}$, $\mathrm{Y_i}$ stand for the X, Y coordinates of node *i* and $\mathrm{X_{DST}}$, $\mathrm{Y_{DST}}$ represent the X, Y coordinates of the destination. It is also assumed that the locations of NSC and MDCs are known. The RSSI localization technique given in [76] is used to calculate the values of $\mathrm{X_i}$, and $\mathrm{Y_i}$ of the node *i*. The values of $\mathrm{T_j}$, $\mathrm{D_{(i,j)}}$, and $\mathrm{E_j}$ are received from Hello packet (hp). The shorter distance ($\mathrm{D_{(i,j)}}$), lower device type ($\mathrm{T_j}$), and higher residual energy ($\mathrm{E_j}$) will generate a lower communication cost ($\mathrm{C_j}$). The node *j* with lowest value of $\mathrm{C_j}$ is the best choice for next hop.

Lines 13-26 of Algorithm 8-1 shows that a new record for the destination is added in neighbor table if the distance from the neighbor node *j* to the destination ($D_{(j,Dst)}$) is less than the distance from the node *i* to the destination i.e. $D_{(j,Dst)}(hp) < D_{(i,Dst)}$.

A new record with the information of the neighbor node *j* is also added with the new calculated values as shown in Algorithm 8-1 from lines 27-38.

The neighbor table constructor algorithm repeats the same process of updating the neighbor table after receiving every new Hello packet.

Algorithm 8 – 1 Neighbor table constructor algorithm for ZEQoS

```
INPUT: Hello Packet, at each node $i$.
$\bar{X}_{\mathrm{i}} = \frac{N_{Acks}(\mathrm{rm})}{N_{Trans}(rm)}$
$\rho_{\mathrm{r}} \leftarrow 0.4$
$\mathrm{R}_{\mathrm{link(i,j)}} = (1 - \rho_{\mathrm{r}}) * \mathrm{R}_{\mathrm{link(i,j)}} + \rho_{\mathrm{r}} * \bar{X}_{\mathrm{i}}$
$\mathrm{R}_{\mathrm{path(i,Dst)}} = \mathrm{R}_{\mathrm{link(i,j)}} + \mathrm{R}_{\mathrm{path(j,Dst)}}(hp)$
$\rho_{\mathrm{d}} \leftarrow 0.2$
$\mathrm{DL}_{\mathrm{queue+channel}} \leftarrow$ First packet delay
$\mathrm{DL}_{\mathrm{queue+channel}} = (1 - \rho_{\mathrm{d}}) * (\mathrm{DL}_{\mathrm{MAC\_queue}}(dm) + \mathrm{DL}_{\mathrm{channel}}(\mathrm{dm}) + \mathrm{DL}_{\mathrm{Net\_queue}}) + \rho_{\mathrm{d}} * (\mathrm{DL}_{\mathrm{MAC\_queue}}(dm) + \mathrm{DL}_{\mathrm{channel}}(\mathrm{dm}) + \mathrm{DL}_{\mathrm{Net\_queue}})$
$\mathrm{DL}_{\mathrm{node(i)}} = \mathrm{DL}_{\mathrm{trans(i)}}(dm) + \mathrm{DL}_{\mathrm{queue+channel}} + \mathrm{DL}_{\mathrm{proc}}$
$\mathrm{DL}_{\mathrm{path(i,Dst)}} = \mathrm{DL}_{\mathrm{node(i)}} + \mathrm{DL}_{\mathrm{path(j,Dst)}}(hp)$
$\mathrm{D}_{(\mathrm{i,j})} = \sqrt{(\mathrm{X_i} - \mathrm{X_j})^2 + (\mathrm{Y_i} - \mathrm{Y_j})^2}$
$\mathrm{C_j} = \frac{\left(\mathrm{T_j}\ (\mathrm{hp}) * \mathrm{D}^2_{(\mathrm{i,j})}(\mathrm{hp})\right)}{\mathrm{E_j}(\mathrm{hp})}$
$\mathrm{D}_{(\mathrm{i,Dst})} = \sqrt{(\mathrm{X_i} - \mathrm{X_{Dst}})^2 + (\mathrm{Y_i} - \mathrm{Y_{Dst}})^2}$
    if $\left(\mathrm{D}_{(\mathrm{j,Dst}}(\mathrm{hp}) < \mathrm{D}_{(\mathrm{i,Dst})}\right)$ then
        (add a new record for the Dst's information in the neighbor table)
        $\mathrm{ID_{Dst}}(\mathrm{nt}) \leftarrow \mathrm{ID_{Dst}}(\mathrm{hp})$
        $\mathrm{ID_j}(\mathrm{nt}) \leftarrow \mathrm{ID_j}(\mathrm{hp})$
        $\mathrm{L_j}(\mathrm{nt}) \leftarrow \mathrm{L_j}(\mathrm{hp})$
        $\mathrm{D}_{(\mathrm{j,Dst})}(\mathrm{nt}) \leftarrow \mathrm{D}_{(\mathrm{j,Dst})}(\mathrm{hp})$
        $\mathrm{D}_{(\mathrm{i,j})}(\mathrm{nt}) \leftarrow \mathrm{D}_{(\mathrm{i,j})}$
        $\mathrm{C_j}(\mathrm{nt}) \leftarrow \mathrm{C_j}$
        $\mathrm{T_j}(\mathrm{nt}) \leftarrow \mathrm{T_j}(\mathrm{hp})$
        $\mathrm{R}_{\mathrm{link(i,j)}}(\mathrm{nt}) \leftarrow \mathrm{R}_{\mathrm{link(i,j)}}$
        $\mathrm{R}_{\mathrm{path(i,Dst)}}(\mathrm{nt}) \leftarrow \mathrm{R}_{\mathrm{path(i,Dst)}}$
        $\mathrm{DL}_{\mathrm{node(i)}}(\mathrm{nt}) \leftarrow \mathrm{DL}_{\mathrm{node(i)}}$
        $\mathrm{DL}_{\mathrm{path(i,Dst)}}(\mathrm{nt}) \leftarrow \mathrm{DL}_{\mathrm{path(i,Dst)}}$
    end if
(add a new record for the neighbor node $j$'s information in the neighbor table)
        $\mathrm{ID_{Dst}}(\mathrm{nt}) \leftarrow \mathrm{ID}_{(\mathrm{Dst})}(\mathrm{hp})$
        $\mathrm{ID_j}(\mathrm{nt}) \leftarrow \mathrm{ID_j}(\mathrm{hp})$
        $\mathrm{L_j}(\mathrm{nt}) \leftarrow \mathrm{L_j}(\mathrm{hp})$
        $\mathrm{D}_{(\mathrm{j,Dst})}(\mathrm{nt}) = 0$
        $\mathrm{D}_{(\mathrm{i,j})}(\mathrm{nt}) \leftarrow \mathrm{D}_{(\mathrm{i,j})}$
        $\mathrm{C_j}(\mathrm{nt}) \leftarrow \mathrm{C_j}$
        $\mathrm{T_j}(\mathrm{nt}) \leftarrow \mathrm{T_j}(\mathrm{hp})$
```

```
$R_{link(i,j)}(nt) \leftarrow R_{link(i,j)}$
$R_{path(i,Dst)}(nt) \leftarrow R_{path(i,Dst)}$
$DL_{node(i)}(nt) \leftarrow DL_{node(i)}$
$DL_{path(i,Dst)}(nt) \leftarrow DL_{path(i,Dst)}$
```

8.4.3. Routing Services Module

The routing services module contains four sub-modules: QoS classifier, routing table constructor algorithm, routing table, and path selector algorithm. The QoS classifier sub-module is responsible to categorize the data packets into Delay-Sensitive Packets (DSPs), Reliability-Sensitive Packets (RSPs), and Ordinary Packets (OPs). The routing table constructor algorithm is used to construct and update the routing table. The routing table sub-module stores the required information of the next hop(s) for the data packets. The routing table structure for node *i* is shown in Figure 8-4. The path selector algorithm chooses the best path(s) for each category (DSP, RSP, or OP) of traffic, based on the QoS requirement.

$\mathbf{ID_{Dst}}$	$\mathbf{L_{Dst}}$	$\mathbf{NH_E}$	$\mathbf{NH_{R1}}$	$\mathbf{NH_{R2}}$	$\mathbf{NH_{R3}}$	$\mathbf{NH_D}$	$\mathbf{R_{option1(i,Dst)}}$	$\mathbf{R_{option2(i,Dst)}}$	$\mathbf{R_{option3(i,Dst)}}$	$\mathbf{DL_{path(i,Dst)}}$

Figure 8-4: Routing table structure

8.4.3.1. Routing Table Constructor Algorithm

The neighbor table entries are used to construct the routing table. Neighbor table contains multiple records for each destination. The routing table constructor Algorithm determines the best next hops OPs, RSPs, and DSPs. It filters the neighbor table, and only chooses an entry with the best values for the routing table. As shown in Algorithm 8-2, a new record is added in the routing table for each destination *Dst* ∈ {MDC, NSC, BAN}. Lines 2-8, 9-27, and 28-34 are used to determine the values related to the OPs, RSPs, and DSPs respectively.

The line 2 checks if the neighbor node and destination node is the same node, the next hop for OPs (NH_E) will be the destination ID (ID_{Dst}). Otherwise a neighbor node *j* with the lowest communication cost (C_j) will be selected as next hop (NH_E).

For RSPs, the routing table constructor algorithm of ZEQoS finds three possible paths to ensure the minimum required reliability. For each destination, the three paths with highest reliabilities ($R_{path1(i,Dst)}$, $R_{path2(i,Dst)}$, $R_{path3(i,Dst)}$) are chosen and their corresponding next hops (NH_{R1}, NH_{R2}, NH_{R3}) are stored in the routing table. The routing table constructor calculates and stores the three options for RSP. Line 9 of Algorithm 8-2 shows that the node *i* identifies the next hop candidates by searching the records which have the same ID_{Dst} in neighbor table and stores them in the variable NH_R. If NH_R is empty, it means there is no next hop stored in NH_R. The node stores NULL to NH_{R1}, NH_{R2}, NH_{R3}, $R_{option1(i,Dst)}$, $R_{option2(i,Dst)}$, and $R_{option3(i,Dst)}$. If NH_R is not empty, the next hop nodes' information are stored in the routing table one after another in descending order of their path reliabilities $R_{path(i,Dst)}$. The first neighbor node *j* with the highest reliability in the routing table is stored as NH_{R1} (line 14). If there are two entries in NH_R then the aggregate reliability of first and second paths ($R_{option2(i,Dst)}$) is calculated (line 17-21). In case of more than two entries in NH_R, the aggregate reliability of first, second, and third paths ($R_{option3(i,Dst)}$) is calculated (line 22-26). In the routing table, the three paths with highest reliabilities ($R_{path1(i,Dst)}$, $R_{path2(i,Dst)}$, $R_{path3(i,Dst)}$) are chosen and their corresponding next hops (NH_{R1}, NH_{R2}, NH_{R3}) are stored for each destination, in the routing table. The routing table constructor calculates and stores the three options for RSP. The detailed calculations of $R_{option1(i,Dst)}$, $R_{option2(i,Dst)}$, and $R_{option3(i,Dst)}$ are discussed earlier in Section 7.3.4 of Chapter 7.

For DSP data, the path delay $DL_{path(i,Dst)}$ has been calculated by using the neighbor table constructor algorithm (line 9 of Algorithm 8-1) and stored in neighbor table for each next hop candidate. The node stores the neighbor node's IDs in the variable NH (line 28). If NH has only one entry, this means there is only one path available. The node stores this entry to NH_D (line 30). Otherwise the node sorts the NH entries in ascending order with respect to the path delay (i.e. $DL_{path(i,Dst)}$) values, and then stores the first entry which has the lowest path delay in NH_D (lines 32-33). The next hop candidate NH_D is then stored with its path delay value ($DL_{path(i,Dst)}$) in the routing table. Algorithm 8-2 (lines 27-38) shows that a new record for the destination *Dst* is added with the calculated values.

The routing table constructor algorithm repeats the same process of updating the routing table after receiving every new Hello packet.

Algorithm 8 − 2 Routing table constructor algorithm for ZEQoS

INPUT: Neighbor table, i's neighbor table records $NH_{(i,Dst)}$, $\forall$ Dst $\in$ {MDC, NSC, BAN}

```
for each destination Dst ∈ {NSC, MDC, BAN} do
    if ($ID_j(nt) == ID_{Dst}(nt)$) then
        $NH_E \leftarrow ID_{Dst}(nt)$
    else
        if ($C_j == \min_{k \in NH_{(i,Dst)}} C_k$) then
            $NH_E \leftarrow ID_j(nt)$
        end if
    end if
    $NH_R = \{$All neighbor nodes $j \in NH_{(i,Dst)}\}$
    if ($NH_R == NULL$) then
        Put NULL in $NH_{R1}, NH_{R2}, NH_{R3}, R_{option1(i,Dst)}, R_{option2(i,Dst)}, R_{option3(i,Dst)}$
    else
        Sort $NH_R$ in descending order of $R_{path(i,Dst)}$
        $NH_{R1} =$ first neighbor node $j \in NH_R$
        $R_{option1(i,Dst)} = R_{path(i,Dst)}$
        $P_{error} = 1 - R_{option1(i,Dst)}$
        if ($|NH_R| > 1$)
            $NH_{R2} =$ second neighbor node $j \in NH_R$
            $P_{error} = P_{error} * (1 - R_{path(i,Dst)})$
            $R_{option2(i,Dst)} = 1 - P_{error}$
        end if
        if ($|NH_R| > 2$)
            $NH_{R3} =$ third neighbor node $j \in NH_R$
            $P_{error} = P_{error} * (1 - R_{path(i,Dst)})$
            $R_{option3(i,Dst)} = 1 - P_{error}$
        end if
    end if
    $NH = \{$All neighbor nodes $j \in NH_{(i,Dst)}\}$
    if ($|NH| == 1$) then
        $NH_D \leftarrow NH$
    else if ($|NH| > 1$) then
        Sort NH in ascending order of $DL_{path(i,Dst)}$
        $NH_D =$ first neighbor node $j \in NH$
    end if
        (add a new record for the Dst's information in the routing table)
        $ID_{Dst} \leftarrow ID_{Dst}(nt)$
        $L_{Dst} \leftarrow L_{Dst}(nt)$
        $NH_E \leftarrow NH_E$
        $NH_{R1} \leftarrow NH_{R1}$
        $NH_{R2} \leftarrow NH_{R2}$
        $NH_{R3} \leftarrow NH_{R3}$
        $NH_D \leftarrow NH_D$
        $NH_{option1(i,Dst)} \leftarrow NH_{option1(i,Dst)}$
        $NH_{option2(i,Dst)} \leftarrow NH_{option2(i,Dst)}$
        $NH_{option3(i,Dst)} \leftarrow NH_{option3(i,Dst)}$
        $DL_{path(i,Dst)} \leftarrow DL_{path(i,Dst)}$
end for
```

8.4.3.2. Path Selector Algorithm

The data packets from both upper layers and packet classifier are received by QoS classifier. The QoS classifier classifies the packets into DSP, RSP, and OP data. For each data packet, the path selector algorithm checks the QoS requirement and chooses the most appropriate next hop(s). Lines 2-7, 8-17, and 18-21 of Algorithm 8-3 are used for the selection of appropriate next hops of DSPs, RSPs, and OPs respectively. The path selector algorithm compares the delay requirement (DL_{req}) with the path delay ($DL_{path(i,Dst)}$) of NH_D which is stored in the routing table. If the path delay ($DL_{path(i,Dst)}$) is lower than required delay (DL_{req}), the packet is sent to NH_D (lines 3-4). Otherwise, the packet is dropped (line 6).

Algorithm 8 − 3 Path selector algorithm for ZEQoS

```
INPUT: Routing table, i's routing table records $NH_{(i,Dst)}$, $\forall$ Dst $\in$
{MDC, NSC, BAN}
1.  for each data packet do
2.    if data packet is delay − sensitive packet (DSP)
3.      if ($DL_{path(i,Dst)} <= DL_{req}$) then
4.        send to $NH_D$
5.      else
6.        drop the packet immediately
7.      end if
8.    else if data packet is reliability − sensitive packet (RSP)
9.      if ($R_{option1(i,Dst)} > R_{req}$)
10.       send to $NH_{R1}$
11.     else if ($R_{option2(i,Dst)} > R_{req}$)
12.       send to $NH_{R1}$ and $NH_{R2}$
13.       else if ($R_{option3(i,Dst)} > R_{req}$)
14.         send to $NH_{R1}$, $NH_{R2}$ and $NH_{R3}$
15.       else
16.         drop the packet immediately
17.     end if
18.   else if data packet is Ordinary Packet (OP)
19.       send to $NH_E$
20.     else
21.       drop the packet immediately
22.   end if
23. end for
```

For RSPs, the path selector algorithm checks if the reliability of a single path exceeds $R_{req,}$ then a single path is used to send these packets through NH_{R1} (lines 9-10). In case the required reliability is greater than the reliability of any single path, then, the path selector selects two paths (by using NH_{R1} and NH_{R2}) whose aggregate reliability is more than the requested R_{req} (lines 11-12). If not, three paths are used as long as their aggregate reliability is greater than the R_{req} (lines 13-14) or else the packet is dropped. For OPs, the path selector algorithm returns the next hop NH_E (lines 18-19). Any unknown packet should be dropped without assigning any next hop (line 21).

8.4.4. QoS-aware Queuing Module

The data packets are sent to the QoS-aware Queuing Module (QQM) after the selection of appropriate next hop(s) by routing services module. QQM receives the data packets and separates these packets in three classes (DSPs, RSPs, and OPs). An individual queue is used for each class of packets. QQM functions are the same as discussed in [33]. The priority of the DSPs queue is higher than that of the RSPs and OPs queues. The RSPs queue has lower priority than DSPs queue. The priority of OPs queue is the lowest. By default, the DSPs queue with highest priority sends the packets first. The packets from lower priority RSP queue will be sent only when the DSPs queue is empty. The OPs need to wait until the DSPs and RSPs queues are empty. However, for fair treatment of OPs data, a timeout is used by all the queues. A queue sends the packets to the MAC layer within the period specified by the timeout for that queue. QQM changes the control from higher priority queue to lower priority queue after the queue timeout occurs.

