REDUCING HANDOVER LATENCY AND IMPROVING TCP PERFORMANCE IN WIRELESS NETWORKS

by

Beizhong Chen

A Dissertation submitted to the

Graduate School-New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in

Electrical and Computer Engineering

written under the direction of

Professor Ivan Marsic

and approved by

New Brunswick, New Jersey

October, 2010

©2010

BEIZHONG CHEN

ALL RIGHTS RESERVED

ABSTRACT OF THE DISSERTATION

Reducing Handover Latency and Improving TCP Performance in Wireless Networks

By BEIZHONG CHEN

Dissertation Director:

Professor Ivan Marsic

Modern network technologies first evolved in wired networks and subsequently entered the wireless network field. Applications may work in pure wired network, pure wireless network or hybrid network. Improving performance in these network infrastructures has been a continuous effort for decades. In this dissertation, we tackle two important challenges: (1) improving handover performance in heterogeneous wireless network, and (2) improving TCP performance in multi-hop wireless network.

In heterogeneous network, users expect uninterrupted services moving from one network to another. IEEE proposed Media Independent Handover (MIH) to make it possible to achieve better handover performance. Currently, Mobile IPv4 (MIP) is the dominant mechanism for mobility management and is expected to persist into the future. However, when multiple interfaces of a mobile client are connected to a Foreign Agent (FA), MIP and its existing improvements do not perform well when the active interface fails unexpectedly. In this dissertation, we propose novel mechanism and MIP extension with MIH support. We prove experimentally that the new approach eliminates the FA-HA latency and achieves much faster handover, compared with existing mechanism. Our method also allows FA-bicasting, which can improve transmission reliability by combining traffic from different links, through the same FA.

In multi-hop wireless networks, the main network factor that affects TCP performance is the medium-access contention, complicated by other factors like hidden terminal. We analyze the TCP congestion window and provide a more accurate estimate of its optimal value than those reported in the prior work. We also show that the much shorter TCP-ACK packets consume comparable channel capacity as the much longer data packets. We therefore propose two methods to improve TCP throughput, as follows. (1) Segregate the flows of data and ACK using static routing in grid wireless network, (2) Develop an improved variant of delayed TCP ACK by minimizing the number of ACK packets. Our evaluation validates the effectiveness of the proposed method, which can enhance TCP performance significantly. In terms of throughput, it achieves up to 204% improvement over the regular TCP in chain-topology wireless networks, and about 35% improvement in a complex grid wireless network. We also propose a new architecture to achieve higher throughput when multiple TCP connections exist.

Acknowledgements

Firstly, I would like to thank Professor Ivan Marsic for his supervision, for his assistance, guidance, support and patience during the course of this research effort. I have enjoyed and benefited very much from working under his supervision.

I also would like to thank Professor David G. Daut, Professor Dario Pompili, and Professor Liang Cheng, for taking time from their busy schedules to serve on my dissertation committee. Their valuable comments and suggestions have greatly aided the completion of this study.

Finally, I would like to thank the great help I have received from David Faucher, Cezar Purzynski, Ray Miller, and Sharma Sameer in Bell Labs, Alcatel-Lucent, and Dr. Huairong Shao in Samsung Electronic Company.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Starting from the ARPAnet, modern network technologies first evolved in wired networks and entered the wireless network field later on. Numerous efforts have been spent on wired networks and large amounts of research and industrial experience have been accumulated since then. Currently, one important technology evolution trend is IP-convergence. In an IP-network, most of mechanisms are first designed for wired network and then reused in wireless network. However, because the characteristics of wireless network are very different from wired network, some existing mechanisms suffer significant performance deterioration in wireless network. Modifying such mechanisms for different wireless scenarios to achieve better performance has presented significant challenges. Another challenge is the need for integrating wired and wireless networks to achieve better performance and improve user experience. In this dissertation, we tackle two important challenges, one is to improve the handover performance in heterogeneous networks, the other is to improve TCP performance in multi-hop wireless network. We propose novel approaches to provide more reliable and faster services in heterogeneous networks and in multi-hop wireless ad-hoc networks.

1.1 Handover in heterogeneous network

Starting from wired network, different computer network technologies have evolved following their distinct paths. As a result, different infrastructures coexist in the current network. Wireless access has seen exponential growth in past decade, and different wired and wireless technologies have been emerging, e.g., Ethernet [1], WiFi [2, 3], 3GPP [4], 3GPP2 [5], WiMax [6, 7], LTE [8-11], etc. Recent trends indicate that future network infrastructures will consist of heterogeneous wired and wireless networks. Different network technologies have their respective strength and are used in different scenarios. For example, IEEE 802.11 offers a high-throughput wireless network in unlicensed spectrum, while 3GPP/3GPP2 provides a much larger area of coverage using a licensed spectrum. To take advantage of the benefits for these different network infrastructures, mobile terminals can be equipped with multiple interfaces which are based on those different wired and wireless technologies. It is a major challenge to provide mechanisms to achieve fast handover when the user moves from one network to another. Although many companies have proposed their proprietary technologies, it is a daunting task, if not impossible, to allow different types of mobile terminals to function properly in heterogeneous network. Therefore, the computer networking industry has been working together to design a standard platform, IEEE 802.21 (medium-independent handover, or MIH), to facilitate the handover in heterogeneous networks. MIH makes it possible for mobile terminals to roam from one type of network to another.

In addition to handover in heterogeneous networks, providing mobility management is another issue. Currently, Mobile IPv4 (MIPv4) is the most popular protocol used to provide uninterrupted IP connection when a mobile terminal changes the access network. Integrating MIH and mobile IP to achieve better handover performance is a challenge. By carefully analyzing the timing diagram of MIP message flow, we first identify a weakness of the Mobile IP protocol in the case when the multiple interfaces of a mobile terminal are connected to a single Foreign Agent (FA). Before a network interface card can be used for data transmission, a set of control messages must be exchanged between different network entities, e.g., Mobile Node (MN), Foreign Agent (FA), Home Agent (HA), etc. The procedure of messages exchanging affects the handover performance significantly. In this work, we propose a novel approach to achieve fast handover by more efficient message flow. Based on that, we propose a new MIP extension. To evaluate our proposal, we design and deploy a complete test network which includes all required network entities, including MN, FA, HA, network simulator, etc.

1.2 TCP in multi-hop wireless network

Our work on improving the network-layer handover performance in heterogeneous networks is complemented by the second topic, which is improving transport-layer (TCP) performance in multi-hop networks. TCP is the most popular protocol used for reliable transmission. However, the TCP behavior in multi-hop wireless networks is very different from that in wired networks. Improving the TCP throughput in multi-hop wireless networks poses a major challenge. Towards this goal, we first focus our work on studying the characteristics of TCP behaviors in multi-hop networks and the impact of different data-packet sizes in wireless networks. We assume that the multi-hop wireless network is based on IEEE 802.11 MAC.

Unlike the wired network case, TCP performance in contention-based multi-hop wireless networks is shaped by two main factors. First, there is a maximum throughput for a given topology and flow pattern, which is obtained at an optimal congestion window. We provide an improved analyzing method and a better result is achieved, compared with previous works. Second, the traffic generated by control packets, e.g., TCP Acknowledgement packet (TCP ACK), uses significantly higher proportion of the channel bandwidth than in the wired case. Our analysis and simulation results show that the much smaller TCP ACK packets consume channel resource comparable to the much longer data packets, over high-speed wireless links. Inspired by these two major insights, we propose methods to improve TCP performance. Based on the first result, we can calculate an optimal congestion window and use it to control the TCP behavior. The second result suggests reducing the interference between data packet traffic and the TCP ACK traffic.

Based on these two hypotheses, we first propose to separate the data traffic and ACK traffic in a grid-topology wireless network using a static routing protocol. To alleviate the negative effect of ACK traffic, a more generic approach is to minimize the number of ACK packet. Therefore, we further propose an improved ACK-delay approach which is shown to improve TCP performance significantly. Our research results also show the relationship between the number of TCP connections and the TCP throughput, and based on this, we propose an architecture to reduce the number of TCP connections and, therefore, improve the overall TCP throughput.

