REDUCING REAL-TIME TRAFFIC PACKET LOSS IN MOBILE IPv6 USING TWO TIER BUFFER

SALIM M. ZAKI

A project report submitted in partial fulfilment of the
requirements for the award of the degree of
Master of Science (Computer science)

Faculty of Computer Science and Information Systems
University Technology Malaysia

OCTOBER 2008

To my family

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my lord Allah, then to family for their unlimited support in my study, and I would like to acknowledge the support and help I got from my supervisor Dr. Shukor. Finally, I want to express my thanks to my colleges, friends and researchers for their technical support.

ABSTRACT

Mobile IPv6 is a network layer protocol for enabling mobility in IPv6 networks. MIPv6 was designed to allow nodes to be reachable and maintain ongoing connections while their location within the topology. A number of implementations of Mobile IPv6 have been done to enhance the functionality of MIPv6. This study proposes a new scheme to reduce real-time traffic packet loss in MIPv6 environment. The Network Simulator ns-2 and its extension MobiWan that supports IPv6 have been used to simulate this scheme and tested its efficiency. The scenarios in this study divided into two parts first five scenarios represent the standard Mobile IPv6 that does not support buffer, and second five scenarios with buffer. The difference between scenario and other in each part is the period of sending data from Correspondent Node to Mobile Node. However, the proposed scheme that uses buffer shows that it could reduce packet loss that occur during Mobile Node handover while the Correspondent Node sending data to Mobile Node. And this metric has been measured and compared with standard Mobile IPv6 in order to compare its efficiency. Some other metrics such as throughput and delay have been tested and compared.

ABSTRAK

Mobil IPv6 merupakan lapisan protokol rangkaian bagi membolehkan mobiliti di dalam rangkaian IPv6. MIPv6 telah direka bagi membenarkan nodus dicapai dan mengekalkan sambungan antara lokasi di dalam topologi. Beberapa perlaksanaan Mobil IPv6 telah dijalankan bagi menambah baik kefungsian MIPv6. Kajian ini mencadangkan cara baru yang akan membawa pengurangan dalam kehilangan paket masa nyata di dalam persekitaran MIPv6. Simulasi rangkaian ns-2 dan sambungannya, MobiWan yang menyokong IPv6 telah digunakan untuk mensimulasi cara ini dan menguji tahap ke cekapannya. Kajian kes telah dipecahkan kepada dua bahagian dimana 5 bahagian pertama kajian kes tidak menyokong buffer dan 5 bahagian yang berikutnya menyokong buffer. Perbezaan yang terdapat di dalam kajian kes adalah tempoh sela masa penghantaran data dari nodus penghantar kepada mobil. Metrik ini telah diuji dan di bandingkan dengan Mobil IPv6 piawai bagi memperolehi hasil perbezaan kecekapan. Kaedah metrik yang berlainan juga seperti “throughput’ dan “delay’ telah dibandingkan bagi menguji kecekapannya.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AR	Access Router
BA	Binding Acknowledgment
BU	Binding Update
cCoA	current Care-of Address
CN	Correspondent Node
CoA	Care-of Address
CR	Central Router
FBU	Fast Finding Update
FMIPv6	Fast Handovers Mobile IPv6
FTP	File Transport Protocol
HA	Home Agent
HI	Handover Initiate
HMIPv6	Hierarchical Mobile IPv6
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPng	Internet Protocol next generation
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
LAN	Local Area Network
LBU	Local Binding Update
LD	Link Delay
MAP	Mobility Anchor Point
Mbps	Mega bits per second
MIPv4	Mobile Internet Protocol version 4

MIPv6	Mobile Internet Protocol version 6
MN	Mobile Node
NAR	New Access Route
newAR	new Access Router
NS-2	Network Simulator
oldAR	old Access Router
OSI	Open System Interconnection Model
PAR	Previous Access Router
QoS	Quality of Service
RA	Router Advertisement
RSVP	Resource Reservation Protocol
Sec	second
SMTP	Simple Mail Transfer Protocol
T	Time
TB	Tunnel Buffering
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
VCoIP	Videoconferencing over Internet Protocol
VoIP	Voice over Internet Protocol
Wireless LAN	Wireless Local Area Network

LIST OF APPENDIECIES

CHAPTER 1

INTRODUCTION

1.1 Introduction

The Internet today is not a simple hierarchical structure. It is made up of many wide and local area networks joined by connecting devices and switching stations. It is difficult to give an accurate representation of the Internet because it is continuously changing (i.e. new networks are being added, existing networks need more address, and networks of defunct companies need to be removed [1]). Today most end users need to use the Internet for different purposes such as IP telephony, video conference, email, and other online operations.

The wireless mobile Internet, which was a dream just a few years ago, is now progressing so fast and revolutionize the whole framework of the telecommunications industry. The wireless mobile Internet is an extension of the Internet into the mobile environment, which gives users access to the Internet services while they are on the move [2].

Most networks on the Internet currently use Internet Protocol version 4. However, this version has significant shortcomings. The primary problem is that the Internet address is only 32 bits in length with the address space divided into different classes. With rapid growth of the Internet, this addressing scheme cannot handle the projected number of users [1]. IETF has designed a new version called version 6. IPv6 also known as IPng (IP next generation), uses 128-bits (16-byte) address, versus the 32-bit (4-byte) address currently used in version 4 [8]. IPv6 can accommodate a larger number of users. In version 6, the packet format has been simplified, yet at same time it is more flexible to allow for future addition of features.

The new version supports authentication, data integrity, and confidentiality at the network later. It is designed to handle the transmission of real-time data such as audio and video, and can carry data from other protocols. IPng can also handle congestion and route discovery better than the current version [1].

Mobile communication has received a lot of attention in the last decades. The interest in mobile communication on the Internet means that IP protocol, originally designed for stationary devices, must be enhanced to allow the use of mobile computers [1]. Mobility has been supported in both Internet protocol versions. Mobile IPv4 allows transparent routing of IP datagrams to mobile nodes in the Internet, and Internet protocol version 6 allows nodes to remain reachable while moving around in the network.

1.2 Problem background

The increase of mobile computing devices and wireless networking products over the past decades has necessitated the support for host mobility on the Internet. In Internet environments, when a host moves and attaches itself to another network, it needs to obtain a new IP address. With this change of IP address, all existing connections to the mobile host terminates, as the IP routing mechanisms cannot deliver the data to the correct end-point. Mobile IPv4 [3] overcomes this by introducing a level of indirection at the network (IP) layer. It deploys a home agent that intercepts packets from the correspondent host and redirects these packets by tunneling them to the mobile node via a foreign agent in the visiting network. This approach ensures correspondent host transparency and only requires the mobile node to update its location to the home agent when changing between networks. However, initializing this indirection requires a timely home network registration process and an address resolution procedure. However, some shortcomings accompany MIPv4 such as triangle routing [4].

Mobile IPv6 has some features that could overcome MIPv4 shortcomings and provide better support for real-time traffic between the correspondent node and the mobile node [4]. Handoff for a mobile node causes packet loss. This problem will be huge during real-time sessions between the mobile node (MN) and the correspondent node (CN). Several solutions have been proposed to overcome this problem in MIPv6 such as fast handover and hierarchical MIPv6, including their extensions, which each of them aims to reduce packet loss as much as possible.

1.3 Problem statement

When the mobile node moves from one subnet to another, it cannot receive packets that the correspondent node sent to it. This unreachable problem remains until the mobile node gets new address and sends its binding updates to inform the correspondent node about its new address. In real-time traffic this situation will be a huge setback.

A new scheme will be proposed to reduce real-time traffic packet loss in MIPv6 handoff procedure. This scheme will be evaluated through simulation experiments to investigate its efficiency compared to other existing techniques.

1.4 Project objectives

The objectives of this study are as follow:

1. To investigate existing schemes that deal with reducing packet loss in MIPv6.
2. To propose an alternative scheme that could reduce real-time packet loss in MIPv6.
3. To evaluate the efficiency of the proposed scheme compared to the existing one.

1.5 Project scope

1. This study will focus only on the implementation of the proposed scheme in MIPv6 environment.
2. The performance evaluation of the proposed scheme will be made through a series of simulations using NS-2 network simulator and its extension mobiwan to support IPv6.

1.6 Importance of the study

Proposing the new scheme, communication in MIPv6 will be made more reliable by reducing the number of packet loss.

1.7 Organization of the report

This chapter provides an introduction and some basic information about the Internet protocol. Apart from that, problem background, problem statement of packet loss in MIPv6, project objectives, and project scope are also discussed in this chapter.

Chapter 2 discusses the relevant background of Mobile IP and gives the concept of TCP/IP and Mobile IPv6. This chapter discusses the handoff procedure and the enhancements of MIPv6 and some existing works.

Chapter 3 introduces the methodology of this study, the framework and the stages of the project included in this chapter.

Chapter 4 shows the implementation of the proposed scheme.

Chapter 5 discusses the network model and the simulation model with its parameters for the proposed technique, and the results with its analysis are discussed at the end of this chapter.

Chapter 6 concludes this study with the summary of the work that has been done and gives a future work.

1.8 Conclusion

Mobile Computing is becoming increasingly important due to the increase in the number of portable computers and the need to have continuous network connectivity to the Internet irrespective of the physical location of the node. The next version of IP, IPv6 is designed to be an evolutionary step from IPv4. Mobility support in IPv6 solves many of the problems of basic Mobile IP. However, there is still a problem in handoff procedure in MIPv6 such as packet loss. This research proposes a new scheme that tries to reduce packet loss in real-time traffic between the mobile node and the correspondent node in MIPv6.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

The popularization of the Internet and moveable computers has become remarkable and wireless communication environment has developed quite common at the same time. With this tendency, the order of users for better service is increasing every year. Users want the same quality audio and real-time services to be delivered in the mobile node. These users want to have indefinite access to the network in fixed IP with their portable devices, transcend the limits of time and place. However, since existing IPv4 has some limitations to provide mobility, users have to change IP address according to access points. Thus, there emerges the necessity of Mobile IPv6 (MIPv6).

This chapter discusses the concept of TCP/IP and defining the mobility support for Mobile IP. Also provides a relevant background about MIPv6 handover procedure and MIPv6 enhancements. At the end the simulation apart of that concept of real-time traffic, QoS, and using buffering technique covered too.

2.2 TCP/IP suite

The Transmission Control Protocol/Internet Protocol (TCP/IP) suite was produced to achieve end-to-end delivery of data [27]. TCP would be responsible for higher level functions such as segmentation, reassembly, and error detection while IP would handle datagram routing.

The Open System Interconnection Model (OSI) describes how the communication system between computers that need to communicate should be designed and OSI model has seven layers [11]. Unlike OSI [1], TCP/IP reference model consists of four layers: Application, Transport, Internet, and Network Interface, as shown in Figure 2.1.in the Application layer users enabled to access the network by providing a few services to the user. Some of the protocols and services available to the user are File Transport Protocol (FTP) for transferring files, Telnet for remote login, and Simple Mail Transfer Protocol (SMTP) for exchanging mail messages. Communication between computers is handled by the Transport layer, which is comprised of Transmission Control Protocol (TCP) and the User Datagram Protocol. The Internet layer is responsible for routing the data packets to the appropriate destination. The most important protocol in this layer is Internet Protocol (IP). The function of the Data Link layer and the Physical layer of the OSI model have been combined into single layer called the Network Interface layer in the TCP/IP reference model [11].

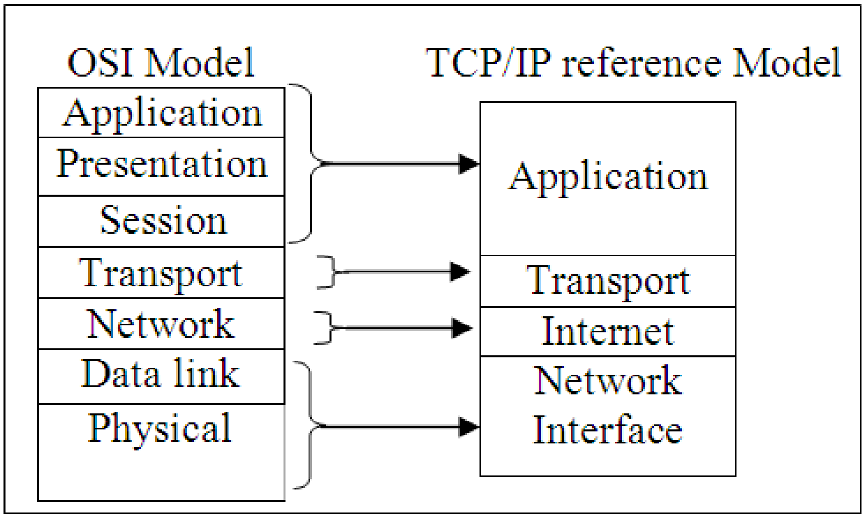


Figure 2.1 OSI Model vs. TCP/IP Model

2.2.1 IP concept

The Internet Protocol (IP) is designed for use in interconnected systems of packet-switched computer communication networks. The internet protocol provides for transmitting blocks of data called datagrams from sources to destinations, where sources and destinations are hosts identified by fixed length addresses [9]. IP is unreliable and connectionless datagram protocol [27] that means IP cannot provide error checking or tracking. Since reliability is important, the IP must paired with a protocol provides reliability such as TCP.

In IP each datagram is handled independently, and each datagram can follow a different route to the destination that is why IP called connectionless protocol [11]. This could cause data that sent from source to destination lost, corruption or arrive out of order.

2.2.2 TCP concept

The Transmission Control Protocol (TCP) is intended for use as a highly reliable host-to-host protocol between hosts in packet-switched computer communication networks, and in interconnected systems of such networks [6]. The TCP provides full transport layer services to applications [1].

TCP protocol must take the bytes it receives from an application and send them using a network layer protocol, which is IP in this case. IP is a message-oriented protocol; it is not stream-oriented. Thus, TCP must take the stream from the application and divide it into discrete messages for IP. These messages are called TCP segments. The segment consists of a 20- to 60-byte header, followed by data from the application program [11].

TCP has many services to the process at the application layer, like full duplex communication, process to process communication using port numbers, and TCP is a reliable stream transport protocol this means connection-oriented so, a connection must be established between both ends of a transmission before either can transmit data.

TCP is a reliable transport protocol because it offers reliable communication service. Thus, a stream of data sent on a TCP connection is delivered reliably and in order at the destination [6], and if it doesn’t delivered, detecting this and resending it.

TCP provides reliability using error control [1]. Error control includes mechanisms for detecting the segments those could be lost, corrupted, out of order and duplicated segments. Error control includes in addition to detecting, error correction mechanisms. Error control achieved in TCP through the use of three tools: checksum Acknowledgement, and time-out. In checksum tool each segment includes a checksum field which is used to check for corrupted segment. The destination TCP will discard the corrupted segment and is considered as lost. In Acknowledgement tool TCP uses acknowledgement to confirm the receipt of data segments.

2.2.3 UDP

Exchange importance data between systems on the network needs a simple protocol that is sits on top of IP and allows the user to launch data and receives replies on the network. That protocol is user datagram protocol (UDP) [30].

UDP is called a connectionless protocol, but it really is not [31]. The operating system maintains information about each active UDP socket, which implies a connection-oriented service. What UDP does not provide is error correction and flow control. There are no acknowledgments sent for received UDP data units; they are assumed received. The application using UDP services is responsible for determining if there have been errors or if data units are missing.

The UDP header is much simpler than that used in TCP as shown in Figure 2.2. There are no sequence numbers, which means no acknowledgements. Likewise, there is no an acknowledgement number field or urgent data processing capability. There is a source and destination port number and length and checksum fields. Nothing else is needed. Obviously, this allow receiving hosts to process UDP data units rather quickly since all that is needed is to send the received data unit to the proper application (identified by the port number).

16 bits	16 bits
UDP Source Format	UDP Destination Port
UDP Message Length	UDP checksum
DATA	

Figure 2.2 UDP Header

2.3 IPv6

IPv6 is a new version of Internet Protocol. It has been designed as an evolutionary, rather than revolutionary, step from IPv4. Functions which are generally seen as working in IPv4 were kept in IPv6. Functions which don't work or are infrequently used were removed or made optional. A few new features were added where the functionality was felt to be necessary [32].

The basic IPv6 header [33] is simpler than the IPv4 header: it has 8 fields instead of12. The IP header length and checksum fields are removed, the flow label field is added and the fragmentation fields are moved in the extinction headers. The size of the basic header is 40 octets, which is double the IPv4 header size. This is due to the 4 fold increase of each address field, from 32 bits to 128 bits each, or from 4 octets to 16 octets each as shown in Figure 2.3.

