Journal of Harbin Engineering University
ISSN: 1006-7043

Vol 46 No. 04
April 2025

Architecture Design for Mode Based Packet Distribution in Multi-Interface Wireless Mesh Network

Sanjeev Thapa[1*], Madhu Sudan Kayastha[1,2]

[1]School of Engineering, Pokhara University, Pokhara-30, 33700, Nepal

[2]Kathmandu Institute of Applied Science, Lalitpur, Nepal

Abstract

Introduction: In this work, an architecture enabled by a multi-interface wireless network based on IEEE802.11g NIC ad-hoc mode built-in OMNeT++/ INETMANeT is designed and then implemented in various Load Balancing (LB) algorithms for packet transmission. The designed simulation positioned an intermediate layer (i.e. cross layer 2.5) between the second and third layers, as defined by the OSI reference model.

Objectives: designing cross-layer architecture, as an intermediate layer in the node architecture between layer 2 and 3, for packet handling in the WMN.

Methods: A new module named packet SCHEDULER, which is a generic C++ structure able to implement MAC mapping of neighbouring host interfaces, and distribute the traffic (i.e. LB) in multi-hop mesh networks, collaborating the implemented routing protocol with two or more interfaces built-in in each node. The symmetric link in a Wireless Mesh Network (WMN) was developed by adopting the Optimized Link State Routing (OLSR) protocol, which was implemented to find the route in the network. Performance parameters such as throughput/end-to-end delay/PER for all LB modes were compared. The results obtained using the node architecture with two interfaces, three interfaces, and a multi-hop single-interface WMN (SI WMN) were analysed.

Results: Implemented LB algorithms in a Multi-interface WMN (MI WMN) enhance the route between all traffic and improve the capacity of links over the mesh networks. An approach to distribute packets in the defined scheme in Multi-interface Multichannel (MIMC) WMN is made possible by designing cross-layer architecture, an intermediate layer named SCHEDULER in the node architecture.

Conclusions: The working of the cross layer justifies the improvement in aggregate capacity and overall performance with an increase in the number of interfaces. The evaluations based on performance measurement have shown that, the packets handling is possible in mesh networks by designing an intermediate layer.

Keywords: Cross-layer, End to end delay, Load balancing, OMNeT++, INETMANeT, Wireless mesh network

1. Introduction

Wireless mesh networks (WMN), which are a unique type of static ad-hoc network, are an alternative to common infrastructure networks. In this example, the network has no routers or access points. Ad hoc networks, like this one, view routing as a collaborative endeavour in which each node assists in directing traffic around other nodes. When there is no fixed network infrastructure, each network node is willing to send packets to other network nodes, even if they don't want to [1].

The quality performance of the entire WMN is based on its radio resource management (i.e., routing, transmission power, and channel assignment) [1],[2],[5],[7]. To enhance performance in WMNs, nodes are equipped with multiple radios that allow the use of multiple channels to increase the system throughput unless some special-purpose load balancing (LB) modes (like fall-back) are implemented. Routing in wireless ad hoc networks is the process of sending data from a source to a destination across the network. The routing protocol is a way for routers to share routing information with one another and for routing to be implemented. It enables routers to retain their own routing tables and share routing information. At the network layer, the routing

protocol is in charge of gathering information about the network's present condition and determining the best transmission path [2].

Since the amount of data a link can deliver in a limited period is the most important issue in WMN, adopting Multi-interface Multi-channel (MIMC) in WMN can enhance the capacity. Further, various Channel Assignment (CA) algorithms can also be deployed to better distribute the channels and interfaces in MIMC networks [4], [25]. Mesh networks are single-channel by nature, and MIMC schemes are not standardized yet. Though a few emulation testbeds and simulation frameworks have been developed to investigate WMNs, they are still not sufficient to handle the multiple interfaces needed to fulfill the research needs of MIMC networks [4], [7]. Hence, the goal of designing a simulation framework for LB (packet routing) was decided to be built into the OMNeT++/INETMANeT framework. Then, to validate the cross-layer design, the performance of the proposed LB modes of MI-WMNs was compared with the SI WMN.

1.1 Background of Study

Before deployment on the real network, the architectures are designed and implemented in a network simulator. The simulation framework designed with a cross-layer is expected to provide balanced packet distribution within a defined protocol to meet the targeted goals of the research work. So, to achieve the below-listed goals, some modifications have been adopted into the INETMANeT framework of the OMNeT++ simulator used:

- The first goal is to design an MIMC WMN in an INETMANeT framework based on IEEE802.11 ad-hoc mode consisting of mesh nodes that are equipped with multiple IEEE802.11g NICs.
- Implement different standardized LB modes such as Round Robin (with two interfaces and three interfaces), Fallback, and QoS into the designed MIMC WMN simulation. An intermediate layer (i.e. cross layer 2.5) scheduler is inserted between layer-2 and layer-3.
- Finally, analyze and evaluate the multi-hop scheduling performance in load-balanced MIMC WMN and compare the results with multi-hop SI WMN.

Load Balancing (LB) and Packet Distribution:

Load in the communication system is defined in terms of channel occupancy. The LB is a process of improving the performance of a system through a redistribution of loads or packets among the multiple interfaces/channels. Several LB modes are being implemented in ad-hoc networks to cover a wide range of commercial applications. The purpose of load balancers in this work is to schedule the packets over parallel interfaces through assigned channels according to the algorithm of the model used. So, to handle the packet scheduling in WMN a scheduler module is proposed at layer 2.5.

The following special-purpose LB modes are implemented to schedule the packet transmission and distribution in MIMC WMN.

- *Round Robin (RR)*: One of the scheduling algorithms used to distribute packets in networking. The concept of algorithms is like every involved interface distributing the load equally.
- *FallBack*: Fallback scheme distributes the load in the network through a single channel (interface) as long as it doesn't face any disturbances along that path. For example, until the number of Collisions (numCollisions: a term used in simulation file) does not reach a specified value in the MAC layer the traffic distribution follows the same channel. The designed framework implements the 'numCollisions' as a parameter of disturbance that the scheduler should take into account before deciding on channel selection.
- *QoS*: The Transport Control Protocol (TCP) and User Datagram Protocol (UDP) can be used to set up end-to-end communication between the mesh clients. QoS scheme is TCP versus UDP traffic distribution in the network. For example, out of two interfaces involved one transmits TCP traffic only and the other UDP only.