1.3 Outline of the dissertation

The remainder of this dissertation is organized as follows. Chapter 2 reviews and discusses heterogeneous network, handover and mobility management. Some critical issues (e.g., MIP procedure, Move detection, etc.) are emphasized. Chapter 3 describes and discusses the details of IEEE 802.11 which are related to our discussion in subsequent chapters. In chapter 4, we tackle the challenge of improving the handover performance in heterogeneous network. A novel mechanism and a new mobile IP protocol extension are presented. We analyze the TCP transmission behaviors in multi-hop wireless networks and then propose novel algorithms and architecture to improve TCP throughput in Chapter 5. Finally, several conclusions are drawn for fast handover in heterogeneous networks and improved TCP throughput in multi-hop wireless network, and these are presented in Chapter 6. Some of the relevant future work to be done as continuation of this dissertation is also discussed in Chapter 6.

Chapter 2

Media Independent Handover and Mobile IP

2.1 Introduction

Nowadays, a diverse range of access technologies are available and in the course of development, including Ethernet [1], WiFi [2, 3], 3GPP [4], 3GPP2 [5], WiMax [6, 7], LTE[8-11] , etc. To provide a fast, seamless connection to different mobile users, growing number of mobile clients are equipped with multiple interfaces that can receive services from multiple service providers, possibly via different access technologies. Different access technologies have their respective strengths. When a client roams from one location to another, due to the availability of different access points (wired or wireless), the changed signal strength in the wireless network, different services provided by different service providers, and other factors, he may prefer one network to another, and therefore, the connection needs to be switched from the old one to the new one. Many companies have proposed their proprietary protocols to handle the mobility management issues. All dual-mode cellular phones nowadays are based on such type of protocols. The biggest drawback of such protocols is that they can be used only in a network which is provided by a service provider adopting the specific protocol and cannot be connected to another network provided by service provider who uses another type of technology.

Media Independent Handover (MIH, IEEE 802.21) [12] is an ongoing, evolving effort to provide a generic solution for mobility across access networks. Collaborating with Layer 3

Mobility Management protocol, MIH makes it possible to achieve better handover performance with respect to latency and packet loss rate than the existing mechanisms in heterogeneous networks. Currently, Mobile IPv4 (MIPv4) is the dominant mechanism for mobility management in IP-network and is expected to persist into the future. In this chapter, we discuss MIH, MIPv4 and related issues.

2.2 Handover use case in heterogeneous network

In the past decades, a diverse range of access technologies have become available, in wired and wireless fields, which include Ethernet[1, 13, 14], WiFi[2, 3], 3GPP[4] (GSM, UMTS), 3GPP2 [5, 15] (CDMA2000), WiMax[6, 7, 11], 4G (LTE) [8-11], etc. These different wired and wireless technologies have evolved down their distinct paths, each providing services with different characteristics. For example, WiFi is mainly deployed indoors providing short-range connectivity while 3GPP provides a much longer-range service outdoors. In order to provide better coverage and improve user experience, some vendors offer proprietary dual-mode mobile terminals that allow users to access two different networks. For example, Samsung SCH-a790 [16] provides access to CDMA and GSM network. RIM BlackBerry 8820 smartphone [17] offers connection to GSM and WiFi network. Figure 2-1 shows a use case in the heterogeneous network. Assume a user is equipped with a mobile terminal which is capable of providing connection through 3GPP/3GPP2/WiMAX/4G cellular network, WiFi network and wired Ethernet network. When the user is working in the visited office, he may prefer to use the Ethernet connection because Ethernet uses wired link and it typically provides faster, more reliable connection,

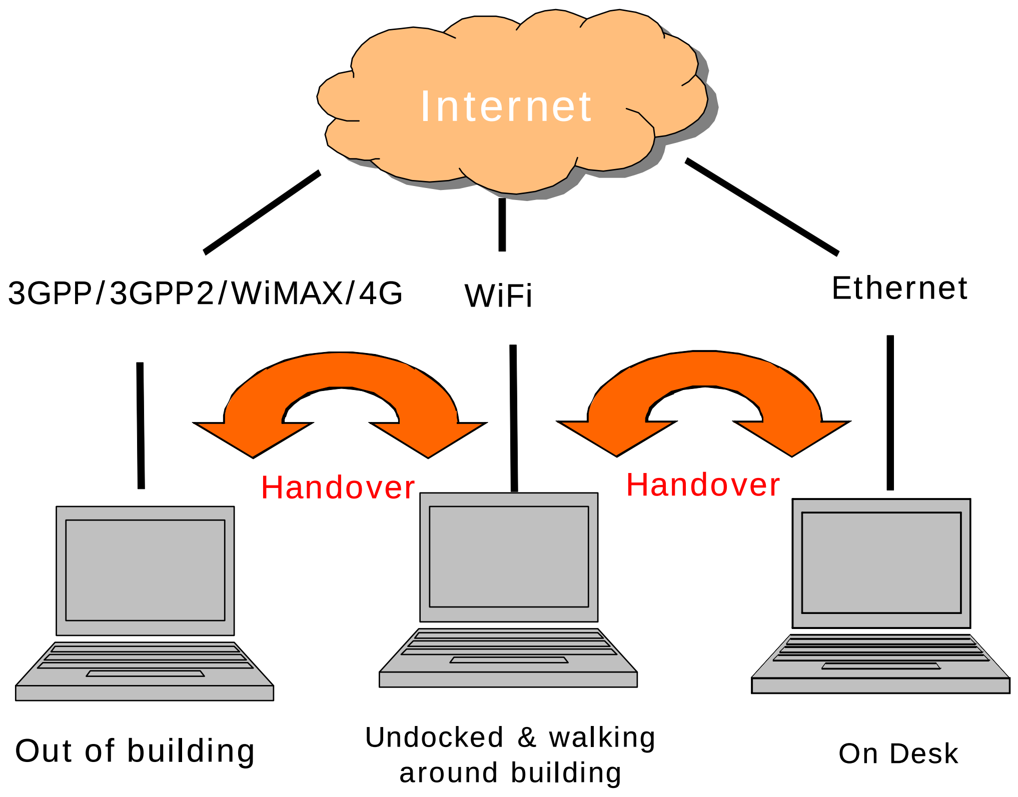


Figure 2-1 Handover use case in heterogeneous network

and higher security. When the user needs to walk around in the building, he then undocks his mobile terminal from the socket. In this case, he may use the WiFi card to connect to the access point (AP) for internet connection. When he leaves the building and goes outside, the 3G/4G network may be the best choice for broad connection. In such case, the connection needs to be switched to the 3G/4G network. During this process, the handover occurs when the mobile terminal changes the connection to different networks.

Because the existing access technologies are designed without the consideration of interoperation between other types of access technologies, there is not exiting mechanism to allow the user to switch from one access network to another. In order to provide best user

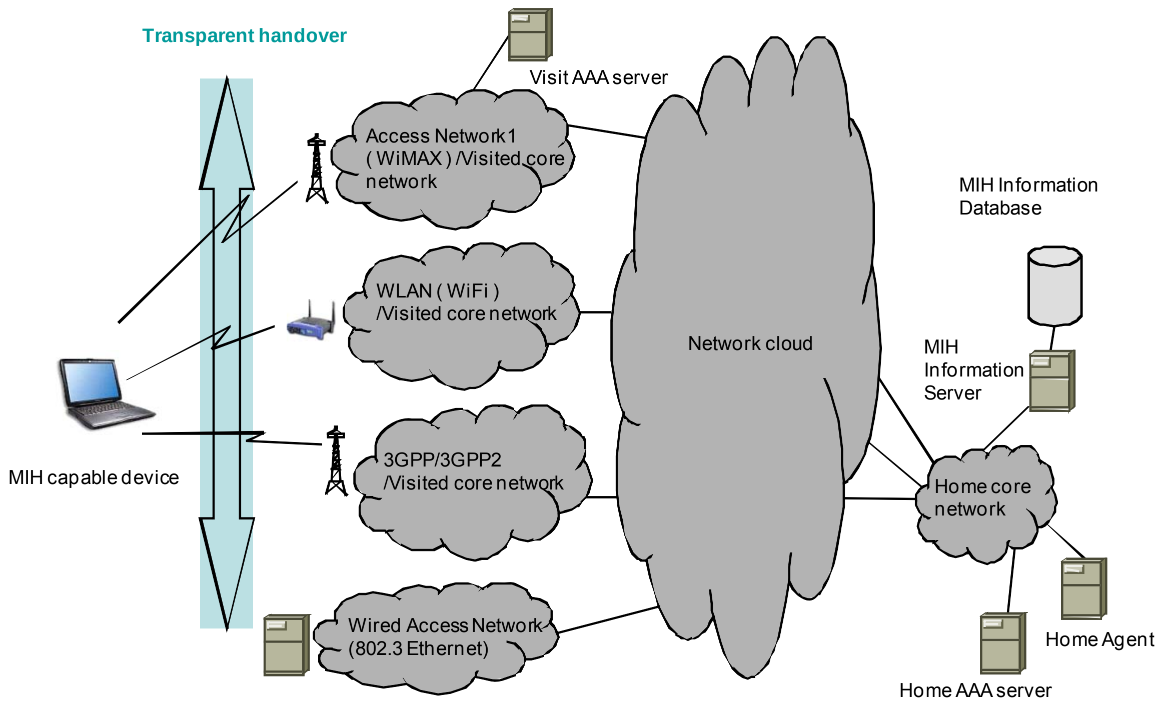


Figure 2-2 MIH reference model

experience, IEEE 802.21 (Media Independent handover, MIH) [18] aims to provide a generic solution for intelligent handover in heterogeneous technologies. Figure 2-2 shows the MIH Network model. The mobile terminal connects to Ethernet, access point or base station in the visited network, which provides necessary services, e.g., visitor AAA. All visited networks are connected with the home core network.