8.5. PERFORMANCE EVALUATION

OMNeT++ based simulator Castalia [78] is used to test the performance of the proposed ZEQoS routing protocol. The simulation results prove that the ZEQoS approach based on end-to-end path delays and reliabilities in addition to the available energy and geographic information of the node is more effective for all data types (i.e. OPs, DSPs, and RSPs). The simulations are done by considering a real 24 bed hospital scenario outlined in Chapter 7. The details about the scenario, parameters information, and performance results for the simulations are provided below.

8.5.1. 49 Nodes in Hospital Environment

A real 24 patient bed hospital with a movable source node is considered for the testing of ZEQoS routing protocol, as shown in Figure 8-5. The approximate measurements used for this hospital environment are similar to the Hematology-Oncology unit of the Children Hospital named IWK Health Centre Halifax, NS, Canada. The approximate area covered by this unit is 16m by 21m. The distance between two beds is 3 meters. Each BAN transmits the data to its respective MDC. All the BANs and MDCs are sending or receiving Hello protocols to/from other nodes and the NSC. The total numbers of nodes used in this scenario are 49 which include 24 BANs, 24 MDCs, and 1 NSC. The NSC is placed on the left side of the deployment area. The patient rooms are in four rows. The room numbers 1-7, 8-12, 13-17, and 10-24 are in rows 1, 2, 3, and 4 respectively. Room number 18 and the nursing station are just in front of all these rows.

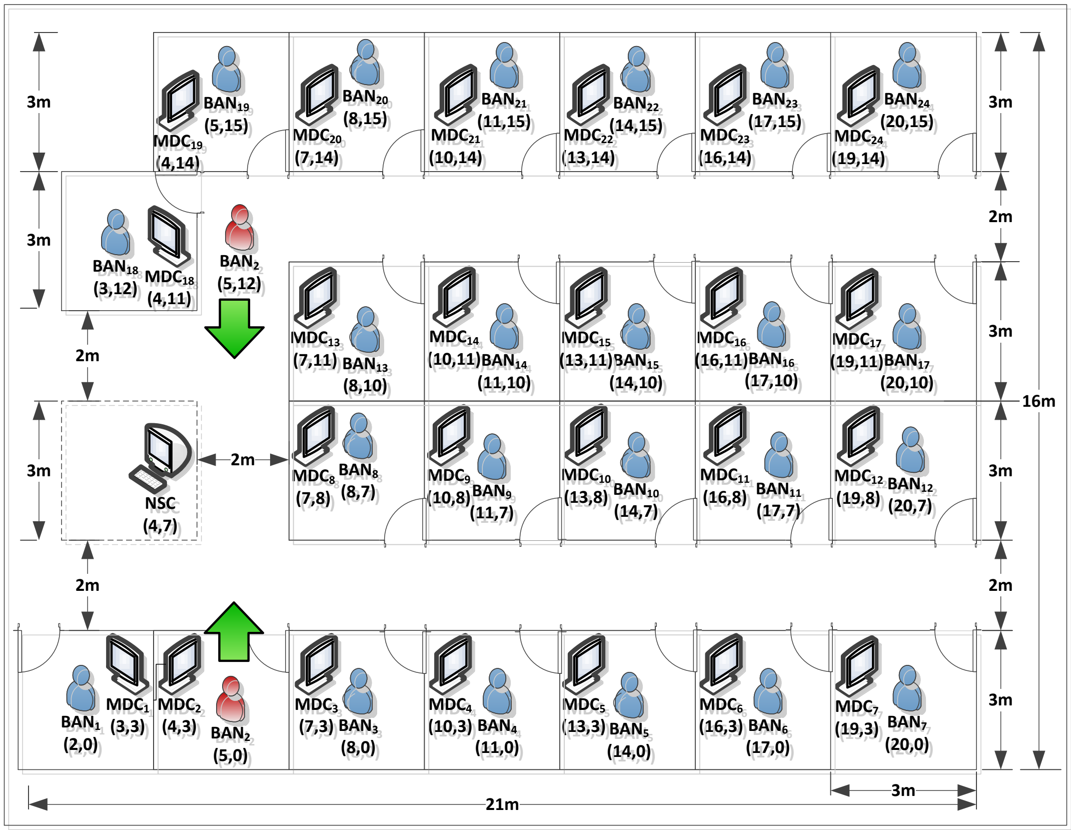


Figure 8-5: Node deployment for 24 patient beds in Hospital environment

The MDCs and BANs are movable but normally a MDC placed in a room moves only within that room. BANs can move freely anywhere. It is assumed that the MDC of one room has a connection with the MDC of the next room. The patient node BAN$_2$ is considered as a movable BAN Coordinator (BANC). As a fast walking patient, the speed of movable BANC is set to 1 meter per second. The node BAN$_2$ moves vertically as shown by green arrows in Figure 8-5. The source node BAN$_2$ displays its data to MDC$_2$.

8.5.2. Parameters Used for Simulations

The transmit power used in simulations is -25dBm. The transmission range of -25dBm is about 3 meters which is the recommended value for BAN communication in hospital environment. The network parameters used in our simulations are shown in Table 8-2.

Table 8-2: Parameters information

Deployment	Area	16m by 21m
	Deployment type	Movable source node BAN$_2$ (shown in Figure 8-5)
	Number of nodes	49 nodes (24 BANs, 24 MDCs, 1 NSC)
	Initial nodes locations	As shown in Figure 8-5
	Initial node energy	18720 J (= 2 AA batteries)
	Buffer size	32 packets
	Link layer trans. Rate	250 Kbps
	Transmit power	−25dBm
Task	Application type	Event − driven
	Max. packet size	32 *Bytes*
	Traffic type	CBR (Constant Bit Rate)
MAC	IEEE 802.15.4	Default values
Simulation	Time	2003 Seconds (3 seconds are setup time)

8.5.3. Performance Results

The source nodes send a total of 95K data packets in the 49 node hospital environment. The above mentioned parameters are calculated after the transmission of every 9.5K packets of all types sent by the source nodes. All types of data packets including OPs, DSPs, and RSPs are sent from source nodes. To achieve a 97% confidence interval for the illustrative results, three runs are simulated in every experiment which may introduce a maximum error of $3x10^{-3}$, based on the error calculation done by Castalia simulator [79]. The below two cases are considered for the same scenario shown in Figure 8-5.

Case 1: A fixed number of DSPs and RSPs but a variable number of OPs are sent from the source nodes. The number of DSPs is 1.2K when 9.5K packets are sent by source nodes. After that 7K DSPs are consistently included in the offered traffic loads by source nodes. The 7K RSPs are included consistently in all offered traffic loads. The OPs are continuously increased from 1K to 81K as with the increase of offered traffic load from 9.5K to 95K respectively. The types of data packets included in the offered traffic load are shown in Figure 8-6.

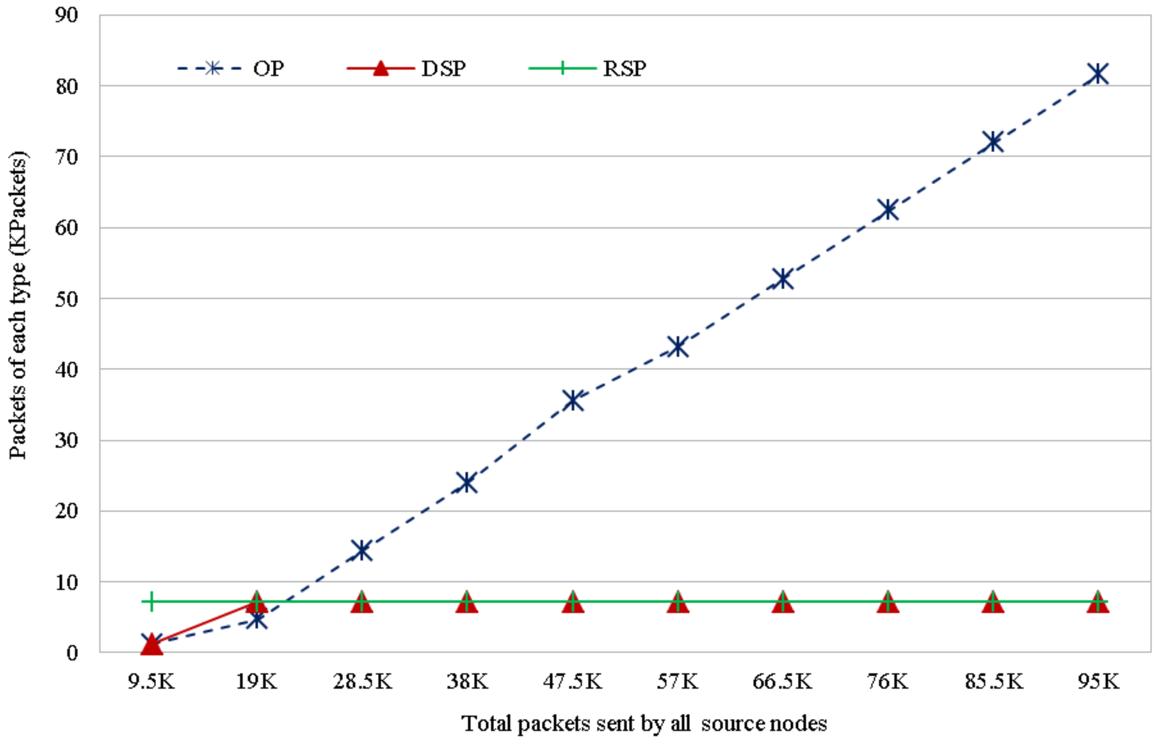


Figure 8-6: Offered traffic by source nodes

Case 2: A variable number of OPs, DSPs, and RSPs are sent with the ratio of 40%, 30%, and 30% respectively. The OPs constitute from 4K to 39.5K packets as the offered traffic load is increased from 9.5K to 95K. Similarly, DSPs and RSPs packets constitute from 2.8K to 28K packets of each type, when the total offered traffic load by source nodes is increased from 9.5K to 95K packets. Figure 8-7 shows the types of packets included in the offered traffic load for Case 2.

The throughput, packets forwarded by intermediate nodes, network traffic, packets dropped at the network layer, packets dropped on MAC layer, and energy consumption are measured. The performance results of each parameter are discussed below.

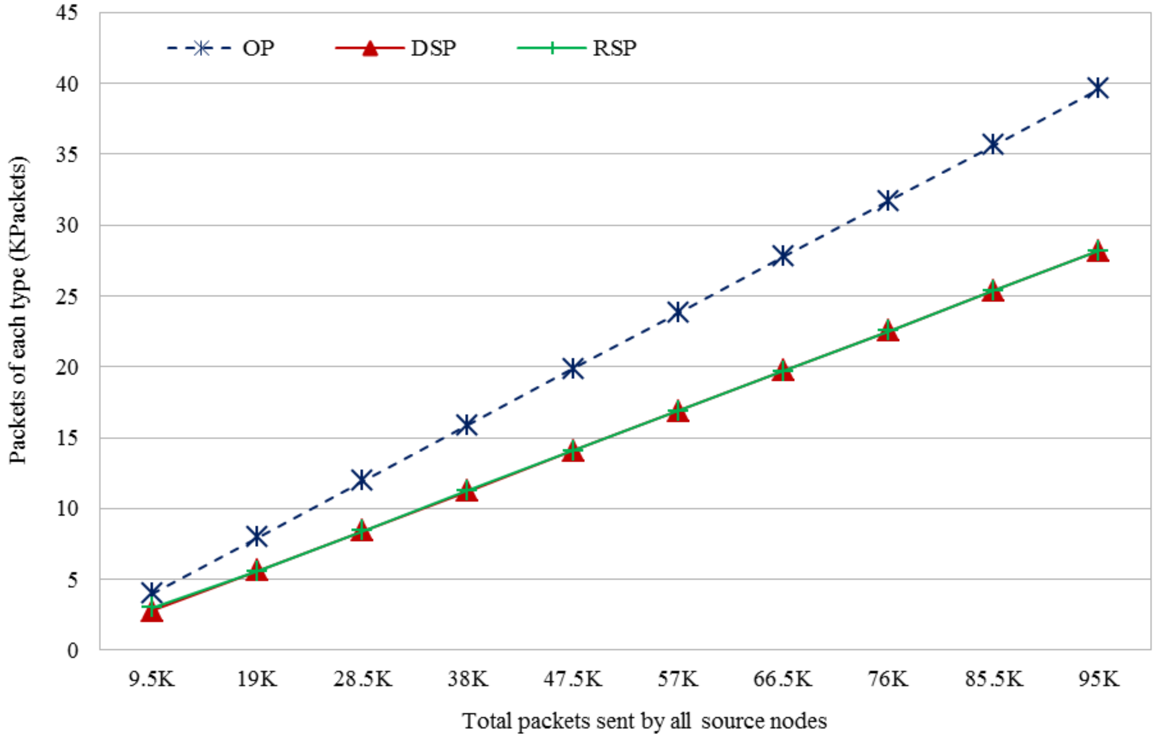


Figure 8-7: Case 2 - Offered traffic by source nodes

8.5.3.1. Throughput

The throughput is measured by calculating the number of packets received successfully at the destination nodes. The successful transmission rate or throughput is measured after the transmission of every 9.5K packets sent by the source. For Case 1, Figure 8-8 shows that ZEQoS provides a consistent reliability which is in excess of 82%, 85%, and 81% for OPs, DSPs, and RSPs respectively. For Case 2, as shown in Figure 8-9, the successful transmission rate of OPs, DSPs, and RSPs are in excess of 88%, 86%, and 75% respectively.

The results from Figures 8-8 and 8-9 show that the mechanism of ZEQoS handles all the data types (i.e. OPs, DSPs, and RSPs) successfully with higher throughput. ZEQoS overcomes the issues of traffic congestion by using the end-to-end path delays and reliabilities for DSPs and RSPs respectively. Also the transmission of RSPs over redundant paths ensures the higher reliability of RSPs packets.

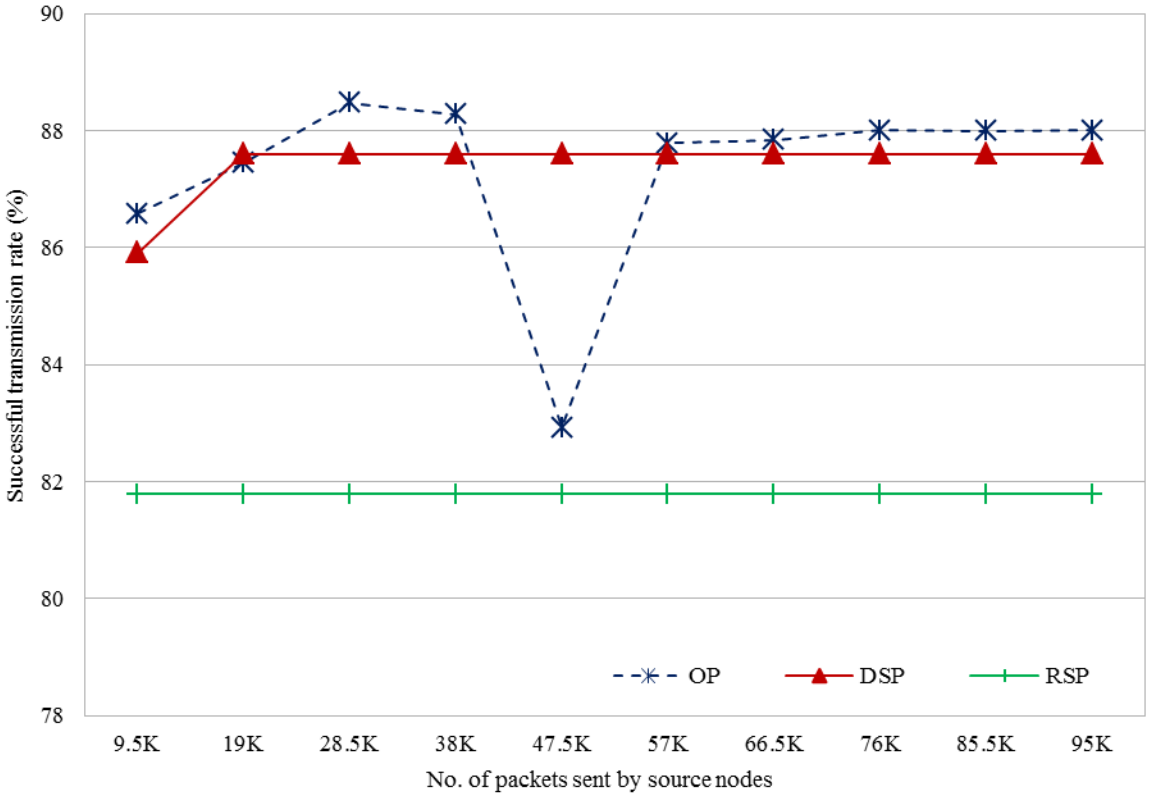


Figure 8-8: Case 1 - Throughput vs. Offered traffic

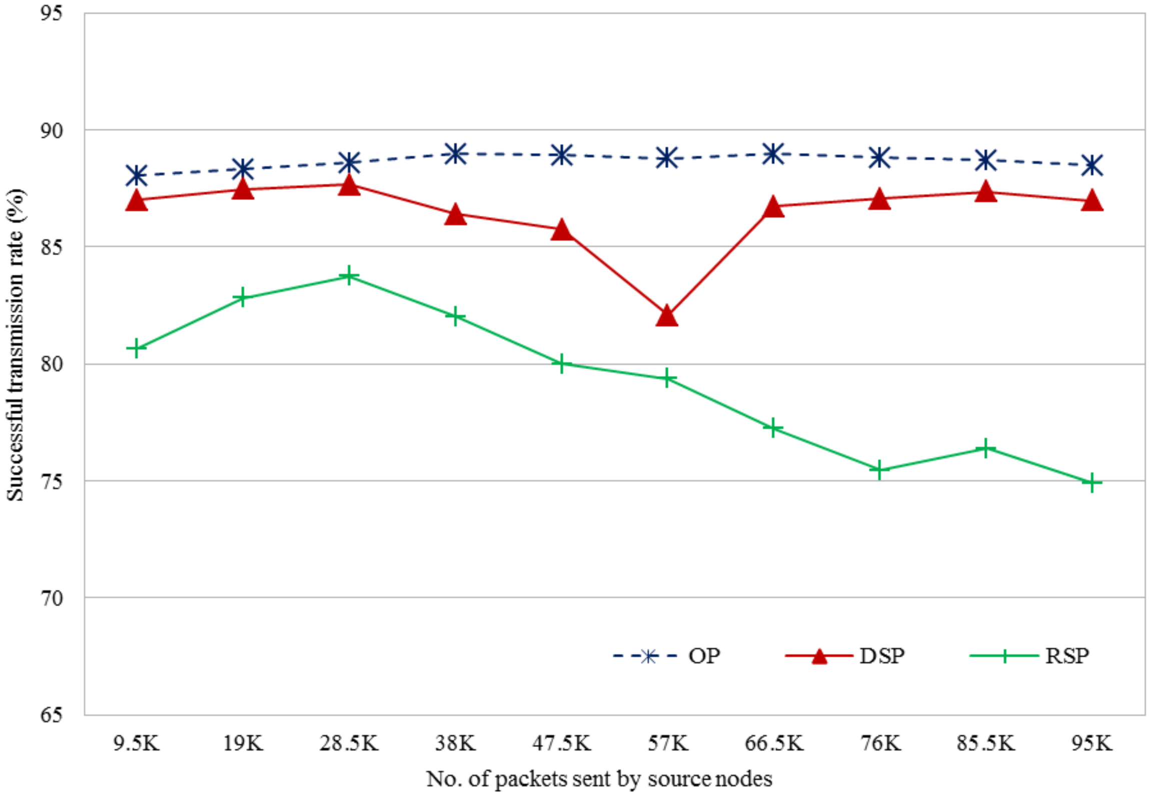


Figure 8-9: Case 2 - Throughput vs. Offered traffic

The path selection mechanism of ZEQoS considers the geographic location, energy availability, end-to-end path delays, and end-to-end path reliabilities for all nodes in the network which helps to improve the overall throughput for all the data types.