WMN requires proactive discovery of the path to reduce packet delays. The OLSR is specially designed to be used in ad-hoc networks. The key concept used to reduce flooding by this protocol is that of multipoint relays (MPR). MPRs have selected nodes that forward broadcast messages during the flooding

process. The set of MPRs covers all two-hop neighbors and only the MPRs are allowed to rebroadcast flooding messages, also nodes elected as MPR keep partial Link-state information which is used for route calculation [5],[12]. Hence, OLSR works fundamentally in three steps:

- *Sensing of Neighbor*: done with the exchange of 'HELLO' message among all nodes.
- *Dissemination of Topology Control Message*: regularly broadcast invitation containing a message for link establishing.
- *Calculating Routing Table*: About 'TC' and 'HELLO' messages obtained from other neighboring nodes, with this one of it can calculate its possible shortest link routes using the Dijkstra's algorithm.

1.2 State of the Art in WMN

WMN is a special kind of ad-hoc network. Basically, due to the problems of being coherent in the heterogeneous type of hardware equipment, the demand for corporate ad-hoc networks is increasing. Lam *et al*. [14] overcome this demand by developing a multi-interface mesh test environment, by creating a software-independent operating system for mesh access. WMNs are a rapidly growing sector and motivate various deployments.

Besides the compatibility issue, several types of research have been carried out for commercialization. Yarali *et al*. [15] stated that publically available ad-hoc type mesh technics have high recovery ability and are measurable, in the issue of energy sensitivity in clog city areas. They have also compared multi-interface ad-hoc networks with a single interface, where the MIWMN approach is found with outstanding performance than the single interface, with the issue of packet delivery speed. Since WMNs offer a huge range of applicability and are therefore highly suitable for commercial applications. Hence, the focus of this research is to design a WMNs based on multiple IEEE802.11 NICs in ad-hoc mode.

Mesh network connectivity promptly improves the performance of the network like defect tolerance, LB, throughput, the efficiency of the protocol used, and cost reduction. Tzu-chich *et al*. [22] purposed a new possibility of designing a cross-structure to support routing protocol in any mesh network environment. Further, the concept shows a clear guideline for traffic routing ability with the selection of power requirements and existing interference while sending the messages in the network. But in the paper, we proposed a possibility of mode-based load (traffic) balancing with a cross-layer design. Further, Yahya *et al*. [6] purposed multi-radio interfaces in WMNs that enabled multichannel communication to enhance the throughput and minimize the delay as well as packet loss rate. The work also dealt with multi-radio multi-channel WMN to show a possibility of mode-based load balancing in MIMC WMNs.

Load Balancing in Ad-Hoc Network:

Load balancing describes the mode based load balancing like Round Robin, Fallback, and QoS, which distribute traffic in the network according to the modes of algorithms, but Rahmawan *et al*. [13] and Talooki *et al*. [16], focus on a different aspect of load distribution for traffic management and energy consumption. Traffic if not balanced might cause an excessive delay due to dropping of packets and congestions decreasing packet decreasing ratio (PDR). To balance the load, the work in paper [13] purposes a so-called load balance dynamic source routing (LBDSR) protocol that uses source routing instead of relying on routing table to perform demand-based route like in DSR while transmitting the requested packets. Whereas L. Ding, Y. Shao, and M. Li [17] have presented an idea of using a directional antenna to handle the traffic/load in ad-hoc networks.

Another paper by N. Goel *et.al*. in [18] explains an "Efficient Weighted Innovative Routing Protocol" (EWiRP) technique for load balancing in ad-hoc networks as a routing approach. As stated the load balancing is performed by choosing multiple weighted paths and bandwidth reservation to enhance the 'Throughput' abruptly. Further the statement by M. Buvana *et.al*. in [19], purposes a load balancing approach by calculating the energy of each node using an energy calculation algorithm and then analyzing the hop to hop distance to make a cluster. Secondly, listening to each adjacent node then stores the information regarding services. Finally, with the help of computing power and distance between nodes the algorithm the load balancing in the topology.

Here, as the state of the art, all the load balancing approaches previously made in ad-hoc networks are either directly implementing some suitable routing protocol or performing distance and energy calculation or following directional antenna implementation approach in order to balance the load in the network. So, this research has proposed a different approach of load balancing in MIMC WMN, which is implemented in the self-designed module between the network layer and MAC layer and used to handle the traffic with the application of various algorithms like RR, QoS, and Fallback modes of load balancing.

2. Objectives

To design a middle layer (SCHEDULER) in between the network layer and the data link layer to perform LB of packet transmission in the network topology. The data link layer is equipped with multiple IEEE802.11 NICs, which can also be manually assigned to multiple channels. So, a wireless network based on IEEE802.11 WLAN ad-hoc mode and consisting of mesh points (nodes) that performs multi-hop bidirectional forwarding between different wireless devices is to be built.

3. Architecture Design and Implementation

The main task of the module designed is to receive the packets from the network and link layer and forward it to another layer by applying a scheduling algorithm. The scheduling module acts as a load balancer and tries to utilize multiple channels available in the network. For example, RR scheduling algorithms were implemented (it means hosts assigned with a different set of channels, heterogeneous). When the packets arrive from the network layer, this module selects the interface (channel) to transmit the data based on the scheduling scheme. Now, from the available installed wireless interfaces, i.e. NIC, it sends the packet through the interface which is assigned the selected channel. Selecting different interfaces in the same host implies using different channels for transmitting data in the network. Also, if the packets are received from the link-layer it simply forwards the packets to the network layer.