2.3 Types of handovers

When a mobile terminal moves from one network to another one, the connection must be moved from the old network to the new network. The process of connection changing is called handover. Based on different criteria, there are different categories of handovers. According to the moment when the new connection is added, the handover can be divided into *hard handover* and *soft handover*.

- *Hard Handover*

Hard handover is also known as *break-before-make*. The mobile terminal first breaks from the current connection before it moves to a new connection. This is a rather simple handover technology. Typically, hard handover will cause connection gap and therefore, result in packet loss. However, if the interruption is short enough, it is not perceptible practically by a mobile device user. Therefore, hard handover can be seamless or non-seamless.

- *Soft Handover*

Soft handover is also known as *make-before-break*. The mobile terminal set up the new connection before the old connection is removed, and therefore, the mobile terminal is always kept connected to the network. The mobile terminal may have two or more link connection with one, two, or more access points or base stations. The soft handover transition is evidently faster than the hard handover and it is possible to achieve packet lossless handover.

The novel mechanism we will introduce in chapter 4 can be categorized as semi-hard and semi-soft handover, in the sense that the new connection has been partially completed before the old connection is down.

According to the type of access networks before and after the handover, the handover can be divided into vertical handover and horizontal handover.

- *Vertical handover* also known as heterogeneous handover. It occurs when a mobile terminal roams from one network into another network served by a different access technology, e.g. from a WLAN network to a 3GPP network. IEEE 802.21 is designed to facilitate this type of handover. This type of handover provides localized mobility.
- *Horizontal handover* also known as homogeneous handover. It occurs when a mobile terminal roams into networks served by the same type of access technology, e.g., from one access point to another access point in a WLAN network. This type of handover provides global mobility.

Due to the protocol stack architecture adopted in all popular access techniques, when handover occurs, it actually occurs on several layers, such as layer 2 and layer 3. Therefore, handovers can also be classified as Layer 2 (L2) or Layer 3 (L3) handover. Typically, a L3 handover occurs after a L2 handover is completed. A handover involves many operations, e.g., network discovery, L2 disconnection detection, L2 reconnection, L3 re-connection, protocol negotiation, reconfiguration, etc. Generally, handover would cause performance deterioration.

2.4 Media Independent Handover (MIH, IEEE 802.21)

2.4.1 Introduction

Briefly speaking, the process of handover contains stages of handover initiation, handover preparation, and handover execution. The handover initiation stage is the one that searches for new link, which may contain (1) network discovery (2) network selection (3) handover negotiation. Handover preparation is the stage to setup a new link. It includes L2 handover and L3 handover. The handover execution is the stage that transfers the connection. It may contain handover signaling, context transfer, packet reception, etc.

MIH (IEEE 802.21) facilitates all handover stages, e.g., Handover Initiation, Network Selection and Interface Activation, in heterogeneous networks. Figure.2-3 shows the placement of the MIH function in the protocol stack, sandwiched between Layer 2 (L2) and Layer 3 (L3). MIH is intended to assist in dissemination of handover-related information (e.g., link states, handover progress, etc.) to the upper layers (via event service), enabling them to monitor and control the handover (via command service). MIH provides three types of services, which are (1) Event service, (2) Command service (3) Information service.

- Event service provides event classification, filtering and reporting. The events are changes of link status, link quality, link modification, etc. For example, a new network interface card is plugged in or unplugged is an event, which need to be passed to upper layer for a fast handover.

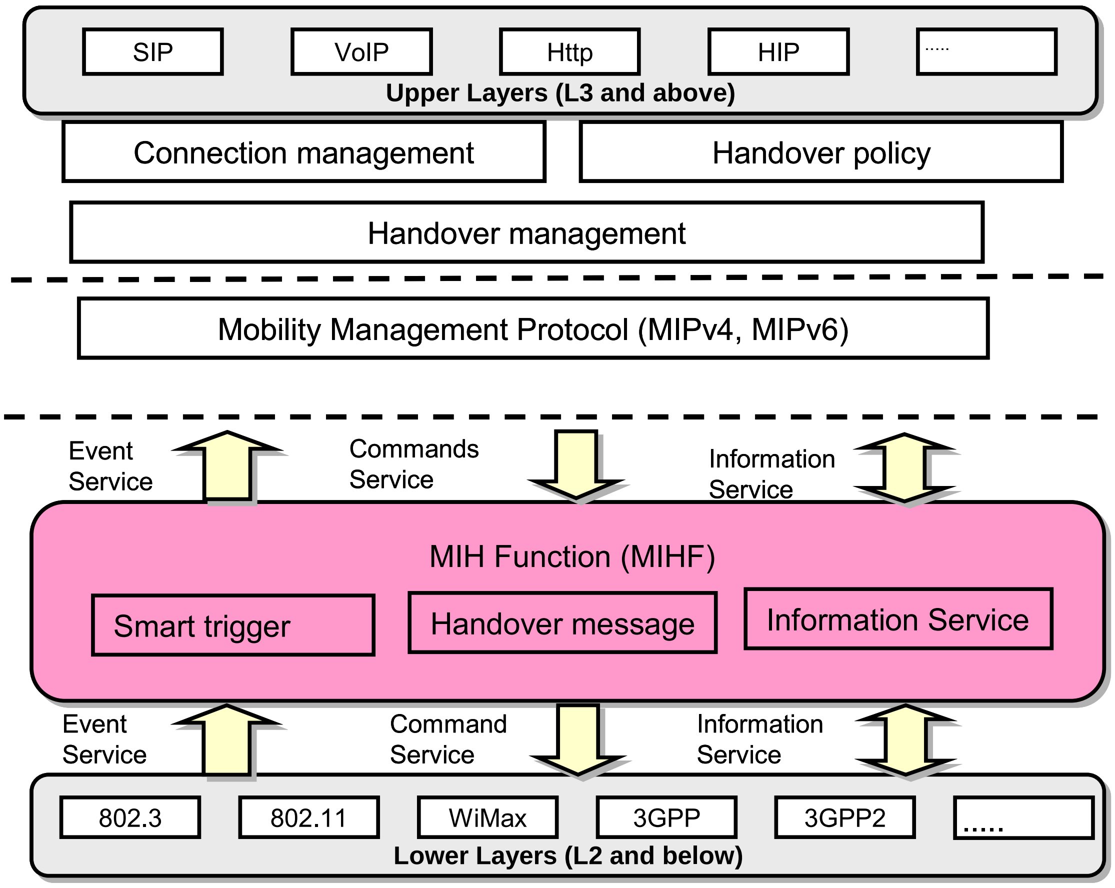


Figure 2-3 MIH Function

- Command service enables MIH users to control and manage the connection based on events, pre-set policies, etc. For example, the MIH user may issue a command to a WiFi card to change the transmission rate.
- Information service provides information of services available in the serving or neighbor network. It also allows the terminal to inquiry necessary information for network selection and decision making.

Figure 2.3 demonstrates the layer where the MIH resides in the protocol stack and the services it provides. It clearly shows that the MIH is different from Mobility Management Protocol. MIH itself provides functions that are used to facilitate handover. However, on IP level, mobility management protocol, e.g., MIPv4, MIPv6, are still needed.

2.4.2 Proposed Test-bed for MIH design and development

MIH is an on-going, evolving IEEE standard. We are the group representing Bell-Labs to participate in the design of this new standard. To design and establish this standard, a numerous issues need to be identified and addressed. Figure 2-4 illustrates our test-bed proposed for MIH establishment. It includes all necessary network entities as shown. Because the test-bed resides in Bell-Labs, for security reason, we need a DMZ (Demilitarized Zone) entity to connect the test-bed to the external Internet. DMZ is a component to expose an organization’s external services to an un-trusted network, i.e., the external Internet outside of the company Intranet.