8.5.3.2. Packets Forwarded by Intermediate Nodes

The approach used in ZEQoS for the selection of the most appropriate next hop is very effective. In the proposed ZEQoS scheme, a BAN coordinator does not send data to other BAN coordinators unless it is necessary. Figures 8-10 and 8-11 show the number of OPs, DSPs, and RSPs forwarded by the intermediate nodes. It is seen from Figures 8-10 and 8-11 that no OPs or DSPs data packets are forwarded by any intermediate nodes. In Case 1, the number of RSPs forwarded by intermediate nodes are only 94 which is negligible when compared to the overall network traffic. In Case 2, from Figure 8-11 it is shows that the intermediate nodes forwarded 85 to 433 RSPs when offered traffic is increased from 9.5K to 95K. The control of Hello packets broadcast also helps to reduce the packets forwarded by intermediate nodes.

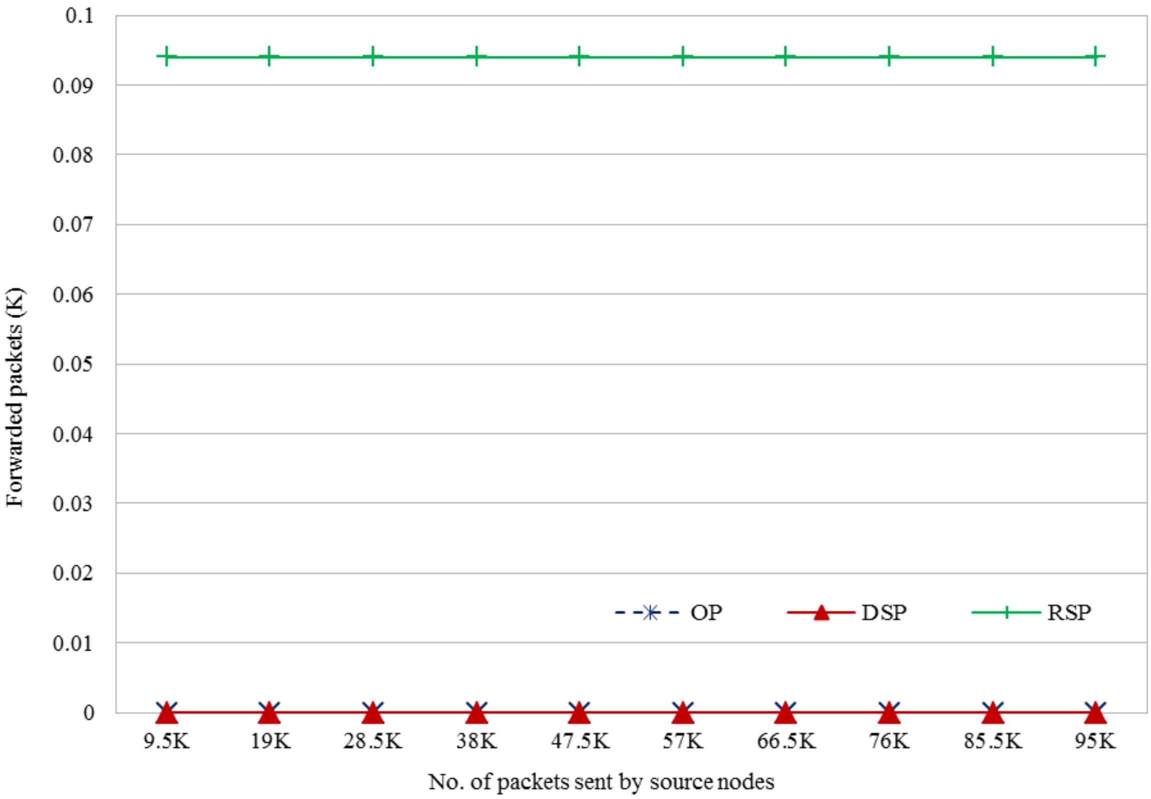


Figure 8-10: Case 1 - Packets forwarded by intermediate nodes

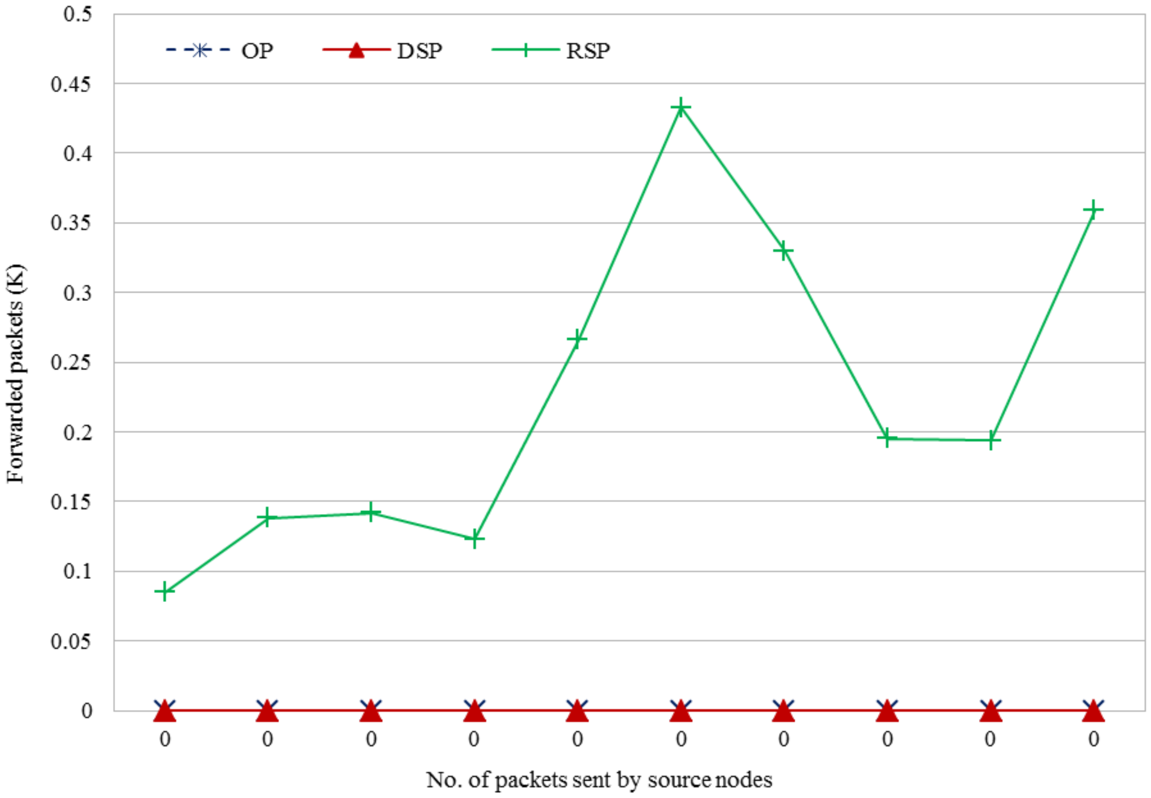


Figure 8-11: Case 2 - Packets forwarded by intermediate nodes

8.5.3.3. Overall Network Traffic

The lower number of forwarded packets as discussed in previous section helps to reduce the overall network traffic. The Hello packets are not added in this network traffic. In Case 1, Figure 8-12 shows that the overall network traffic due to OPs, DSPs, and RSPs are almost 7K, 7K, and 1K to 81K respectively. The numbers of Hello packets are 179K to 2198K when 9.5K to 95K offered traffic load is applied from the source nodes respectively. In Case 2, the overall network traffic due to OPs, DSPs, and RSPs are almost 4K to 39K, 2.5K to 28K, and 3K to 28.5K respectively as shown in Figure 8-13. In addition to data packets, 182K to 2171.5K Hello packets are also part of overall network traffic when 9.5K to 95.5K packets are sent by source nodes, respectively.

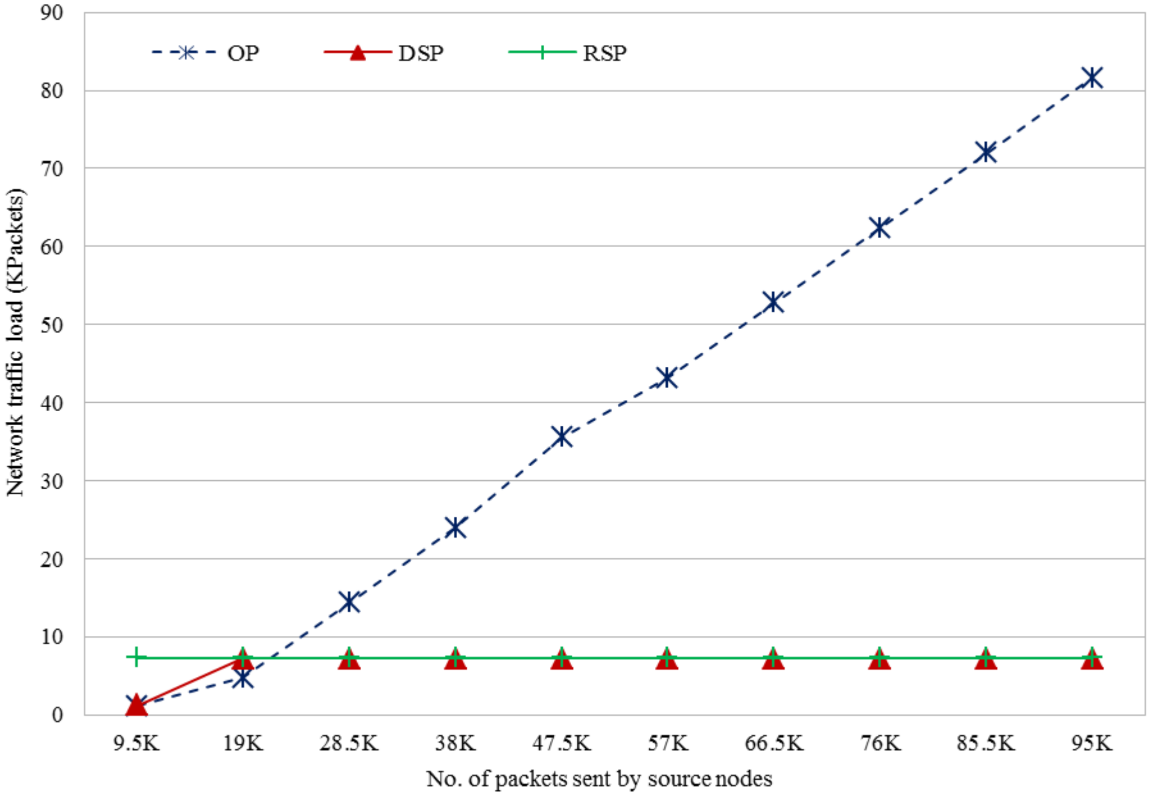


Figure 8-12: Case 1 - Overall network traffic vs. Offered load

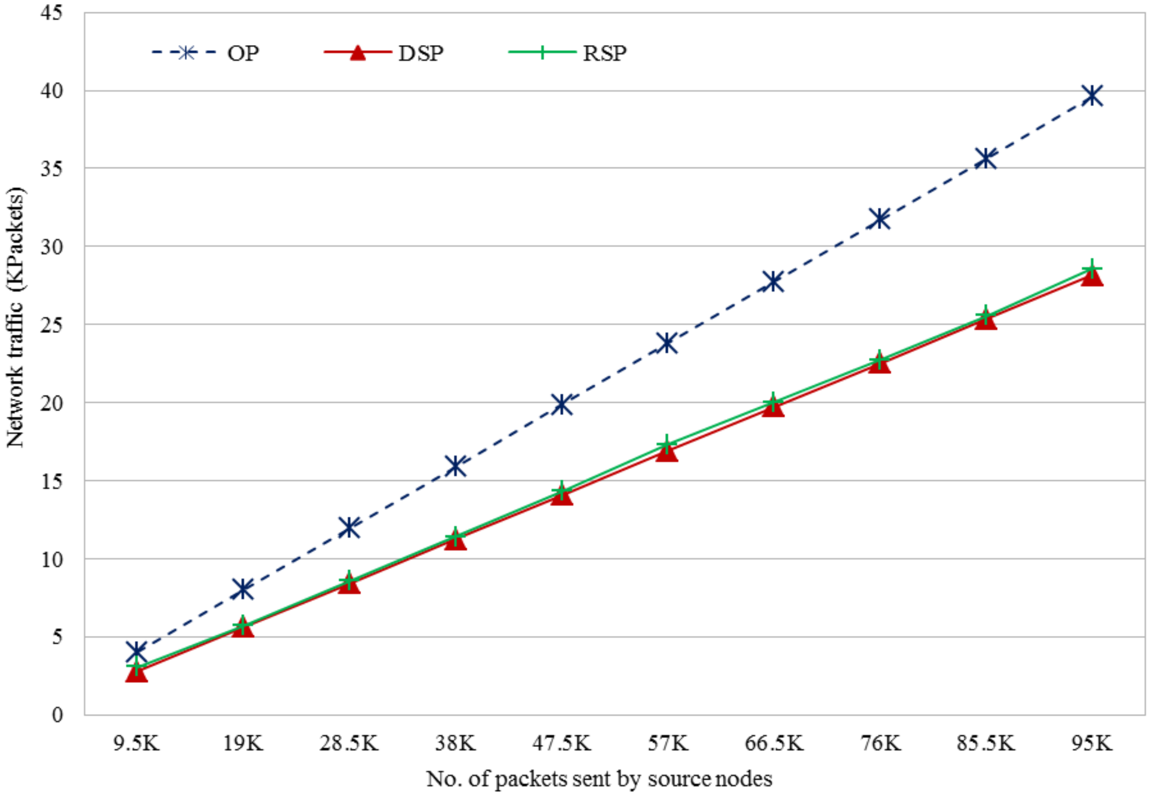


Figure 8-13: Case 2 - Overall network traffic vs. Offered load

8.5.3.4. Packets Dropped at the Network Layer

In previous protocols like DMQoS [33], the source nodes calculate the hop-by-hop delay and reliability of the next hop nodes for the DSPs and RSPs respectively and send the data to the best next hop which has lowest delay for DSPs and highest reliability for RSPs. The next hop then calculates the delays or reliabilities of its upstream nodes. The packets are dropped in case of not meeting the requested delay or reliability by all neighboring upstream nodes. ZEQoS resolves this problem by using the end-to-end path delays and reliabilities for DSPs and RSPs respectively. Also the use of three redundant paths for RSPs in ZEQoS ensures better transmission rate. In Case 1, ZEQoS drops 23 DSPs and 714 RSPs data packets for all the traffic loads as shown in Figure 8-14. In Case 2, Figure 8-15 shows that the DSPs and RSPs dropped at the network layer due to not meeting the requested reliability and delay requirements are average 0.2% and 4.4% respectively.