3.1. System Architecture

MIMC WMN simulation framework may need the support of multi-channel and multi-radio protocols and CA protocol and lots of administrative efforts [23]. In our approach of MIMC WMN design CA is mainly fixed into the configuration file of the INETMANeT framework. The load balancing is performed by the new cross-layer module inserted between the network layer and the MAC layer.

Packet scheduling and interface bundling are the major focus of the approach. While implementing, each interface is assigned with a specific single channel, so channel/interface bundling is applicable here. Only the primary need is to equip all the nodes with at least two NICs, so that enabled 'two-1-hop' neighbors can tune both interfacing surfaces towards the single channel from each side scheduling the packets simultaneously in parallel on all the interfacing surfaces. A very simple view of the architecture of a system which is presented in Fig.1 is designed to fulfill the goal of depicting cross-layer applications to distribute packets in the multi-interface networks.

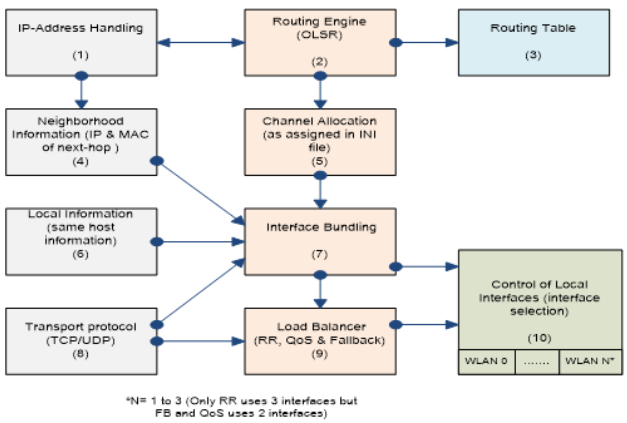

Fig. 1. Simplified overview of the system architecture designed to fulfill the load-balancing goal, which depicts a cross-layer (i.e. middle layer 2.5).

Though the system is modular and it can support any ad-hoc routing protocols and routing metrics, for the route discovery OLSR protocol is implemented, a proactive type of route protocol, so that, it floods a topology table or routing table of its neighbors to all nodes in the network and serves to compute optimal forwarding paths locally by calculating the next hop towards each destination and thus build up a routing table.

For the topology discovery, the information on the local node status [14] regarding their interface IP, corresponding next-hop IP, MAC address (4), and their state (idle/sleep/occupied) are important. The messages coming from the 1-hop neighbor (1) are further processed through the signaling information from the routing engine (2) i.e. "HELLO" message from the OLSR protocol as an output build routing table (3). Since, this research statistically implemented CA strategies in the user configuration file (5), which fixes a separate channel to all interfaces built-in a node (7), so no advanced method is defined in the scheduler module (9) to dynamically assign a channel to each local (wireless NIC) WNIC/MAC addresses, rather simply set up link via the assigned channel between hop to hop interfaces by following host to MAC mapping method (1) in scheduler module, that distributes packets towards the appropriate interfaces selected (10) according to the load definition of balancing modes implemented.

Now, with the help of local information and information from 1-hop neighborhood (6), the traffic type (TCP or UDP) (8) is processed in the network via the channel allocated in (5). Finally, in order to schedule packets with predefined load balancing modes, load balancer (9) schedules packets in each bundle, and interface bundling (7) decides about the next-hop neighbor to share information.

3.2. Node Architecture

In order to overcome the design challenges and to implement the various load balancing modes as explained above, an intermediate layer named SCHEDULER is placed between the network layer (L3) and data link layer (L2), that introduces the MIMC-WMN simulation framework to support the packet routing in the WMN. Hence, this framework defines a new node (host) structure as shown in Fig. 2a. The new node structure developed is able to handle packets in the aforementioned fashion, as defined in the scheduler module section above. The features and functionality of this new node structure are presented. The node structure can work with different IP and MAC protocols and is compatible to work and supporting the network layer of the existing INETMANeT framework.

If this new node architecture is compared with the standard node (e.g. mobile routing host) of the INETMANeT framework, the major difference in the SCHEDULER module inserted between two layers is to perform packet handling and scheduling in a predefined fashion while resolving IP and MAC addresses. In addition, this module can be recorded inside its C++ code structure as the project requirement. Further, the module can be hardcoded to coordinate the number of NICs (e.g. additional experiments with 4-interfaces for a heterogeneous set of channels) specified in the network.

Packets from the network layer are received in the SCHEDULER module and transmitted through different wireless interfaces based on the scheduling scheme. Also, packets from the lower layer are received and forwarded to the network layer for further processing. To apply different load balancing schemes, the scheduler module stores information about its neighbor nodes. This information contains a number of interfaces and their MAC address in the neighbor host. Users can configure the number of interfaces (e.g.: maxInterfaces=3, OMNeT defines at ManetRoutingBase.h, until and unless modified to other values) to be installed in mobile/fixed host and the scheduling scheme of their choices and scheduler performs its function accordingly. In the scheduler module following configuration parameters are provided which can be configured in the '.ini' file:

- *numMIRadios*: The parameter that defines the total number of wireless interfaces, i.e. NIC, in the host.
- *numMIHosts*: The parameter which defines the total number of hosts present in the network. This is used to collect information about the neighbor host.

- *schedulingAlgorithm*: This defines the scheduling algorithm. Possible values for this field are RR, Fallback, QoS, and random_channel.

- ***manetrouting.manetmanager.routingProtocol*= "OLSR"

This router module should be attached to the network layer of the node that participates in MANET routing. It contains a 'Manet Manager' which instantiates the requested routing protocol. The routing protocol should be configured to remain the same throughout the whole network. For the implementation place the above configuration statements in the written INI file inside the INETMANeT framework [11].

- *numChannels*: this parameter defines the ChanelControlExtended module defined in NED file that signifies the number of radio channels (frequencies) used in the network and also supports the multiple radio interface per host.

- *numTcpApp & numUdpApp*: this parameter defines the type of traffic to be generated in the network. The value '1' configured corresponding to the traffic app signifies the active mode of the traffic type to be generated.