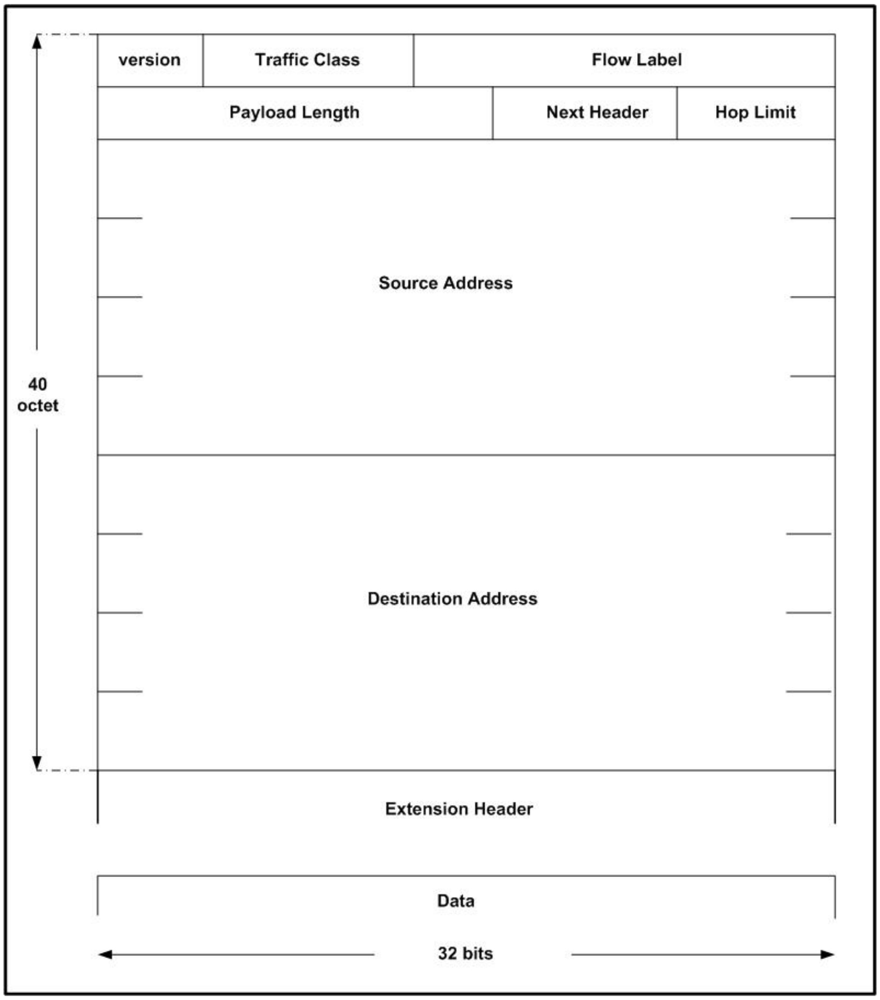


Figure 2.3 IPv6 Header

2.4 Addressing in IPv6

As stated earlier, the IPv6 addresses are of length 128 bits, which will increase the address space greatly [36]. Addresses are assigned to individual interfaces on nodes, not to nodes themselves. This means that a node can be reached by more than one address, since the interface is the node’s attachment to the link, and a node may have more than one interface. Also, a single interface might have multiple unique unicast addresses. The huge address space also allows more flexibility in terms of address allocation.

IPv6 addresses are divided into three categories:

1. Unicast addresses, which identify a single network interface.
2. Multicast addresses, which identify a group of network interfaces, typically at different locations. A packet will be sent to all network interfaces in the group.
3. Anycast addresses, which also identify a group of addresses. However, a packet will be sent to only one network interface in the group, usually the nearest one. Thus the name anycast, as the packet may be delivered to any interface in the group.

The combination of long addresses and multiple addresses per interface enables improved routing efficiency over IPv4. In IPv4, addresses generally do not have a structure that assists routing. Therefore, a router may need to maintain large tables of routing paths. Longer internet addresses allow for aggregating addresses by hierarchies of network, access provider, geography, and corporation and so on. Such aggregation should make for smaller routing tables and faster router table look-ups. The IPv6 address format is therefore built up hierarchically, as shown in sections below.

2.4.1 Unicast addressing

Unicast addresses have been structured in a number of ways:

- Provider-based Global unicast address
- Link-Local unicast address
- Site-Local unicast address
- Embedded IPv4 unicast address
- Loopback unicast address

A provider-based global unicast address provides has a hierarchical structure, and has the following five fields: Registry ID, which identifies the registration authority that assigns the provider part of the address; Provider ID, which identifies the internet service provider, who in turn assigns the subscriber port of the address; Subscriber ID, which distinguishes among multiple subscribers attached to the provider portion of the address; Subnet ID, which identifies the local subnetwork within the subscriber network; and finally, the Node ID, which identifies a single node interface in the addressed subnetwork [36].

Furthermore, two types of local-use addresses have been defined; Link-local and Site-local addresses. Link-local addresses are to be used for addressing on a single subnetwork, and cannot be integrated into the global addressing scheme. Examples of their use include auto-address configuration and neighbor router discovery. Such an address is for local link use only, so routers are not allowed to retransmit IPv6 packets that have a Link Local address as a source address.

Site-local addresses are designed for local use, but formatted in such a way that they can later be integrated into the global addressing scheme. The advantage of this is that the addresses can be used immediately by an organization that expects transition to the use of global addresses.

2.4.2 Multicast addressing

IPv6 includes the capability of addressing a predefined group of interfaces using a single multicast address. A packet with a multicast address in the destination address field is to be delivered to all members of the group.

The multicast address starts with a 8-bit long format prefix of only ones, followed by a 4-bit flags and a 112-bit group ID as shown in Figure 2.4. The flags field indicates whether the multicast field, a 4-bit scope field, address is permanently assigned or not. The scope field limits the scope of the multicast group, specifying if the multicast group is global, organizational-local, site-local, link-local or node-local.

Figure 2.4 Multicast address

2.4.3 Anycast addressing

An anycast address enables a source to specify that it wants to contact any node from a group of nodes via a single address. A packet with such an address will be routed to the nearest interface in the group, according to the router's measure of distance. One particular form of anycast address, the subnet-router anycast address is defined.

2.4.4 Address notation in IPv6

Since the address length has been increased from 32 to 128 bit in IPv6 compared to IPv4, [34]. That theoretically allows for as many as 3.4 x 1038 addresses. Even when used with the same efficiency as the current version IPv4 address space. That would still allow for 50,000 addresses per square

meter of land of earth. IPv6 addresses are represented in form of eight hexadecimal numbers divided by colons, for example:

FE80:0000:0000:0000:0001:0800:23e7:f5db

The auto configuration mechanism creates an address where the network part is provided by the router advertisements and the host part is created by the host, based on its link-layer address. DHCP instead is a centralized way of handling node configuration where the DHCP server gives all information, both the network and the host part, to the node. It is also capable of sending additional information, such as DNS servers, time servers, printers, font server, etc. not available in the route advertisements. DHCP in IPv6 does the same function as DHCP in IPv4. The DHCP server sends IP address, DNS server address and another possible data to the DHCP client and the client configures itself accordingly [35].

2.4.5 Address autoconfiguration

Typically, in an IPv4 environment, users or network managers must manually configure IPv4 addresses on nodes [36]. This is a resource-consuming task, and in IPv6 address autoconfiguration is a feature that is a part of the specification. This enables a host to configure automatically one or more addresses per interface. The aim of this feature is to support “plug-and-play” capability, which will allow a user to attach a host to a subnetwork and obtain IPv6 addresses automatically. Three models of address assignment have been defined:

- Local scope model. The local scope model is designed for use on a network without router, isolated from other networks.

- Stateless server model. With the stateless server model, a new device sends a request packet to a local well-known multicast address.

- Stateful server model. The Stateful server model support greater administrative support, by retaining information about retaining information about address-assignment transactions.

2.5 Internet Control Message Protocol for IPv6 – ICMPv6

The Internet Control Message Protocol for IPv6 – ICMPv6, although it is not a part of the IPv6 protocol as such, but a management protocol that uses IPv6 as a bearer, it is a necessary component for traffic management control messages and error reporting in IPv6.

In the IPv4 standard a separate protocol, ICMP, is defined for exchange of control messages across the Internet, with the purpose to provide feedback about problems in the communication environment. This is an essential part of the TCP/IP protocol suite; it is absolutely necessary to make routers and hosts able to report error situations to one another. Examples of such error messages can be that a datagram cannot reach its destination, time-to-live (TTL) for a datagram has exceeded, or that a router has not got enough buffer capacity to forward a datagram. Another category of ICMP messages, are messages of informational character. These are used for hosts to exchange information for diagnostic and management purposes. A well-known function that uses ICMP is the Ping command, which is useful to measure the roundtrip delay between two hosts.
As for IPv4, IPv6 requires the use of ICMP. The original version does not meet the requirements of IPv6, so a new ICMPv6 protocol has been specified [38]. The key features of ICMPv6 are as follows [39]:

- ICMPv6 uses a new protocol number to distinguish it from the ICMP that works with IPv4.

- Both protocols use the same header format, which is of 64 bits length.
- Some little-used ICMP messages have been omitted from IPv6.
- The maximum size of ICMPv6 messages are larger, to exploit the increased size of packets that IPv6 guarantees will be transmitted without fragmentation.

ICMPv6 can be divided into two categories of control messages; messages for error reporting and messages for informational purposes.

2.6 Mobile IP

The Internet Protocol (IP) with its both versions IPv6 and IPv4 define how data is sent from one computer to another computer over packet-switched networks such as the Internet [7]. Supporting mobility in IP provides an efficient, scalable mechanism for roaming within the Internet. Using Mobile IP, nodes may change their point-of-attachment to the Internet without changing their IP address. This allows them to maintain transport and higher-layer connections while moving.

2.6.1 Mobile IPv4

Mobile IPv4 allows transparent routing of IP datagrams to mobile nodes in the Internet. Each mobile node is always identified by its home address, regardless of its current point of attachment to the Internet. While situated away from its home, a mobile node is also associated with a care-of address, which provides information about its current point of attachment to the Internet. The protocol provides for registering the care-of address with a home agent. The home agent sends datagrams destined for the mobile node through a tunnel to the care-of address. After arriving at the end of the tunnel, each datagram is then delivered to the mobile node [3].

2.6.2 Mobile IPv6

IPv6 is the new version of the current the Internet Protocol, designed as the successor to IP version 4 (IPv4) [8]. IPv6 addresses the main problem of IPv4, that is, the exhaustion of addresses to connect computers or host in a packet-switched network. IPv6 has a very large address space and consists of 128 bits as compared to 32 bits in IPv4. IPV6 brings quality of service that is required for several new applications such as IP telephony, video/audio, interactive games or ecommerce. Whereas IPv4 is a best effort service, IPv6 ensures QoS, a set of service requirements to deliver performance guarantee while transporting traffic over the network. For networking traffic, the quality refers to data loss, latency (jitter) or bandwidth. In order to implement QoS marking, IPv6 provides a traffic-class field (8 bits) in the IPv6 header. It also has a 20-bit flow label [7]. Overall, IPv6 contains addressing and control information to route packets for the next generation Internet.

Mobile IPv6 allows nodes to remain reachable while moving around in the IPv6 Internet as shown in Figure 2.5. Each mobile node is always identified by its home address, regardless of its current point of attachment to the Internet. While situated away from its home, a mobile node is also associated with a care-of address, which provides information about the mobile node's current location. IPv6 packets addressed to a mobile node's home address are transparently routed to its care-of address. The protocol enables IPv6 nodes to cache the binding of a mobile node's home address with its care-of address, and to then send any packets destined for the mobile node directly to it at this care-of address. To support this operation, Mobile IPv6 defines a new IPv6 protocol and a new destination option. All IPv6 nodes, whether mobile or stationary, can communicate with mobile nodes [4].

Mobile IPv6 has three main elements [10], each of which has its own function:

- The function "mobile node": this function is installed in the mobile node (MN). It fulfils the functions of movement detection (movements from a sub-network IP to another) and of current localization update concerning its movements towards its "Home Agent" and its correspondents.
- The function "Home Agent" (HA): this function is activated in the router who connects the home sub-network of the mobile. It takes care to maintain information concerning the current localization of the mobile and to intercept the datagrams addressed to the mobile before to retransmitting them towards the MN current position.
- The correspondent node's mobility function: this function is activated in the correspondent. It takes care to maintain information concerning the current localization of the mobile and to transmit the MN's intended datagrams directly to the MN.

The delivery of multimedia and other services over wireless laptops and cellular telephones devices depends on the ability of the mobile Internet to support real-time traffic flows with guaranteed quality of service (QoS). The current mobility protocol for the Internet, Mobile IP, was designed for best effort packet delivery, and as a result of being on IPv4, has inefficient handoff routing procedure that limits its performance for multimedia traffic. IPv6 has new features, such as security support and increased address space that can be used to achieve more efficient handoff routing techniques [9].

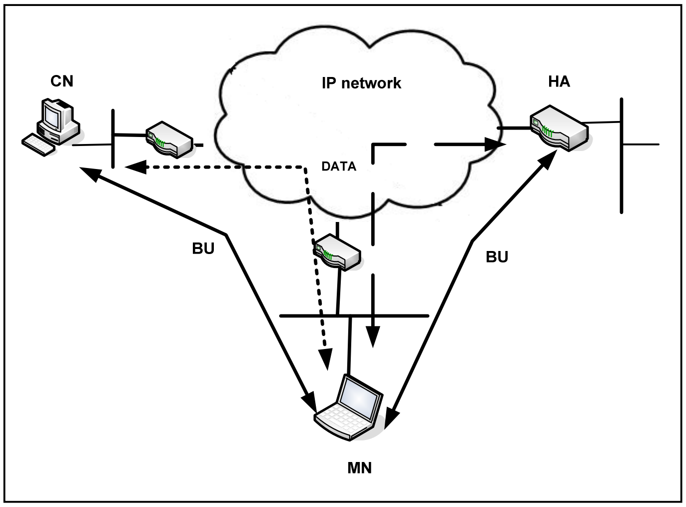


Figure 2.5 overview of Mobile IPv6

2.7 Handoff Procedure

A mobile node undergoes an IP-layer handoff, or simply a handoff, when it changes IP connectivity. This begins with a change in link-layer attachment, also referred to as a link-layer handoff, and is followed by as shown in Figure 2.6 the discovery of new routers, movement detection, address configuration getting care-of-address, and finally Mobile IPv6 registrations [16].

The mobile can use the "Neighbour Discovery" protocol mechanisms to learn about local routers and on-link prefixes during router discovery [10]. Routers send out Router Advertisement (RA) messages periodically, or in response to Router Solicitations [17]. Thus, a Mobile Node periodically receives RA of it's HA and determines that it is on its home network.

To detect its movement, the mobile can use the "Neighbour Discovery" protocol mechanisms. This method is based on the reception of RA messages emitted by the current routers present on the sub-networks. By receiving, RA message emitted by the router, the mobile maintains lists of prefixes and of default routers. When the mobile detects that it does not have any more contact with its current router, it chooses another one in its default router list and creates its IP address starting from the corresponding prefix. Within the IP mobility framework, this address is known as "care-of address" [10].

After movement detection and the MN choose its care-of address and registers its CoA with home agent and correspondent node. This establishes a binding between the care-of address and the mobile node's home address, which has a prefix from the home agent's network and remains stable across movements [16]. Such message called "Binding Update" (BU). A mobile node can ask for any of its correspondents (HA or other) to acknowledge its BU. Such message called "Binding Acknowledgment" (BA).

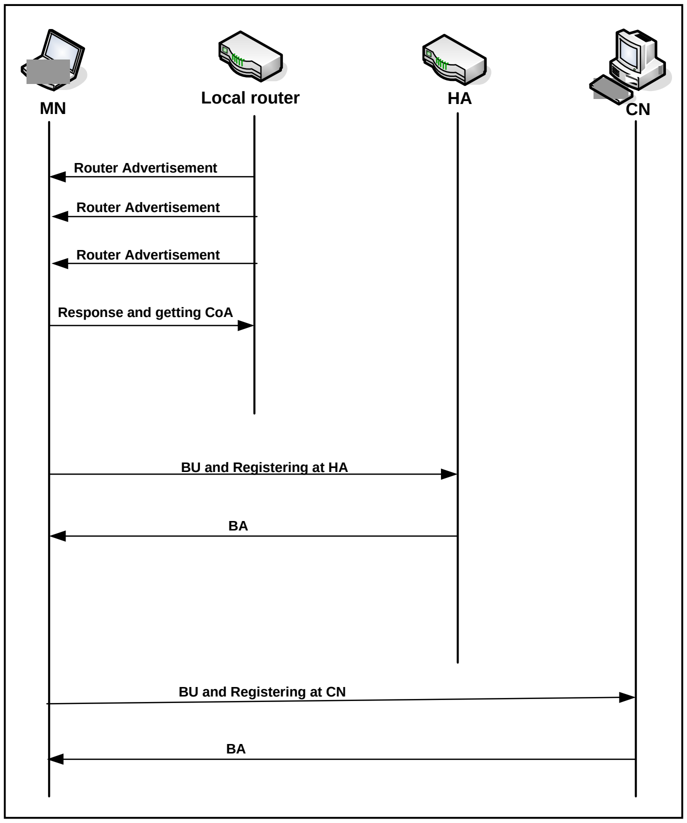


Figure 2.6: Handoff procedure

2.8 Mobile IPv6 Security

Mobile IPv6 is a new protocol that attempts to do something that has not been done before on the internet: redirect traffic between a mobile node and other correspondent nodes from one address to another. The signalling for such redirection is done the mobile and correspondent nodes. This signalling utilizes several messages that require different levels of security. When dealing with the mobile node-home agent messages, it is possible to use manual configuration to set up an IPsec security association [36].

2.9 MIPv6 enhancements

A certain number of mobility protocols were presented to improve the performances of Mobile IP and in particular the Handoff process. These protocols are made for environments where the MIPv6 specifications are insufficient: signalisation overload, packets loss, and data delivery delay. These delays are directly related to the round-trip time of registration messages [19]. Fast Handovers for Mobile IPv6 and Hierarchical Mobile IPv6 are both enhancements to MIPv6.

2.9.1 Fast Handovers for Mobile IPv6

Protocol enhancements that enable mobile nodes to move quickly become connected at new points of attachment to the Internet. These protocol enhancements are intended to minimize the time during which the mobile node is unable to send or receive IPv6 packets (i.e., the handover latency). Mobile IPv6 is considered to be the base protocol, but Mobile IPv6 does not mandate any mechanism which gives assurances about minimizing the handover latency. Fast Handover considered as an enhancement for Mobile IPv6 [18].

The goal of this protocol is to allow a MN (Mobile Node) to configure an new Care-of-Address (CoA, or current CoA as cCoA) before it moves towards a new sub-network with the aim of being able to use it straight away after its connection to the new Access Router. The solution of Fast Handover is the minimization of the handover latency. The principle is to establish a new temporary address before to break the Mobile Node connection with its old Access Router (oldAR). Then, when the mobile is attached to the new Access Router (newAR), it can continue its communications with its new already-known address. If the anticipated registration fails, the Mobile Node can always carry out a "traditional" Handoff process. Moreover the "Fast Handover" sets up a packets forwarding system between the oldAR and the newAR [19].