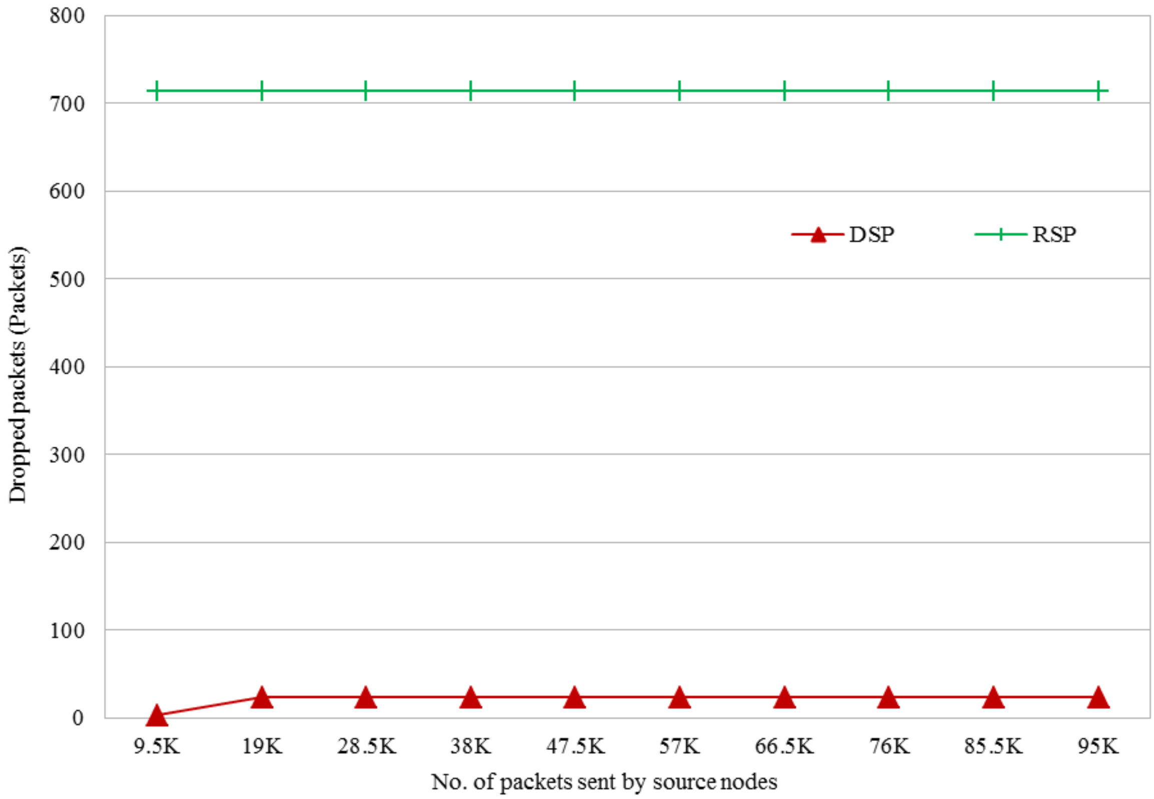


Figure 8-14: Case 1 - Packets dropped at the network layer due to lower delay or reliability requirements

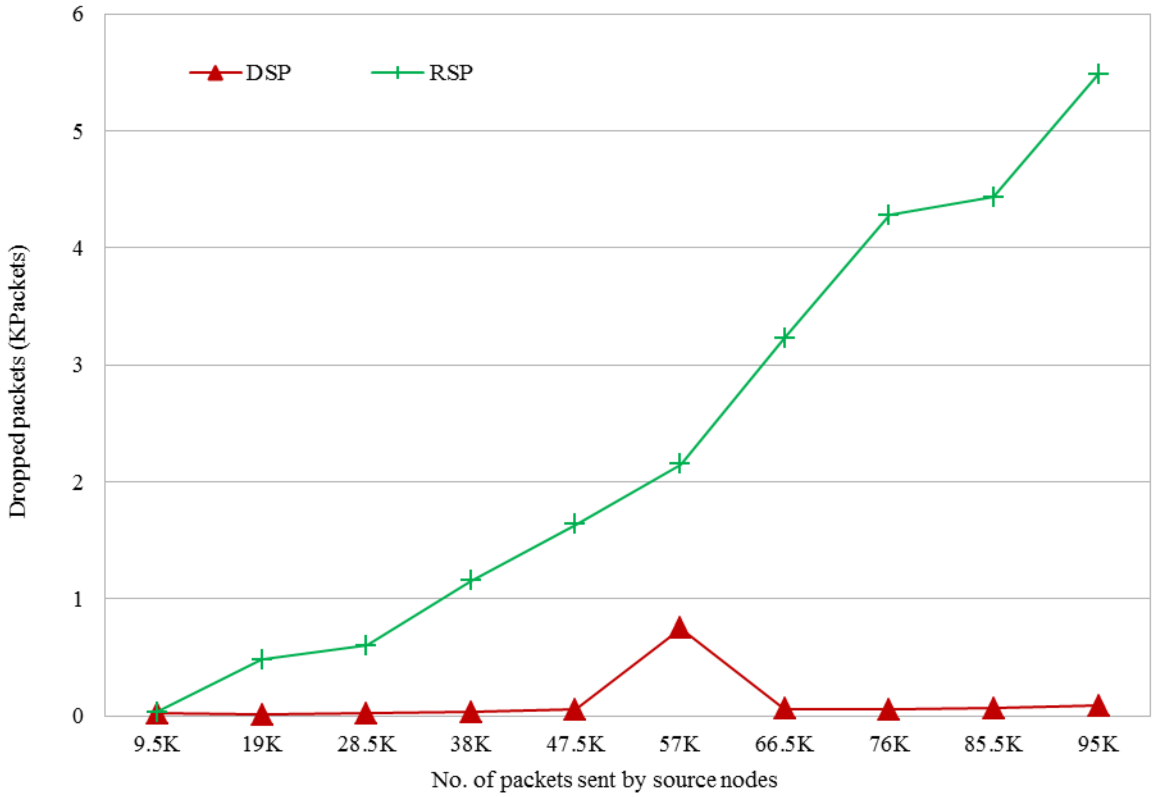


Figure 8-15: Case 2 - Packets dropped at the network layer due to lower delay or reliability requirements

8.5.3.5. Packets Dropped by the MAC Layer

The total number of packets dropped by the MAC layer due to buffer overflow, busy channel, and no acknowledgements are measured. Figures 8-16 and 8-17 show the packets dropped by MAC layer for Case 1 and Case 2 respectively. The total offered traffic including Hello packets are 188K to 2294K and 192K to 2267K for Case 1 and Case 2, respectively. No data packets are dropped due to busy channel in both cases. Also the packets dropped due to no acknowledgments increases from 1K to 11K in both cases. It is seen from the Figures 8-16 and 8-17 that packets dropped due to the MAC buffer overflow are very high. In Case 1, the packets dropped due to the buffer overflow are 16K to 209K. Whereas, in Case 2, the average packets dropped due to buffer overflow are 8.4%.

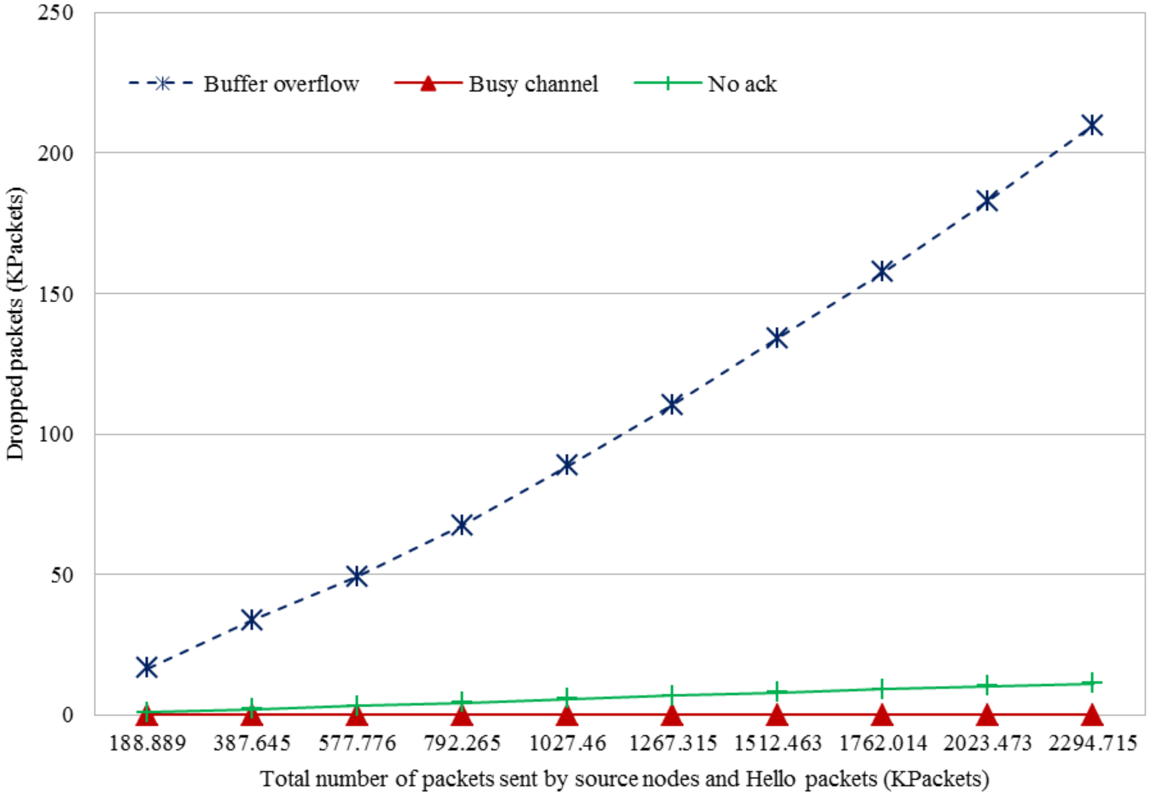


Figure 8-16: Packets dropped by the MAC layer

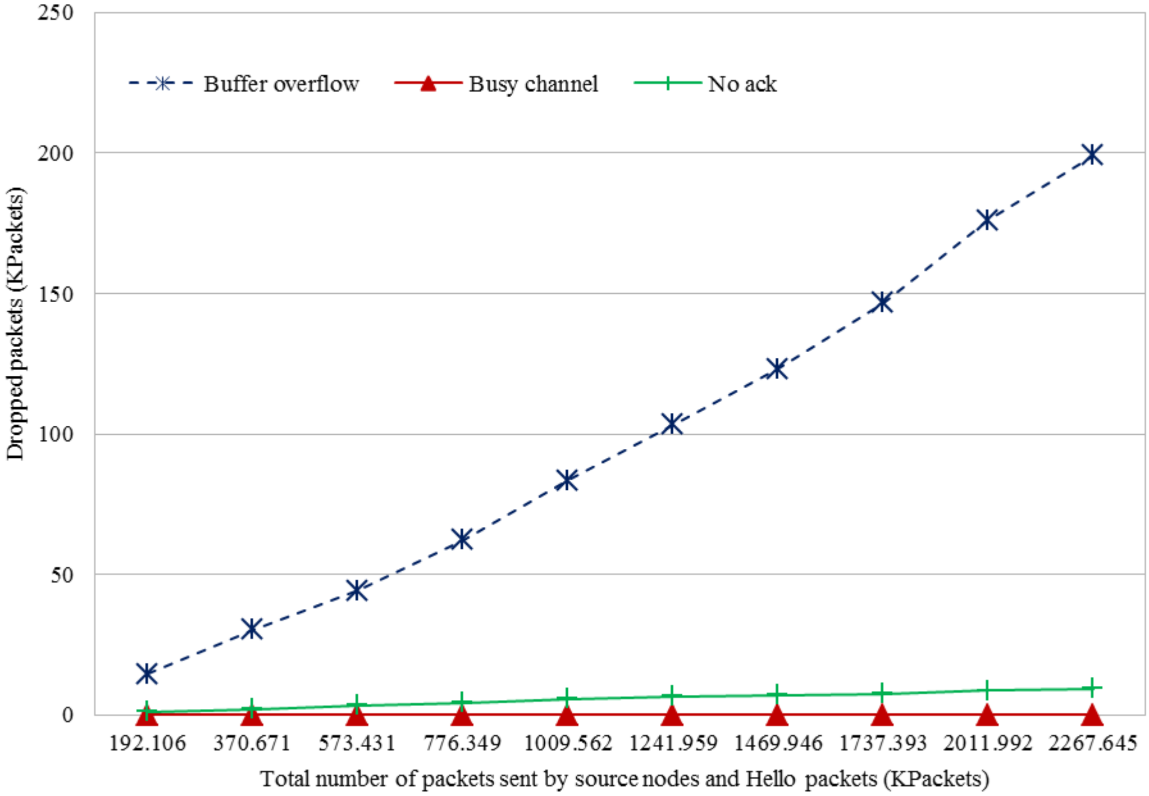


Figure 8-17: Case 2 - Packets dropped by the MAC layer

8.5.3.6. Overall Energy Consumption

The overall energy consumption in both cases for ZEQoS is discussed in this section. It shows that ZEQoS provides a consistent and more reliable delivery of all three types of data packets (OPs, DSPs, and RSPs) as previously discussed in Section 8.5.3.1. The energy consumptions of both cases are similar as shown in Figure 8-18 and Figure 8-19. The figures show that ZEQoS consumes 112 to 118 Joules of energy when the offered load is 9.5K to 95K data packets as sent by source nodes. The drawback of ZEQoS is to consume much higher energy as compared to the energy consumption of the protocols (EPR, QPRD, and QPRR) which are not handling all three data types OPs, DSPs, and RSPs at a time.

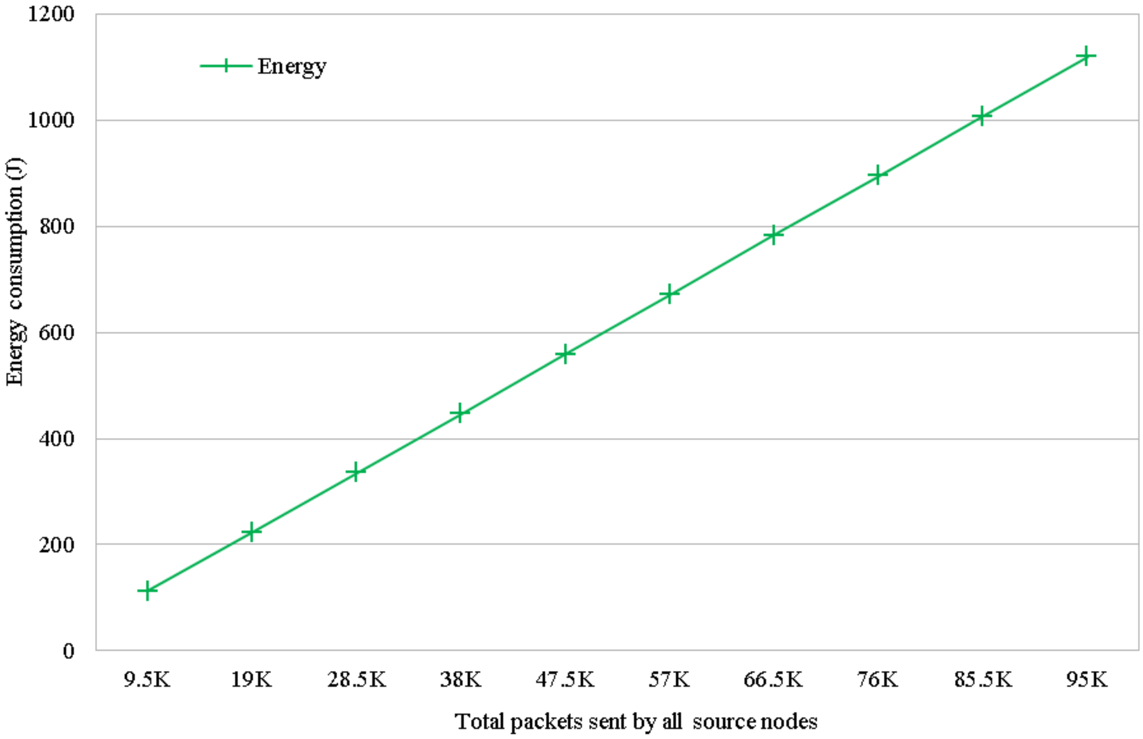


Figure 8-18: Case 1 – Overall energy consumption

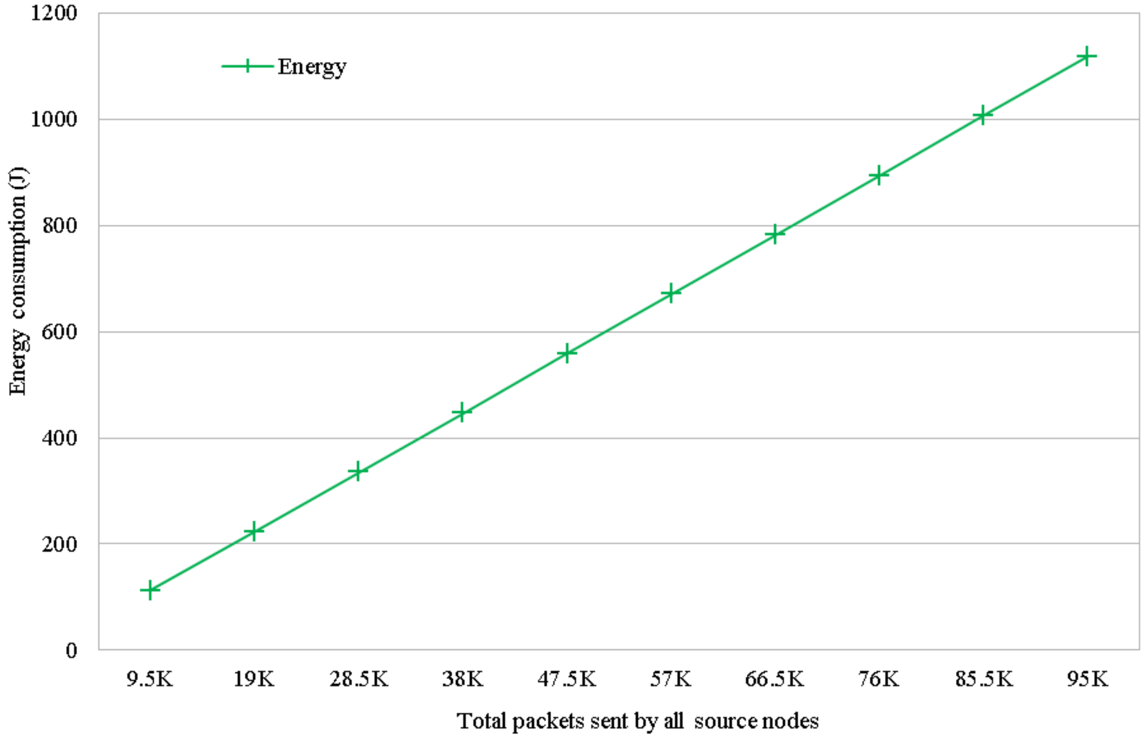


Figure 8-19: Case 2 – Overall energy consumption

8.6. Summary

A new modular energy and QoS aware routing protocol (ZEQoS) for hospital BAN communication is proposed in this chapter. The modules of new protocol are divided into two main types: MAC layer modules and network layer modules. MAC layer modules include the MAC receiver, the reliability module, the delay module, and the MAC transmitter. The packet classifier, the Hello protocol module, the routing services module, and the QoS-aware queuing module are included in network layer modules.

The proposed routing protocol provides a mechanism with the help of neighbor table constructor algorithm, routing table constructor algorithm, and path selector algorithm to calculate the communication costs, end-to-end path delays, and end-to-end path reliabilities of all possible paths from a source to destination, and then decides on the best possible path(s) with the consideration of QoS requirement of the OPs, RSPs, and DSPs.

OMNeT++ based simulator Castalia 3.2 [78] was used to test the performance of the proposed protocol. The simulations were performed by considering a real hospital scenario when a source node was movable. All three types of data packets OPs, RSPs, and DSPs were sent from the source nodes. Both fixed and variable numbers of OPs, DSPs, and RSPs were considered. The simulation results showed that the ZEQoS had in excess of 81% and 75% throughput for all classes of packets in fixed and variable cases respectively when offered a traffic load of 9.5K to 95K packets was used.

CHAPTER 9

CONCLUSIONS AND FUTURE WORK

In this chapter, the major contributions of the thesis are summarized and some key directions for future work are suggested. Section 9.1 discusses the thesis summary, and Section 9.2 provides future research directions.