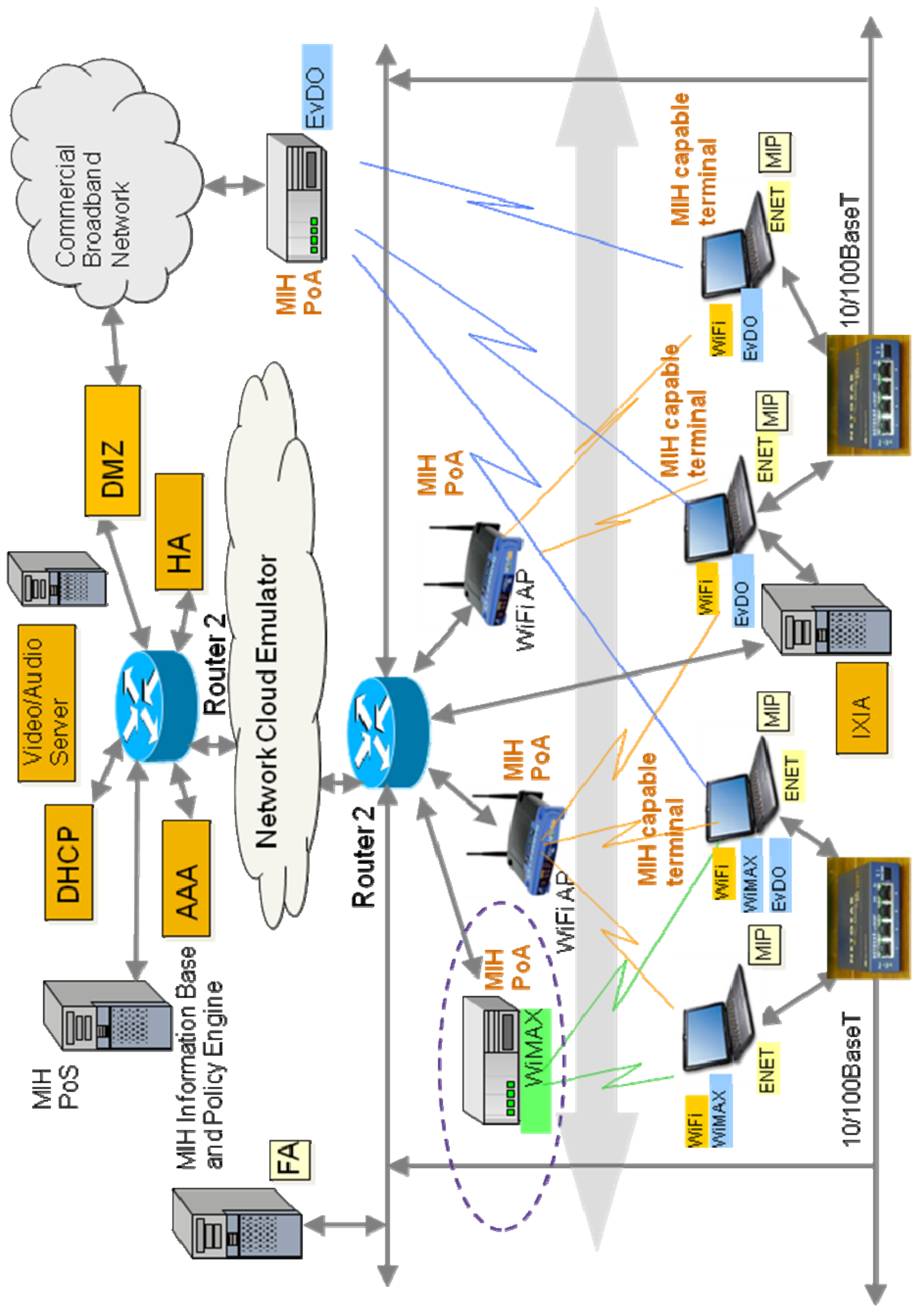


Figure 2-4 Test-bed designed for MIH development

2.5 Mobile IP

2.5.1 Introduction

Mobile IP is a protocol residing on the Layer3 to provide mobility management to the mobile terminal. It allows location independent routing of datagram to a mobile terminal which is identified by its home address. The home address of a mobile client will not change no matter which visited network the mobile terminal is connected to. When the mobile terminal roams to a visited network, the visited network will assign a care-of address to the mobile terminal. The care-of-address is a temporary IP given to the mobile client in the visited network. The information of this care-of address is sent back to the Home Agent (HA) in the home network. The home agent keeps the association of the care-of address and the mobile terminal's home address. An IP tunnel may be built to connect the mobile terminal and the home agent through the Internet cloud. For any packets received in the home agent, the home agent will forward them to the mobile terminal through the tunnel. Mobile IP provides an efficient mechanism for roaming in the Internet. Using Mobile IP, nodes may keep its connection to the Internet without changing their home IP address. Therefore, the location changing in the network is transparent to the corresponding node (CN). Node mobility is realized without the need to propagate the changed location on the network.

2.5.2 Mobile IP concepts

- Mobile Node (MN). This is the mobile terminal that roams to a visited network. We use mobile node and mobile terminal interchangeably in the subsequent chapters.
- Care-of IP address (CoA). Any node which implements MIP has two IP addresses, one is the home address which is assigned in its home network, and another is the care-of address which is assigned by the visited network. The home address is the permanent address and is used to identify the MN in the internet. The care-of address is used for MIP management and is unknown to outside world.
- Home Agent (HA). This is the agent in the home network which takes charge of the management of MN. It forwards all packets destined to MN, using the Care-of-address.
- Foreign Agent (FA). This is the agent that resides in the visited network. It takes charge of the MN management in the visited network. Its tasks may includes assigning care-of address to the MN, passing the control messages between the MN and the HA, forwarding and receiving the datagram for the MN, etc.
- Correspondent node (CN). This is the peer node which communicates with MN. It does not need to know the actual location of the MN.
- Security Parameter Index (SPI). This is the index used to identify the security context between a pair of nodes.
- Mobility binding. This is the association of the home address with the MN's care-of address. HA keeps this record. When it receives packets directing to a MN,

which is identified by its home address, the HA uses this binding to find out the care-of address and then forward the traffic.

- Bi-casting. A MN may register multiple care-of addresses in the home agent. This typically adopted during the handover period. In this scenario, the HA may select to duplicate and forward the coming traffics to multiple care-of addresses, see figure 2-5. Because the same packet is transmitted through two different paths, packets lost in one path may be received by other path successfully. Therefore, this mechanism can help to mitigate packet lose during the handover transition. When this technique is used, the upper layer takes the responsibility to remove the duplicated packets received. The bi-casting can be achieved by simply duplicating the packets or using some coding techniques, e.g., FEC bi-casting. [19]

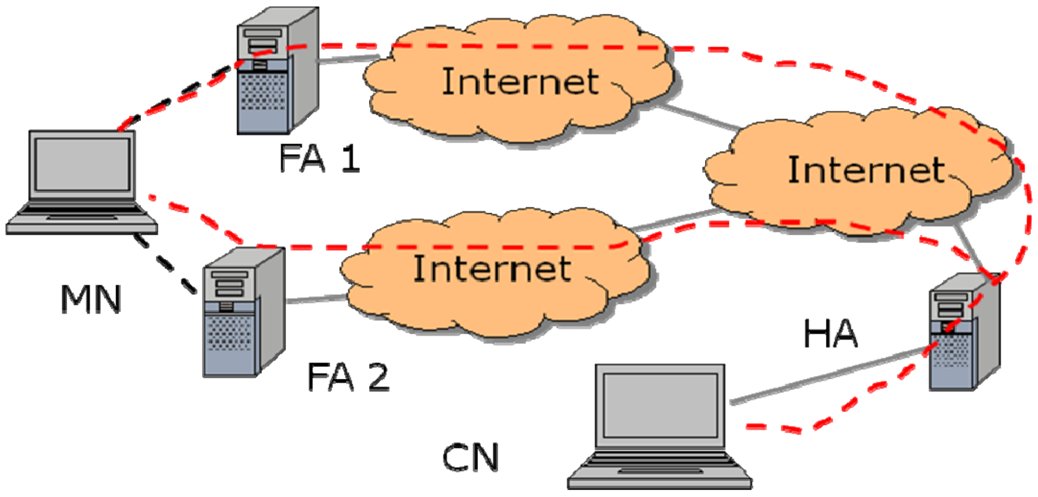


Figure 2-5 Bi-casting in MIP

2.5.3 Mobile IP procedure

When a MN roams to a new network, it first need to find out a new FA and then finishes the necessary message exchange to setup a new connection, which includes the connection to the access network and notification to the home network. The basic related procedure is given as follows.