2.9.2 Hierarchical Mobile IPv6

It is a well-known observation that MNs moving quickly as well as far away from their respective home domain or correspondent nodes produce significant BU signalling traffic and will suffer from handoff latency and packet losses when no extension to the base-line Mobile IP protocol is used. Hierarchical Mobile IPv6 (HMIPv6) is a localized mobility management proposal that aims to reduce the signalling load due to user mobility [20].

A new Mobile IPv6 node, called the Mobility Anchor Point (MAP), is used and can be located at any level in a hierarchical network of routers, including the Access Router (AR). The MAP will limit the amount of Mobile IPv6 signalling outside the local domain. The introduction of the MAP provides a solution to the issues outlined earlier in the following way:

- The mobile node sends Binding Updates to the local MAP rather than the HA (which is typically further away) and CNs
- Only one Binding Update message needs to be transmitted by the MN before traffic from the HA and all CNs is re-routed to its new location. This

is independent of the number of CNs that the MN is communicating with [21].

2.10 Proposed schemes for MIPv6 and its enhancements

The authors in [22] proposed an interoperable enhancement to Mobile IPv6 to reduce packet loss during movement, and to avoid retransmission when movement is complete. Tunnel Buffering (TB) scheme does this by 'pausing' the packets which are to be send over a MobileIPv6 or HMIPv6 tunnel until movement is complete. The MN cannot generally inform the Home Agent (HA) or the Mobility Anchor Point (MAP) of its new location until it has arrived there, and so the HA/MAP may forward packets to the old location during movement.

Tunnel Buffering (TB) considers the situation where a MN anticipates that it is about to move, but does not know where it is about to move to. In this case, the MN can send a special (L)BU requesting that its traffic be buffered until it has determined its new link CoA and can send another (L)BU.

In [26] the author assumes the forwarded packets from Previous Access Router (PAR) may arrive at New Access Route (NAR) before the MN is able to attach to it. These packets will be lost unless they are buffered by the NAR. Similarly, if the MN attaches to NAR and then sends a Fast Binding Update (FBU) message, packets arriving at PAR will be lost unless they are buffered. It provides an option to indicate request for buffering at the NAR in the Handover Initiate (HI) message. When the PAR does request this feature, it should also provide its own support for buffering. The problem here is generating triangle routing through the PAR, the NAR and the CN.

In [14], the authors proposed an enhanced buffer management scheme for Fast handover. The proposed scheme has two parts. First, while the original Fast Handover protocol only buffers packets in the New Access Router (NAR), they use buffers in both the Previous Access Router (PAR) and the NAR during a handoff process as shown in Figure 2.7. This helps improve the total buffer utilization in the network. Second, they define three types of services in the handoff process so that packets can be treated differently based on their traffic characteristics. Buffering packets will reduce packet loss. But if the number of mobile nodes those attached to the router increased. The overhead on the router will affect its efficiency.

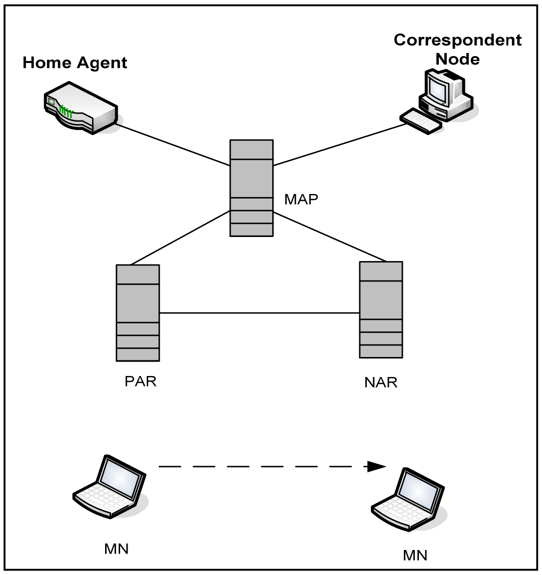


Figure 2.7 Buffering on PAR and NAR in FMIPv6 scenario

The authors in [23] proposed a new two-path hop handoff technique; The CN has the option of using two different Resource Reservation Protocol (RSVP) paths to reach the MN. The trust path is called the mobile path, which is a direct RSVP path to any new location of the MN. Since the mobile path does not depend on forwarded packets from the HA, it is the more efficient and preferred path to use. However, while the MN is moving to a new location, incoming packets may be lost or may arrive out of order. Therefore, this technique periodically use the home path, i.e., from the CN to the HA to the MN, to ensure uniform and ordered forwarding of incoming packets, and to reduce packet loss. The incoming packets for the MN are buffered in the HA until the new path between MN and HA can be established. The problem here, increasing the overhead on the Home Agent especially when many MNs are attached to HA.

A comparison between afford mentioned schemes are listed in Table 2.1.

2.11 Quality of Service (QoS)

QoS is becoming an increasingly important element of any communications system. In the simplest sense, QoS means providing consistent, predictable data delivery service, in other words satisfying the customer application requirements. Providing QoS means providing real-time (e.g., voice) as well as non-real-time services [28].

2.11.1 QoS Parameters

There are some QoS parameters those are relevant to multimedia applications like; Throughput, Delay, and Loss [29].

Case	Buffer place	Advantages	Limitations
1	-On MAP when HMIPv6 is used. - Or on HA when MIPv6 is used.	-Reducing packet loss. -Avoid retransmission when movement is complete.	-Overhead on the access router. -the previous router may forward packets to new router before MN attached to it.
2	On both PAR and NAR	-Reducing packet loss. -Retain packets those could be lost when forwarded from PAR to NAR.	-Increasing overhead on ARs when more than one MN attached to it. - Also it is generating a triangle routing through the PAR, the NAR and the CN.
3	On both NAR and PAR	-Reducing packet loss. -Proposing a management scheme to manage the different kind of buffered packets.	-The overhead on PAR and NAR will increase when MN numbers increased.
4	On the Home Agent (HA)	-Reducing packet loss. - Uniform and ordered forwarding of incoming packets from CN to MN by buffering them on HA.	-Increasing overhead on the Home Agent when many MNs attached to it.

Table 2.1 comparison between proposed buffer schemes in MIPv6

2.11.1.1 Throughput

Form the application perspective, throughput refers to the data rate (bits per second) generated by the application. Throughput measured in the number of bits per second, some time is called bit rate or bandwidth.

There is one characteristic of an application data traffic generation process: the data traffic generation rate (constant and variable bit rate). It shows how the traffic generation process determines the applications required throughput.

The constant bit rate (CBR) applications generate data traffic with constant data rate. While variable bit rate (VBR) applications generate data traffic with variable data rate.

2.11.1.2 Packet Loss

Packet loss directly affects the perceived quality of the application. It compromises the integrity of the data or disrupts the services. At the network level, packet loss can be caused by network congestion, which results in dropped packets. Another cause of loss is caused by bit error that occurs due to a noisy communication channel. Such loss will most likely occur in a wireless channel.

2.11.1.3 Delay

Real-time applications require the delivery of information from the source to the destination within certain period of time as shown in Figure2.8 Long delays may cause incidents such as data missing the playback points, and that suffers real-time data. There are many sources across the network cause the delay between source and destination:

1. Source-processing delay (digitization and packetizing delay).
2. Transmission delay: the transmission time of a packet is a function of the packet size and transmission speed.
3. Network delay: propagation delay, protocol delay, and output queuing delay.
4. Destination processing delay.

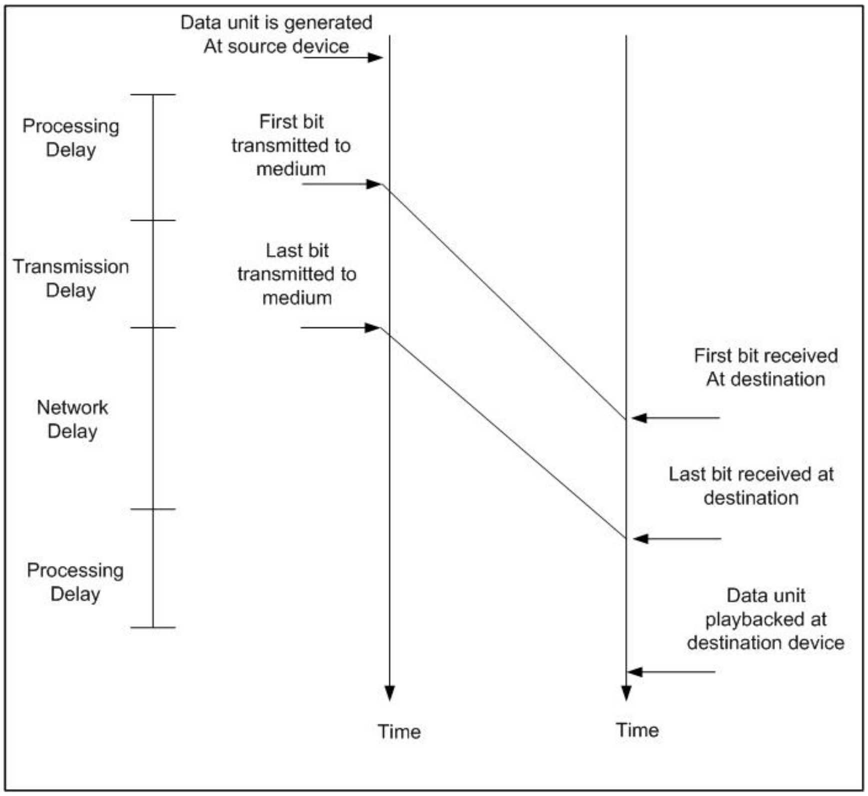


Figure 2.8 End-to-end Delay diagram

2.12 Real-time traffic

The vision of nomadic users at roaming devices performing synchronous communication, such as voice or videoconferencing over IP (VoIP/VCoIP), is around and raises new challenges for the Internet infrastructure [25]. Streaming media applications have requirements that distinguish them from other types of TCP/IP traffic. Most applications, such as e-mail and Web traffic, are not very sensitive to network conditions such as delay and bandwidth. They are, however, very sensitive to lost or corrupted packets. In contrast, streaming media is very sensitive to network characteristics such as delay and bandwidth and may be in some cases, be able to tolerate lost or corrupted packets. These unique requirements, combined with their popularity, make streaming media applications interesting and relevant to examine in a mobile environment [24].

2.13 Buffering

Real-time communication imposes stern quality of services requirements on the underlying network infrastructure. 100 ms real-time carry relevant information, a spoken syllable for instance in the audio case [15].

A feasible solution to avoid packet loss during the link down time is to buffer those packets [14]. During MN handoff process packet loss especially in stream traffic is a huge problem. Therefore, buffering technique is used to store data and uses it for future need. However, the art of queuing is the art of managing congested queues in order to give specific bandwidth, latency and loss to some flows through each node. Several technologies exist, which can be grouped into two categories [37]. The first take care of packet ordering in the output queues:

-FIFO (first in first out), also called first come first served (FCFS), simply outputs the packets in the order in which they have been received;

-Class-based queuing (also called custom queuing by CISCO) assigns packets to prioritized logical queues;

-Fair queuing and weighted fair queuing (WFQ) algorithms.

Those techniques can be combined with any packet loss management technique of the second category:

-Simple overflow.

-Random early detection and weighted random early detection.

2.14 The simulation tool ns-2

Network simulator ns-2 is a discrete event simulator targeted at networking research. Ns provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless (local and satellite) networks [12]. The ns simulator covers a very large number of applications, protocols, network types, network elements, and traffic models. Ns-2 is a standard experiment environment in research community.

Ns-2 is an object-oriented simulator, written in C++, with an OTcl interpreter as a front end user programming language. Two programming languages are used for the reason that the implementation of packet handling, byte manipulation and routing protocols requires efficient implementation of simulator code in C++. On the other hand, a more simple and user-friendly language is desired to be able to write simulation scripts without having to create long and complex C++ programs. This is done with the script language OTcl. Ns is primarily developed for Unix, but also exists for Windows.

The network animator, NAM, is a visualization tool that comes with the ns-2 package. It is capable of animating the packet traffic to give the user a visual representation of the trace file events. It supports topology layout, packet level animation, and it also provides a simple trace functionality to trace a packet from source to destination.

Ns-2 doesn't support MIPv6; therefore, a simulation tool (MobiWan) was developed to support MIPv6. MobiWan based on ns-2 meant to simulate Mobile IPv6 under large Wide-Area Networks (both local-area mobility and global-area mobility) [13]. Mobiwan contains extensions to simulate Mobile IPv6 as well as an extension to manipulate and configure large network topologies called Topoman. Topoman is a tool for automatic configuration of larger topologies, which can be quite time-consuming if it is done manually.

Mobiwan comprises mobile IPv6 available in two modes; local mobility (i.e. mobile node movement within an administrative domain) and global mobility (i.e. mobile node movement across administrative domains). The latter is also meant to support hierarchical mobile IPv6, but this is not included in the distribution.

2.15 Conclusion

The Internet infrastructure is built on top of a collection of protocols, called the TCP/IP protocol suite. Transmission Control Protocol (TCP) and Internet Protocol (IP) are the core protocols in this suite. IP requires the location of any host connected to the Internet to be uniquely identified by an assigned IP address. This raise one of the most important issues in mobility, because when a host moves to another network, it has to change its IP address. However, the higher level protocols require IP address of a host to be fixed for identifying connections. Mobile IPv6 is an extension to the Internet Protocol proposed by the Internet Engineering Task Force (IETF) that addresses this issue. It enables mobile hosts to stay connected to the Internet not considering their location and without changing their IP address.

The ns simulator covers a very large number of applications, protocols, and other network objects. Widely used by researchers in network field. MobiWan is a simulation tool makes ns to support MIPv6.

CHAPTER 3

METHODOLOGY

3.1 Introduction

Many techniques have been proposed to enhance MIPv6 protocol and overcome as much as possible the problems that occur during MN handoff. Some of them focused on reducing packet loss during real-time traffic. Each of these techniques has its own pros and cons as mentioned in Chapter 2.

This research aims to propose a new scheme to reduce packet loss in real-time traffic between the CN and the MN in MIPv6 environment.

The performance evaluation of the proposed scheme will be made through a series of simulations using network simulator (ns-2) and its extension MobiWan that supports MIPv6.

3.2 Stages of the work

This study is divided into four stages as shown in Figure 3.1 at the first studying existing techniques that were proposed to reduce packet loss in MIPv6 and analyzing their performance and efficiency.

New scheme will be proposed in the second stage by designing the network topology for this study and the simulation model for the ns-2 implementation. Evaluating the performance for the new scheme and comparing the results with the existing scheme will be in the last two stages.

The conclusion of this study is presented in Chapter 6.

3.2.1 First stage

The first stage consists of two parts: first one focuses on the literature review and studying the existing schemes that were proposed to reduce packet loss in MIPv6 protocol.

In the second part investigating and analyzing the existing work will determine the limitations and weakness for each scheme.

3.2.2 Second stage

The first part of this stage includes framework design for the proposed scheme in MIPv6 environment.

The second part presents the simulation model design for the proposed scheme using the ns-2 and MobiWan and setting the parameters for the simulation such as links bandwidth and delay time etc. These parameters are subject to change according to testing and redesigning the new scheme. At the end of this stage simulating the proposed scheme will be done.

3.2.3 Third stage

Evaluating the performance for the proposed scheme is presented in this stage. Enhancing and redesign the proposed scheme until achieve best performance for reducing packet loss with this scheme.

3.2.4 Fourth stage

Discussing and analyzing the results of the proposed scheme with the existing work to evaluate its efficiency compared with the existing one.

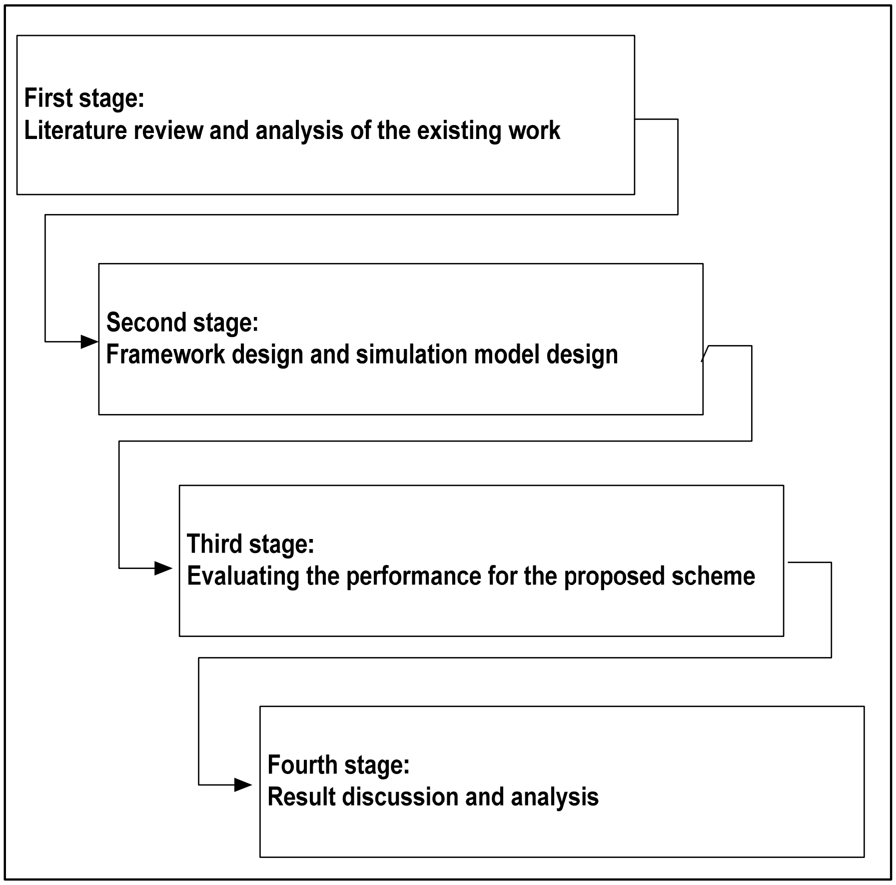


Figure 3.1 Stages of the work

3.3 Conclusion

Reducing packet loss in MIPv6 is an important thing. Therefore, many proposed techniques have been tested in the simulation by using ns-2. A new scheme has been proposed to reduce the real-time packet loss in this study.