9.1. THESIS SUMMARY

The research work in this thesis provides five significant contributions to the hospital BAN communication. The main goal of this research is to facilitate patient monitoring in the hospital by introducing a new patient monitoring framework and associated routing protocols with energy and QoS aware features.

The first contribution of this thesis is the proposal of a new patient monitoring framework for hospital BAN communication. The framework aims to display the real-time patient data on the display units with the consideration of data privacy and reliability. According to the framework, the classification of real communication devices used in a hospital is based on their residual energy levels. The mechanism of the framework uses both centralized and distributed modes of communication. In the centralized mode, all the information of the patients and display units are stored on the nursing station computer, which improves data privacy and helps to better manage patient records. The communication between the BAN coordinators and medical display units are in distributed fashion. In distributed mode, the nodes can directly communicate to each other without contacting the central computer, which helps to reduce the overall network traffic and increases the successful transmission rate in addition to providing the ease of node mobility.

The second significant contribution involved the design of a new energy-aware peering routing protocol (EPR). The choice of next hop in EPR is based on the residual energy and geographic information of the neighbor nodes. Hello packets are used to share a node's information with other nodes. The Hello packet received by a node *i* contains information about the destination device ID (ID_{Dst}), destination location (L_{Dst}), sender's ID (ID_j), distance from sender node *j* to the destination ($D_{(j,Dst)}$), residual energy (E_j), and device type (T_j). The values of T_j, $D_{(i,j)}$, and E_j are used to find the communication cost (C_j). According to the mechanism used in EPR, a shorter distance ($D_{(i,j)}$), lower device type (T_j), and higher residual energy (E_j) will generate a lower communication cost (C_j). The node *j* with the lowest value of C_j is the best choice for next hop. In DMQoS [33], every node broadcasts its Hello packets after a specific period of time. A disadvantage of [33] is that the method used for sending the Hello packets and creating the routing table results in increased network traffic, thereby increasing BAN energy consumption. Unlike [33], in EPR, only NSC and MDCs broadcast Hello packets periodically, and the BAN broadcasts its Hello packet only at the reception of other nodes' Hello packets, which contain the NSC or MDC information. The mechanism provides the details of who and when the Hello packets are broadcasted. This mechanism results in a reduced number of broadcasted Hello packets, which reduces overall network traffic and energy consumption. The neighbor table constructor calculates the communication cost and updates the neighbor table periodically after receiving every new Hello packet. The simulation results given in Chapter 5 show that EPR outperforms DMQoS and noRouting protocols with 20% higher throughput and 44% reduced network traffic.

The third significant contribution was the design of a noval modular QoS-aware routing protocol (QPRD) to handle the ordinary and delay-sensitive data for hospital BAN communication. The network layer uses the information from the delay module to calculate the node delay ($DL_{node(i)}$). The delay module used in QPRD monitors the time required to capture the channel ($DL_{channel(i)}$), MAC layer queuing delay ($DL_{MAC\_queue(i)}$), and transmission time ($DL_{trans(i)}$) of a packet. The node delay ($DL_{node(i)}$) is then used to find the end-to-end path delay ($DL_{path(i,Dst)}$). The proposed QPRD selects and chooses the next hop device based on the lowest end-to-end path delay from the source node *i* to the destination *Dst*. Extensive simulations in the OMNeT++ based Castalia 3.2 simulator show the better performance of QPRD than DMQoS and noRouting protocols, as discussed in Chapter 6. QPRD improves throughput by 40% and reduces the overall network traffic by 25%.

The fourth contribution involved the development of a novel modular QoS-aware routing protocol (QPRR) [8] to handle the ordinary and reliability-sensitive data in hospital BAN communication. The source node in QPRR sends the data packets on the redundant paths to achieve the required reliability condition. QPRR calculates the path reliabilities of all possible paths from the source node to the destination and then determines the degree of duplication for sending reliability-sensitive packets. For each destination, QPRR chooses the three paths with the three highest reliabilities ($R_{path1(i,Dst)}$, $R_{path2(i,Dst)}$, $R_{path3(i,Dst)}$) and their corresponding next hops (NH_{R1}, NH_{R2}, NH_{R3}). QPRR sends the data packets on path 1 if the reliability of path 1 ($R_{path1(i,Dst)}$) is greater than the required reliability. Otherwise QPRR sends the duplicate packets on paths 1 and 2 if the combined effect of path 1 and path 2 reliabilities are greater than the required reliability. If not, three redundant paths are used to send the RSPs data packets. The experimental results in Chapter 7 reveal that QPRR performs better than DMQoS and noRouting protocols, in all kinds of cases such as low, medium or large scale networks even when the source nodes are mobile.

The fifth significant contribution was the design of a new integrated energy and QoS aware routing protocol (ZEQoS) to deal with all three data types: Ordinary Packets (OPs), Delay-Sensitive Packets (DSPs), and Reliability-Sensitive Packets (RSPs). ZEQoS provides a mechanism to combine the functionalities of EPR, QPRD, and QPRR protocols. Chapter 9 explains the results of this protocol. The data traffic constitutes OPs, DSPs, and RSPs. The simulation results show the high performance level of ZEQoS for all three data types.

9.2. FUTURE RESEARCH DIRECTIONS

A few interesting future research directions are presented here that are either the extension of this research work or are motivated by using the proposed algorithms to improve reliable and efficient communication of the BANs.

- The novel patient monitoring framework introduced in this thesis helps to enhance data privacy and improves the control on the BAN Coordinators (BANCs) and Medical Display Coordinators (MDCs) by using the hybrid mode of communication. Further improvements in data privacy can be made to introduce the additional data privacy features on the BANCs or MDCs. Moreover, patient privacy can be improved by considering techniques that help protect the patient's location privacy.

- An extension of the routing protocols could cover the inclusion of the fault detection capability of the sensor nodes, particularly the BANC due to its crucial role for the proper operation of the BAN. In this regard, the algorithms used for protocols can be modified such that the BANCs can send the message of node malfunction or failure to the nursing station computer. Also a mechanism is needed for the cases when a BANC or MDC fails.

- The proposed routing protocols are assumed to work only for indoor hospital environments (i.e. indoor BAN communication). An enhancement of the protocols may be made by incorporating the features to work in the scenarios of other indoor (i.e. home environment) or outdoor (i.e. street level) BAN communication. In this case, the BANCs can also be made compatible with the WiFi or cellular system.

BIBLIOGRAPHY

[1] M. Patel and J. Wang, "Applications, challenges, and prospective in emerging body area networking technologies," *IEEE Wireless Communications,* vol. 17, no. 1, pp. 80-88, 2010.

[2] B. Zhen, M. Patel, S. Lee, E. T. Won and A. Astrin, "TG6-technical-requirements," IEEE Project: IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs), 17 March 2011. [Online]. Available: https://mentor.ieee.org/802.15/dcn/11/15-11-0307-00-0006-tg6-closing-report-march-2011.ppt. [Accessed 14 April 2013].

[3] "IEEE 802.15 WPAN™ Task Group 6 (TG6) Body Area Networks," IEEE 802.15, November 2007. [Online]. Available: http:// ieee802.org/15/pub/TG6.html. [Accessed 14 April 2013].

[4] M. Chen, S. Gonzalez, A. Vasilakos, H. Cao and V. C. M.Leung, "Body Area Networks: A Survey," *ACM/Springer Mobile Networks and Applications (MONET),* vol. 16, no. 2, pp. 171-193, 2010.

[5] Z. Khan, S. Sivakumar, W. Phillips and N. Aslam, "A new patient monitoring framework and Energy-aware Peering Routing Protocol (EPR) for Body Area Network Communication," *Journal of Ambient Intelligence and Humanized Computing (JAIHC), Springer,* (in press), Invited paper, 2013.

[6] Z. Khan, N. Aslam, S. Sivakumar and W. Phillips, "Energy-aware Peering Routing Protocol for indoor hospital Body Area Network Communication," *Elsevier, Procedia Computer Science,* vol. 10, no. 0, pp. 188-196, 2012.

[7] Z. Khan, S. Sivakumar, W. Phillips and B. Robertson, "QPRD: QoS-aware Peering Routing Protocol for Delay Sensitive Data in hospital Body Area Network Communication," in *Seventh International Conference on Broadband and Wireless Computing, Communication and Applications (IEEE BWCCA)*, University of Victoria, Victoria, Canada pp. 178-185, November 12-14, 2012.

[8] Z. Khan, S. Sivakumar, W. Phillips and B. Robertson, "A QoS-aware Routing Protocol for Reliability Sensitive Data in Hospital Body Area Networks," *Elsevier, Procedia computer science,* vol. 19, pp. 171-179, 2013.

[9] T. Zimmerman, "Personal Area networks: near-field intrabody communication, , 35 (3 & 4) (1996)," *IBM Systems Journal,* vol. 35, no. 3.4, pp. 609-617, 1996.

[10] K. v. Dam, S. Pitchers and M. Barnard, "From PAN to BAN: Why Body Area Networks?," in *the Wireless World Research Forum (WWRF) Second Meeting, Nokia Research Centre*, Helsinki, Finland, May 10-11, 2001.

[11] T. Zasowski, *A System Concept for Ultra Wideband (UWB) Body Area Networks,* Logos Verlag Berlin: PhD Thesis, ETH Zürich, No. 17259, 2007.

[12] M. Y. Guang-Zhong Yang, Body Sensor Networks, Springer, 2006.

[13] B. Li, Q. Wang, Y. Yang and J. Wang, "Optimal distribution of redundant sensor nodes for wireless sensor networks," in *2006 IEEE International Conference on Industrial Informatics*, Singapore, 2006.

[14] T. Frey, "Invasion of the Digital Body Cloud," FuturistSpeaker.com, 21 October 2011. [Online]. Available: http://www.futuristspeaker.com/2011/10/invasion-of-the-digital-body-cloud/. [Accessed 16 March 2013].

[15] H.-J. Yoo, J. Yoo and L. Yan, "Wireless Fabric Patch Sensors for Wearable Healthcare," in *32nd Annual International Conference of the IEEE EMBS* , Buenos Aires, Argentina, 2010.

[16] R. H. Jacobsen, F. O. Hansen, J. K. Madsen, H. Karstoft, P. H. Mikkelsen, T. A. Skogberg, E. S. Rasmussen, C. Andersen, M. Alrøe and T. S. Toftegaard, "A modular platform for wireless body area network research an real-life experiments.," *International Journal On Advances in Networks and Services, 4(3 & 4).,* vol. 4, no. 3 & 4, pp. 257-277, 2011.

[17] "MCP430 Ultra-low power Microcontroller Manual," Texas Instruments, 2012. [Online]. Available: http://www.ti.com/lit/sg/slab034v/slab034v.pdf. [Accessed 16 March 2013].

[18] B. Lo, S. Thiemjarus, R. King and G.-Z. Yang, "Body Sensor Network–A Wireless Sensor Platform for Pervasive Healthcare Monitoring," *PERVASIVE 2005. LNCS, Springer,* vol. 3468, 2005.

[19] "Chipcon AS SmartRF CC2420 Preliminary Datasheet (rev 1.2)," Barkley, 9 June 2004. [Online]. Available: http://inst.eecs.berkeley.edu/~cs150/Documents/CC2420.pdf. [Accessed 16 March 2013].

[20] "TinyOS," 14 March 2013. [Online]. Available: http://www.tinyos.net/. [Accessed 17 March 2013].

[21] "Types of ZigBee Networks," Software technologies group, 2009. [Online]. Available: http://www.stg.com/wireless/ZigBee_netw.html. [Accessed 22 July 2012].

[22] B. Zhen, M. Patel, S. Lee, E. T. Won and A. Astrin, "15-08-0644-09-0006-tg6-technical-requirements," IEEE Project: IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs), 17 March 2011. [Online]. Available: https://mentor.ieee.org/802.15/dcn/11/15-11-0307-00-0006-tg6-closing-report-march-2011.ppt. [Accessed 4 August 2012].

[23] B. Latré, "Reliable and Energy Efficient Network Protocols for Wireless Body Area Networks," PhD Thesis, 2008.

[24] M. Hanson, H. Powell, A. Barth, K. Ringgenberg, B. Calhoun, J. Aylor and J. Lach, "Body Area Sensor Networks: Challenges and Opportunities," *IEEE Computer Society,* vol. 42, no. 1, pp. 58-65, 2009.

[25] "IEEE Standards Association," 2003. [Online]. Available: http://standards.ieee.org/findstds/standard/802.15.2-2003.html. [Accessed 20 July 2012].

[26] P. Johansson, M. Kazantzidis, R. Kapoor and M. Gerla, "Bluetooth: an enabler for personal area networking," *IEEE Network,* vol. 15, no. 5, p. 28–37, 2001.

[27] B. Heile, "IEEE 802.15 Working Group for WPAN," IEEE, 8 February 2013. [Online]. Available: http://www.ieee802.org/15/. [Accessed 8 February 2013].

[28] L. Hanlen and D. Smith, "Wireless Body-Area-Networks: toward a wearable intranet," 7 February 2011. [Online]. Available: www.nicta.com.au/pub?doc=4690. [Accessed 8 February 2013].

[29] A. Halteren, R. Bults, K. Wac, D. Konstantas, I. Widya, N. Dokovsky, G. Koprinkov, V. Jones and R. Herzog, "Mobile Patient Monitoring: The MobiHealth System," *The Journal on Information Technology in Healthcare,* vol. 2, no. 5, pp. 365-373, 2004.

[30] "HealthService 24: Continuous Mobile Services for HealthCare," Ericsson Enterprise AB (SE), University of Twente (NL), University of Cyprus (CY), Hospital Clinic Provincial de Barcelona (E), Medisch Spectrum Twente (NL), LITO POLYCLINIC PARALIMNI LTD (CY), TMS International B.V. (NL), Yucat B.V. (NL), 2005. [Online]. Available: http://www.healthservice24.com. [Accessed 11 June 2013].

[31] X. Liang and I. Balasingham, "A QoS-aware Routing Service Framework for Biomedical Sensor Networks," in *4th International Symposium on Wireless Communication Systems (ISWCS 2007).* , Trondheim, Norway, 2007.

[32] X. Liang, I. Balasingham and S.-S. Byun, "A reinforcement learning based routing protocol with QoS support for biomedical sensor networks," in *First International Symposium on Applied Sciences on Biomedical and Communication Technologies, 2008. ISABEL '08.*, Aalborg, Denmark, 2008.

[33] M. A. Razzaque, C. S. Hong and S. Lee, "Data-centric Multiobjective QoS-aware Routing Protocol for Body Sensor Networks," *Sensors,* vol. 11, no. 1, p. 917–937, 2011.

[34] A. Wood, G. Virone, T. Doan, Q. Cao, L. Selavo, Y. Wu, L. Fang, Z. He, S. Lin and J. Stankovic, "ALARM-NET: wireless sensor networks for assisted-living and residential monitoring," Department of Computer Science, University of Virginia, Virgina, USA, Technical Report CS-2006-11.

[35] D. Curtis, E. Shih, J. Waterman, J. Guttag, J. Bailey, T. Stair, R. A. Greenes and L. Ohno-Machado, "Physiological signal monitoring in the waiting areas of an emergency room," in *Proceedings of the ICST 3rd international conference on Body area networks (BodyNets '08)*, Arizona, USA, 2008.

[36] T. Gao, T. Massey, L. Selavo, D. Crawford, B.-r. Chen, K. Lorincz, V. Shnayder, L. Hauenstein, F. Dabiri, J. Jeng, A. Chanmugam, D. White, M. Sarrafzadeh and M. Welsh, "The Advanced Health and Disaster Aid Network: A Light-Weight Wireless Medical System for Triage," *Biomedical Circuits and Systems, IEEE Transactions on,* vol. 1, no. 3, pp. 203-216, 2007.

[37] S. Ullah, H. Higgins, B. Braem, B. Latre, C. Blondia, I. Moerman, S. Saleem, Z. Rahman and K. Kwak, "A Comprehensive Survey of Wireless Body Area Networks: On PHY, MAC, and Network Layers Solutions, ,," *Journal of Medical Systems,* vol. 36, no. 3, pp. 1065-1094, 2012.

[38] Z. A. Khan, Advanced Zonal Rectangular LEACH (AZR-LEACH): An Energy Efficient Routing Protocol For Wireless Sensor Networks, Halifax, NS, Canada: Thesis, Dalhousie University, 2012.

[39] P. J. Riu and K. R. Foster, " Heating of tissue by near-field exposure to a dipole: A model analysis," *IEEE Transactions on Biomedical Engineering,* vol. 46, no. 8, pp. 911-917, 1999.

[40] D. Djenouri and I. Balasingham, "New QoS and geographical routing in wireless biomedical sensor networks," in *Sixth International Conference on Broadband Communications, Networks, and Systems (BROADNETS)*, Trondheim, Norway, 2009.

[41] M. Chen, T. Kwon and a. Y. Choi, "Energy-efficient Differentiated Directed Diffusion (EDDD) for Real-Time Traffic in Wireless Sensor Networks," *Elsevier Computer Communications,* vol. 29, no. 2, pp. 231-245, 2006.

[42] M. Chen, V. Leung, S. Mao and Y. Yuan, "Directional Geographical Routing for Real-Time Video Communications in Wireless Sensor Networks," *Computer Communications (JCC),* vol. 30, no. 17, p. 3368–3383, 2007.

[43] X. Huang and Y. Fang, "Multiconstrained QoS multipath routing in wireless sensor networks," *Wireless Networks,* vol. 14, no. 4, p. 465–478, 2008.

[44] E. Felemban, C. G. Lee and E. Ekici, "MMSPEED: Multipath multi-SPEED protocol for QoS guarantee of reliability and timeliness in wireless sensor networks," *IEEE Trans. Mobile Comput.,* vol. 5, no. 6, p. 738–754, 2006.

[45] M. Chen, T. Kwon, S. Mao, Y. Yuan and V. Leung, "Reliable and energy-efficient routing protocol in dense wireless sensor networks," *International Journal on Sensor Networks,* vol. 4, no. 1/2, pp. 104-117 , 2008.

[46] M. Razzaque, M. Alam, M. Rashid and C. Hong, "Multi-Constrained QoS Geographic Routing for Heterogeneous Traffic in Sensor Networks," in *the 5th IEEE Consumer Communications and Networking Conference (CCNC 2008)*, Kyung Hee Univ., Seoul, 2008.

[47] M. Chen, V. Leung, S. Mao, Y. Xiao and I. Chlamtac, "Hybrid Geographical Routing for Flexible Energy-Delay Trade-Offs," *IEEE Transactions on Vehicular Technology,* vol. 58, no. 9, pp. 4976-4988, 2009.