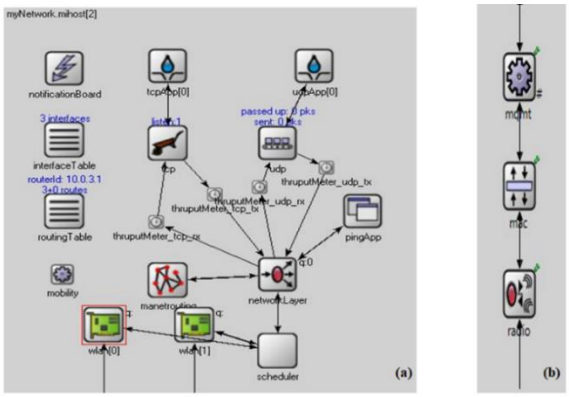

Fig.2: Node architecture of an MIMC WMN simulation (a) Node structure (b) WLAN structure

The WLAN module Fig. 2b represents the wireless NIC for sending and receiving messages. It has three submodules as shown in its internal structure; radio, 'mgmt', and MAC. Radio module works in the physical layer, modulates and demodulates transmitted packets, and processes radio signals. As seen in the WLAN structure the 'mgmt' is simply a command module to radio module, so that it can work in a specified channel. In this thesis, the module designed can work with the homogenous set (combination of 0 & 1 or 1 & 2) of channels only, for all successive nodes. MAC modules are used to implement mac protocols. In general, NICs are identified with MAC address and node/hosts are identified with their IP address although each node can have multiple IP addresses each for the interfaces assigned.

Finally, to coordinate and hence establish a link among all nodes involved in the network. The channel control module defined in the NED (network description) file of the OMNeT++ framework keeps information regarding the position and motion of the nodes, and hence makes calculations of the nodes within the communication or interference range. This information is used by the radio interfaces (NICs) of the nodes during the transmission time. Further to implement multiple radios interfaces per node channel control extended module adapted in the NED file.

3.3. Designed Architecture for Simulation of MIMC WMN

The SCHEDULER module is implemented in INETMANET with OMNET++ 4.1 simulation framework, integrated between network and link layer is developed. The module becomes active when it receives messages from the network or link layer. The flow of control inside the module is explained in Fig. 3.

The functionality of the module as explained in the process flow chart can be further explained as when the simulation starts it first collect information of all neighbor hosts. It constructs mapping of the host to MAC addresses. It contains a mapping from the host to multiple interfaces. The messages may arrive in the scheduler from the link-layer or network layer. Scheduling is performed only on the messages arriving from the network layer. When the message arrives from the network layer, it first selects the scheduling algorithm as configured by the user. This algorithm decides the next channel to use for packet transfer. If the message is the broadcast message it is forwarded to the wireless interface responsible for the channel as decided by the scheduling scheme. If the message is not broadcast type, then modification in control information is done.

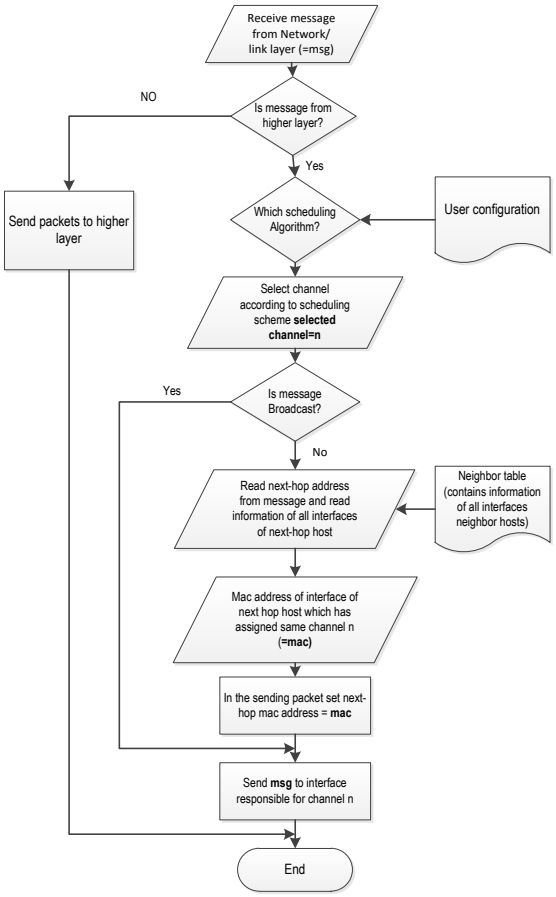

Fig. 3: Process flow chart of SCHEDULER

The first next-hop address from the Ieee802Ctrl object attached to the message is read. This is the MAC address of the interface of the next-hop host. INETMANET always set this field to the MAC address of the first interface installed in the mobile host even though there are multiple interfaces in the host. The scheduler module first finds the host from the next-hop MAC address. Then it identifies which interface in receiving host is assigned the same channel number as selected by the scheduling scheme. For instance, if the scheduler of sending host selects channel 1 to transfer data then the next-hop address field in control information of the packet is set to the MAC address of interface at the receiving side which is also assigned with channel number1. The message is now forwarded to interface with the selected channel and it will finally transmit the data to the destination.

3.3.1. Scheduling Algorithms

- *Round Robin*: Round Robin scheduling scheme distribute the load equally among available channels. If there are n interfaces in a mobile host and total *N* packets are transmitted from the host, each of *n* interfaces transmits *N/n* messages. In this way, it truly balances the traffic in the network and hence improves the throughput. The algorithm for the RR scheduling scheme is listed below:

Definition:

chnr: channel number

neighbor_map: map with key = hostname and value = mac address list.