CHAPTER 4

IMPLEMENTATION

4.1 Introduction

The MIPv6 has been proposed to overcome MIPv4 drawbacks. MIPv6 also has some limitations; many schemes have been proposed to enhance the quality of service in MIPv6 especially during real-time traffic. This project proposed a new scheme to reduce real-time traffic packet loss in MIPv6. This chapter describes this scheme and explains the implementation of it using the network simulator ns-2 with its extension mobiwan.

4.2 The Proposed Architecture

In the proposed architecture, the CN has two tier buffers to retain the forwarded real-time traffic packet as shown in Figure 4.1. The first buffer used to retain a copy of the sent packets to the MN. However, while the MN

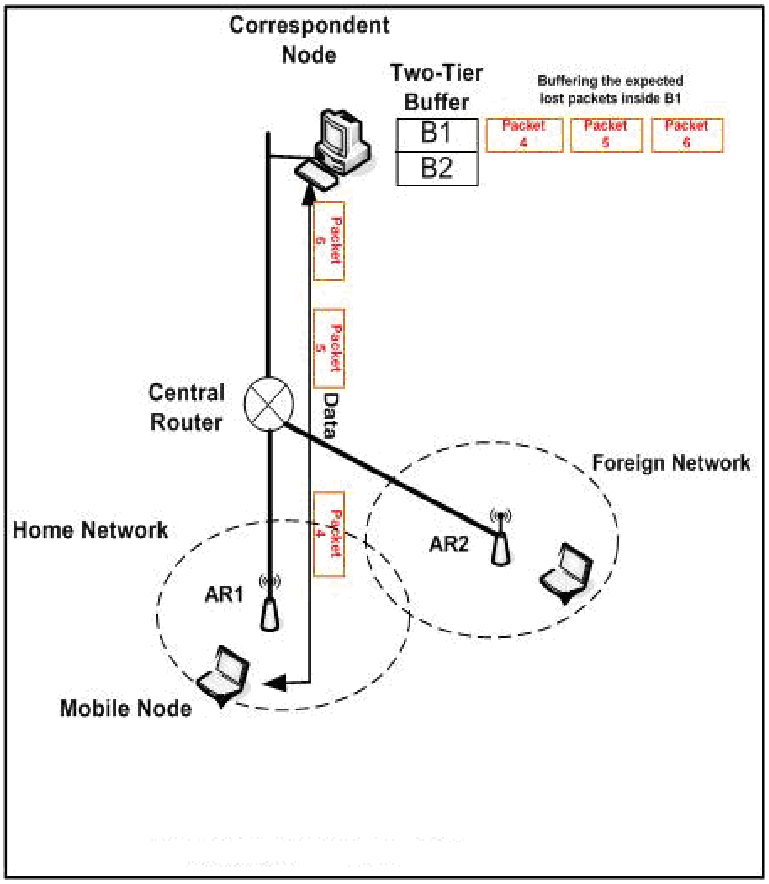


Figure 4.1 MN receiving real-time packets

Therefore, this study proposed to periodically use the first buffer on the CN to retain the packets that already sent before MN handoff procedure starts, as illustrated in Figure 4.2. When the MN attached to a new location and sending binding updates (BU) to the correspondent node, the CN will send the first buffer contents to the Mobile node. The contents of this buffer cleared after MN receives them and substituted by new packets that the CN wants to send to the MN and so on until the real-time session finish between the two nodes. The size of the buffer depends on the amount of packets that need to be sent. This approach will reduce packet loss during the MN handoff procedure and will reduce overhead on the routers those used to buffer real-time packets for MNs.

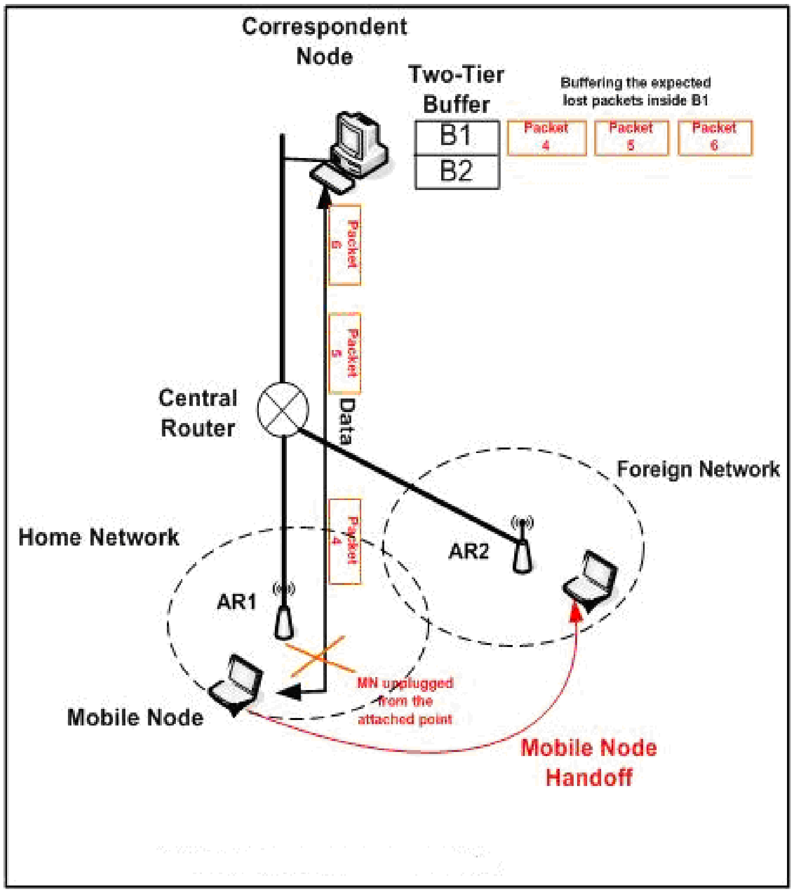


Figure 4.2 MN handoff and buffering packets

It is possible the MN moves again while the CN sending the buffered packets. This could make the forwarded packet unable to reach the MN. To avoid this problem, duplication for the first buffer contents into second tier of the buffer has been proposed as shown in Figure 4.3.

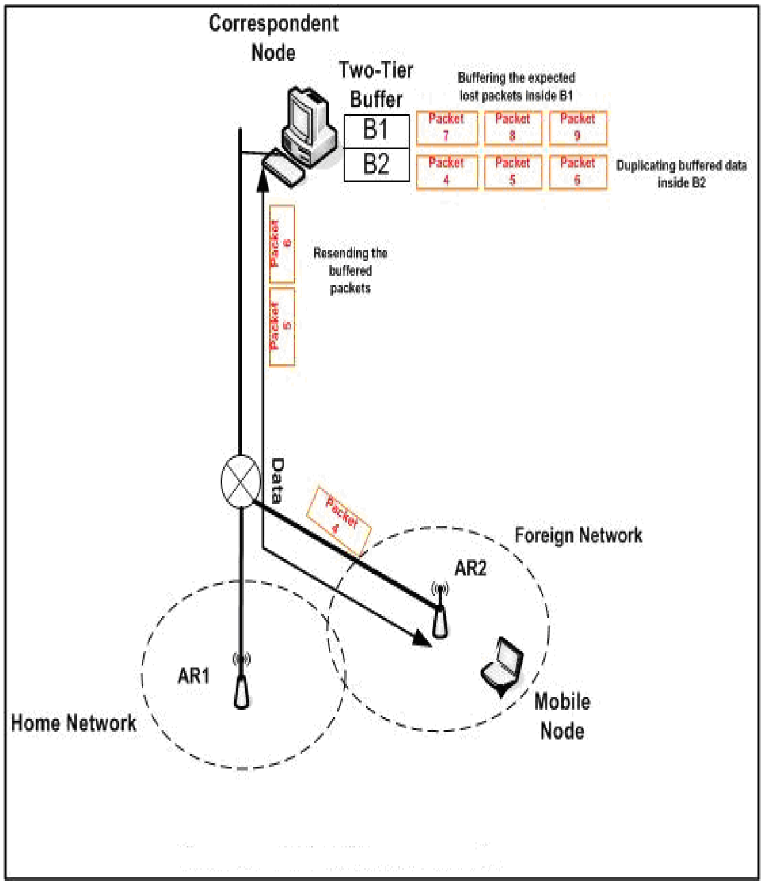


Figure 4.3 Duplicating and sending buffered packets

The aim of this buffer is to retain the packets for the first buffer until the CN be sure that the MN got the packets that were retained in the first buffer. Once the MN received the packets, the second buffer will be cleared as illustrated in Figure 4.4.

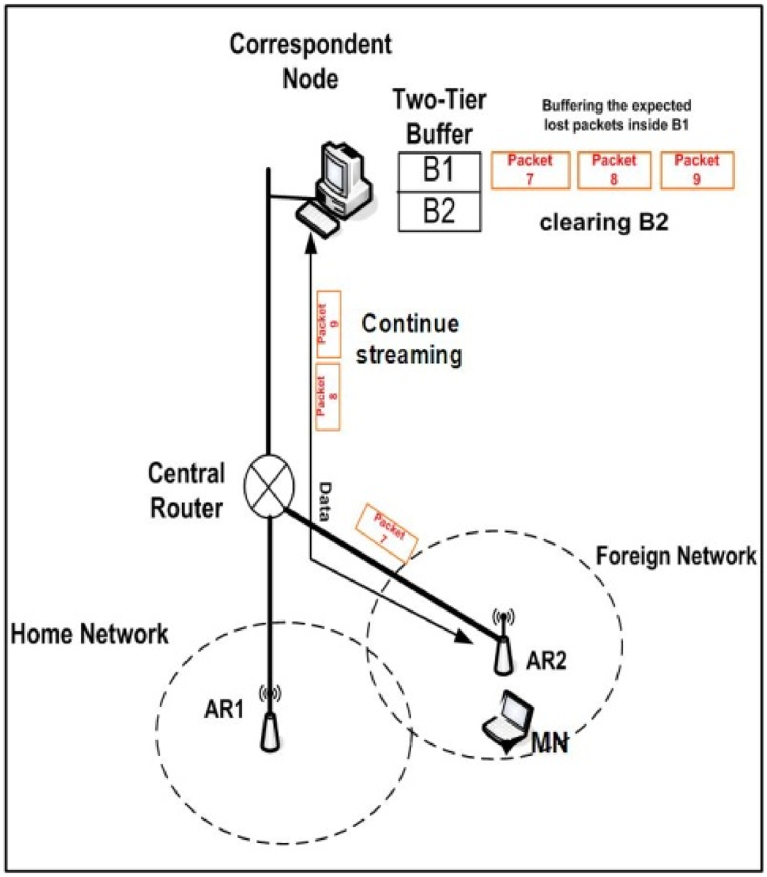
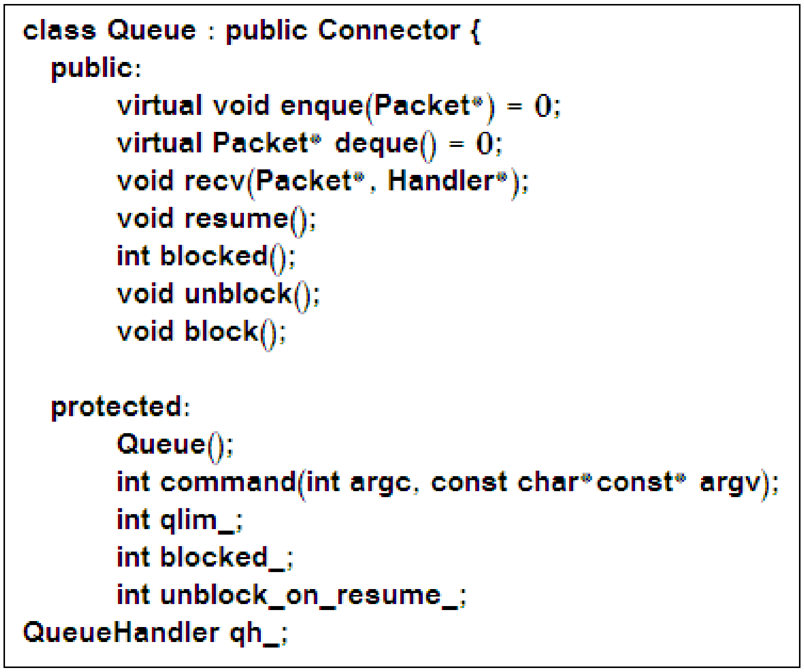


Figure 4.4 Clearing B2 and sending new packets

4.3 Implementing the Proposed Architecture in ns-2

The network simulator ns-2 supports different kinds of queues, which includes drop-tail (FIFO) queuing, RED buffer management, CBQ (including a priority and round-robin scheduler), and variants of Fair Queuing including, Fair Queuing (FQ), Stochastic Fair Queuing (SFQ), and Deficit Round-Robin (DRR).

The Queue class in ns-2 is written with C++ and is derived from a Connector base class. It provides a base class used by particular types of (derived) queue classes, as well as a call-back function to implement blocking. As shown in Figure 4.5, a piece of code is provided in queue.h file.

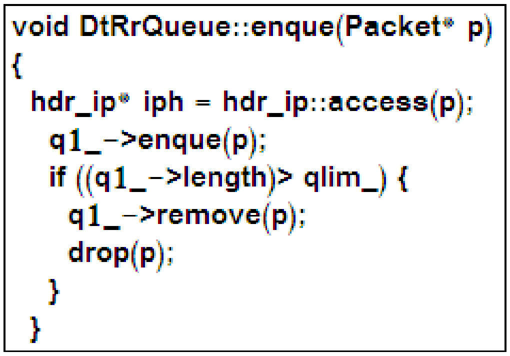


```
class Queue : public Connector {
  public:
        virtual void enque(Packet*) = 0;
        virtual Packet* deque() = 0;
        void recv(Packet*, Handler*);
        void resume();
        int blocked();
        void unblock();
        void block();

  protected:
        Queue();
        int command(int argc, const char*const* argv);
        int qlim_;
        int blocked_;
        int unblock_on_resume_;
QueueHandler qh_;
```

Figure 4.5 Queue class

To build a two tier drop-tail (FIFO) buffer, it is necessary to use the Queue Class provided by ns-2 to be the base class used by the new queue class. The queue has two logical FIFO queues; first logical queue and second logical queue, of which the total size is equal to the size of physical queue, Figure 4.6 shows a piece of the code.



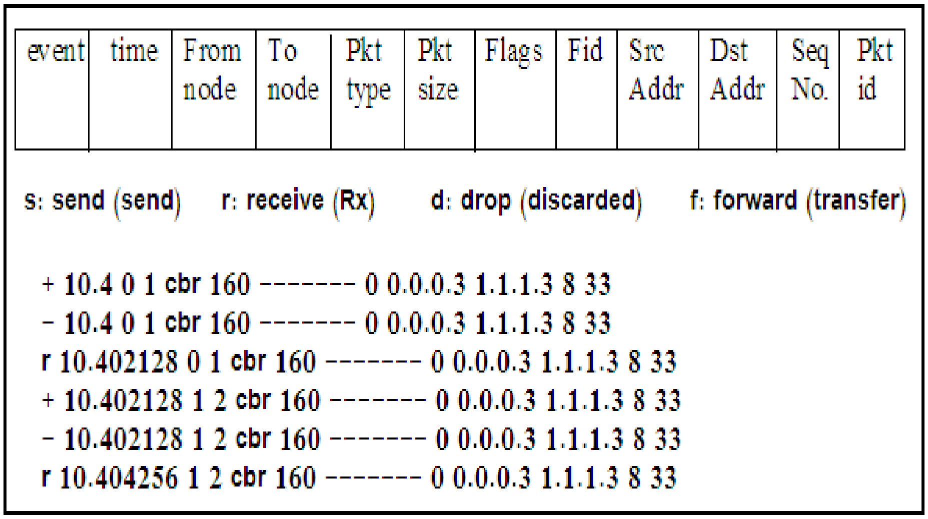
```
void DtRrQueue::enque(Packet* p)
{
  hdr_ip* iph = hdr_ip::access(p);
    q1_->enque(p);
    if ((q1_->length)> qlim_) {
      q1_->remove(p);
      drop(p);
    }
  }
```

Figure 4.6 Enqueing packets

The name of the C++ file for this queue objects "DtRrQueue" (Drop-Tail Round-Robin Queue) that is derived from "Queue" class. The matching OTcl name is "Queue/DTRR". When the "recv" member function that is implemented in the "Queue" class (in "queue.cc") receives a packet, it invokes the "enqueue" member function of the queue object and invokes "dequeue" if the link object is not blocked. When the link came from a blocked state, it also calls the "dequeue" member function of its queue object. Therefore, it was needed to write "enqueue" and "dequeue" member functions of the "DtRrQueue" object.