[48] A. Hirata, G. Ushio and T. Shiozawa, "Calculation of temperature rises in the human eye for exposure to EM waves in the ISM frequency bands.," *IEICE Transactions communications,* vol. E83, no. B, pp. 541-548, 2000.

[49] I. E. Commission, "Medical Electrical Equipment, Part 2-33: Particular Requirement for the Safety of Magnetic Resonance Systems for Medical Diagnosis," *IEC:,* pp. 60601-2-33. 2nd edn., 1995.

[50] D. D. Arumugam, A. Gautham, G. Narayanaswamy and D. W. Engels, "Impacts of RF radiation on the human body in a passive wireless healthcare environment.," in *Second International Conference on Pervasive Computing Technologies for Healthcare* , 2008.

[51] H. Ren and M. Q.-H. Meng, "Rate Control to Reduce Bioeffects in Wireless Biomedical Sensor Networks," in *3rd Annual International Conference on Mobile and Ubiquitous Systems - Workshops*, pp.1-7, San Jose, CA, 2006.

[52] Q. Tang, N. Tummala, S. Gupta and L. Schwiebert, "TARA: thermal-aware routing algorithm for implanted sensor networks," in *1st IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS'05)*, Marina Del Rey, CA, 2005.

[53] A. Bag and M. A. Bassiouni, "Energy efficient thermal aware routing algorithms for embedded biomedical sensor networks," in *IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS 2006)*, Vancouver, BC, 2006.

[54] D. Takahashi, Y. Xiao, F. Hu, J. Chen and Y. and Sun, "Temperature-aware routing for telemedicine applications in embedded biomedical sensor networks," *EURASIP Journal on Wireless Communications and Networking,* vol. 2008, no. Article ID 572636, pp. 1-11, 2008.

[55] A. Bag and M. Bassiouni, "Hotspot Preventing Routing Algorithm for Delay-Sensitive Biomedical Sensor Networks," in *IEEE International Conference on Portable Information Devices (PORTABLE07)* , Univ. of Central Florida, Orlando, 2007.

[56] A. Bag and M. Bassiouni, "Routing Algorithm for network of homogeneous and Idless biomedical sensor Nodes (RAIN)," in *IEEE Sensors Applications Symposium (SAS 2008).*, Univ. of Central Florida, Orlando, 2008.

[57] F. Ahourai, M. Tabandeh, M. Jahed and S. Moradi, "A Thermal-aware Shortest Hop Routing Algorithm for in vivo Biomedical Sensor Networks," in *Sixth International Conference on Information Technology: New Generations*, Las Vegas, Nevada, 2009.

[58] W. Heinzelman, A. Chandrakasan and H. Balakrishnan, "Energy-efficient routing protocols for wireless microsensor networks," in *Proc. 33rd Hawaii International Conference System Sciences (HICSS)*, Maui, HI, 2000.

[59] Z. A. Khan and S. Sampalli, "AZR-LEACH: An Energy Efficient Routing Protocol For Wireless Sensor Networks," *International Journal of Communications, Network and System Sciences (IJCNS),* vol. 5, no. 11, pp. 785-795, 2012.

[60] J. Culpepper, L. Dung and M. Moh, "Hybrid indirect transmissions (HIT) for data gathering in wireless micro sensor networks with biomedical applications," in *IEEE Annual Workshop on Computer Communications*, California, USA, 2003.

[61] T. Watteyne, S. Auge-Blum, M. Dohler and D. Barthel, "Anybody: a self-organization protocol for body area networks," in *Second International Conference on Body Area Networks (BODYNETS 2007)*, Florence, Italy, 2007.

[62] N. Mitton and E. Fleury, "Distributed node location in clustered multi-hop wireless networks," in *The First Asian Internet Engineering conference on Technologies for Advanced Heterogeneous Networks*, Bangkok, Thailand, 2005.

[63] A. G. Ruzzelli, R. Jurdak, G. OHare and P. V. D. Stok, "Energy-efficient multi-hop medical sensor networking," in *1st ACM SIGMOBILE international workshop on Systems and networking support for healthcare and assisted living environments*, New York, NY, USA, 2007.

[64] B. Braem, B. Latre, I. Moerman, C. Blondia and P. and Demeester, "The wireless autonomous spanning tree protocol for multihop wireless body area networks," in *3rd International Conference on Mobile and Ubiquitous Systems: Networking and Services*, San Jose, CA, 2006.

[65] B. Latre, B. Braem, I. Moerman, C. Blondia, E. Reusens, W. Joseph and P. and Demeester, "A low-delay protocol for multihop wireless body area networks," in *4th International Conference on Mobile and Ubiquitous Systems: Networking and Services*, Philadelphia, PA, USA, 2007.

[66] D. Singelée, B. Latré, B. Braem, M. Peeters, M. De, P. Cleyn, B. Preneel, I. Moerman and C. Blondia, "A Secure Cross-layer Protocol for Multi-hop Wireless Body Area Networks," in *7th international conference on Ad-hoc, Mobile and Wireless Networks*, Sophia-Antipolis, France, 2008.

[67] A. Bag and M. A. Bassiouni, "BIOCOMM: a cross-layer medium access control (MAC) and routing protocol co-design for biomedical sensor networks," *International Journal of Parallel, Emergent and Distributed Systems,* vol. 24, no. 1, pp. 85-103, 2009.

[68] S. Jiang, Y. Cao, S. Lyengar, P. Kuryloski, R. Jafari, Y. Xue, R. Bajcsy and S. Wicker, "CareNet: an integrated wireless sensor networking environment for remote healthcare.," in *Proceeding of ICST 3rd international conference on body area networks (BodyNets '08).*, Tempe, Arizona, USA, 2008.

[69] V. Shnayder, B. Chen, K. Lorincz, T. Fulford-Jones and a. M. Welsh, "Sensor Networks for Medical Care Sensor Networks for Medical Care," TR-08-05, Cambridge, MA., 2005.

[70] J. Ko, C. Lu, M. Srivastava, J. Stankovic, A. Terzis and M. Welsh, "Wireless Sensor Networks for Healthcare," *Proceedings of the IEEE,* vol. 98, no. 11, pp. 1947-1960, 2010.

[71] "IEEE 802.15 WPAN™ Task Group 6 (TG6) Body Area Networks," IEEE standards, November 2007. [Online]. Available: http://www.ieee802.org/15/pub/TG6.html. [Accessed 20 November 2012].

[72] A. Wood, G. Virone, T. Doan, Q. Cao, L. Selavo, Y. Wu, L. Fang, Z. He, S. Lin and J. Stankovic, "ALARM-NET: wireless sensor networks for assisted-living and residential monitoring," Department of Computer Science, University of Virginia, Virgina, USA, 2006.

[73] S. Agarwal, Divya and G.N.Pandey, "SVM based context awareness using body area sensor network for pervasive healthcare monitoring," in *Proceedings of the First International Conference on Intelligent Interactive Technologies and multimedia (IITM '10)*, Allahabad, India, 2010.

[74] D.-Y. Kim and J. Cho, "WBAN meets WBAN: Smart Mobile Space over Wireless Body Area Networks," in *IEEE 70th Vehicular Technology Conference Fall (VTC 2009-Fall)*, Anchorage, Alaska, USA, 2009.

[75] C. Hedrick, "Routing Information Protocol - rfc1058," Network Working Group, June 1988. [Online]. Available: http://tools.ietf.org/html/rfc1058. [Accessed 9 April 2013].

[76] J. Xu, W. Liu, F. Lang, Y. Zhang and C. Wang, "Distance Measurement Model Based on RSSI in WSN," *Wireless Sensor Network,* pp. 606-611, 2010.

[77] M. Jevti, N. Zogovi and G. Dimić, "Evaluation of Wireless Sensor Network Simulators," in *17th Telecommunications Forum (TELFOR 2009)*, Belgrade, Serbia, 2009.

[78] NICTA, "Castalia," National ICT Australia, March 2011. [Online]. Available: http://castalia.npc.nicta.com.au. [Accessed 28 May 2013].

[79] A. Boulis, "Castalia, Wireless Sensor Network Simulator, NICTA," March 2011. [Online]. Available: http://castalia.npc.nicta.com.au/pdfs/Castalia%20-%20User%20Manual.pdf. [Accessed 15 May 2013].

[80] M. Chen, S. Gonzalez, A. Vasilakos, H. Cao and V. C. M.Leung, *Mobile Networks and Applications,* vol. 16, no. 2, pp. 171-193, April 2011 .

[81] Y. Xue, H. S. Lee, M. Yang, P. Kumarawadu, H. Ghenniwa and W. Shen, "Performance Evaluation of NS-2 Simulator for Wireless Sensor Networks," in *Canadian Conference on Electrical and Computer Engineering, 2007 (CCECE 2007)*, Vancouver, BC, 2007.

APPENDIX A

This section contains the brief information of routing modules used in OMNeT++ based Castalia 3.2 simulator.

EhnPBRouting: The main program to perform the routing function.

The routing module consists of three components.

EhnRoutingTable: Routing table

MulObjQosQ: Multi-object QoS Queue

EhnHelloPacket: Hello packet

Functions in EhnPBRouting

startup() -> initializes the parameters with the value assigned in .ned file and assigns default values to all other parameters. Moreover, creates an EhnroutingTable object and MulobjQosQ object. Sends Hello packets if the node is sink. Set timer to update routing table.

sendHelloPacket() -> Broadcasts a Hello packet. In this function, it puts its location and the sink's information into the Hello packet. If it is a sink and the Hello packet originates from it, the sink location is the same with the node location. Otherwise, the node location field is the location of current node and the sink location field is the location of the node which the Hello packet originates from.

initRoutingTable() -> Creates an object of EhnRoutingTable
initMultiObjQ() -> Creates an object of MulObjQosQ

fromApplicationLayer() -> This function reacts to a data packet received from application layer. What we do here is to encapsulate the packet, look up the routing table (calling lookfrNextHop()) and get the next hop(s). Then enqueue to MulObjQosQ (calling enqMulObjQosQ()).

lookfrNextHop() -> This function is to check the next hop from the routing table. In this function, we check the priority of the packet first, then check the next hop from routing table for each class.

processBufferPacket() -> This function tries to send all the packets in the MulObjQosQ. It process from high priority to low priority and FIFO for the packets with the same priority.

fromMacLayer() -> This function reacts to the data packet received from MAC layer. What we do here is to check the destination first. If the destination is this node or broadcast address, pass the packet to upper layer. Otherwise, check if it is the next hop of the packet. If yes, check the next hop from routing table, put the packet into MulObjQosQ and keep processing the packets in the MulOBJQosQ. If the packet is Hello Packet, it checks the distance to the sink. It only process the Hello packet from the node has shorter distance to the sink.

handleMacControlMessage()-> This function reacts to the control message received from MAC layer. There are 2 kinds of control message in our protocol. The first one is 'Sent time notification' and the second one is 'Ack notification'. The first thing this function do is to check what type of message it is. If it is ACK notification, update the ACK count in RelTableACK. If it is sent time notification, record the sent time for that packet.

Functions in EhnRoutingTable
initRoutingTbl() -> Initialize all the data structures and assign default value to the variables.

updateNeighborInfo() -> In this function, it checks that if it is a new sink first (By checking if the sink in the routing table). If yes, add a new record into routing table. Then check if it is a new neighbor. If yes, add a record into neighbor table. Otherwise, update the neighbor information.

updateRoutingTable() -> This function is invoked periodically to update the routing table. The loop in this function is to find the neighbor through which can reach the certain sink and the neighbor must has shorter distance to the sink. Then add the neighbors into temp table and pass to calEAGF() for further process. This function update ERP next hop through calling calEAGF(), update reliability next hop through calling relControl() update delay next hop through calling delayControl().

calEAGF() -> This function is invoked periodically to update the EPR next hops in routing table. After getting the temp table from updateRoutingTable(), it finds the neighbor with lowest communication cost.

relControl() -> This function is invoked periodically to update the QPRR next hops in the routing table. The first loop is to update the reliabilities of all the records in neighbor table. The second and third loops are to sort the records in neighbor table by destination and path reliability. The last loop is to update the reliability next hop 1, reliability next hop 2 and reliability next hop 3 for all the records in routing table.

delayControl() -> This function is invoked periodiclally to update the QPRD next hops in routing table. This function is simple. It just find the neighbor with the lowest deday for the each sink and update the routing table.

EAGFNextHop() -> Return the EPR next hop for certain sink.

reliabilityNextHop() -> This function is called to return the QPRR next hops for certain sink with reliability requirement. If single route can satisfy the reliability requirement, return the next hop with highest reliatility. If two routes can satisfy the reliability requirement, the 2 next hops with higher reliability will be return. If it needs three routes to guarantee the reliability, all three next hops will be return. If even the aggregate reliability of all the three routes are still lower the requirement, error code(999) will be return.

delayNextHop() -> This function is called to return the QPRD next hop with can satisfy the delay requirement. This function search the routing table and return the next hop through which the packet can be transmitted successfully before it times out.

Functions in MulObjQosQ
enqMulObjQosQ() -> This function put the data packets into corresponding queue.

deqMulObjQosQ() -> This function dequeue a data packet from Multi Object Qos Queue. It scan the the queues from the one with highest priority to the one with lowest priority and return the first packet it find in the queues.

delElement() -> delete the first packet from the queue with given priority.

initMulObjQosQ() -> Initialize all the queues and assign default values to those variables.

empty() -> This function checks if all the queues are empty. If yes, return true. Otherwise, return false.

calCPDelay() -> This function calculates the delay of critical packet.

calRPDelay() -> This function calculates the delay of QPRR packet.

calDPDelay() -> This function calculates the delay of QPRD packet.

calOPDelay() -> This function calculates the delay of EPR packet.

APPENDIX B

Appendix B provides the code of the important functions of the routing protocol modules used in Castalia 3.2 simulator. Section B.1 gives the code of EhnPBRouting.cc function. Sections B.2 and B.3 contain the codes of EhnPBRouting.h and ehnHelloPacket_m.cc functions.