Algorithm:

a. Initialize: chnr = 0

b. Read next-hop MAC address from message to be transmitted and find destination host.

c. Get the list of MAC addresses of all interfaces installed in the destination host (in step 2) from the neighbor map.

d. Choose destination interface which is assigned the same channel, chnr.

e. Set next hop mac address in control message (Ieee802Ctrl) of the message to be transmitted to MAC address of interface chosen from step (4).

f. Send a message through the interface which is assigned channel, chnr.

g. Increment chnr by 1.

h. If chnr > number of interfaces in host:chnr = 0;

- *Fallback:* Fallback scheduling scheme uses the same channel to transmit messages until some performance parameter is comprised. For instance, in our implementation, if the number of collisions in one of the channels exceeds the predefined value, the next available channel is selected. In this way, it may improve network reliability. The Fallback scheduling algorithm is listed below:

Definition:

Chnr: channel number

numSentArr[NUM_RADIO]: an array containing the number of messages sent through each channel

neighbor_map: map with key = hostname and value = mac address list.

Algorithm:

a. Initialize: chnr = 0.

b. Get interface which is assigned channel chnr.

c. Get the number of collisions in channel chnr.

d. Calculate the ratio, r as:

$r = (number\ of\ collision\ in\ channel, chnr)/$ (number of messages sent through the channel, (numSentArr[chnr])

e. If r < 0.1:

Goto step 6

Else

for each channel:

chnr = chnr+1;

Goto step 2.

f. Read next-hop MAC address from message to be transmitted and find destination host.

g. Get the MAC address of all interfaces installed in the destination host (in step 6) from neighbor_map.

h. Choose the destination interface which is assigned the same channel,' chnr'.

i. Set next hop mac address in control message (Ieee802Ctrl) of the message to be transmitted to the MAC address of interface chosen from step (8).

j. Send a message through the interface which is assigned channel, chnr.

k. Increment the number of messages sent through channel, chnr:

l. numSentArr[chnr] = numSentArr[chnr] +1

- *QoS (TCP vs. UDP)*: QoS scheduling scheme distributes load by selecting different channels to transmit messages based on the type of packet to transmit. It uses separate channels to transmit TCP and UDP packets. The QoS scheduling algorithm is listed below:

Definition:

chnr: channel number

neighbor_map: map with key = hostname and value = mac address list.

Algorithm:

a. Check if the message to be transmitted is TCP or UDP.

b. If TCP packet:

chnr=1;

c. If UDP packet:

chnr = 0;

d. Read next-hop MAC address from message to be transmitted and find destination host.

e. Get the MAC address of all interfaces installed in the destination host (in step 3) from neighbor_map.

f. Choose destination interface which is assigned the same channel, chnr.

g. Set next hop mac address in control message (Ieee802Ctrl) of the message to be transmitted to the mac address of interface chosen from step (5).

h. Send a message through the interface which is assigned channel, chnr.

3.4. Modifications in INETMANeT Framework

To design MIMC supporting simulation framework and hence to implement the load balancing schemes as proposed in the research, the existing INETMANeT framework has been investigated and necessary modifications are made to overcome the existing limitations in the framework which are explained below:

New method is added in;

src/Networklayer/contract/IPAddressResolver.cc:

A Module of INETMANeT framework

std::map<std::string,std::vector<MACAddress>>
IPAddressResolver::addressesOf(const char *s,int addrType)

{

std::vector<MACAddress> mac;

std::map<std::string,std::vector<MACAddress>>it_map;

This method returns MAC address of all the interfaces present in the given host.

Other method,

IPvXAddress address Of (cModule*host, int addrType), in the IPAddressResolver returns the MAC address of only one interface present in the given host. But, in multiple interface implementations, we need the MAC Address of all the interfaces built in a host. This method is a modification to the existing method in IPAddressResolver to do so. It returns the map where the key is hostname and value is the vector containing the MAC address of all interfaces in the host.

Modifications were adopted in order to implement the Fallback mode of Load Balancing to read the number of collisions from the MAC layer. Modified Modules are:

1) Networklayer/common/Interfaceentry.h

Variable: long numCollision; //to store number of collision

Methods:long getNumCollision () const

{

return numCollision ;

}

virtual void setNumCollision (long nc)

{

numCollision = nc;

}

InterfaceEntry class represent wireless interface and contains all information related to interface. Scheduler module can easily access InterfaceEntry object by following code:

InterfaceTable*ift=InterfaceTableAccess ().getIfExists ();

InterfaceEntry*e=ift->getInterfaceByName ("wlan0");

So, numCollision variable is added in InterfaceEntry class. Also, the number of collision values is available in Ieee80211egMac.cc class. When this value is changed in the MAC layer, numCollision value in InterfaceEntry is also updated. This is done in Ieee80211egMac.cc class as shown below.

2) Src/linklayer/iee80211/mac/Ieee80211egmac.cc :

In method Ieee80211egMac::handleWithFSM () following piece of code is added to the source code inside FSMA State (RECEIVE) main function to update the number of collision values in InterfaceEntry.

FSMA_State (RECEIVE)

{

//update number of collision in interface entry

IInterfaceTable*ift=InterfaceTableAccess ().getIfExists ();

InterfaceEntry*e=ift->getInterfaceByName (ifName);

e->setNumCollision(numCollision); //addition: code block end

if (fixFSM)

FinishReception ();

else

ResetStateVariables ();

}

Now, the SCHEDULER module can always read the latest value for the number of collisions. There is no other way to read the number of collisions in the Ieee80211egMac.cc class of the existing INETMANeT framework. So, this approach is found useful. It is also possible to test other parameters by applying a similar approach.

Further, for delay measurement for UDP traffic type and TCP traffic type simple modifications are performed inside the application layer (src/application/tcpApp/TCPSinkApp.cc & application/udpApp/UDPSink.cc), after which delay can be read directly from scalars of analysis file (.anf).

4. Simulation Results and Analysis

A working module by placing a cross-layer named 'SCHEDULER' in between the Data link layer and Network layer is successfully designed. Then, to verify the working of architecture designed and

perform the analysis, Grid (4×4) types of topologies have been defined with and without the introduction of background traffic. Background traffic is self-generated and easily receivable even by the secure machine, the concept that helps to create a virtual reality simulation environment, introducing some interference and noise which could be experienced in real mesh networks [20]. Then the topology is implemented with all three load balancing algorithms and results are compared. Selected test parameters of interest for measurements and hence to confirm the working of cross-layer implemented are throughput, end to end delay, and PER. Each topology is implemented with all three load balancing algorithms for packet distribution and then the results obtained are compared.