- Agent discovery
  - In this phrase, the MN needs to look for a FA. To achieve this, the MN may broadcast agent solicitation messages for new FA, or listen to the advertisement periodically sent by the mobility agent (FA or HA). The MN can detect whether it has moved to another network by monitoring those advertisements. Based on the messages it has received, the MN can decide which network it can attach to.
  - After a FA is found, the MN can obtain a care-of address from the FA, which may connect to a separate DHCP server.
- Registration
  - Mobile IP registration allows the mobile terminal to notify the Home Agent for its current reachability. This occurs after the MN obtains the care-of address. The MN sends messages to the HA to register its care-of address.
  - Two registration procedures are defined, one via a FA which relays the message to the HA, and another allows the MN to connect to the HA directly, without going through a FA. After this step, the HA has the

location information of the MN. In the former case, typically, a tunnel is setup between the HA and the FA. The traffic between the MN and the CN will be passed through this tunnel. In this step, some securities check, e.g., authentication, will be performed.

- After a MN has been authenticated and registered successfully, a message is sent back to the MN through the FA. The FA then keeps a record of the MN in its visitor list, and uses this information to forward the packets to the specific MN.

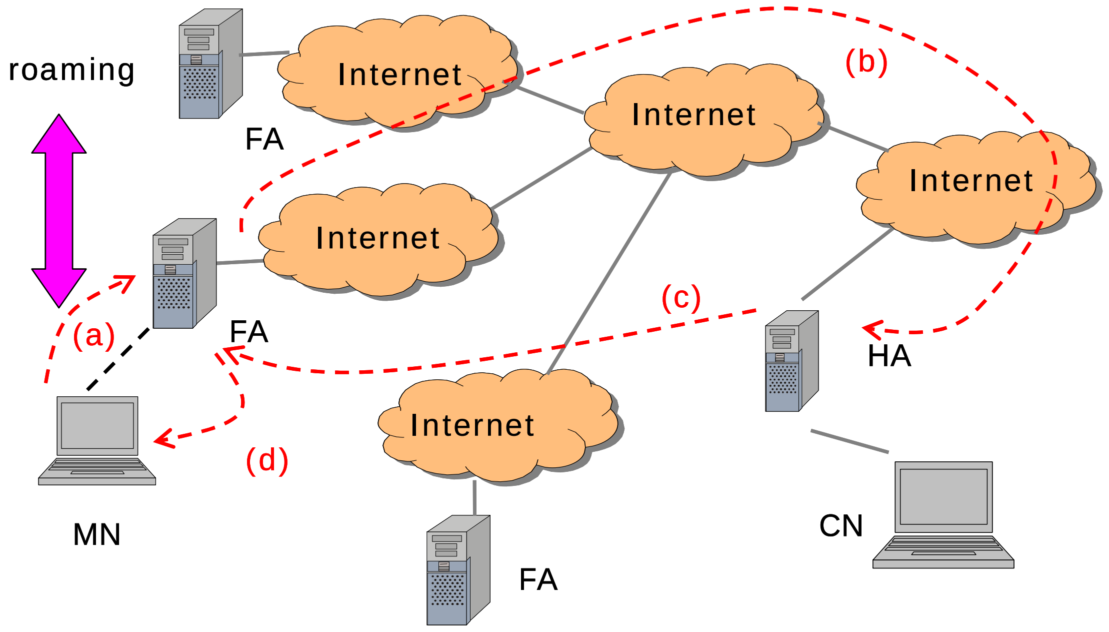


Figure 2-6 Flow of registration messages in MIPv4

Figure 2-6 shows a typical registration message flow between a mobile node with a Foreign Agent care-of address and its home agent, in the case of a foreign agent care-of address is used. The MN sends a registration request to the FA, in the (a) step. The FA does some validity checks and then relay the registration request to the HA, in the step (b). The

FA needs to maintain the information about the MN, e.g., the link layer address, IP address, etc. In the registration request sent to the HA, the FA also includes its own address as the IP source address. After receiving the message, the HA need to perform the authentication procedure on the requested service from the FA. If the request is valid, the HA will update its records of the MN and FA in its binding table. The authentication can be done in the HA or through an AAA server separated from the HA. When this process is finished, the HA sends back a reply with authentication information to the FA who then relays it to the MH, in the (c), (d) steps. If no reply indicates any failures, data packets can be started to transmit between the MH and CN.

The Mobile IP provides L3 mobility management and the MIH provides services that facilitate the handover management. We can see the benefit obtained by combining these two types of techniques in chapter 4.

2.5.4 Move detection

The Mobile IP is designed to allow a mobile terminal to move from one subnet to another in different IP domain, which is transparent to the corresponding node. Therefore, a mechanism to detect the change of subnet is required. Because Mobile IP is on top of IP, it does not have the information in physical layer to detect the change of subnets. Two algorithms are provided in MIP for move detection.

The first method is based on the Lifetime field in the Agent Advertisement message. In every Agent Advertisement, a parameter of Lifetime is included. A mobile terminal keeps the Lifetime in record and run an associated timer. When this timer expires, if the mobile terminal fails to receive an update of Agent Advertisement, it concludes that it has lost contact with this agent. In this case, the mobile terminal attempts to discover a new agent to register to.

The second approach uses network prefixes, which is similar to the netmask used in the IP routing. MIP provides a Prefix-Lengths Extenstion which allows the mobile terminal to detect the change of subnet. If the prefix it receives is changed, it may assume that it has moved to another subnet.

Both methods are based on the information in the message exchanging between mobile terminal and access network (visited network or home network). These messages are used for connection management purpose and do not contribute to the amount of data packet transmitted. In other words, they are overhead and consume the effective channel capacity for data transmission. Therefore, they should not be transmitted too frequently. The time interval between two control messages is in the order of seconds, or even minutes, depending on the network configuration. One consequence is that the move detection latency could be very long, up to minutes. This is unacceptable for some applications, for example, real time application like VoIP.

2.5.5 Security issues

For any communication, security is an important issue which must be addressed. Before a mobile terminal is allowed to be connected to a network, it must be authenticated first by the network. The typical issues are authentication, authorization and accounting (AAA). Different access networks have their distinct security mechanism design. In the case when a mobile terminal needs to connect to different access networks, the mobile terminal should support those different designs and select the security mechanism accordingly. This is an important feature we need to take into account when we design new protocols.

2.5.6 Improved MIP variants

A complete MIP procedure involves a series of messages forwarding. In different scenarios, the factors that dominate the latency are different. Based on the dominant factors, some MIP variants have been proposed to reduce the overall latency.

- Hierarchical Mobile IP[18]. This is a Mobile IP extension that supports hierarchical of FAs between MN and the HA.
- Fast Handoff [20]. This is similar to the hierarchical MIP but provides improved handover mechanism.
- TeleMIP[21]. This protocol adds some balancing features.

All the improved MIP variants mentions above are based on the assumption that the FAs are changed before and after the handover. This is different from the problem we will tackle in chapter 4.

Chapter 3

Wireless Network and IEEE 802.11

3.1 Multi-access channel

There are two modes to utilize the transmission medium. The first one is the *point-to-point* mode, where the medium is shared by two stations. The second one is the *broadcast* mode, where multiple stations share the same medium to communicate with each other. In the latter mode, stations in the hearing area may hear the communication between other stations. To occupy the medium for transmission, a station needs to compete for it or a coordinator must present to arrange the access to the medium for stations, based on some pre-set rules. IEEE 802.11 belongs to the second case. In literature, broadcast channels are sometimes referred to as *multi-access* channels or *random access* channels.

Systems in which multiple stations share a common medium in a way that can lead to confliction are known as *contention* systems. In such a system, the medium may be in one of the following 3 states, (1) only one station is transmitting (2) two or more stations are transmitting (3) no station is transmitting. Because the medium is shared, a successful transmission will occur only if exactly one station transmits at any given time. If two or more stations transmit, the reception is garbled and therefore, the transmission fails. If none is transmitting, the channel is idle. The protocols used to determine whose turn is next on a multi-access channel belong to a sub-layer of the link layer in the OSI reference architecture, called the *MAC (Medium Access Control)* sub-layer.