B.1. Routing protocol module (EhnPBRouting.cc)

```
/*****************************************************************************
* Routing protocol module                                                    *
* OMNeT++ based simulator Castalia 3.2 is used for simulations               *
* Castalia website: http://castalia.research.nicta.com.au/index.php/en/      *
* By Zahoor A. Khan                                                          */
/*****************************************************************************
*//EhnPBRouting.cc                                                           *
******************************************************************************/
#include "EhnPBRouting.h"

Define_Module(EhnPBRouting);

void EhnPBRouting::startup()                          // read parameters from ned file
{
	CPPacketCounter = 0;
	DPPacketCounter = 0;
	RPPacketCounter = 0;
	OPPacketCounter = 0;
	recOPPktCounter = 0;
	recRPPktCounter = 0;
	recDPPktCounter = 0;
	recCPPktCounter = 0;
	frOPPktCounter = 0;
	frRPPktCounter = 0;
	frDPPktCounter = 0;
	frCPPktCounter = 0;
	helloCounter = 0;
	otCounter = 0;
	pktSeqNumber = 0;

	helloTimeout = (double)par("helloTimeout") / 1000.0;
	upRTableTimeout = (double)par("upRTableTimeout") / 1000.0;
	pbRoutingFrameOverhead = par("pbRoutingFrameOverhead");
	avSmthFac = (double)par("avSmthFac")/10;
	msrWin = (int)par("msrWin");
	sink = par("sink");
	mdc = par("mdc");
	deviceLevel = par("deviceLevel");

	cModule *nodeModule = getParentModule()->getParentModule();
	xCoor = nodeModule->par("xCoor");
	yCoor = nodeModule->par("yCoor");

	cModule *rmModule = getParentModule()->getParentModule()-
>getSubmodule("ResourceManager");
	initEnergy = rmModule->par("initialEnergy");

	rsrcManager = check_and_cast<ResourceManager*>(getParentModule()-
>getParentModule()->getSubmodule("ResourceManager"));

	initRoutingTable();
	initMultiObjQ();

	trace() << "sink" << sink;
	if (sink) {
		sendHelloPacket();
	}

	setTimer(UPDATE_ROUTING_TABLE,upRTableTimeout);
}

void EhnPBRouting::sendHelloPacket()
{
	trace() << "sending hello packet";
	EhnHelloPacket *helloPkt = new EhnHelloPacket("Sink Hello Packet",
NETWORK_LAYER_PACKET);

	helloPkt->setSinkID(atoi(SELF_NETWORK_ADDRESS));
	helloPkt->setSource(SELF_NETWORK_ADDRESS);
	helloPkt->setDestination(BROADCAST_NETWORK_ADDRESS);
	helloPkt->setSinkXCoor(xCoor);
	helloPkt->setSinkYCoor(yCoor);
	helloPkt->setNodeXCoor(xCoor);
	helloPkt->setNodeYCoor(yCoor);
	helloPkt->setResEnergy(1);
	helloPkt->setDelay(0);
	helloPkt->setSinkDistance(0);
	helloPkt->setPathReliability(1);
	helloPkt->setDeviceLevel(deviceLevel);
	helloCounter++;
```

```
        toMacLayer(helloPkt, BROADCAST_MAC_ADDRESS);
        setTimer(HELLO_TIMEOUT,helloTimeout);
}

void EhnPBRouting::initRoutingTable()
{
        cModule *nModule = getParentModule()->getParentModule();
        rTable = new EhnRoutingTable(nModule->par("xCoor"),nModule-
>par("yCoor"));
        rTable->initRoutingTbl();
        rTable->setSmoothFactor(avSmthFac);
}

void EhnPBRouting::initMultiObjQ()
{
        mOQueue = new MulObjQosQ();
        mOQueue->initMulObjQosQ();
}

void EhnPBRouting::fromApplicationLayer(cPacket * pkt, const char *destination)
{
        if (sink)
                return;

        PBRoutingPacket *netPacket = new PBRoutingPacket("EPBRouting packet",
NETWORK_LAYER_PACKET);
        netPacket->setSource(SELF_NETWORK_ADDRESS);

//      stringstream st;
//      int destIDInt = 999;
//      st << destIDInt;
//      string destIDStr = st.str();
//      netPacket->setDestination(destIDStr.c_str());
        netPacket->setDestination(destination);

        ProjectBanPacket * PBPkt = check_and_cast<ProjectBanPacket *>(pkt);
        netPacket->setPktClassfier(PBPkt->getPacketClassfier());
        netPacket->setReliability(PBPkt->getPacketReliability());
        netPacket->setLifeTime(PBPkt->getPacketDelay());
        netPacket->setArriveTime(SIMTIME_DBL(getClock()));
        netPacket->setSeqNumber(++pktSeqNumber);
        encapsulatePacket(netPacket, pkt);

        lookfrNextHop(netPacket);
        processBufferPacket();
```

```
    switch(PBPkt->getPacketClassfier()) {
        case OP: {
            OPPacketCounter++;
            break;
        }
        case RP: {
            RPPacketCounter++;
            break;
        }
        case DP: {
            DPPacketCounter++;
            break;
        }
        case CP: {
            CPPacketCounter++;
            break;
        }
    }
}


void EhnPBRouting::fromMacLayer(cPacket * pkt, int srcMacAddress, double rssi,
double lqi)
{
    PBRoutingPacket *netPacket = dynamic_cast <PBRoutingPacket*>(pkt);

    if (netPacket) {
        string destination(netPacket->getDestination());
//      trace() <<"receive routing packet with destination: " << destination << "
from node " << netPacket->getSource();
        trace() << "arrive time: " << netPacket->getArriveTime() << " lifetime : "
<< netPacket->getLifeTime();

        int selfNetworkAddr = atoi(SELF_NETWORK_ADDRESS);
        int nextHop  = netPacket->getNextHop();
        int pktClass = netPacket->getPktClassfier();

        if (destination.compare(SELF_NETWORK_ADDRESS) == 0){
            switch(pktClass) {
                case OP: {
                    trace() <<"to app OP " << netPacket-
>getSeqNumber();
                    recOPPktCounter++;
                    toApplicationLayer(decapsulatePacket(pkt));
                    break;
                }
                case RP: {
                    trace() <<"to app RP " << netPacket-
>getSeqNumber();
                    recRPPktCounter++;
                    toApplicationLayer(decapsulatePacket(pkt));
                    break;
                }
                case DP: {
                    if (netPacket->getLifeTime() < 0) {
                        otCounter++;
                        trace() << "DP packet time out, drop in sink
point";
                        break;
                    }
                    trace() <<"to app DP " << netPacket-
>getSeqNumber();
                    recDPPktCounter++;
                    toApplicationLayer(decapsulatePacket(pkt));
                    break;
                }
                case CP: {
                    trace() <<"to app CP " << netPacket-
>getSeqNumber();
                    recCPPktCounter++;
                    toApplicationLayer(decapsulatePacket(pkt));
                    break;
                }
            }

        } else if (destination.compare(BROADCAST_NETWORK_ADDRESS)
== 0){
            trace() << "selfNetworkAddr = broadcast address";

        }
        else if (selfNetworkAddr == nextHop ){
            trace() << "selfNetworkAddr = nextHop";

            switch(pktClass) {
                case OP: {
                    frOPPktCounter++;
                    break;
                }
                case RP: {
                    frRPPktCounter++;
                    break;
                }
                case DP: {
                    frDPPktCounter++;
                    break;
                }
                case CP: {
                    frCPPktCounter++;
                    break;
                }
            }

            PBRoutingPacket *nPacket = netPacket->dup();
            lookfrNextHop(nPacket);
            processBufferPacket();
        }
    } else if (!sink) {
        EhnHelloPacket *recHelloPkt = dynamic_cast <EhnHelloPacket*>(pkt);
        if (recHelloPkt){
            trace() << "recieved sink brocast from node " << recHelloPkt-
>getSource() << " path delay: " << recHelloPkt->getDelay();

            int sinkXCoor = recHelloPkt->getSinkXCoor();
            int sinkYCoor = recHelloPkt->getSinkYCoor();
            int sinkID = recHelloPkt->getSinkID();
            double sinkDist = sqrt((xCoor - sinkXCoor) * (xCoor - sinkXCoor)
+ (yCoor - sinkYCoor) * (yCoor - sinkYCoor));

            if(sinkDist > recHelloPkt->getSinkDistance()){
                rTable->updateNeighborInfo(recHelloPkt);
//              showRoutingTable();
                showNeighborCtrlTbl();
                trace() << "rebrocast sink brocast from node " <<
recHelloPkt->getSource();
                double reEng =  1 - rsrcManager->getSpentEnergy() /
initEnergy;
                EhnHelloPacket *helloPacket = recHelloPkt->dup();
                helloPacket->setSource(SELF_NETWORK_ADDRESS);

                helloPacket->setSinkDistance(sinkDist);
                helloPacket->setPathReliability(rTable-
>getPathReliability(sinkID));
                helloPacket->setResEnergy(reEng);
                helloPacket->setNodeXCoor(xCoor);
                helloPacket->setNodeYCoor(yCoor);
                helloPacket->setDelay(rTable->getPathDelay(sinkID));
                helloPacket->setDeviceLevel(deviceLevel);
                helloCounter++;
                toMacLayer(helloPacket,
BROADCAST_MAC_ADDRESS);
            }
        } else {
            trace () << "Received Packet from MAC error in EhnPBRouting"
<< "source: " << srcMacAddress;
        }
    }
}

void EhnPBRouting::timerFiredCallback(int timerIndex)
{
    switch (timerIndex) {
        case HELLO_TIMEOUT:{
            sendHelloPacket();
            break;
        }
        case UPDATE_ROUTING_TABLE:{
//          showRoutingTable();
//          rTable->setDPDelay(mOQueue->getDPDelay());
            rTable->updateRoutingTable();

//          showTempTable();
//          double aD = rTable->calAvDistance();
//          trace() << "av distancd: " << aD;
//          trace() << "next hop: " << rTable->calAvEnergy(aD);
//          showNeighborCtrlTbl();
            showRoutingTable();
//          showNeighborCtrlTbl();
            setTimer(UPDATE_ROUTING_TABLE,upRTableTimeout);
            break;
        }
    }
}

void EhnPBRouting::showRoutingTable()
{
    routingRecord rRec;
    trace() << "     Routing Table ";
    for(int i = 0; i < (int)rTable->getSize(); i++) {
        rRec = rTable->showRoutingTbl(i);
        trace() << "Sink " << rRec.sinkID << " Next Hop: " <<
rRec.EAGFNextHop << " Dist: " << rRec.distance
        << " X: " << rRec.sinkXCoor << " Y: " << rRec.sinkYCoor;
        trace() << " reliability: " << rRec.totReliability << " " <<
rRec.secReliability << " " << rRec.topReliability
            << " " << rRec.relCtrlNextHop1 << " " << rRec.relCtrlNextHop2
<< " " << rRec.relCtrlNextHop3;
        trace() << " delay: " << rRec.pathDelay << " next hop: " <<
rRec.delayCtrlNextHop;
        trace() << " cr reliability " << rRec.crReliability << " cr delay: " <<
rRec.crDelay << " " << rRec.crNextHop1
        << " " << rRec.crNextHop2  << " " << rRec.crNextHop3;
    }
}

void EhnPBRouting::handleMacControlMessage(cMessage *msg)
{
    MacLayerMessage *macMsg = dynamic_cast <MacLayerMessage*>(msg);
    int msgKind = 999;
    if (macMsg) {
        msgKind = macMsg->getMacLayerMessageKind();
    }
    switch (msgKind) {
        case ACK_NOTIFICATION: {
            int hopNum = macMsg->getSrcID();
            int sinkID = macMsg->getSinkID();
            int seqNumber = macMsg->getPktSeqNumber();
            trace() <<"get ACK from MAC layer from " << hopNum << " "
<< sinkID << " seq " << seqNumber;

            // update reliability table
            int temp = rTable->updateRelTableACK(sinkID,hopNum);
            if (temp == 999)
                trace() << "got unexpected ACK packet";
            showNeighborCtrlTbl();
            break;
        }

        case SENT_TIME_NOTIFICATION: {
            int rValue;
            int pktClass = macMsg->getPktClass();
            trace() << "SENT_TIME_NOTIFICATION for class " << pktClass
<< " packet" << " seq: " << macMsg->getPktSeqNumber();
            switch (pktClass) {
                case CP: {
                    rValue = mOQueue->calCPDelay(macMsg-
>getPktSeqNumber(),macMsg->getSentTime());
                    if (rValue == 999) {
                        trace() << "Calculate CP Packet Delay error,
Seq: " << macMsg->getPktSeqNumber();
                    }
                    trace() << "CP packet delay: " << mOQueue-
>getCPDelay();
                    break;
                }
                case DP: {
                    trace() << "got sent time notification, sent time = "
<< macMsg->getSentTime();
                    rValue = mOQueue->calDPDelay(macMsg-
>getPktSeqNumber(),macMsg->getSentTime());
                    if (rValue == 999) {
                        trace() << "Calculate DP Packet Delay error,
Seq: " << macMsg->getPktSeqNumber();
                    }
                    rTable->setDPDelay(mOQueue->getDPDelay());
                    trace() << "DP packet delay: " << mOQueue-
>getDPDelay();
                    break;
                }
                case RP: {
                    rValue = mOQueue->calRPDelay(macMsg-
>getPktSeqNumber(),macMsg->getSentTime());
                    if (rValue == 999) {
                        trace() << "Calculate RP Packet Delay error,
Seq: " << macMsg->getPktSeqNumber();
                    }
                    trace() << "RP packet delay: " << mOQueue-
>getRPDelay();
                    break;
                }
                case OP: {
                    rValue = mOQueue->calOPDelay(macMsg-
>getPktSeqNumber(),macMsg->getSentTime());
                    if (rValue == 999) {
                        trace() << "Calculate OP Packet Delay error,
Seq: " << macMsg->getPktSeqNumber();
                    }
                    trace() << "OP packet delay: " << mOQueue-
>getOPDelay();
                    break;
                }
            }
            break;
        }
```

```
    }
}

int EhnPBRouting::lookfrNextHop(PBRoutingPacket * netPacket)
{
    int classfier = netPacket->getPktClassfier();

    switch(classfier) {
        case RP:{
            int destinationID = atoi(netPacket->getDestination());
            if(destinationID != 999) {
                int nh = rTable->getReliabilityNxHop(destinationID);
                if (nh == 999)
                    trace() << "no RP destination : " << destinationID;
                else {
                    netPacket->setNextHop(nh);
                    mOQueue-
>enqMulObjQosQ(netPacket,SIMTIME_DBL(getClock()));
                    trace() << "RP next hop : " << netPacket-
>getNextHop();
                }
                break;
            }

            float rRel = netPacket->getReliability();
            trace() << "reliability required: " << rRel;
            vector<int> nextHopIDs = rTable->reliabilityNextHop(rRel);
            vector<int>::iterator it;
            it = nextHopIDs.begin();
            if (*it == 999){
                trace() << "No reliable next hop, drop packet";
            }
            else {
                pktSeqNumber--;
                int destinationID;
                do {
                    if ((*it) == 998) {
                        it++;
                        destinationID = (*it);
                        it++;
                    }

                    do {
                        PBRoutingPacket* RPPacket = netPacket-
>dup();
                        RPPacket->setNextHop(*it);
                        stringstream st;
                        st << destinationID;
                        string destIDStr = st.str();

                        RPPacket-
>setDestination(destIDStr.c_str());

                        RPPacket-
>setSeqNumber(++pktSeqNumber);
                        trace() << "reliability ctrl next hop: " <<
RPPacket->getNextHop() << " destination: " << RPPacket->getDestination();
                        mOQueue-
>enqMulObjQosQ(RPPacket,SIMTIME_DBL(getClock()));
                        it++;
                    }
                    while (((*it) != 998) && (it != nextHopIDs.end()));

                }
                while (it < nextHopIDs.end());
            }
            break;
        } case CP:{
            int destinationID = atoi(netPacket->getDestination());
            if(destinationID != 999) {
//              EAGF(netPacket);
                trace() << "CP destination : " << destinationID;
                break;
            }

            float rRel = netPacket->getReliability();
            trace() << "reliability required: " << rRel;
            vector<int> nextHopIDs = rTable->criticalNextHop(netPacket-
>getLifeTime(),rRel);
            vector<int>::iterator it;
            it = nextHopIDs.begin();
            if (*it == 999){
                trace() << "No critical next hop, drop packet";
            }
            else {
                pktSeqNumber--;
                int destinationID;
                do {
                    if (*it == 998) {
                        it++;
                        destinationID = *it;
                        it++;
                    }

                    do {
                        PBRoutingPacket* CPPacket = netPacket-
>dup();
                        CPPacket->setNextHop(*it);

                        stringstream st;
                        st << destinationID;
                        string destIDStr = st.str();

                        CPPacket-
>setDestination(destIDStr.c_str());

                        CPPacket-
>setSeqNumber(++pktSeqNumber);
                        trace() << "critical ctrl next hop: " <<
CPPacket->getNextHop() << " destination: " << CPPacket->getDestination();
                        mOQueue-
>enqMulObjQosQ(CPPacket,SIMTIME_DBL(getClock()));
                        it++;
                    }
                    while ((*it != 998) && (it != nextHopIDs.end()));

                }
                while (it < nextHopIDs.end());
            }
            break;

        }
        case DP:{
            int destinationID = atoi(netPacket->getDestination());
            if(destinationID != 999) {
                int nh = rTable->getDelayNxHop(destinationID,netPacket-
>getLifeTime());
                if (nh == 999)
                    trace() << "no DP destination : " << destinationID;
                else if (nh == 998) {
                    trace() << "DP times out";
                    otCounter++;
                }
                else {
                    netPacket->setNextHop(nh);
                    mOQueue-
>enqMulObjQosQ(netPacket,SIMTIME_DBL(getClock()));
```

```
                    trace() << "DP next hop : " << netPacket-
>getNextHop();
                }
//              break;
            }
            else
                trace() << "DP next hop error ";

//          vector<int> nHop = rTable->delayNextHop(netPacket-
>getLifeTime());
//          int nextHop = nHop.front();
//          trace() << "processing DP packet with Destination: " <<
nHop.back() << " lifetime: " << netPacket->getLifeTime()
//          << " delay: " << mOQueue->getDPDelay() << " next hop: " <<
nextHop;

//          if (nextHop == 999){
//              trace() << "no next hop for DP";
//          }
//          else if (nextHop == 998) {
//              trace() << "DP time out";
//              otCounter++;
//          }
//          else {
//              stringstream st;
//              int destIDInt = nHop.back();
//              st << destIDInt;
//              string destIDStr = st.str();
//              netPacket->setDestination(destIDStr.c_str());
//              netPacket->setNextHop(nextHop);

//              mOQueue-
>enqMulObjQosQ(netPacket,SIMTIME_DBL(getClock()));
//          }

            break;
        }
        case OP:{
            int destination = atoi(netPacket->getDestination());
            vector<int> nextHop = rTable->EAGFNextHop(destination);
            trace() << "get next hop from routing table: node " <<
nextHop.front() << " destination: " << nextHop.back();
            if ((int)nextHop.front() == 999)
                trace() << "NO EAGF next hop";
            else {
                int destID = (int)nextHop.back();
                stringstream st;
                st << destID;
                string destIDStr = st.str();
//              trace() << "set destination ID: " << destIDStr << " " <<
destID;
                netPacket->setNextHop((int)nextHop.front());
                netPacket->setDestination(destIDStr.c_str());

//              trace() << "enq packet with seq Num: " << netPacket-
>getSeqNumber();
                mOQueue-
>enqMulObjQosQ(netPacket,SIMTIME_DBL(getClock()));
            }
            break;
        }
    }
    return 1;
}

void EhnPBRouting::showTempTable()
{
    list<tempRecord> tTable = rTable->getTempTbl();
    list<tempRecord>::iterator it;

    for(it = tTable.begin(); it != tTable.end(); it++) {
        trace() << "temp table: " << (*it).nodeID << " " << (*it).distance << " "
<< (*it).engLvl;
    }
}

void EhnPBRouting::showNeighborCtrlTbl()
{
    vector<neighborCtrlRec> nbCtrlTbl = rTable->getNbCtrlTable();
    vector<neighborCtrlRec>::iterator it;
    for(it = nbCtrlTbl.begin(); it != nbCtrlTbl.end(); it++) {
        trace() << "nb ctrl tbl: " << (*it).sinkID << " " << (*it).neighborID << " "
<< (*it).neighborDistance << " " << (*it).sinkDistance
        << " " << (*it).engLevel << " " << (*it).pathDelay
        << " " << (*it).ACKCounter << " " << (*it).txCounter << " " <<
(*it).linkReliability << " " << (*it).pathReliability;
    }
}


void EhnPBRouting::processBufferPacket()
{
    while(!mOQueue->empty()) {
        PBRoutingPacket *tempPacket = mOQueue->deqMulObjQosQ();

        if(tempPacket){
            trace() << "processing buffer packet from node " << tempPacket-
>getSource() << " seq " << tempPacket->getSeqNumber();
            int destinationID = atoi(tempPacket->getDestination());
            int nxHop = tempPacket->getNextHop();

            toMacLayer(tempPacket, nxHop);

            // for the estimation of link reliability
//          relCtrlTbl[nxHop].txCounter++;
            rTable->updateRelTableTx(destinationID,nxHop);
            mOQueue->delElement(tempPacket->getPktClassfier());
//          showNeighborCtrlTbl();
        }
        else {
            trace() << "Dequeue Error";
        }
    }
}

void EhnPBRouting::finishSpecific()
{
    declareOutput("Packet sent");
    collectOutput("Packet sent", "CP", CPPacketCounter);
    collectOutput("Packet sent", "DP", DPPacketCounter);
    collectOutput("Packet sent", "RP", RPPacketCounter);
    collectOutput("Packet sent", "OP", OPPacketCounter);

    declareOutput("Packet received");
    collectOutput("Packet received", "CP", recCPPktCounter);
    collectOutput("Packet received", "DP", recDPPktCounter);
    collectOutput("Packet received", "RP", recRPPktCounter);
    collectOutput("Packet received", "OP", recOPPktCounter);


    declareOutput("Packet forwarded");
    collectOutput("Packet forwarded", "CP", frCPPktCounter);
    collectOutput("Packet forwarded", "DP", frDPPktCounter);
    collectOutput("Packet forwarded", "RP", frRPPktCounter);
    collectOutput("Packet forwarded", "OP", frOPPktCounter);

    declareOutput("Packet droped");
    collectOutput("Packet droped", "DPOverTime", otCounter);
```

```
//          declareOutput("Hello packet");
            collectOutput("Packet droped", "Hello packet", helloCounter);
}
```