4.1. MIMC WMN Test Configuration

The WLAN Model is configured to simulate an IEEE802.11g wireless connection. The set of static parameters is fixed for the whole simulation period. The simulator used is the INETMANeT framework of OMNeT++ 4.1 version. The simulation area is defined by the playground size as set in the configuration file. Fixed-mobile MANET routing was selected using IEEE802.11g MAC protocol and UDP/TCP as a transport protocol. The study establishes using the following primary OMNeT++ parameters; simulation time of 100s, the number of channels two and number of radios two, the basic rate of data transmission is 2 Mbps for MAC, the propagation model used for radio is free space path loss, the antenna used is omnidirectional and OLSR as routing protocol, in all topology.

4.2. Measurement Procedures

The measurements are performed from the outputs of the simulation recorded inside the 'General.anf' files as scalars and vectors, displayed in the 'throughput meter'. Each individual load balancing mode (RR, FB, and QoS) are configured into the INI file. The desired LB mode is selected and the simulation is performed one after another, each last for 200s. Throughput can directly be analyzed from an average of the curve (vector) recorded in the '.anf' file (throughput meter) or directly read from the corresponding average value. For delay measurement a computation method is managed inside the 'UDPSinkApp.cc'and 'TCPSinkApp.cc' files of OMNeT++, after which the delay for UDP & TCP is recorded as scalars inside the 'General.anf' files by the variables name "Mean Delay" and "system Delay" respectively.

4.3. The Performance Metrics of Measurements

- *Throughput*: The ratio of the data transmitted without failure measured per second. Throughputs in OMNeT++ are directly read from the throughput meter placed between the network layer and application layer, which are recorded into the vector files of the '.anf' file. i.e. read average from, myNetwork.mihost [*].thruputMeter_tcp (or udp)_tx & myNetwork.mihost [*].thruputMeter_tcp(or udp)_rx.

- *Average End to end Delay*: Defined as the time taken by the packet to reach the receiver as it starts moving from the host node. The average delay of any mesh network is found by calculating the average of the total packets and all the pairs involved in transmitting and receiving sections.

The end to end delay for UDP traffic and TCP traffic simply means the time taken to process a packet from source to destination For TCP delay measurements we apply:

Delay = message reception time (simTime) – message creation time

This gives the overall system delay or the time taken by the network to send whole packets (according to the data send bytes fixed in the INI file) to the destinations. This gives the average time taken by a complete message to reach TCPSinkApp from TCPApp in implication to OMNeT++. Hence, TCP end to end delay is recorded as "System delay" (i.e. read the corresponding value to; myNetwork.mihost [*].tcpApp[0]), obtained from .anf file of OMNeT++ framework. The value is then divided by total packets received at the destination host which is also possible to read from .anf scalar output file in the myNetwork.mihost[*]. ThruputMeter _ tcp_rx.

But for UDP; UDPBasicBrustApp is used. The source type of UDP application is UDPBasicBurst. UDPBasicBurst generates many messages. Due to multiple messages received at the receiver application, the mean value need to be calculated.

Delay+ = message reception time (simTime) – message creation time

Which can be read as "Mean Delay" from the scalar outputs recorded in '.anf' file after simulation? (i.e. read value from, myNetwork.mihost[*].udpApp[0])

However, Delays always depend on the topology type and the routing protocol used.

- *The PER (Probability of error rate)*: PER is measured by calculating the ratio of the bits lost per second and the total bits transmitted per second, or from the ratio of packets lost to packets received, which are also possible to read from the recorded scalars values in the '.anf' files during the simulation time then calculate PER implying;

PER = (Bits lost per sec / Bits transmitted per sec) * 100% or

PER = (total packets lost/ Total packets received)* 100%

Since TCP is connection-oriented and UDP is connectionless, so PER for UDP is based on transmitted and received bits and TCP based on packets loss.

4.4. Result With Grid Topology

Only a few parameters are configured uniquely to realize the effect of grid topology built with 16 MI Hosts as shown in Fig. 4. The playground size of the topology is fixed to 600×500 where each node is located at a transmission range of 150 m from one another. Similarly; simulation time is set to 200 sec and send bytes for TCP is 1 Mb. These settings can be compared from the configuration files of INeTMANET framework.

Fig.4. A 4X4 grid topology simulation setup.

In order to perform the experiments, a diagonal link from host (0) to host (12) is assumed as a mainstream link. This same topology is analyzed with background traffic. The few hosts except in the mainstream are assigned with single-channel and further two TCP links are set up inside the topology. Host (4) and host (1) have one TCP link and also there is a TCP link set up between host (9) and host (5) to generate background traffic.

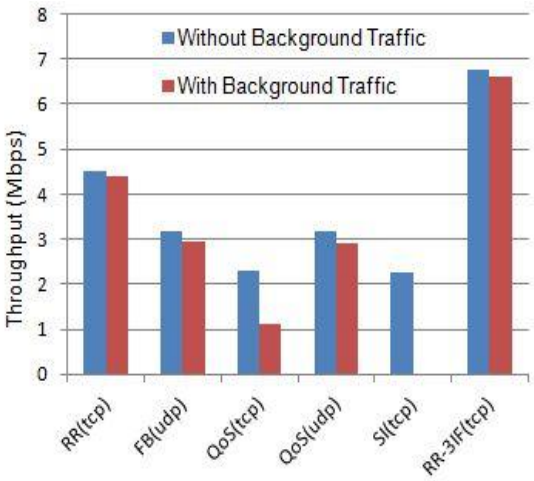

Fig.5. Throughput in a 4 x 4 grid topology.

The simulation results of both topologies with and without background traffic are later investigated for all scheduling schemes and analyzed with the help of the performance bar diagram shown in Fig. 5.