3.2 IEEE 802.11 MAC

Carrier sense multiple access with collision avoidance (CSMA/CA) is the basic MAC protocol adopted in IEEE 802.11. Two coordination functions are defined. The first one is the *Distributed Coordination Function* (DCF) which allows the coordination decision to be distributed over all the stations in the network. The second one is a centralized MAC algorithm, called *Point Coordination Function* (PCF), which provides contention-free service and is primarily intended for scenarios where a central access point is desirable, e.g., an access point connected to the wired backbone provides connection services to wireless clients. This function also makes it possible to provide different service quality based on the packet types. This is especially useful if some of the data is time sensitive or high priority. They are summarized as follows.

Distributed Coordination Function (DCF):

- CSMA/CA used for access control
- When a collision is detected, the sender waits a randomly selected backoff time before the next transmission attempt
- The receiver returns an acknowledgement frame for any data frame it has received (to be explained below)

Point Coordination Function (PCF):

- Build on top of DCF
- Based on pre-set rules, a point coordinator determines the station which has the right to transmit

DCF and PCF can be configured simultaneously in a wireless network. This means that some stations can work in the PCF mode while others work in the DCF mode.

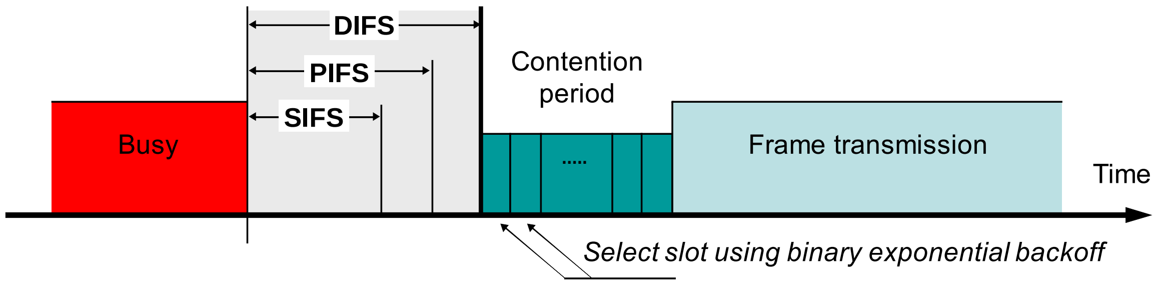


Figure 3-1 802.11 IEEE interframe spacing relationship

To allow a station to detect the current medium state, i.e., to determine whether the medium is busy or idle, IEEE 802.11 defines Inter-frame space (IFS) which is the time interval between frames (figure 3-1). Once the IFS has elapsed, the station may start transmission. To assist with interoperability between different data rates, the inter-frame space is a fixed amount of time, independent of the physical layer bit rate. This implies that for higher data transmission rate, the relative overhead will be smaller. There are basic intervals determined by the PHY: the *short interframe space* (SIFS) and the *slot time*, which is slightly longer than SIFS. Using the slot time, a station is always capable of determining if other station has accessed the medium at the beginning of the previous slot.

The four different types of IFSs in IEEE 802.11 are summarized below.

SIFS: Short inter-frame space is the shortest *IFS* and always elapses earlier than other type of *IFS.* Therefore, it can be used for atomic transmission (e.g., Frame-fragment–ACK), or highest priority transmissions like control frames. This value is a fixed value per PHY and is calculated in such a way that the transmitting station will be able to switch back to receive mode and be capable of decoding the incoming packet. For example, for the IEEE 802.11 FH PHY this value is set to 28 microseconds.

PIFS: PCF (or priority) interframe space is used by the PCF during contention-free operation. PIFS is equal to SIFS plus one slot time. In this dissertation, we do not focus on centralized wireless network and will not consider this *IFS* in subsequent sections.

DIFS: DCF (or distributed) interframe space is the minimum medium idle time for asynchronous frames contending for access. Stations may have immediate access to the medium if it has been free for a period longer than the DIFS. DIFS is equal to SIFS plus two slot times.

EIFS: Extended interframe space is much longer than any of the other intervals. It is used by any station that has received a frame containing errors that it could not understand (not shown in the figure 3-1). We will also not consider this *IFS* in subsequent sections.

Please notice that all interframe spaces are independent of the data transmission rate. This feature affects the transmission effectiveness under different transmission rate.

To give an example, we show some important parameters of the IEEE 802.11b system in table 3-1. The values shown are for the 1Mbps channel bit rate and some of them are different for other bit rates.

Parameter	**Value for 1Mbps channel bit rate**
Slot time	20 μsec
DIFS	50 μsec (DIFS = SIFS + 2 × Slot time)
SIFS	10 μsec
CW_{max}	1024
CW_{min}	32
MAC data header	28 bytes = 224 bits
PHY_preamble	144 bits (144 μsec)
PHY_header	48 bits (48 μsec)
ACK	14 bytes + PHY_preamble + PHY_header = 304 bits (304 μsec)
RTS	20 bytes + PHY_preamble + PHY_header = 352 bits (352 μsec)
CTS	14 bytes + PHY_preamble + PHY_header = 304 bits (304 μsec)

Table 3-1 IEEE 802.11b system parameters

Due to the difficulty and high cost of manufacturing transceivers which can transmit and receive simultaneously, and also because of the inherent hidden stations problem in wireless network (to be explained in section 3.3), IEEE 802.11 employs *virtual carrier sensing* functions for medium state detection, in addition to the physical carrier sensing. If either one of them indicates that the medium is busy, the station defers the transmission.

Virtual carrier sensing is provided by the *network allocation vector* (NAV). Because stations within the hearing range of the transmitter can overhear the frames, if the frames carry the transmission duration information, the other stations can obtain this information and then defer their transmission accordingly. Based on this idea, Most IEEE 802.11 frames carry a duration field (in microseconds) used to reserve the medium for a time period, according to the time for which they expect to use the medium, including any frames necessary to complete the current operation. Other stations that receive the frame set their *NAV timer*, which is a timer that indicates the amount of time the medium will be reserved by another station, and count down from the NAV to 0. When the NAV is nonzero, the station believes that the medium is busy; when the NAV reaches 0, the station considers that the medium is idle. By examining the NAV, a station may avoid transmitting even when the medium appears idle according to the physical carrier sense.

Due to the noise, interference, and other propagation effects, the wireless link is significantly more unreliable than a wired one, even with error-correction codes. To overcome the unreliability transmission of the radio signal, for every packet, the IEEE 802.11 adapts ARQ (automatic repeat request) retransmission strategy, which is essentially a *stop-and-wait* protocol (see Figure 3-2). This protocol makes sure that the current packet has been received successful before the next packet is being sent, or drops the current packet after a couple of transmission attempts (details will be given below). Notice that the data frame and acknowledgement frame (MAC ACK) are separated by SIFS, which is shorter than other IFS. The SIFS interval is independent of the data transmission rate.

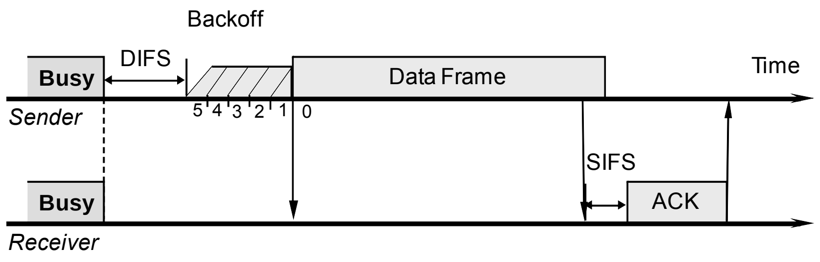


Figure 3-2 IEEE 802.11 basic transmission mode

For the purpose of ARQ, IEEE 802.11 includes a frame exchange protocol. When a station receives a data frame from another station, it returns an acknowledgement (ACK) frame to notify the source station. This exchange can be treated as an *atomic unit* because SIFS is used. The shorter interval of SIFS allows the receiver to grab the medium before another type of frame can be transmitted.

The source will retransmit the frame a couple of times if it does not receive an ACK within a short period of time. Therefore, the source sets up a retry counter which will increase for every transmission failure. Each time the retry counter increases, the contention window is expanded to the next greatest power of two. Contention window sizes are 1 less than a power of 2, e.g., 31, 63,127, 255, 511, 1023. The maximum contention window, *CWmax*, is limited by the physical layer. If the sender fails to transmit a packet in *CWmax,* it will randomly pick a time in the *CWmax* again, without increasing the contention window. The contention window will be reset to CW_0 if one of the events listed below occur.