4.4 Trace file analysis

User scripts are normally written in tcl-code, with file extension .tcl. When the tcl scripts are run through the simulator, an output trace file is created. The trace file has the extension .tr and simulation events are collected here. The trace file contains very detailed information of the discrete-time events of the simulation.



event	time	From node	To node	Pkt type	Pkt size	Flags	Fid	Src Addr	Dst Addr	Seq No.	Pkt id

s: send (send) r: receive (Rx) d: drop (discarded) f: forward (transfer)

```
+ 10.4 0 1 cbr 160 ------- 0 0.0.0.3 1.1.1.3 8 33
- 10.4 0 1 cbr 160 ------- 0 0.0.0.3 1.1.1.3 8 33
r 10.402128 0 1 cbr 160 ------- 0 0.0.0.3 1.1.1.3 8 33
+ 10.402128 1 2 cbr 160 ------- 0 0.0.0.3 1.1.1.3 8 33
- 10.402128 1 2 cbr 160 ------- 0 0.0.0.3 1.1.1.3 8 33
r 10.404256 1 2 cbr 160 ------- 0 0.0.0.3 1.1.1.3 8 33
```

Figure 4.7 Trace file format

Figure 4.7 shows the trace file format and a piece of the trace file taken from one of these project scenarios. Each trace line starts with an event (+, -, d, r) descriptor followed by the simulation time (in seconds) of that event, and from and to node, which identify the link on which the event occurred. The next information in the line before flags (appeared as "------" since no flag is set) is packet type and size (in Bytes). The next field is flow id (fid) of IPv6 that a user can set for each flow at the input OTcl script. Even though fid field may not be used in a simulation, users can use this field for analysis purposes. The fid field is also used when specifying stream color for the NAM display. The next two fields are source and destination address in forms of "node**.**port". The next field shows the network layer protocol's packet sequence number. Note that even though UDP implementations do not use sequence number, NS keeps track of UDP packet sequence number for analysis purposes. The last field shows the unique id of the packet.

Perhaps the biggest challenge with using Ns-2 is to extract the desired information from the trace file. These files can be very large, so an intelligent way of manipulating the trace file is necessary in order to be able to analyze the wanted data. More sophisticated manipulation techniques can be implemented by writing scripts in Perl or in Awk, which are separate programming languages that can be utilized for processing trace files. For analysis of the trace files for this project, awk used in order to extract desired data and to calculate some metrics such as packet loss.

4.5 Conclusion

Several schemes have been proposed to enhance Mobile IPv6 performance. Most of them focused on reducing packet loss during handover, and this project proposed a scheme to enhance MIPv6 performance in term of reducing packet loss. Some of the implementation using ns-2 has been showed in this chapter.

CHAPTER 5

EXPERIMENTS AND RESULTS

5.1 Introduction

To accomplish this project, a simulation for the proposed scheme will be done by using ns-2 and MobiWan. After studying the original MIPv6 and existing schemes those proposed to enhance MIPv6, some of them have good features to reduce packet loss while others have some limitations. Comparing the result of the original MIPv6 with the proposed scheme in this project will test its quality among the original one.

This study evaluates the performance of the proposed scheme using ns-2 and its extension MobiWan that supports IPv6. This study includes designing the simulation model and setting the parameters for it. Real-time packets have been used in order to evaluate the rate of reducing packet loss in the proposed scheme. Also the .awk files have been used to get results from simulation trace files.

5.2 The network model

The network model is as shown in Figure 5.1 with five nodes, consists of a central router in the middle and one CN attached to CR by a wired network LAN, while two access routers are connected to the Central Router (CR). Function of these two access points is to connect the wired topology to the wireless nodes, and they also have routing functionality in the sense that they exchange routing information with other nodes and they route packets towards the correct destination. The MN will be attached by wireless connection to one of the access routers in the beginning of the simulation, serving as a Home Agent.

In Ns-2, wireless simulations require the nodes to be located with x, y and z coordinates in the simulation topography grid in order to simulate motion and radio range between nodes. The locations of the access routers and the mobile node are the only relevant positions for the wireless simulation. The positions of router R1 and the CN are not specified in the simulation script, as they are redundant.

The movement of the MN is linear. It starts its movement from the position (230,100) at t=10s, and moves with constant speed towards the destination (230,500) where it eventually stops. The home agent access router, AR-1, is located in position (x= 100, y =100). The other access router, AR-2 is located in the grid in position (x= 100, y=550), giving the two nodes a separation of 450 meters. This was enough to give us handovers about midway between them with our radio parameter settings for transmitter power and antenna types.

The wireless coverage area of each AR is 250m and both coverage areas are overlapped within a few meters, and the MN moves between them at the speed of 10 meter/second (36km/Hr). The CN will be used to send real-time traffic to the MN.

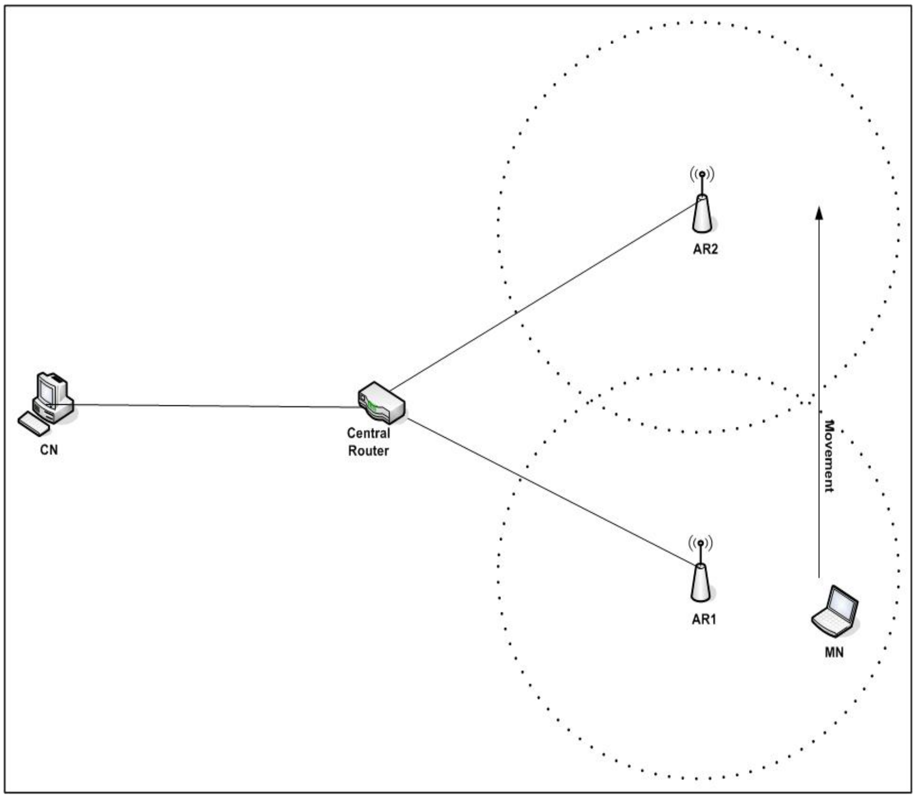


Figure 5.1: The network model

5.3 The simulation model

The performance of the new scheme will be evaluated by simulating the movement of the Mobile Node between two access routers (AR) as shown in Figure 5.4. This model is small enough to be handled efficiently within ns-2. The type of the network that the MN attached to the ARs is 5Mbps Wireless LAN 802.11 provided in ns-2. The links between the nodes are set to a transmission rate of 10Mbit/s and with a delay of 2ms on each link as shown in Figure 5.2 a tcl piece of code:

```
$ns duplex-link $cn_ $router_  10Mb 2.0ms DropTail
$ns duplex-link $router_ $bs1_ 10Mb 2.0ms DropTail
$ns duplex-link $router_ $bs2_ 10Mb 2.0ms DropTail
```

Figure 5.2 Define links between nodes.

Mobility is simply carried out by letting the mobile node move along a straight line with constant velocity until reaches a foreign subnet. When the MN is located within the coverage area of the other access router, AR-2, it performs handover to this cell. Once the MN detects it is on a foreign subnet, it does a binding update towards its home agent to inform it with the new address (CoA). Furthermore, while the MN is receiving packets from the CN, it also sends BU to this node. And this period is critical for packet loss.

The addressing structure is for the topology built up hierarchically, dividing the topology into domains, clusters and nodes. The addresses as shown in Figure 5-2 consist of 3 parts, where the first number indicates domain, the second indicates cluster and the last one indicates node number. This gives us two domains, where the CN is located in domain 0 and all other nodes belongs to domain 1. Furthermore, there are 4 clusters – cluster 0 contains the router and each of the 2 base stations is located in a separate cluster (cluster 1.1 to 1.2). In this way the movement of the MN goes through different clusters of the topology since each cell represents a new cluster. The last number is the node numbering, which restarts within every cluster.

In the simulation script it must be defined how many domains there are, how many clusters there are within every domain and finally how many nodes there are in each cluster. The topology and addressing structure as shown in Figure 5.3 are defined in this way in the script.

```
AddrParams set domain_num_ 2
AddrParams set cluster_num_ {1 3}
AddrParams set nodes_num_ {1  1 2 1}
```

Figure 5.3 Topology and addressing structure

The first line defines two domains; the second line defines one cluster in the first domain and three in the second. The third line defines the number of nodes within each cluster, starting from the lowest numbered cluster.

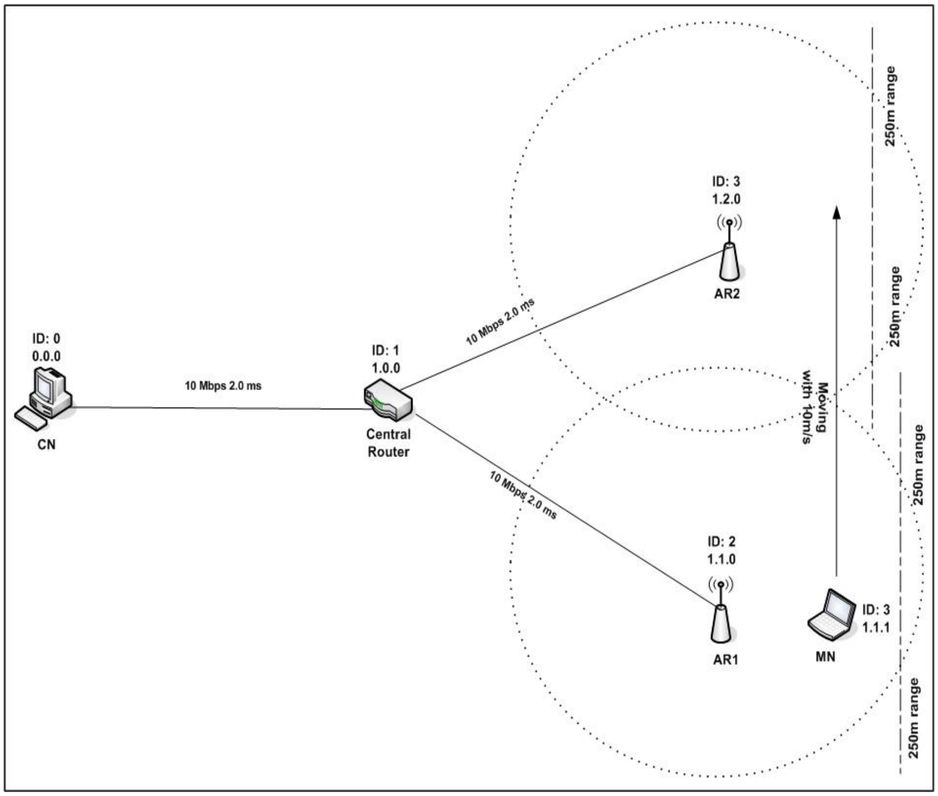


Figure 5.4 the Simulation model

5.4 Simulation tools

To achieve this project some simulation tools have been used and text processing language to analyze the output files.

5.4.1 Network Simulator ns-2

Network Simulator (NS) is an object-oriented, discrete event simulator for networking research. NS provides substantial support for simulation of TCP, UDP, routing and multicast protocols over wired and wireless networks this simulator ns-2 has been used to simulate the simulation topology of this project, over that some C++ files have been added in order to support some features not included or supported by the old version of ns-2 that used to simulate the project's simulation.

5.4.2 Mobiwan

Ns-2 does not support MIPv6; therefore, a simulation tool (MobiWan) was used in this project to support MIPv6. This extension of MobiWan based on ns-2 meant to simulate Mobile IPv6 under large Wide-Area Networks (both local-area mobility and global-area mobility).

5.4.3 AWK

Awk is a full-featured text processing language with syntax reminiscent of C. Awk makes it possible to handle simple data-reformatting jobs with just a few lines of code. It is used to analyze the output trace files those generated from running the simulation, and give numerical results. These data are used to plot the charts.

5.5 Simulation scenarios

The scenarios in this project are divided into two categories; first one five scenarios without buffer, and second one five scenarios with buffer. In the first category each scenario has the same parameters as shown in Table 5.1. The difference between them is the interval time of sending data from CN to MN. The first scenario start to send at t=10s and stop at t=15s is defined in the script in Figure 5.5

```
$ns at 10.0 "$src start"
$ns at 15.0 "$src stop"
```

Figure 5.5 First scenario start and stop time

While the second one continue to send from t=20s to t=25s, the third scenario start at t=10 and stops at t=15, and from t=20s till t=25s, and from t=30s till t=35s as Figure 5.6 shows.

```
$ns at 10.0 "$src start"
$ns at 15.0 "$src stop"
$ns at 20.0 "$src start"
$ns at 25.0 "$src stop"
$ns at 30.0 "$src start"
$ns at 35.0 "$src stop"
```

Figure 5.6 Third scenario start and stop time

while the fourth scenario same as third one but increase the sending periods with one more from t=40s till t=45s, the last one also start at t=10 like the anteceded scenarios but send an extra period from t=50s till second t=55 as Figure 5.7 shows.

```
$ns at 10.0 "$src start"
$ns at 15.0 "$src stop"
$ns at 20.0 "$src start"
$ns at 25.0 "$src stop"
$ns at 30.0 "$src start"
$ns at 35.0 "$src stop"
$ns at 40.0 "$src start"
$ns at 45.0 "$src stop"
$ns at 50.0 "$src start"
$ns at 55.0 "$src stop"
```

Figure 5.7 Fifth scenario start and stop time

In the second category the parameters just like the first category and they have the same periods of sending but with an exception these scenarios have buffers.

Parameters	**Value**
Transmission range	250m
Simulation time	100s
Topology size	800x800
Number of Mobile nodes	1
Number of sources	1
Traffic type	Constant bit rat
Interval	0.05 ms
Packet size	160 byte
Pause time	5 s
Maximum speed	10m/s

Table 5.1 the parameters used in all scenarios

5.6 Simulation metrics

The goal of this project is to measure some metrics of the standard simulated topology and the proposed scheme, there are 3 basic metrics of interest to describe how the mobile node performs:

5.6.1 Throughput

Throughput refers to the data rate (bits per second) generated by the application. Throughput measured in the number of bits per second, some time is called bit rate or bandwidth. The throughput during subnet change is of interest because it will indicate how much and for how long the throughput will drop when the MN performs a change of subnet and thereafter goes through the phases of binding update (BU) and route optimization (RO).

5.6.2 Delay

End-to-end delay refers to the time taken for a packet to be transmitted across a network from source to destination. Long delays may cause incidents such as data missing the playback points, and that suffers real-time data. Overall there are many sources across the network cause the delay between source and destination; source-processing delay, transmission delay, network delay, and destination processing delay.

5.6.3 Packet loss

The packet loss is the number of packets that are lost during the handover process. In general, in wireless and mobile networks packet loss is mostly caused by bit errors in an error-prone wireless channel, congestion in the network, or due to handover. The main reason for packet loss caused by handover is the fact that packets are routed to the old access point while the link to the old access point is already broken. These packets might be dropped by the old access point.

5.7 Results discussion

This section lists the result from both the standard and proposed scheme simulation scenarios. First, an overview of the test procedure for the standard topology simulation and then the results are presented. The same thing is done for the proposed scheme simulation scenario. All the results are derived from the trace files that were generated from each simulation scenario using .awk files.

5.7.1 Results of handover without buffer scenarios

It assumed only one mobile node roaming in the given topology; the MN will receive stream of data sent from the CN starting from t=5s. The UDP protocol used with Constant Bit Rate (CBR) traffic generator to generate packets with 160 byte size with 0.05ms interval (64-kb/s audio), and the total simulation run time is t=100s as shown in the tcl piece of code in Figure 5.8.

```
set opt(stop) 100
………..
Set src[new Application/Traffic/CBR]
$src set packetSize_ 160
$src set rate_ 64k
$src set interval_ 0.05
$src attach-agent $udp
```

Figure 5.8 Setting simulation parameters

From the Figure 5.9 it is clear that the delay increased from scenario one to scenario five, and as mentioned before each scenario sends data for different period of time and is increased gradually from first one to the last one. It can be seen that delay increases from an average of around 7.02ms when the MN on the home subnet, and while the handoff starts the duration of sending going higher the delay increases to be around 10.02ms in fifth scenario.

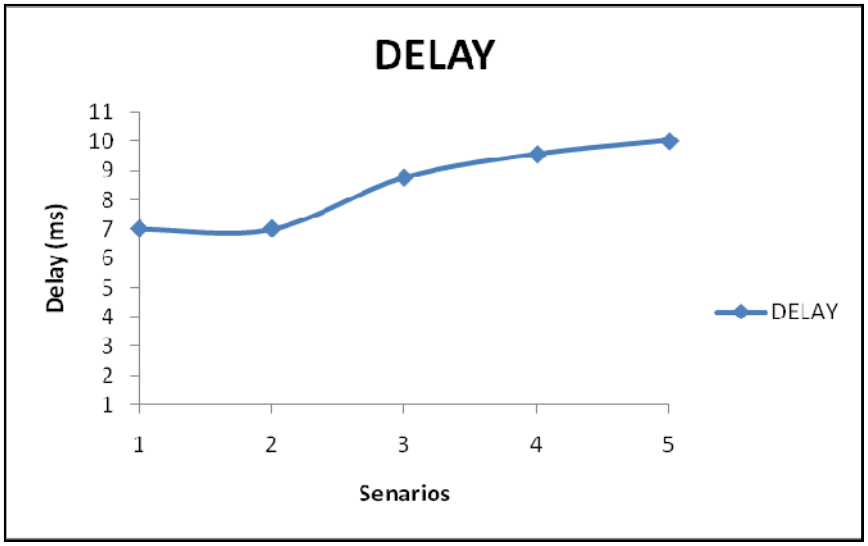


Figure 5.9 average delays for without buffer scenarios

In Figure 5.10 the throughput starting high around 25.6kbps and going down when the handoff of the MN starts in companion of increasing the sending periods from scenario to another and till the last one the throughput is 13.3kbps.