While investigating the multi-hop scenarios, the impact of interference on throughput with multiple APs should be considered. As the number of nodes in an area increases, the scope for interference between nodes also increases. The throughputs are maximum in three interfaces RR LB mode because the traffic is distributed along three alternative channels assigned. The RR 2-IF has less value of throughput than RR 3-IF, but greater than FB, QoS, and SI. In the case of FB and QoS mode, the channels assigned are two, but these schemes either use a single channel or only one type of traffic is sent through one specific channel according to the algorithm used. As a result, the throughputs lie more or less in the same range as SI mode holds, which affects the increased delay due to high collision in a single channel. Though the throughput with background traffics is slightly less than without background traffic, the nature of variation in the

value of throughputs in different LB modes looks similar.

The increase in the number of nodes increases interference and the tendencies towards packet loss also increase due to collisions. When the collision occurs, the resending of packets is done until the whole packet is received or is identical to its original. This increases the chance of latencies. Although the protocols for routing calculate the shortest path to the final receiver, the route area increases with the increment in the number of nodes.

Fig.6. End-to-end delay with UDP traffic in the grid topology.

Results presented in Fig. 6 clearly show the value of end-to-end delays measured for various LB modes in the grid of WMN. The value of delay in SI modes, FB, and QoS is almost in the same range as these schemes either use a single channel or only one type of traffic is sent through one specific channel according to the algorithm used. The significantly least value of delay is measured in a grid with RR 3IF (i.e with higher interface possibility). Since there was a minimum delay and high throughputs with multi-interfaces, so it was obvious to have errorless communication. The minimum delay means no retransmissions and no retransmissions means no or little chances of collisions, so it will definitely provide a channel with the lowest probability of error rate that was experienced in grid topology both with TCP and UDP traffic transmission.

For TCP transmission end-to-end delay was extremely low as usual for the highest number of interfaces on offer as shown in Fig.7. QoS mode as always transmitting both TCP and UDP packets have got the highest end-to-end delay as mainstream TCP and background TCP are assigned with common channels.

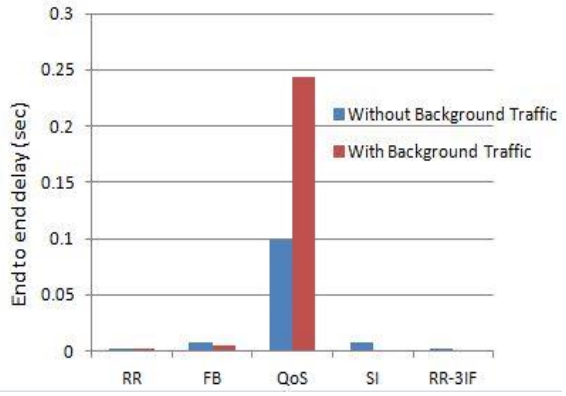

Fig.7. End-to-end delay with TCP traffic in a grid topology.

The results shown in Fig.8 verify that the PER always exists high in SI modes of transmissions than in multiple interfaces. Due to its connectionless property, UDP bears potentially unbounded loss and does no retransmission, so the UDP bit losses are always more maximum than TCP which is connection-oriented. This property was shown obvious with the experiments performed as well.

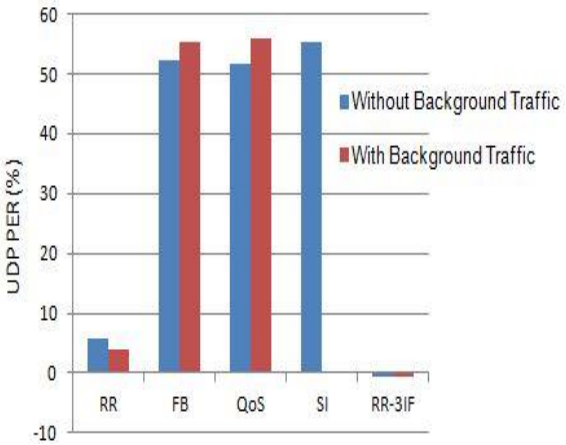

Fig.8. PER based on UDP transmission in a grid topology.

The TCP traffic type PER as shown in Fig.9 is unexpectedly high in QoS mode with background traffic as traffic used in the background was also TCP type, transmitted through the common channel-0 leaving channel-1 free for UDP. It is found that there was no packets loss in the case of multiple interfaces basically in 2-IF RR and 3-IF RR, as all interfaces used are in service to allow a path for TCP only.

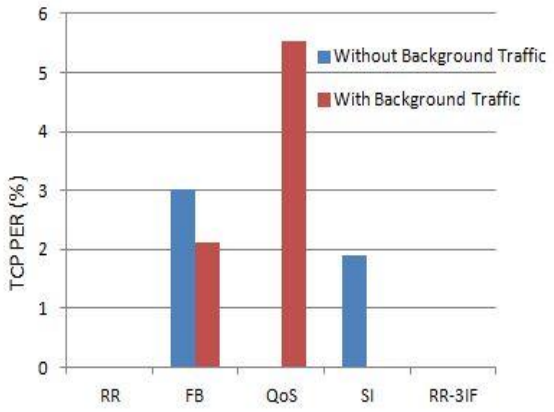

Fig.9. PER based on TCP transmission (Packets) in a grid topology

5. Discussion

This research designs a third layer (layer 2.5 or cross-layer) architecture named SCHEDULER based on IEEE802.11g Ad hoc mode for a modular load balancing in multi-interface WMN. It distributes incoming packets from an upper layer to each of the interfaces based on the configured scheduling scheme through the assigned channel. The operation of the scheduler module enables an application to utilize the available interfaces and the channels according to the scheduling scheme configured by the user. Here, the module supports the homogeneous set of channels allocated in the configuration file.

Using various scenarios in a grid topology, it is shown that the proposed model can be utilized to distribute packets in the MIMC WMN implementing various modes of load balancing. The designed framework provides an extensible and flexible C++ code structure that effectively supports traffic distribution in a defined mode. New load-balancing schemes can be implemented in the scheduler module by extending its base classes. The module is achieved after simple modifications in the existing INETMANeT framework to gather the information of the neighbor node's interfaces and their corresponding MAC addresses. The evaluations based on performance measurement have shown, that packets handling is possible in mesh networks by designing an intermediate layer.