B.2. Routing protocol module (EhnPBRouting.h)

```
/*******************************************************************
* Routing protocol module  (header file)                           *
* OMNeT++ based simulator Castalia 3.2 is used for simulations     *
* Castalia website: http://castalia.research.nicta.com.au/index.php/en/   *
* By Zahoor A. Khan                                                */
/*******************************************************************
*//EhnPBRouting.h                                                  *
********************************************************************/

#ifndef _EHNPBROUTING_H_
#define _EHNPBROUTING_H_

#include <map>
#include "VirtualRouting.h"
#include "EhnHelloPacket_m.h"
#include "ProjectBanPacket_m.h"
#include "PBRoutingPacket_m.h"
#include "MacLayerMessage_m.h"
#include "EhnRoutingTable.h"
#include "MulObjQosQ.h"

using namespace std;

enum PacketClass {
	CP = 1,
	DP = 2,
	RP = 3,
	OP = 4,
};

enum HelloTimers {
	HELLO_TIMEOUT = 1,
	UPDATE_ROUTING_TABLE = 2,
};


class EhnPBRouting: public VirtualRouting {
private:
	double helloTimeout;
	double upRTableTimeout;
	bool sink;
	bool mdc;
	int pbRoutingFrameOverhead;
	double avSmthFac;
```

```
        int msrWin;
        EhnRoutingTable *rTable;
        MulObjQosQ *mOQueue;
        int xCoor;
        int yCoor;
        ResourceManager * rsrcManager;
        double initEnergy;
        int deviceLevel;
        int OPPacketCounter;
        int RPPacketCounter;
        int DPPacketCounter;
        int CPPacketCounter;
        int recCPPktCounter;
        int recDPPktCounter;
        int recRPPktCounter;
        int recOPPktCounter;
        int frCPPktCounter;
        int frDPPktCounter;
        int frRPPktCounter;
        int frOPPktCounter;
        int helloCounter;
        int otCounter;
        int pktSeqNumber;


protected:
        void startup();
        void handleMacControlMessage(cMessage *msg);
        void fromApplicationLayer(cPacket *, const char *);
        void fromMacLayer(cPacket *, int, double, double);
        void processBufferPacket();
        void timerFiredCallback(int);
        void initRoutingTable();
        void initMultiObjQ();
        void sendHelloPacket();
        void showRoutingTable();
        void showNeighborCtrlTbl();
        void showTempTable();
        int lookfrNextHop(PBRoutingPacket *);
        void finishSpecific();
};

#endif                              //EHNPBROUTINGMODULE
```

B.3. Routing protocol module (ehnHelloPacket_m.cc)

```
/*******************************************************************
* Routing protocol module                                          *
* OMNeT++ based simulator Castalia 3.2 is used for simulations     *
* Castalia website: http://castalia.research.nicta.com.au/index.php/en/  *
* By Zahoor A. Khan                                               */
/*******************************************************************
*//ehnHelloPacket_m.cc                                             *
********************************************************************/

// Generated file, do not edit! Created by opp_msgc 4.1 from
src/node/communication/routing/ehnPBRouting/EhnHelloPacket.msg.
//

// Disable warnings about unused variables, empty switch stmts, etc:
#ifdef _MSC_VER
#  pragma warning(disable:4101)
#  pragma warning(disable:4065)
#endif

#include <iostream>
#include <sstream>
#include "EhnHelloPacket_m.h"

// Template rule which fires if a struct or class doesn't have operator<<
template<typename T>
std::ostream& operator<<(std::ostream& out,const T&) {return out;}

// Another default rule (prevents compiler from choosing base class' doPacking())
template<typename T>
void doPacking(cCommBuffer *, T& t) {
   throw cRuntimeError("Parsim error: no doPacking() function for type %s or its base
class (check .msg and _m.cc/h files!)",opp_typename(typeid(t)));
}

template<typename T>
void doUnpacking(cCommBuffer *, T& t) {
   throw cRuntimeError("Parsim error: no doUnpacking() function for type %s or its base
class (check .msg and _m.cc/h files!)",opp_typename(typeid(t)));
}



Register_Class(EhnHelloPacket);
```

```
EhnHelloPacket::EhnHelloPacket(const char *name, int kind) :
RoutingPacket(name,kind)
{
   this->sinkID_var = 0;
   this->pathReliability_var = 0;
   this->sinkXCoor_var = 0;
   this->sinkYCoor_var = 0;
   this->nodeXCoor_var = 0;
   this->nodeYCoor_var = 0;
   this->resEnergy_var = 0;
   this->sinkDistance_var = 0;
   this->delay_var = 0;
   this->deviceLevel_var = 0;
}

EhnHelloPacket::EhnHelloPacket(const EhnHelloPacket& other) : RoutingPacket()
{
   setName(other.getName());
   operator=(other);
}

EhnHelloPacket::~EhnHelloPacket()
{
}

EhnHelloPacket& EhnHelloPacket::operator=(const EhnHelloPacket& other)
{
   if (this==&other) return *this;
   RoutingPacket::operator=(other);
   this->sinkID_var = other.sinkID_var;
   this->pathReliability_var = other.pathReliability_var;
   this->sinkXCoor_var = other.sinkXCoor_var;
   this->sinkYCoor_var = other.sinkYCoor_var;
   this->nodeXCoor_var = other.nodeXCoor_var;
   this->nodeYCoor_var = other.nodeYCoor_var;
   this->resEnergy_var = other.resEnergy_var;
   this->sinkDistance_var = other.sinkDistance_var;
   this->delay_var = other.delay_var;
   this->deviceLevel_var = other.deviceLevel_var;
   return *this;
}

void EhnHelloPacket::parsimPack(cCommBuffer *b)
{
   RoutingPacket::parsimPack(b);
```

```
    doPacking(b,this->sinkID_var);
    doPacking(b,this->pathReliability_var);
    doPacking(b,this->sinkXCoor_var);
    doPacking(b,this->sinkYCoor_var);
    doPacking(b,this->nodeXCoor_var);
    doPacking(b,this->nodeYCoor_var);
    doPacking(b,this->resEnergy_var);
    doPacking(b,this->sinkDistance_var);
    doPacking(b,this->delay_var);
    doPacking(b,this->deviceLevel_var);
}

void EhnHelloPacket::parsimUnpack(cCommBuffer *b)
{
    RoutingPacket::parsimUnpack(b);
    doUnpacking(b,this->sinkID_var);
    doUnpacking(b,this->pathReliability_var);
    doUnpacking(b,this->sinkXCoor_var);
    doUnpacking(b,this->sinkYCoor_var);
    doUnpacking(b,this->nodeXCoor_var);
    doUnpacking(b,this->nodeYCoor_var);
    doUnpacking(b,this->resEnergy_var);
    doUnpacking(b,this->sinkDistance_var);
    doUnpacking(b,this->delay_var);
    doUnpacking(b,this->deviceLevel_var);
}

int EhnHelloPacket::getSinkID() const
{
    return sinkID_var;
}

void EhnHelloPacket::setSinkID(int sinkID_var)
{
    this->sinkID_var = sinkID_var;
}

float EhnHelloPacket::getPathReliability() const
{
    return pathReliability_var;
}

void EhnHelloPacket::setPathReliability(float pathReliability_var)
{
    this->pathReliability_var = pathReliability_var;
}
```

```
int EhnHelloPacket::getSinkXCoor() const
{
   return sinkXCoor_var;
}

void EhnHelloPacket::setSinkXCoor(int sinkXCoor_var)
{
   this->sinkXCoor_var = sinkXCoor_var;
}

int EhnHelloPacket::getSinkYCoor() const
{
   return sinkYCoor_var;
}

void EhnHelloPacket::setSinkYCoor(int sinkYCoor_var)
{
   this->sinkYCoor_var = sinkYCoor_var;
}

int EhnHelloPacket::getNodeXCoor() const
{
   return nodeXCoor_var;
}

void EhnHelloPacket::setNodeXCoor(int nodeXCoor_var)
{
   this->nodeXCoor_var = nodeXCoor_var;
}

int EhnHelloPacket::getNodeYCoor() const
{
   return nodeYCoor_var;
}

void EhnHelloPacket::setNodeYCoor(int nodeYCoor_var)
{
   this->nodeYCoor_var = nodeYCoor_var;
}

double EhnHelloPacket::getResEnergy() const
{
   return resEnergy_var;
}
```

```
void EhnHelloPacket::setResEnergy(double resEnergy_var)
{
   this->resEnergy_var = resEnergy_var;
}

double EhnHelloPacket::getSinkDistance() const
{
   return sinkDistance_var;
}

void EhnHelloPacket::setSinkDistance(double sinkDistance_var)
{
   this->sinkDistance_var = sinkDistance_var;
}

double EhnHelloPacket::getDelay() const
{
   return delay_var;
}

void EhnHelloPacket::setDelay(double delay_var)
{
   this->delay_var = delay_var;
}

int EhnHelloPacket::getDeviceLevel() const
{
   return deviceLevel_var;
}

void EhnHelloPacket::setDeviceLevel(int deviceLevel_var)
{
   this->deviceLevel_var = deviceLevel_var;
}

class EhnHelloPacketDescriptor : public cClassDescriptor
{
  public:
   EhnHelloPacketDescriptor();
   virtual ~EhnHelloPacketDescriptor();

   virtual bool doesSupport(cObject *obj) const;
   virtual const char *getProperty(const char *propertyname) const;
   virtual int getFieldCount(void *object) const;
   virtual const char *getFieldName(void *object, int field) const;
   virtual int findField(void *object, const char *fieldName) const;
   virtual unsigned int getFieldTypeFlags(void *object, int field) const;
   virtual const char *getFieldTypeString(void *object, int field) const;
   virtual const char *getFieldProperty(void *object, int field, const char *propertyname)
const;
   virtual int getArraySize(void *object, int field) const;

   virtual std::string getFieldAsString(void *object, int field, int i) const;
   virtual bool setFieldAsString(void *object, int field, int i, const char *value) const;

   virtual const char *getFieldStructName(void *object, int field) const;
   virtual void *getFieldStructPointer(void *object, int field, int i) const;
};

Register_ClassDescriptor(EhnHelloPacketDescriptor);

EhnHelloPacketDescriptor::EhnHelloPacketDescriptor() :
cClassDescriptor("EhnHelloPacket", "RoutingPacket")
{
}

EhnHelloPacketDescriptor::~EhnHelloPacketDescriptor()
{
}

bool EhnHelloPacketDescriptor::doesSupport(cObject *obj) const
{
   return dynamic_cast<EhnHelloPacket *>(obj)!=NULL;
}

const char *EhnHelloPacketDescriptor::getProperty(const char *propertyname) const
{
   cClassDescriptor *basedesc = getBaseClassDescriptor();
   return basedesc ? basedesc->getProperty(propertyname) : NULL;
}

int EhnHelloPacketDescriptor::getFieldCount(void *object) const
{
   cClassDescriptor *basedesc = getBaseClassDescriptor();
   return basedesc ? 10+basedesc->getFieldCount(object) : 10;
}

unsigned int EhnHelloPacketDescriptor::getFieldTypeFlags(void *object, int field) const
{
   cClassDescriptor *basedesc = getBaseClassDescriptor();
   if (basedesc) {
      if (field < basedesc->getFieldCount(object))
```

```
            return basedesc->getFieldTypeFlags(object, field);
        field -= basedesc->getFieldCount(object);
    }
    static unsigned int fieldTypeFlags[] = {
        FD_ISEDITABLE,
        FD_ISEDITABLE,
        FD_ISEDITABLE,
        FD_ISEDITABLE,
        FD_ISEDITABLE,
        FD_ISEDITABLE,
        FD_ISEDITABLE,
        FD_ISEDITABLE,
        FD_ISEDITABLE,
        FD_ISEDITABLE,
    };
    return (field>=0 && field<10) ? fieldTypeFlags[field] : 0;
}

const char *EhnHelloPacketDescriptor::getFieldName(void *object, int field) const
{
    cClassDescriptor *basedesc = getBaseClassDescriptor();
    if (basedesc) {
        if (field < basedesc->getFieldCount(object))
            return basedesc->getFieldName(object, field);
        field -= basedesc->getFieldCount(object);
    }
    static const char *fieldNames[] = {
        "sinkID",
        "pathReliability",
        "sinkXCoor",
        "sinkYCoor",
        "nodeXCoor",
        "nodeYCoor",
        "resEnergy",
        "sinkDistance",
        "delay",
        "deviceLevel",
    };
    return (field>=0 && field<10) ? fieldNames[field] : NULL;
}

int EhnHelloPacketDescriptor::findField(void *object, const char *fieldName) const
{
    cClassDescriptor *basedesc = getBaseClassDescriptor();
    int base = basedesc ? basedesc->getFieldCount(object) : 0;
    if (fieldName[0]=='s' && strcmp(fieldName, "sinkID")==0) return base+0;
    if (fieldName[0]=='p' && strcmp(fieldName, "pathReliability")==0) return base+1;
    if (fieldName[0]=='s' && strcmp(fieldName, "sinkXCoor")==0) return base+2;
    if (fieldName[0]=='s' && strcmp(fieldName, "sinkYCoor")==0) return base+3;
    if (fieldName[0]=='n' && strcmp(fieldName, "nodeXCoor")==0) return base+4;
    if (fieldName[0]=='n' && strcmp(fieldName, "nodeYCoor")==0) return base+5;
    if (fieldName[0]=='r' && strcmp(fieldName, "resEnergy")==0) return base+6;
    if (fieldName[0]=='s' && strcmp(fieldName, "sinkDistance")==0) return base+7;
    if (fieldName[0]=='d' && strcmp(fieldName, "delay")==0) return base+8;
    if (fieldName[0]=='d' && strcmp(fieldName, "deviceLevel")==0) return base+9;
    return basedesc ? basedesc->findField(object, fieldName) : -1;
}

const char *EhnHelloPacketDescriptor::getFieldTypeString(void *object, int field) const
{
    cClassDescriptor *basedesc = getBaseClassDescriptor();
    if (basedesc) {
        if (field < basedesc->getFieldCount(object))
            return basedesc->getFieldTypeString(object, field);
        field -= basedesc->getFieldCount(object);
    }
    static const char *fieldTypeStrings[] = {
        "int",
        "float",
        "int",
        "int",
        "int",
        "int",
        "double",
        "double",
        "double",
        "int",
    };
    return (field>=0 && field<10) ? fieldTypeStrings[field] : NULL;
}

const char *EhnHelloPacketDescriptor::getFieldProperty(void *object, int field, const
char *propertyname) const
{
    cClassDescriptor *basedesc = getBaseClassDescriptor();
    if (basedesc) {
        if (field < basedesc->getFieldCount(object))
            return basedesc->getFieldProperty(object, field, propertyname);
        field -= basedesc->getFieldCount(object);
    }
    switch (field) {
        default: return NULL;
    }
}

int EhnHelloPacketDescriptor::getArraySize(void *object, int field) const
{
    cClassDescriptor *basedesc = getBaseClassDescriptor();
    if (basedesc) {
        if (field < basedesc->getFieldCount(object))
            return basedesc->getArraySize(object, field);
        field -= basedesc->getFieldCount(object);
    }
    EhnHelloPacket *pp = (EhnHelloPacket *)object; (void)pp;
    switch (field) {
        default: return 0;
    }
}

std::string EhnHelloPacketDescriptor::getFieldAsString(void *object, int field, int i)
const
{
    cClassDescriptor *basedesc = getBaseClassDescriptor();
    if (basedesc) {
        if (field < basedesc->getFieldCount(object))
            return basedesc->getFieldAsString(object,field,i);
        field -= basedesc->getFieldCount(object);
    }
    EhnHelloPacket *pp = (EhnHelloPacket *)object; (void)pp;
    switch (field) {
        case 0: return long2string(pp->getSinkID());
        case 1: return double2string(pp->getPathReliability());
        case 2: return long2string(pp->getSinkXCoor());
        case 3: return long2string(pp->getSinkYCoor());
        case 4: return long2string(pp->getNodeXCoor());
        case 5: return long2string(pp->getNodeYCoor());
        case 6: return double2string(pp->getResEnergy());
        case 7: return double2string(pp->getSinkDistance());
        case 8: return double2string(pp->getDelay());
        case 9: return long2string(pp->getDeviceLevel());
        default: return "";
    }
}

bool EhnHelloPacketDescriptor::setFieldAsString(void *object, int field, int i, const char
*value) const
{
    cClassDescriptor *basedesc = getBaseClassDescriptor();
    if (basedesc) {
        if (field < basedesc->getFieldCount(object))
            return basedesc->setFieldAsString(object,field,i,value);
        field -= basedesc->getFieldCount(object);
    }
    EhnHelloPacket *pp = (EhnHelloPacket *)object; (void)pp;
    switch (field) {
        case 0: pp->setSinkID(string2long(value)); return true;
        case 1: pp->setPathReliability(string2double(value)); return true;
        case 2: pp->setSinkXCoor(string2long(value)); return true;
        case 3: pp->setSinkYCoor(string2long(value)); return true;
        case 4: pp->setNodeXCoor(string2long(value)); return true;
        case 5: pp->setNodeYCoor(string2long(value)); return true;
        case 6: pp->setResEnergy(string2double(value)); return true;
        case 7: pp->setSinkDistance(string2double(value)); return true;
        case 8: pp->setDelay(string2double(value)); return true;
        case 9: pp->setDeviceLevel(string2long(value)); return true;
        default: return false;
    }
}

const char *EhnHelloPacketDescriptor::getFieldStructName(void *object, int field) const
{
    cClassDescriptor *basedesc = getBaseClassDescriptor();
    if (basedesc) {
        if (field < basedesc->getFieldCount(object))
            return basedesc->getFieldStructName(object, field);
        field -= basedesc->getFieldCount(object);
    }
    static const char *fieldStructNames[] = {
        NULL,
        NULL,
        NULL,
        NULL,
        NULL,
        NULL,
        NULL,
        NULL,
        NULL,
        NULL,
    };
    return (field>=0 && field<10) ? fieldStructNames[field] : NULL;
}

void *EhnHelloPacketDescriptor::getFieldStructPointer(void *object, int field, int i)
const
```

```
{
   cClassDescriptor *basedesc = getBaseClassDescriptor();
   if (basedesc) {
      if (field < basedesc->getFieldCount(object))
         return basedesc->getFieldStructPointer(object, field, i);
      field -= basedesc->getFieldCount(object);
   }
   EhnHelloPacket *pp = (EhnHelloPacket *)object; (void)pp;
   switch (field) {
      default: return NULL;
   }
}
```