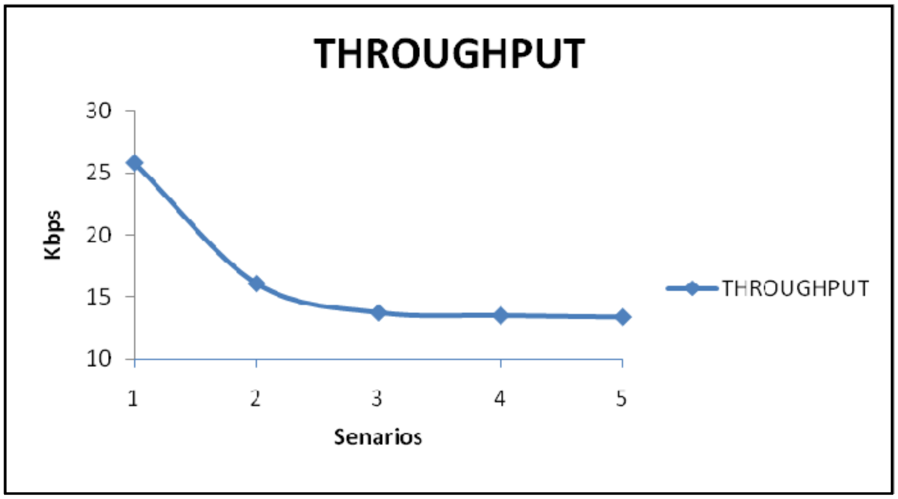


Figure 5.10 average throughput for without buffer scenarios

The ratio of packet loss in the standard simulation topology is as shown in Figure 5.11 where the ratio is zero (i.e.no packet loss) because the MN receives the sent packets while still attached to its Home Agent. In second scenario it is moving to another subnet the packet loss will occur, because the handover starts at t=23s and the mobile node receives 167 packets during sending periods. Thus the ratio becomes 16.5 percent and it is the highest ratio in all the scenarios. During third scenario still the handover occurs but because the periods increased so the received packets number increased as well and becomes 268 packets and the ratio went down to be 10.9635 percent because of MN attached itself to AP-2 and sent its BU to HA and CN. In fourth scenario the number of received packets increased to be 369 packets but the ratio went down and became 8.20896 percent. However, the numbers of sent packets are still increase till it became 470 packets in fifth scenario because the periods of sending packets from CN to MN increased and the packet loss ratio became in this scenario 6.56064 percent and this ratio the lowest one among other scenarios.

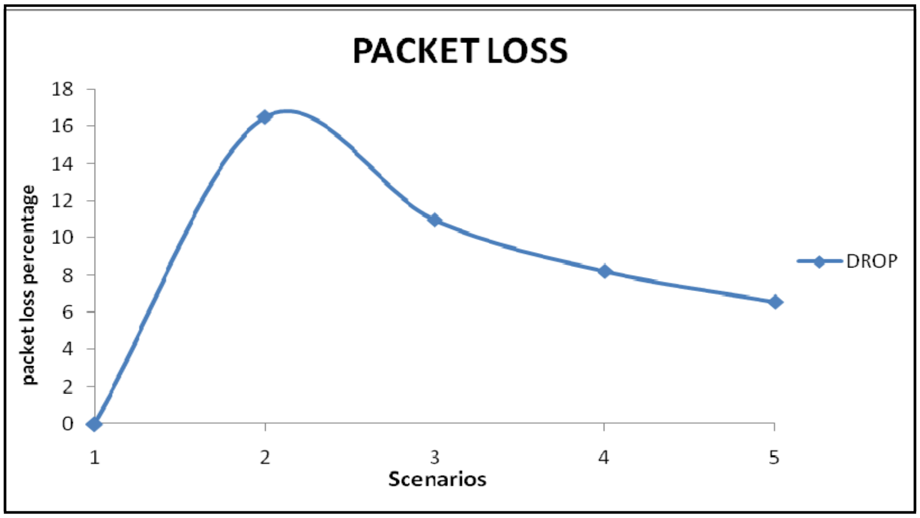


Figure 5.11 packet losses ratio for without buffer scenarios

5.7.2 Results of handover with buffer scenarios

As the same parameters for the last scenarios, these scenarios will test the metrics but with using buffer.

In Figure 5.12 the delay starts at 7.02ms and it will start getting higher a little bit when MN moving out of its Home Agent coverage area and starting the handoff time till MN attaches itself to new Access Router. At the last scenario the delay was around 7.26ms.

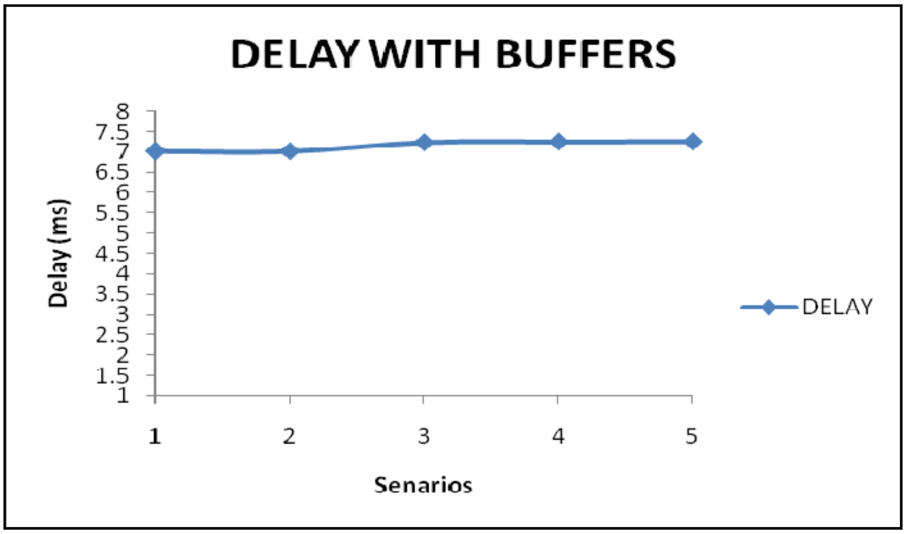


Figure 5.12 average delay for with buffer scenarios

As shown in Figure 5.13 the throughput of buffer scenarios starting at highest point around 25.8kbps, when the handoff of the MN starts the throughput will be lower around 16.03kbps , but when the periods of sending data from CN to MN increased the throughput starts to go higher till it gets 21.7kbps in the last scenario.

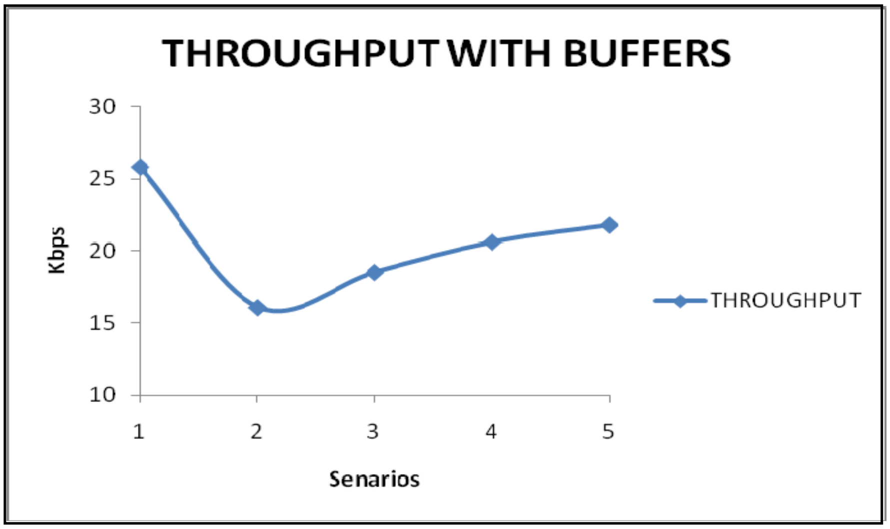


Figure 5.13 average throughput for with buffer scenarios

The Figure 5.14 shows the packet loss ratio with buffer. It starts with zero while MN still attached to its Home Agent and the handover procedure does not start yet. But the lost packets ratio increased when MN starts moving to another subnet and the handover starts at t=23s in second scenario and the packet loss ratio becomes 4.0229 percent. In third scenario the number of sending periods increased so the number of sent packets is increased as well but the ratio of packet loss going down to be 2.5454 percent that reduce in percentage because the MN attached to AP-2 and sent the BU to its HA and CN. During fourth scenario the ratio decreased to be 1.8617 percent. And finally in fifth scenario the number of sending periods increased and number of sent packets also increased and the ratio of lost packets become 1.4675 percent.

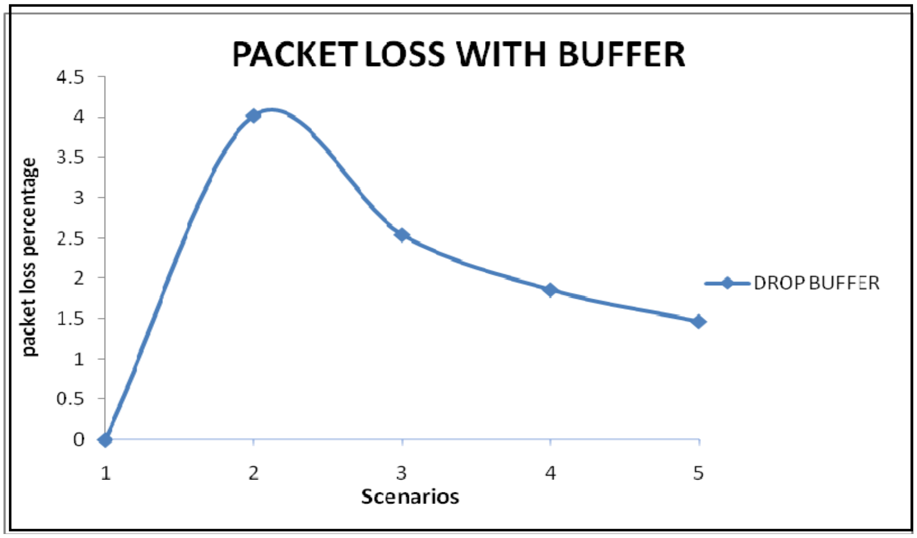


Figure 5.14 packet losses ratio with buffer scenarios

5.7.3 Results comparison

Five scenarios were compared in this study, some of them without buffer and others with buffer.

It can be observed in Figure 5.15 that the average delay for without buffer scenarios increasing when the MN starts moving and going outside its Home Agent coverage area and going higher while the sending periods increasing. With buffer the average delay starts as without buffer schemes and also increased but with a very low rate compared to the standard scenarios. That means buffering packets saves the delay.

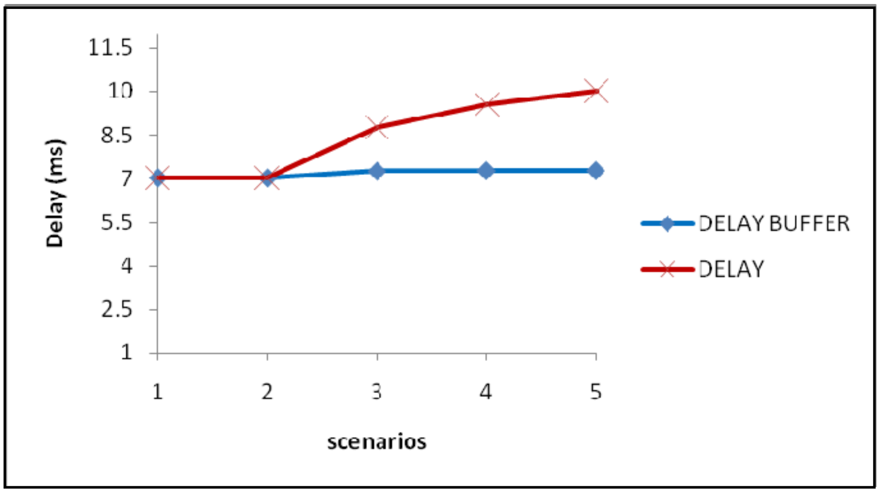


Figure 5.15 Delay comparison

The Figure 5.16 shows that the average throughput decreased significantly while MN moves out of its home network, but the average throughput will increase significantly by using buffer and the increases start when MN moves and the periods of sending data from CN to the destination (MN) increases.

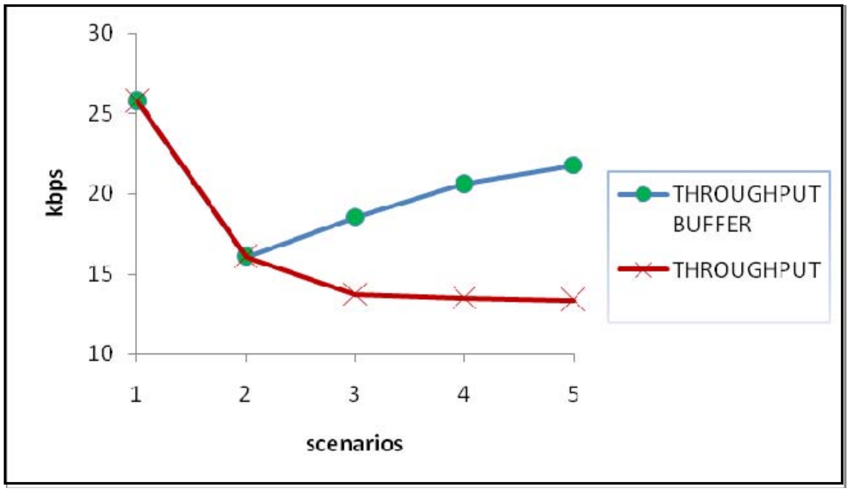


Figure 5.16 the average throughput comparison

The ratio of lost packets in both schemes starts with zero because MN does not start its handoff as shown in Figure 5.17. Due to the handoff of MN and since the MIPv6 does not support buffering, so the ratio of lost packets was high about 16.5 percent in second scenario and going down in other scenarios till last scenario it becomes around 6.56 percent. And this ratio still high and that’s really suffers the real-time sessions.

With buffer scheme also the ratio starts at zero and increased starting from second scenario to be around 4.02 percent but it is going down in the remaining scenarios third, fourth, and fifth scenario. To be in the fifth scenario around 1.46 percent and that really reduces the packet loss ratio in compare to the standard scenarios. The packet loss decreased due to buffering these packets until the MN sends the BU to CN the sender in this case to resume sending packets. In fact the number of lost packet in last four standard scenarios is 33 packets and in last four buffer scenarios is 7 packets but the ratio is different due to the number of received packets because the period of sending data increased from one scenario to another one.

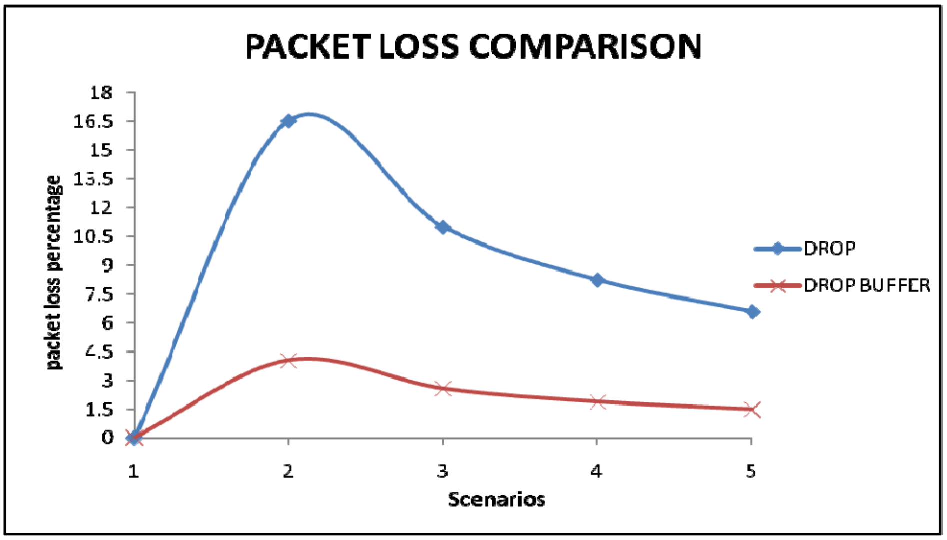


Figure 5.17 Packet loss comparisons

5.8 Conclusion

The MIPv6 has been proposed to overcome MIPv4 drawbacks. MIPv6 also has some limitations; many schemes have been proposed to enhance the quality of service in MIPv6 especially during real-time traffic. Each of these schemes has its advantages and disadvantages. This project proposed a new scheme to reduce real-time traffic packet loss in MIPv6.

Network simulator ns-2 has been used by the researchers who are interested in networks because it is open source software and has the capability to simulate networks objects. This project used with its extension mobiwan that supports IPv6 to simulate the new scheme and test its efficiency.

The results of the standard scheme and proposed scheme have been discussed in this chapter and a comparison between them was shown as well.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Introduction

This chapter concludes this study that deals with proposing a new scheme to reduce real-time traffic packet loss in MIPv6 environment. Finally a future work for this study has been given in the end of this chapter.

6.2 Achievements

This study presented a new scheme to reduce real-time packet loss in Mobile IPv6 environment. In order to achieve its goal, this study is divided into four stages. The first stage focuses on the literature review, investigating and analyzing the existing work in order to determine the advantages and limitations of each existing scheme.

Second stage includes framework design for the proposed scheme and presents the simulation model design for the proposed scheme using the ns-2 and MobiWan and setting the parameters for the simulation such as links bandwidth and the delay time. Implementing the scheme and evaluating the performance for it and redesigning it to achieve best performance are presented in the third stage. Discussing and analyzing the results of the proposed scheme has been done in the fourth stage.