Further, it is recommended, the model can be redesigned to support routing based on dynamic channel assignment strategies in MIMC WMN, multiple interfaces application in its intermediate nodes and hence investigate the performances.

Conflict of interest

The authors declare no conflict of interest exists for this work.

References

[1] Agrawal R., Faujdar N., Romero C. A. T., Sharma O., Abdulsahib G. M., Khalaf O. I., Mansoor R. F., & Ghoneim O. A. (2023). Classification and comparison of ad hoc networks: A review. Egyptian Informatics Journal, 24(1), 1-25. https://doi.org/10.1016/j.eij.2022.10.004

[2] Abdulhakim M., Abdulhakim A., Sain M., & Lee H. (2020). Moving Ad Hoc Networks—A Comparative Study. *Sustainability*, *13*(11), 6187. https://doi.org/10.3390/su13116187

[3] Rahman M., Agarwal A., & Alsarahn A. (2008). Capacity based channel assignment in multi-interface wireless mesh networks. 2008 11th International Conference on Computer and Information Technology. https://doi.org/10.1109/iccitechn.2008.4803127

[4] Kim H., Gu Q., Yu M., Zang W., & Liu, P. (2010). A simulation framework for performance analysis of multi-interface and multi-channel wireless networks in INET/OMNET++. Proceedings of the 2010 Spring Simulation Multiconference on - SpringSim '10. https://doi.org/10.1145/1878537.1878643

[5] Misra, Subhas & Woungang, Isaac & Misra, Sudip. (2009). Guide to Wireless Sensor Networks. 10.1007/978-1-84882-218-4

[6] Al-hazmi and H.de Meer "Virtualization of 802.11 interfaces for wireless mesh networks", Eighth International Conference on Wireless On-Demand Network Systems and Services, 2011, pp. 44-51, DOI: 10.1109/WONS.2011.5720199

[7] Rasheed, T.M. (2009). Wireless Mesh Network Simulation Framework for Omnet++", create-net technical report CN-

TR-200700016 2007, accessed on: May 2021.

[8] Munawar M. A. (2004). Multi-interface Multi-channel wireless mesh networks. UWSpace. Retrieved from http://hdl.handle.net/10012/875

[9] Passos D., G. O. Passos F., Dos Santos Silva B., & Albuquerque C. (2021). Modeling the performance of the link quality hypothesis test estimator mechanism in wireless networks. *Wireless Networks*, *27*(6), 4065-4081. https://doi.org/10.1007/s11276-021-02717-9

[10] Bhushan S., Singh A. K., & Vij S. (2019). Comparative study and analysis of wireless mesh networks on AODV and DSR. 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU). https://doi.org/10.1109/iot-siu.2019.8777466

[11] INETMANeT framework, Accessed and installed on 2012

[12] Optimized link state routing protocol (OLSR). (2003). https://doi.org/10.17487/rfc3626

[13] Rahmawan H., & Gondokaryono Y. S. (2009). The simulation of static load balancing algorithms. 2009 International Conference on Electrical Engineering and Informatics. https://doi.org/10.1109/iceei.2009.5254739

[14] J.H. Lam, S.G. Lee and W.K. Tan "Multi-channel wireless mesh networks test-bed with embedded systems", 2012, 26th International Conference on Advanced Information Networking and Applications Workshops, pp.533-537, DOI: 10.1109/WAINA.2012.68.

[15] Yarali A. (2008). Wireless mesh networking technology for commercial and industrial customers. 2008 Canadian Conference on Electrical and Computer Engineering. https://doi.org/10.1109/ccece.2008.4564493

[16] Talooki V. N., Rodriguez J., & Sadeghi R. (2009). A load balanced aware routing protocol for wireless ad hoc networks. *2009* International Conference on Telecommunications. https://doi.org/10.1109/ictel.2009.5158613

[17] Ding L., Shao Y., Li M., & Wu M. (2007). Load balanced broadcasting in ad hoc wireless networks using directional antennas. 2007 Second International Conference on Communications and Networking in China. https://doi.org/10.1109/chinacom.2007.4469553

[18] Goel N., Sangwan S., & Jangra A. (2012, March). Efficient weighted innovative routing protocol (EWiRP) to balance load in mobile ad hoc networks (MANETs). In IEEE-International Conference On Advances In Engineering, Science And Management (ICAESM-2012) (pp. 278-284). IEEE.

[19] Buvana M., Suganthi M., & Muthumayil K. (2011). Novel architecture for cluster based service discovery and load balancing in mobile ad hoc network. 2011 Third International Conference on Advanced Computing. https://doi.org/10.1109/icoac.2011.6165191

[20] Binh, L. H., & Duong, T. T. (2021). Load balancing routing under constraints of quality of transmission in mesh wireless network based on software defined networking. Journal of Communications and Networks, 23(1), 12-22. https://doi.org/10.23919/jcn.2021.000004

[21] Amnai M., Fakhri Y., & Abouchabaka J. (2011). QOS routing and performance evaluation for mobile ad hoc networks using OLSR protocol. International Journal of Ad hoc, Sensor & Ubiquitous Computing, *2*(2), 12-23. https://doi.org/10.5121/ijasuc.2011.2202.

[22] Tsai T., Tsai S.,& Liu T. (2009). Cross-layer design for multi-power, multi-interface routing in wireless mesh networks. 2009 Second International Conference on Advances in Mesh Networks. https://doi.org/10.1109/mesh.2009.27

[23] C. Kobel, W. Balaji, and J. Haberman, "A survey on multi-channel based, digital media-driven 802.11 wireless mesh network", In ICWMC 2011: 7th International Conference on Wireless and Mobile Communication, 2011, pp. 169-175.

[24] Köbel Christian & Baluja García, Walter & Habermann Joachim. (2011). A Survey on Multi-Channel Based, Digital Media- Driven 802.11 Wireless Mesh Networks

.