(1) The packet is transmitted successfully

(2) The number of retransmissions reaches the associated retry limit and the packet is dropped

In IEEE 802.11, reception of acknowledge frame is the only indication of a successful transmission. The IEEE 802.11 standard requires that a data frame is discarded by the transmitter's MAC after a certain number of unsuccessful transmission attempts. The maximum attempts are determined by the frame length. Frames that are shorter than a pre-defined threshold are considered to be short; frames longer than this threshold are long. Depending on the length of the frame, it is associated with a short or long retry counter. Defined in the IEEE 802.11 standard, the default values of `short retry counter` and `long retry counter` are 7 and 4, respectively.

3.2.1 Backoff procedure with DCF

IEEE 802.11defines the transmission procedure as follows.

(1) The higher-layer protocol passes a packet to MAC for transmission

(2) The sender senses the channel

a. If the medium is found idle, the station waits for a DIFS period. If the medium remains idle, the station transmits immediately. The station goes back to the step (1).

b. If the medium is found busy, the station first waits for the medium to become idle for DIFS. After this, it sets the backoff timer to a random integer number of slots that is drawn from a uniform distribution over [0, *w*-1], where value *w* the number of slots in the *contention window* or *backoff window*.

(3) The sender enters the backoff stage.

In the backoff stage, the station that picks the lowest random number attains the medium. The stations that lose the competition enter the backoff stage again and will retry the transmission for a couple of times. The packet will be discarded after a certain number ℓ of retransmission attempts. The *backoff stage k* is the round of re-transmission $k \in \{0, 1, \ldots, \ell\}$, which determines the contention window size. Each time the retry counter *k* increases, the contention window CW_k is doubled. The maximum size of the contention window is determined by the physical layer. For example, for the DSSS physical layer $CW_{\min} = 32$ and $CW_{\max} = 1024$. The contention window is reset to its minimum size when the frame is transmitted successfully, or the associated retry counter is reached, and the frame is discarded.

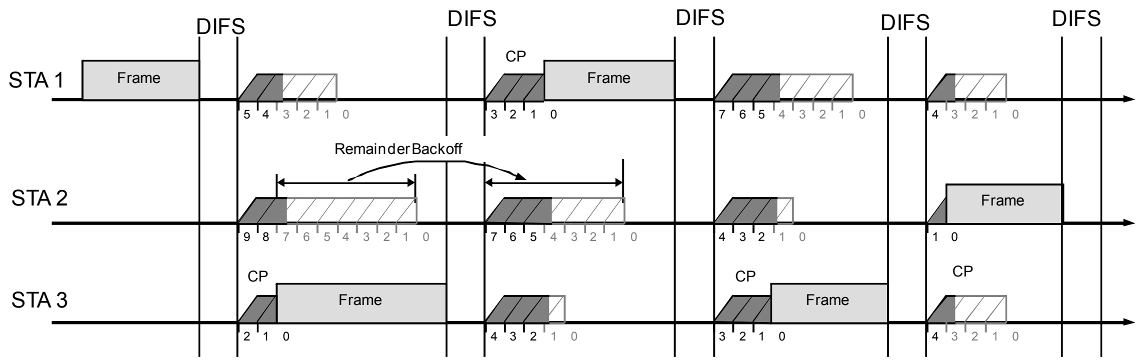


Figure 3-3 The backoff mechanism of the 802.11 MAC.

Figure 3-3 illustrates the procedure of the backoff mechanism. With multiple stations contending for the channel, once the channel is sensed idle for a DIFS, every station with a packet to transmit decrements its backoff timer. The station whose timer expires first begins transmission and the remaining stations freeze their countdown and defer their transmission. Once the current station finishes transmission, the process repeats again. The stations that were preempted during the countdown start decrementing their timer from where they left off rather than selecting a new countdown period.

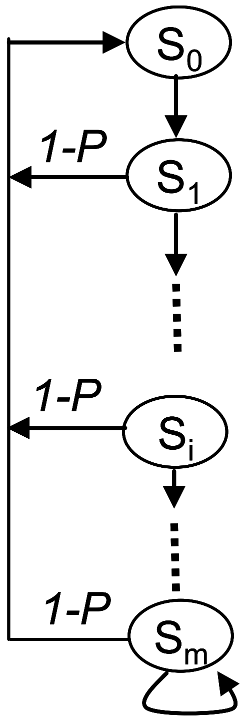


Figure 3-4 Simple Markov chain representing the backoff procedure

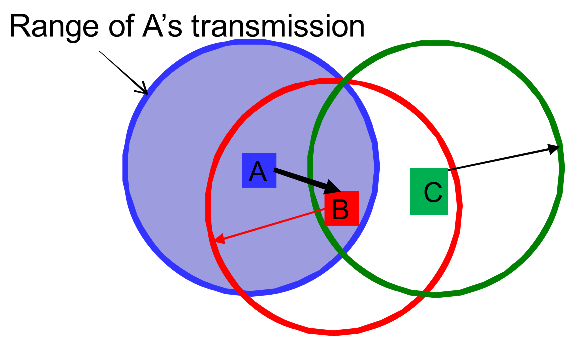


Figure 3-5 Illustration of the hidden terminal problem

The backoff procedure can be represented by a simple Markov chain, as shown in figure 3-4. *Si* represents the different back-off stages, and *p* is the collision probability and therefore, *1-p* is the probability of a successful transmission. Due to the different size of *CW* in different backoff stages, a more complicated Markov chain is needed to provide accurate analysis. This will be introduced in section 3.6.

3.3 Link-layer interference

Due to the limited radio transmission range, the air medium is partitioned into broadcast regions, rather than being a single broadcast medium. In IEEE 802.11, the partitions are called BSSs. Characteristics of wireless network based on IEEE 802.11 are listed below.

(i) Not all stations within a partition can necessarily hear each other

(ii) The broadcast regions can overlap. The former causes the hidden station problem and the latter causes the exposed station problem.

Different from the wired broadcast medium, the transitivity of connectivity does not apply any more. In wired broadcast networks, such as Ethernet, the maximum network size has been defined in a way that all stations in the network can hear from each other within a given maximum time period. Therefore, if station *A* can hear station *B* and station *B* can hear station *C*, then station *A* can hear station *C*. This is not always the case in wireless broadcast networks. Figure 3-5 demonstrates a typically scenario. The circles indicate the transmission ranges of stations. It can been seen that station *C* cannot hear station *A*'s transmissions and may mistakenly conclude that the medium is idle and start transmitting. The radio signal from C will interfere at *B*, wiping out the packet sent from *A*. This phenomenon is called *hidden station problem*. Generally, a station *Y* is considered to be hidden from another station *X* in the same receiver's area of coverage if the transmission coverage of the transceivers at *X* and *Y* do not overlap. The basic CSMA procedure is shown in the figure 3-6 (a), which demonstrates the vulnerable period.