6.3 Challenges and Limitations

This project faced some challenges during the execution. Finding the compatability between the version of mobiwan and the version of ns-2 that supports mobiwan was one of the challenges. Because the original mobiwan was implemented to work with ns-2.1b6 and this old version does not work properly on any distribution of linux, but on old versions of linux. And it does not support some new features that added to new versions of ns-2.

6.4 Future work

During the work with this project, we have seen many areas where further work and exploration might be of great interest.

Chapter 2 has provided a brief introduction to the topic hierarchical mobile IPv6 operation. As Mobiwan has no support for this, it is not possible to carry out any experiments with hierarchical mobile IP structures. This is an area where much research effort is spent currently, and it has been claimed that the performance of hierarchical mobile IPv6 is performing better than normal mIPv6 with respect to handover performance. It is an interesting topic for research. In this project the scenario was as simple as to carry out the proposed scheme and evaluate its efficiency. It is good to extend the scenario with many mobile nodes, to evaluate the scheme efficiency.

After experiencing limitations and some incorrectness of Mobiwan with ns-2, this study would like to suggest modeling and simulations of mobile IPv6 with other tools.

REFERENCES

[1] Forouzan, B. A. (2006), “TCP/IP Protocol Suite” third edition, Mc graw – hill.

[2] Jamalipour, A. (2002), “The Wireless Mobile Internet Archetecture, Protocols and Services”, Wiley.

[3] Perkins, C. (1996), “IP Mobility Support”, RFC 2002, Network Working Group.

[4] Johnson, D. (2003), Perkins, C., Arkko, J., “Mobility Support in Ipv6”, internet draft, ietf.

[5] Postel, J. (1981), “Internet Protocol”, RFC 791, http://www.ietf.org/rfc/rfc791.txt.

[6] Postel, J. (1981), “Transmission Control Protocol”, RFC 793, http://www.ietf.org/rfc/rfc793.txt.

[7] InterntProtocolv6, http://www.ipv6.com/articles/general/ipv6-the-next generation-internet.htm.

[8] “Internet Protocol, Version 6 (IPv6) Specification”, RFC 2460 http://www.ietf.org/rfc/rfc2460.txt

[9] Perkins, C. (1998), “Mobile Networking in the Internet”, ACM/Blatzer Mobile Networks and Applications (MONET) Journal, vol.3: pp.319-334.

[10] Beloeil Luc (2001), “Overview and Introduction to the Mobile IPv6 Protocol”, EURESCOM Technical Information, 3-15.

[11] Shanmugam R., Padamini R., Nivedita S. (2002), “Using TCP/IP, 2nd edition”.

[12] Network Simulator ns-2, http://nsnam.isi.edu/nsnam.

[13] Ernest, T. (2001), “MobiWan: A NS-2.1b6 simulation platform for Mobile IPv6 in Wide Area Networks”, MobiWan official Manual.

[14] Wei-Men Yao, and Yaw-Chung Chen (2004), “An Enhanced Buffer Mnagement Scheme for Fast Handover Protocol”, IEEE.

[15] T.C. Schmidt, M. Wahlisch, H.L. Cycon, M. Palkow (2003), “ Mobility Support in Real-time Video Communication”, the society for Modeling and Simulation International, California, 72-77.

[16] Vogt, C. (2006), “A Comprehensive and Efficient Handoff Procedure for IPv6 Mobility Support”, Proceedings of the 2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'06), IEEE.

[17] “Neighbor Discovery for IP version 6 (IPv6)”, RFC 4861, http://tools.ietf.org/html/rfc4861.

[18] Perkins, C. (2000), “Fast Handovers for Mobile IPv6”, internet draft, ietf.

[19] “Fast handovers for Mobile IPv6”, EURESCOM Technical Information, 2002. 1-15.

[20] Costa, X. P., Moreno, M. T., Hartestein, H., “A Performance Comparison of Mobile IPv6, Hierarchical Mobile IPv6, Fast Handovers for Mobile IPv6 and their Combination”, Mobile Computing and Communications Review, Volume 7, Number 4.5-19.

[21] “Hierarchical Mobile IPv6 Mobility Management (HMIPv6)”, internet draft, ietf, November 2007.

[22] “Tunnel Buffering for Mobile IPv6”, internet draft, Mobility Optimizations Group, 09 July 2004.

[23] McNair, J., Akyildiz, I. F., Bender, M. D. (2001), “Handoffs for Real-Time Traffic in Mobile IP Version 6 Networks", IEEE, 3464-3467.

[24] Sessini,P., Leventer, M. and Mahanti, A.(2007), “ Video to Go: The Effiects of Mobility on Streaming Media in a CDMA2000 1xEV-DO Network", Multimedia Computing and Networking.

[25] Schmidt, T. C., Wahlisch, M.(2003), “Roaming Real-Time Applications - Mobility Services in IPv6 Networks", Proceedings TERENA Networking Conference Zagreb, 1-15.

[26] Koodli, R. (2002), “Fast Handovers for Mobile IPv6", Internet Draft.

[27] Bentham, J. (2002), “TCP/IP LEAN Web Servers for Embedded Systems", 2nd edition, CMP Books.

[28] Parsad, Anand., Neeli., (2005) “802.11 WLANs and IP Networking security, QoS, and mobility”,1st edition, Artech House.

[29] Ganz, A., Ganz, Z., and Wongthavarawat, K.,(2004),”Multimedia Wireless Networks technologies, standards, and QoS”1st edition, Prentice Hall.

[30] Bentham, J., (2002),”TCP/IP Lean, web servers for embedded systems”, 2nd edition, CMP Books.

[31] Russell, R., (2000), ”Telecommunications Protocols”, 2nd edition, McGraw-Hill.

[32] “The Recommendation for the IP Next Generation Protocol”, RFC1752, ftp://ftp.isi.edu/in-notes/rfc1752.txt.

[33] “Internet Protocol, Version 6 (IPv6) Specification”, RFC 2460, http://www.ietf.org/rfc/rfc2460.txt.

[34] Ammann, P.,(2000),”Managing Dynamic IP networks”1st edition, McGrow-Hill.

[35] Blanchet, M.,”Migrating to IPv6, A practical guide to implementing IPv6 in mobile and fixed networks”,1st edition, Wiley.

[36] Soliman, H., (2004),”Mobile IPv6 Mobility in a Wireless Internet”,1st edition, Addison-Wesley.

[37] Hersent, O., Gurle, D., Petit, J.,(2000)“IP Telephony Packer-based multimedia communications systems”, 1st edition, Addison-Wesley.

[38] “Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification”; IETF RFC 1885, www.faqs.org/rfcs/rfc1885.html.

[39] W. Stallings (1997), “Data and computer communications”, fifth edition, Prentice-Hall.

APPENDIX A

TCL CODE

```
# FIRST SCENARIO WITHOUT BUFFER
# Basic Mobile IPv6 example without using ns-topoman
# Needs proc defined in file proc-mipv6-config.tcl
#
#
#
#                                            AR2
#                                             /                  ^
#                                            /                   |
#                                          /10Mbps/2ms           |
#                    10Mbps/2ms          /                       |
#            CN-------------------CR                             |Movment
#                                        \                       |Direction
#                                          \10Mbps/2ms           |
#                                            \                   |
#                                             \                  |
#                                             AR1                MN
#
###########################################################################
####


Agent/MIPv6/MN set bs_forwarding_     0       ; # 1 if forwarding from previous BS
####################################################################
proc log-mn-movement_no_topo { } {
  global logtimer ns
  Class LogTimer -superclass Timer
  LogTimer instproc timeout {} {
```

```
          global mobile_
       $mobile_ log-movement
       $self sched 1
  }
  set logtimer [new LogTimer]
  $logtimer sched 1
}


##################################################################
# Create Topology
##################################################################

proc create-my-topo {} {
  global ns opt topo mobile_ cn_ mnn_nodes_

  # Create and define topography
  set topo       [new Topography]
  #   set prop       [new $opt(prop)]
  #   $prop topography $topo
  $topo load_flatgrid  800 800

  # god is a necessary object when wireless is used
  # set to a value equal to the number of mobile nodes
  create-god 5

  # Call node-config
  $ns node-config \
       -addressType hierarchical \
          -agentTrace On \
          -routerTrace Off\


  # Set NS Addressing
  AddrParams set domain_num_ 2
  AddrParams set cluster_num_ {1 3}
  AddrParams set nodes_num_ {1 1 2 1}

  # Create Nodes
  set cn_ [create-router 0.0.0]
  set router_ [create-router 1.0.0]
  set bs1_ [create-base-station 1.1.0 1.0.0 100 100 0]
  set bs2_ [create-base-station 1.2.0 1.0.0 100 550 0]
  set mobile_ [create-mobile 1.1.1 1.1.0 230 100 0 0 0.01]
```

```
  # Create Links
   $ns simplex-link $cn_ $router_ 10Mb 2.0ms DropTail
   $ns duplex-link $router_ $bs1_ 10Mb 2.0ms DropTail
   $ns duplex-link $router_ $bs2_ 10Mb 2.0ms DropTail


   display_ns_addr_domain
}

################################################################
# End of Simulation
################################################################

proc finish { } {
  global tracef ns namf opt mobile_ cn_

  puts "Simulation finished"
  # Dump the Binding Update List of MN and Binding Cache of HA
  [[$mobile_ set ha_] set regagent_] dump
  [$cn_ set regagent_] dump
  [$mobile_ set regagent_] dump

  $ns flush-trace
  flush $tracef
  close $tracef
  close $namf
  puts "running nam with $opt(namfile) ... "
  #exec nam $opt(namfile) &


  exit 0
}


################################################################
# Main
################################################################

proc main { } {
   global opt ns TOPOM namf n tracef mobile_ cn_
   # Source Files
   # source set-def-options.tcl
   # set home/salim to your own correct path
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mipv6-config.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-tools.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-topo.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/ns-topoman.tcl
```

```
source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mobi-global.tcl
source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mobi-config.tcl
source ~/ns-allinone-2.28/ns-2.28/tcl/mobility/timer.tcl

set NAMF out.nam
set TRACEF out.tr
set INFOF out.info

set opt(mactrace) ON
set opt(NAM) 1
set opt(namfile) $NAMF
set opt(stop) 100
set opt(tracefile) $TRACEF

#>--------------- Extract options from command line ---------------<
#Getopt        ; # Get option from the command line
#DisplayCommandLine

#>---------------------- Simulator Settings ----------------------<
set ns [new Simulator]
#>------------------------ Open trace files ----------------------<
exec rm -f $opt(tracefile)
set tracef [open $opt(tracefile) w]
#... dump the file

#set new trace file for wireless
$ns use-newtrace

$ns trace-all $tracef

set namf [open $opt(namfile) w]
$ns namtrace-all $namf

#>------------- Protocol and Topology Settings -------------------<

create-my-topo
log-mn-movement_no_topo
#############
set-cbr
 # set-ping-int 0.1 $cn_ $mobile_ 10 $opt(stop)

#start movement to pos(x,y) with velocity v
#$ns at 10.0 "$mobile_ setdest 700 400 10"
#MN move again to BS2#####
$ns at 2.0 "$mobile_ setdest 230 500 10"
```

```
   #>---------------------- Run Simulation ------------------------<

   $ns at $opt(stop) "finish"
   $ns run

   $ns dump-topology $namf
   close $namf
     puts "running nam with $opt(namfile) ... "
     exec nam $opt(namfile) &
}

 proc set-cbr { } {
    global ns cn_ mobile_
    set udp [new Agent/UDP]
    $ns attach-agent $cn_ $udp

    set dst [new Agent/Null]
    $ns attach-agent $mobile_ $dst
    $ns connect $udp $dst

    set src [new Application/Traffic/CBR]
    $src set packetSize_ 160
    $src set rate_ 64k
    $src set interval_ 0.05
    $src attach-agent $udp

    $ns at 10.0 "$src start"
    $ns at 15.0 "$src stop"



 }

main
```

```
# SECOND SCENARIO-WITHOUT BUFFER
# Basic Mobile IPv6 example without using ns-topoman
# Needs proc defined in file proc-mipv6-config.tcl
#
#
#
#                                   AR2
#                                   /             ^
#                                  /              |
#                                 /10Mbps/2ms     |
#               10Mbps/2ms       /                |
#          CN-------------------CR                |Movment
#                                \                |Direction
#                                 \10Mbps/2ms     |
#                                  \              |
#                                   \             |
#                                   AR1           MN
#
##########################################################################
####


Agent/MIPv6/MN set bs_forwarding_     0       ; # 1 if forwarding from previous BS
###################################################################
proc log-mn-movement_no_topo { } {
  global logtimer ns
  Class LogTimer -superclass Timer
  LogTimer instproc timeout {} {
        global mobile_
      $mobile_ log-movement
      $self sched 1
  }
  set logtimer [new LogTimer]
  $logtimer sched 1
}


###################################################################
# Create Topology
###################################################################
proc create-my-topo {} {
  global ns opt topo mobile_ cn_ mnn_nodes_

  # Create and define topography
  set topo       [new Topography]
```

```
  #   set prop        [new $opt(prop)]
  #   $prop topography $topo
  $topo load_flatgrid  800 800

  # god is a necessary object when wireless is used
  # set to a value equal to the number of mobile nodes
  create-god 5

  # Call node-config
  $ns node-config \
        -addressType hierarchical \
            -agentTrace On \
            -routerTrace Off\


  # Set NS Addressing
  AddrParams set domain_num_ 2
  AddrParams set cluster_num_ {1 3}
  AddrParams set nodes_num_ {1 1 2 1}

  # Create Nodes
  set cn_ [create-router 0.0.0]
  set router_ [create-router 1.0.0]
  set bs1_ [create-base-station 1.1.0 1.0.0 100 100 0]
  set bs2_ [create-base-station 1.2.0 1.0.0 100 550 0]
  set mobile_ [create-mobile 1.1.1 1.1.0 230 100 0 0 0.01]


  # Create Links
   $ns simplex-link $cn_ $router_ 10Mb 2.0ms DropTail
   $ns duplex-link $router_ $bs1_ 10Mb 2.0ms DropTail
   $ns duplex-link $router_ $bs2_ 10Mb 2.0ms DropTail


   display_ns_addr_domain
}

#################################################################
# End of Simulation
#################################################################
proc finish { } {
  global tracef ns namf opt mobile_ cn_

  puts "Simulation finished"
  # Dump the Binding Update List of MN and Binding Cache of HA
  [[$mobile_ set ha_] set regagent_] dump
  [$cn_ set regagent_] dump
```

```
 [$mobile_ set regagent_] dump

 $ns flush-trace
 flush $tracef
 close $tracef
 close $namf
 puts "running nam with $opt(namfile) ... "
 #exec nam $opt(namfile) &


 exit 0
}


##################################################################
# Main
##################################################################
proc main { } {
  global opt ns TOPOM namf n tracef mobile_ cn_
  # Source Files
  # source set-def-options.tcl
  # set home/salim to your own correct path
  source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mipv6-config.tcl
  source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-tools.tcl
  source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-topo.tcl
  source ~/ns-allinone-2.28/ns-2.28/tcl/lib/ns-topoman.tcl
  source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mobi-global.tcl
  source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mobi-config.tcl
  source ~/ns-allinone-2.28/ns-2.28/tcl/mobility/timer.tcl

  set NAMF out.nam
  set TRACEF out.tr
  set INFOF out.info

  set opt(mactrace) ON
  set opt(NAM) 1
  set opt(namfile) $NAMF
  set opt(stop) 100
  set opt(tracefile) $TRACEF

  #>--------------- Extract options from command line ---------------<
  #Getopt        ; # Get option from the command line
  #DisplayCommandLine

  #>---------------------- Simulator Settings ----------------------<
  set ns [new Simulator]
  #>------------------------ Open trace files ----------------------<
```

```
   exec rm -f $opt(tracefile)
   set tracef [open $opt(tracefile) w]
   #... dump the file

   #set new trace file for wireless
   $ns use-newtrace

   $ns trace-all $tracef

   set namf [open $opt(namfile) w]
   $ns namtrace-all $namf

   #>------------- Protocol and Topology Settings -------------------<
   create-my-topo
   log-mn-movement_no_topo
   #############
   set-cbr
    # set-ping-int 0.1 $cn_ $mobile_ 10 $opt(stop)

   #start movement to pos(x,y) with velocity v
   #$ns at 10.0 "$mobile_ setdest 700 400 10"
   #MN move again to BS2#####
   $ns at 2.0 "$mobile_ setdest 230 500 10"



   #>---------------------- Run Simulation ------------------------<
   $ns at $opt(stop) "finish"
   $ns run

   $ns dump-topology $namf
   close $namf
   # puts "running nam with $opt(namfile) ... "
  #  exec nam $opt(namfile) &
}

 proc set-cbr { } {
    global ns cn_ mobile_
    set udp [new Agent/UDP]
    $ns attach-agent $cn_ $udp

    set dst [new Agent/Null]
    $ns attach-agent $mobile_ $dst
    $ns connect $udp $dst

    set src [new Application/Traffic/CBR]
    $src set packetSize_ 160
```

```
   $src set rate_ 64k
   $src set interval_ 0.05
   $src attach-agent $udp

   $ns at 10.0 "$src start"
   $ns at 15.0 "$src stop"
   $ns at 20.0 "$src start"
   $ns at 25.0 "$src stop"


 }

main





#THIRD SCENARIO-WITHOUT BUFFER
# Basic Mobile IPv6 example without using ns-topoman
# Needs proc defined in file proc-mipv6-config.tcl
#
#
#
#                            AR2
#                            /           ^
#                           /             |
#                          /10Mbps/2ms    |
#            10Mbps/2ms   /               |
#        CN-------------------CR          |Movment
#                          \              |Direction
#                           \10Mbps/2ms   |
#                            \            |
#                            \            |
#                            AR1         MN
#
################################################################################

Agent/MIPv6/MN set bs_forwarding_     0       ; # 1 if forwarding from previous BS
####################################################################
proc log-mn-movement_no_topo { } {
  global logtimer ns
  Class LogTimer -superclass Timer
  LogTimer instproc timeout {} {
```

```
           global mobile_
       $mobile_ log-movement
       $self sched 1
  }
  set logtimer [new LogTimer]
  $logtimer sched 1
}


###############################################################
# Create Topology
###############################################################
proc create-my-topo {} {
  global ns opt topo mobile_ cn_ mnn_nodes_

  # Create and define topography
  set topo       [new Topography]
  #   set prop       [new $opt(prop)]
  #   $prop topography $topo
  $topo load_flatgrid  800 800

  # god is a necessary object when wireless is used
  # set to a value equal to the number of mobile nodes
  create-god 5

  # Call node-config
  $ns node-config \
       -addressType hierarchical \
           -agentTrace On \
           -routerTrace Off\


  # Set NS Addressing
  AddrParams set domain_num_ 2
  AddrParams set cluster_num_ {1 3}
  AddrParams set nodes_num_ {1 1 2 1}

  # Create Nodes
  set cn_ [create-router 0.0.0]
  set router_ [create-router 1.0.0]
  set bs1_ [create-base-station 1.1.0 1.0.0 100 100 0]
  set bs2_ [create-base-station 1.2.0 1.0.0 100 550 0]
  set mobile_ [create-mobile 1.1.1 1.1.0 230 100 0 0 0.01]
```

```
  # Create Links
   $ns simplex-link $cn_ $router_ 10Mb 2.0ms DropTail
   $ns duplex-link $router_ $bs1_ 10Mb 2.0ms DropTail
   $ns duplex-link $router_ $bs2_ 10Mb 2.0ms DropTail


  display_ns_addr_domain
}

###################################################################
# End of Simulation
###################################################################
proc finish { } {
  global tracef ns namf opt mobile_ cn_

  puts "Simulation finished"
  # Dump the Binding Update List of MN and Binding Cache of HA
  [[$mobile_ set ha_] set regagent_] dump
  [$cn_ set regagent_] dump
  [$mobile_ set regagent_] dump

  $ns flush-trace
  flush $tracef
  close $tracef
  close $namf
  puts "running nam with $opt(namfile) ... "
   #exec nam $opt(namfile) &


 #exec awk -f  loss.awk flow=0 src=0 dst=4 out.tr > loss.xgr
 #exec awk -f all.awk out.tr > All.xgr
  #exec awk -f mn.awk  out.tr > MN.xgr
# exec awk -f mn2.awk out.tr > MN2.xgr
 #exec awk -f bs2.awk out.tr > BS2.xgr
# exec awk -f bs1.awk out.tr > BS1.xgr
 #exec awk -f all_results.awk out.tr > allresults.xgr
#exec awk -f throughput.awk flow=0 src=0 dst=4 out.tr > thr.xgr
 # exec xgraph All.xgr MN.xgr &
#  BS1.xgr BS2.xgr thr.xgr &
# exec xgraph allresults.xgr &

  exit 0
}
```

```
####################################################################
# Main
####################################################################
proc main { } {
   global opt ns TOPOM namf n tracef mobile_ cn_
   # Source Files
   # source set-def-options.tcl
   # set home/salim to your own correct path
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mipv6-config.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-tools.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-topo.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/ns-topoman.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mobi-global.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mobi-config.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/mobility/timer.tcl

   set NAMF out.nam
   set TRACEF out.tr
   set INFOF out.info

   set opt(mactrace) ON
   set opt(NAM) 1
   set opt(namfile) $NAMF
   set opt(stop) 100
   set opt(tracefile) $TRACEF

   #>--------------- Extract options from command line ---------------<
   #Getopt        ; # Get option from the command line
   #DisplayCommandLine

   #>---------------------- Simulator Settings ----------------------<
   set ns [new Simulator]
   #>------------------------ Open trace files ----------------------<
   exec rm -f $opt(tracefile)
   set tracef [open $opt(tracefile) w]
   #... dump the file

#set new trace file for wireless
   $ns use-newtrace

   $ns trace-all $tracef

   set namf [open $opt(namfile) w]
   $ns namtrace-all $namf
```

```
   #>------------- Protocol and Topology Settings -------------------<
   create-my-topo
   log-mn-movement_no_topo
   #############
   set-cbr
    # set-ping-int 0.1 $cn_ $mobile_ 10 $opt(stop)

   #start movement to pos(x,y) with velocity v
   #$ns at 10.0 "$mobile_ setdest 700 400 10"
   #MN move again to BS2#####
   $ns at 2.0 "$mobile_ setdest 230 500 10"



   #>---------------------- Run Simulation ------------------------<
   $ns at $opt(stop) "finish"
   $ns run

   $ns dump-topology $namf
   close $namf
   # puts "running nam with $opt(namfile) ... "
  #  exec nam $opt(namfile) &
}

 proc set-cbr { } {
    global ns cn_ mobile_
    set udp [new Agent/UDP]
    $ns attach-agent $cn_ $udp

    set dst [new Agent/Null]
    $ns attach-agent $mobile_ $dst
    $ns connect $udp $dst

    set src [new Application/Traffic/CBR]
    $src set packetSize_ 160
    $src set rate_ 64k
    $src set interval_ 0.05
    $src attach-agent $udp
     $ns at 10.0 "$src start"
     $ns at 15.0 "$src stop"
     $ns at 20.0 "$src start"
     $ns at 25.0 "$src stop"
     $ns at 30.0 "$src start"
     $ns at 35.0 "$src stop"

     }
main
```

```
#FOURTH SCENARIO-WITHOUT BUFFER
# Basic Mobile IPv6 example without using ns-topoman
# Needs proc defined in file proc-mipv6-config.tcl
#
#
##                                  AR2
#                                  /              ^
#                                 /               |
#                                /10Mbps/2ms      |
#              10Mbps/2ms    /                    |
#          CN-------------------CR                |Movment
#                              \                  |Direction
#                               \10Mbps/2ms       |
#                                 \               |
#                                  \              |
#                                  AR1            MN
#
############################################################################
####

Agent/MIPv6/MN set bs_forwarding_     0       ; # 1 if forwarding from previous BS
####################################################################
proc log-mn-movement_no_topo { } {
  global logtimer ns
  Class LogTimer -superclass Timer
  LogTimer instproc timeout {} {
	global mobile_
       $mobile_ log-movement
       $self sched 1
  }
  set logtimer [new LogTimer]
  $logtimer sched 1
}


####################################################################
# Create Topology
####################################################################
proc create-my-topo {} {
  global ns opt topo mobile_ cn_ mnn_nodes_

  # Create and define topography
  set topo        [new Topography]
  #   set prop        [new $opt(prop)]
  #   $prop topography $topo
  $topo load_flatgrid  800 800
```

```
  # god is a necessary object when wireless is used
  # set to a value equal to the number of mobile nodes
  create-god 5

  # Call node-config
  $ns node-config \
        -addressType hierarchical \
            -agentTrace On \
            -routerTrace Off\


  # Set NS Addressing
  AddrParams set domain_num_ 2
  AddrParams set cluster_num_ {1 3}
  AddrParams set nodes_num_ {1 1 2 1}

  # Create Nodes
  set cn_ [create-router 0.0.0]
  set router_ [create-router 1.0.0]
  set bs1_ [create-base-station 1.1.0 1.0.0 100 100 0]
  set bs2_ [create-base-station 1.2.0 1.0.0 100 550 0]
  set mobile_ [create-mobile 1.1.1 1.1.0 230 100 0 0 0.01]


  # Create Links
   $ns simplex-link $cn_ $router_ 10Mb 2.0ms DropTail
   $ns duplex-link $router_ $bs1_ 10Mb 2.0ms DropTail
   $ns duplex-link $router_ $bs2_ 10Mb 2.0ms DropTail


  display_ns_addr_domain
}

##################################################################
# End of Simulation
##################################################################
proc finish { } {
  global tracef ns namf opt mobile_ cn_

  puts "Simulation finished"
  # Dump the Binding Update List of MN and Binding Cache of HA
  [[$mobile_ set ha_] set regagent_] dump
  [$cn_ set regagent_] dump
  [$mobile_ set regagent_] dump
```

```
  $ns flush-trace
  flush $tracef
  close $tracef
  close $namf
  puts "running nam with $opt(namfile) ... "
   #exec nam $opt(namfile) &


  #exec awk -f  loss.awk flow=0 src=0 dst=4 out.tr > loss.xgr
  #exec awk -f all.awk out.tr > All.xgr
  #exec awk -f mn.awk  out.tr > MN.xgr
  # exec awk -f mn2.awk out.tr > MN2.xgr
  #exec awk -f bs2.awk out.tr > BS2.xgr
  # exec awk -f bs1.awk out.tr > BS1.xgr
  #exec awk -f all_results.awk out.tr > allresults.xgr
  #exec awk -f throughput.awk flow=0 src=0 dst=4 out.tr > thr.xgr
  # exec xgraph All.xgr MN.xgr &
  #  BS1.xgr BS2.xgr thr.xgr &
  # exec xgraph allresults.xgr &

  exit 0
}


###################################################################
# Main
###################################################################
proc main { } {
   global opt ns TOPOM namf n tracef mobile_ cn_
   # Source Files
   # source set-def-options.tcl
   # set home/salim to your own correct path
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mipv6-config.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-tools.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-topo.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/ns-topoman.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mobi-global.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mobi-config.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/mobility/timer.tcl

   set NAMF out.nam
   set TRACEF out.tr
   set INFOF out.info

   set opt(mactrace) ON
   set opt(NAM) 1
   set opt(namfile) $NAMF
```

```
   set opt(stop) 100
   set opt(tracefile) $TRACEF

   #>--------------- Extract options from command line ---------------<
   #Getopt        ; # Get option from the command line
   #DisplayCommandLine

   #>---------------------- Simulator Settings ----------------------<
   set ns [new Simulator]
   #>------------------------ Open trace files ----------------------<
   exec rm -f $opt(tracefile)
   set tracef [open $opt(tracefile) w]
   #... dump the file

#set new trace file for wireless
   $ns use-newtrace

   $ns trace-all $tracef

   set namf [open $opt(namfile) w]
   $ns namtrace-all $namf

   #>------------- Protocol and Topology Settings -------------------<
   create-my-topo
   log-mn-movement_no_topo
   #############
   set-cbr
    # set-ping-int 0.1 $cn_ $mobile_ 10 $opt(stop)

   #start movement to pos(x,y) with velocity v
   #$ns at 10.0 "$mobile_ setdest 700 400 10"
   #MN move again to BS2#####
   $ns at 2.0 "$mobile_ setdest 230 500 10"



   #>---------------------- Run Simulation ------------------------<
   $ns at $opt(stop) "finish"
   $ns run

   $ns dump-topology $namf
   close $namf
   # puts "running nam with $opt(namfile) ... "
  #  exec nam $opt(namfile) &
}

 proc set-cbr { } {
```

```
    global ns cn_ mobile_
    set udp [new Agent/UDP]
    $ns attach-agent $cn_ $udp

    set dst [new Agent/Null]
    $ns attach-agent $mobile_ $dst
    $ns connect $udp $dst

    set src [new Application/Traffic/CBR]
    $src set packetSize_ 160
    $src set rate_ 64k
    $src set interval_ 0.05
    $src attach-agent $udp
     $ns at 10.0 "$src start"
     $ns at 15.0 "$src stop"
     $ns at 20.0 "$src start"
     $ns at 25.0 "$src stop"
     $ns at 30.0 "$src start"
     $ns at 35.0 "$src stop"
     $ns at 40.0 "$src start"
     $ns at 45.0 "$src stop"


 }

main
```

```
#FIFTH SCENARIO-WITHOUT BUFFER
# Basic Mobile IPv6 example without using ns-topoman
# Needs proc defined in file proc-mipv6-config.tcl
#
#
#
#                               AR2
#                               /              ^
#                              /               |
#                             /10Mbps/2ms      |
#              10Mbps/2ms    /                 |
#         CN-------------------CR              |Movment
#                             \                |Direction
#                              \10Mbps/2ms     |
#                               \              |
#                                \             |
#                               AR1            MN
#
################################################################################
####

Agent/MIPv6/MN set bs_forwarding_     0       ; # 1 if forwarding from previous BS
#######################################################################
proc log-mn-movement_no_topo { } {
  global logtimer ns
  Class LogTimer -superclass Timer
  LogTimer instproc timeout {} {
          global mobile_
      $mobile_ log-movement
      $self sched 1
  }
  set logtimer [new LogTimer]
  $logtimer sched 1
}


#######################################################################
# Create Topology
#######################################################################
proc create-my-topo {} {
  global ns opt topo mobile_ cn_ mnn_nodes_

  # Create and define topography
  set topo        [new Topography]
  #   set prop        [new $opt(prop)]
  #   $prop topography $topo
  $topo load_flatgrid  800 800
```

```
  # god is a necessary object when wireless is used
  # set to a value equal to the number of mobile nodes
  create-god 5

  # Call node-config
  $ns node-config \
        -addressType hierarchical \
            -agentTrace On \
            -routerTrace Off\


  # Set NS Addressing
  AddrParams set domain_num_ 2
  AddrParams set cluster_num_ {1 3}
  AddrParams set nodes_num_ {1 1 2 1}

  # Create Nodes
  set cn_ [create-router 0.0.0]
  set router_ [create-router 1.0.0]
  set bs1_ [create-base-station 1.1.0 1.0.0 100 100 0]
  set bs2_ [create-base-station 1.2.0 1.0.0 100 550 0]
  set mobile_ [create-mobile 1.1.1 1.1.0 230 100 0 0 0.01]


  # Create Links
   $ns simplex-link $cn_ $router_ 10Mb 2.0ms DropTail
   $ns duplex-link $router_ $bs1_ 10Mb 2.0ms DropTail
   $ns duplex-link $router_ $bs2_ 10Mb 2.0ms DropTail


  display_ns_addr_domain
}

#################################################################
# End of Simulation
#################################################################
proc finish { } {
  global tracef ns namf opt mobile_ cn_

  puts "Simulation finished"
  # Dump the Binding Update List of MN and Binding Cache of HA
  [[$mobile_ set ha_] set regagent_] dump
  [$cn_ set regagent_] dump
  [$mobile_ set regagent_] dump

  $ns flush-trace
```

```
  flush $tracef
  close $tracef
  close $namf
  puts "running nam with $opt(namfile) ... "
   #exec nam $opt(namfile) &


 #exec awk -f  loss.awk flow=0 src=0 dst=4 out.tr > loss.xgr
 #exec awk -f all.awk out.tr > All.xgr
  #exec awk -f mn.awk  out.tr > MN.xgr
# exec awk -f mn2.awk out.tr > MN2.xgr
 #exec awk -f bs2.awk out.tr > BS2.xgr
# exec awk -f bs1.awk out.tr > BS1.xgr
 #exec awk -f all_results.awk out.tr > allresults.xgr
#exec awk -f throughput.awk flow=0 src=0 dst=4 out.tr > thr.xgr
 # exec xgraph All.xgr MN.xgr &
#  BS1.xgr BS2.xgr thr.xgr &
# exec xgraph allresults.xgr &

  exit 0
}


###################################################################
# Main
###################################################################
proc main { } {
   global opt ns TOPOM namf n tracef mobile_ cn_
   # Source Files
   # source set-def-options.tcl
   # set home/salim to your own correct path
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mipv6-config.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-tools.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-topo.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/ns-topoman.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mobi-global.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/lib/proc-mobi-config.tcl
   source ~/ns-allinone-2.28/ns-2.28/tcl/mobility/timer.tcl

   set NAMF out.nam
   set TRACEF out.tr
   set INFOF out.info

   set opt(mactrace) ON
   set opt(NAM) 1
   set opt(namfile) $NAMF
   set opt(stop) 100
```

```
   set opt(tracefile) $TRACEF

   #>--------------- Extract options from command line ---------------<
   #Getopt        ; # Get option from the command line
   #DisplayCommandLine

   #>---------------------- Simulator Settings ----------------------<
   set ns [new Simulator]
   #>------------------------ Open trace files ----------------------<
   exec rm -f $opt(tracefile)
   set tracef [open $opt(tracefile) w]
   #... dump the file

#set new trace file for wireless
   $ns use-newtrace

   $ns trace-all $tracef

   set namf [open $opt(namfile) w]
   $ns namtrace-all $namf

   #>------------- Protocol and Topology Settings -------------------<
   create-my-topo
   log-mn-movement_no_topo
   #############
   set-cbr
    # set-ping-int 0.1 $cn_ $mobile_ 10 $opt(stop)

   #start movement to pos(x,y) with velocity v
   #$ns at 10.0 "$mobile_ setdest 700 400 10"
   #MN move again to BS2#####
   $ns at 2.0 "$mobile_ setdest 230 500 10"



   #>----------------------- Run Simulation -------------------------<
   $ns at $opt(stop) "finish"
   $ns run

   $ns dump-topology $namf
   close $namf
   # puts "running nam with $opt(namfile) ... "
  #  exec nam $opt(namfile) &
}

 proc set-cbr { } {
    global ns cn_ mobile_
```

```
   set udp [new Agent/UDP]
   $ns attach-agent $cn_ $udp

   set dst [new Agent/Null]
   $ns attach-agent $mobile_ $dst
   $ns connect $udp $dst

   set src [new Application/Traffic/CBR]
   $src set packetSize_ 160
   $src set rate_ 64k
   $src set interval_ 0.05
   $src attach-agent $udp
    $ns at 10.0 "$src start"
    $ns at 15.0 "$src stop"
    $ns at 20.0 "$src start"
    $ns at 25.0 "$src stop"
    $ns at 30.0 "$src start"
    $ns at 35.0 "$src stop"
    $ns at 40.0 "$src start"
    $ns at 45.0 "$src stop"
    $ns at 50.0 "$src start"
    $ns at 55.0 "$src stop"

 }

main
```