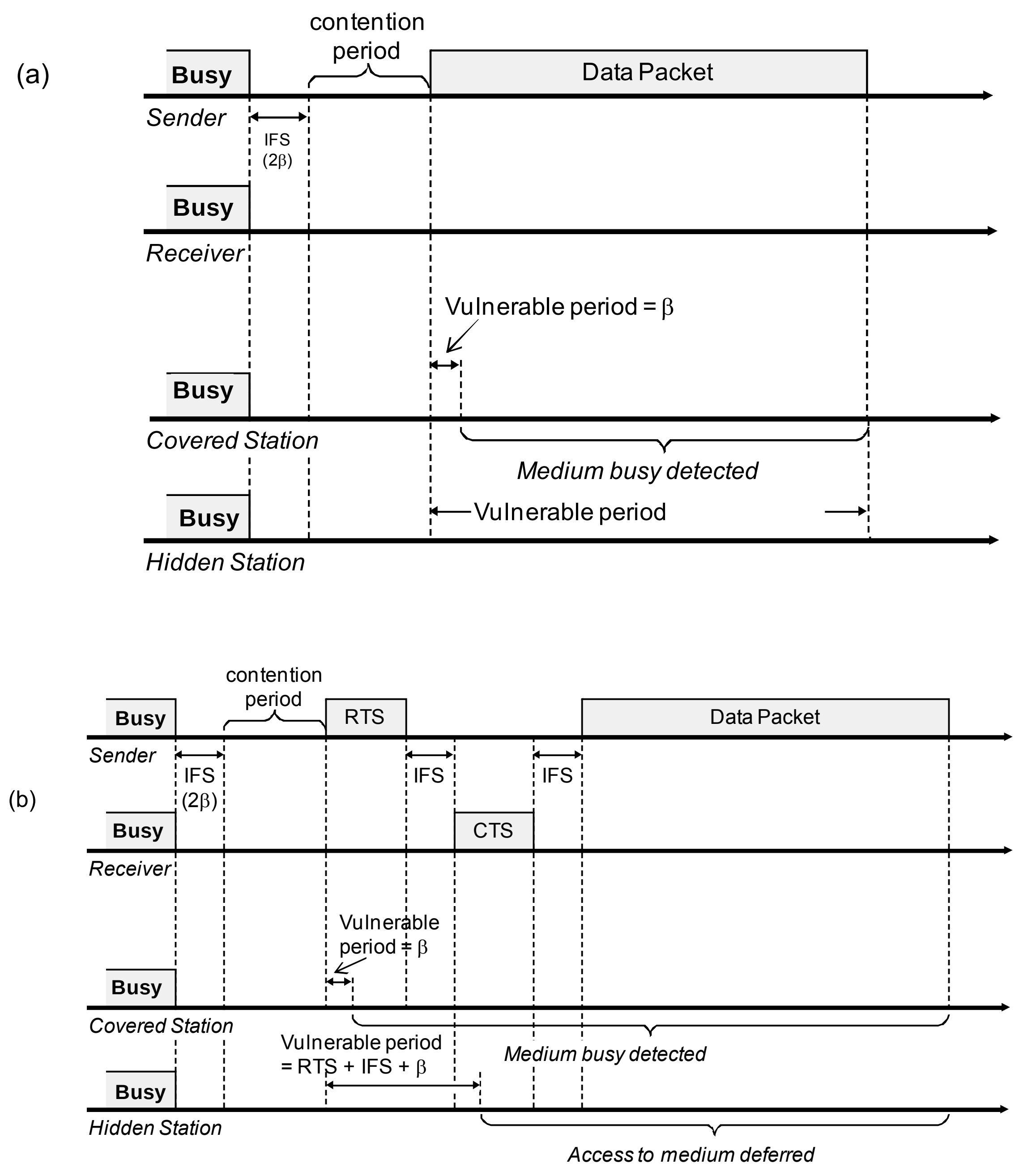


Figure 3-6 (a) Basic CSMA transmission mode. (b) The MACA protocol atomic unit exchange (three packets: RTS, CTS, and Data involved)

To alleviate the hidden station problem, the key insight is that collisions are located at receiver. A solution is to use the receiver's medium state to determine transmitter behavior. IEEE 802.11 specifies *Multiple Access with Collision Avoidance for Wireless* (MACAW) for which includes three-frame atomic exchange shown in figure 3-6 (b). The basic idea behind this scheme is that the sender and receiver exchange 2 short frames (RTS/CTS) before sending the data frame to allow more stations to hear the transmission, and therefore, avoid transmitting for the duration of the upcoming (large) data frame. Through this indirection, the sender performs "floor acquisition" so it can transmit successfully since all other stations which hear the RTS/CTS will remain silent for the duration of transmission. The rationale is, should a collision happen, it should last only a short period. This objective is achieved by allowing collisions only on RTS packets. Since RTS and CTS packets are very short, unlike data packets which can be very long, the collision duration is minimized.

3.4 Solution for Hidden Station in IEEE 802.11

IEEE 802.11 DCF protocol is based on the MACAW protocol for solving the problem of hidden and exposed stations. The key differences arise due to using the strategies of stop-and-wait ARQ and virtual carrier sensing.

The RTS/CTS exchange cannot eliminate the hidden station problem but partially solve it. The 4-way handshake of the RTS/CTS/DATA/ACK exchange of the IEEE 802.11 DCF protocol (Figure 3-7) requires that the roles of sender and receiver are interchanged several

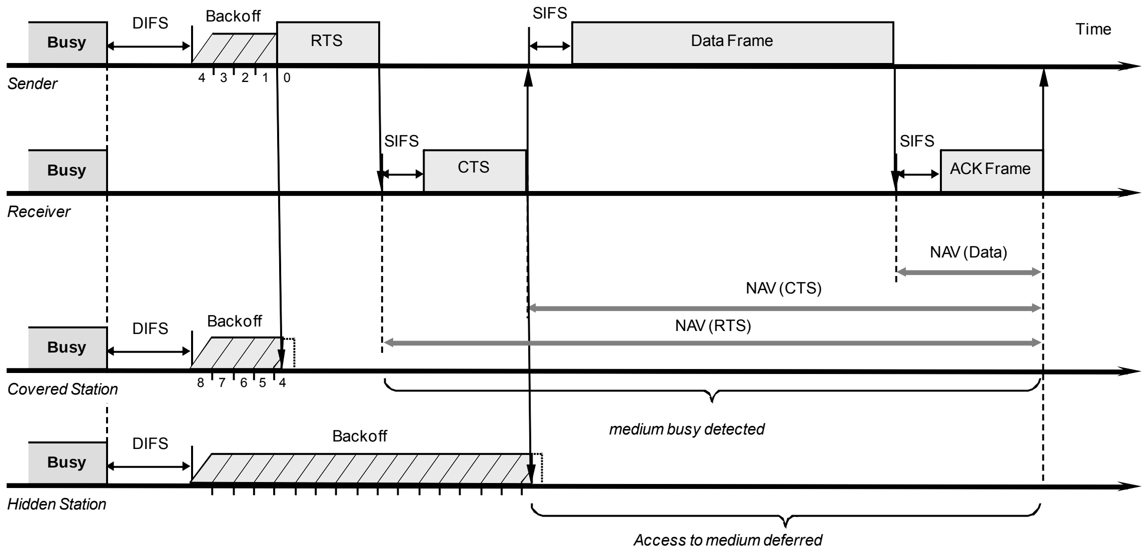


Figure 3-7 The IEEE 802.11 protocol atomic unit exchange in RTS/CTS transmission mode (four frames: RTS, CTS, Data, and ACK involved)

times between pairs of communicating nodes, so neighbors of the sender and receiver must remain silent *during the entire exchange*. This is achieved by using SIFS. Meanwhile, the RTS/CTS also carry the transmission duration information. The neighboring nodes update their Network Allocation Vector (NAV) values accordingly. By using SIFS and NAV, the stations ensure that atomic operations are not interrupted. The NAV time duration is carried in the frame headers on the RTS, CTS, data and ACK frames.

3.5 IEEE 802.11b PHY

In IEEE 802.11 [22], the wireless physical layer is split into two parts, called PLCP (Physical Layer Convergence Protocol) and PMD (Physical Medium Dependent)

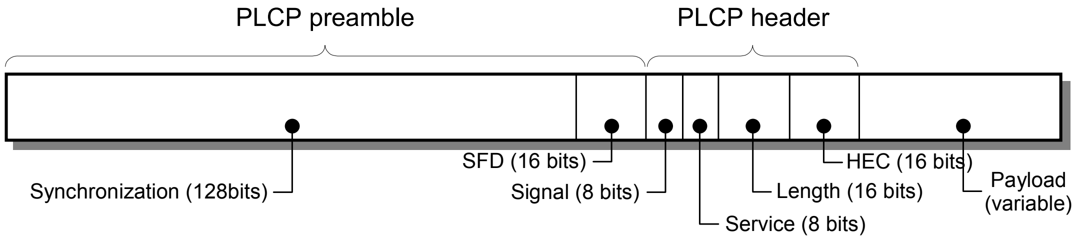


Figure 3-8 IEEE 802.11b PHY frame using DSSS.

sub-layer, as shown in figure 3-8. The PLCP presents a common interface for higher-level drivers to write to and provides carrier sense and CCA (*Clear Channel Assessment*), which is the signal that the MAC layer uses to determine whether the medium is currently busy.

The PLCP contains PLCP preamble and PLCP header. The preamble consists of a 144-bit that is used for synchronization to determine radio gain and to establish CCA. The next 48 bits are collectively known as the PLCP header. The header contains four fields: signal, service, length and HEC (header error check). The signal field indicates how fast the payload will be transmitted (1, 2, 5.5 or 11 Mbps). Independent from the payload transmission rate, in IEEE 802.11b, the PLCP is always transmitted at 1 Mbps. Thus, 24 bytes of each packet are sent at 1 Mbps. The PLCP introduces 24 bytes of overhead. Figure 3-8 shows the IEEE 802.11b PHY. Generally, the PHY layer and MAC layer overhead (e.g., header, RTS/CTS/ACK, etc.) are transmitted using a much lower transmission rate than the payload.

3.6 Throughput analysis of IEEE 802.11

We have discussed the IEEE 802.11 mechanism in details above. In this section, we introduce how to analyze the throughput.

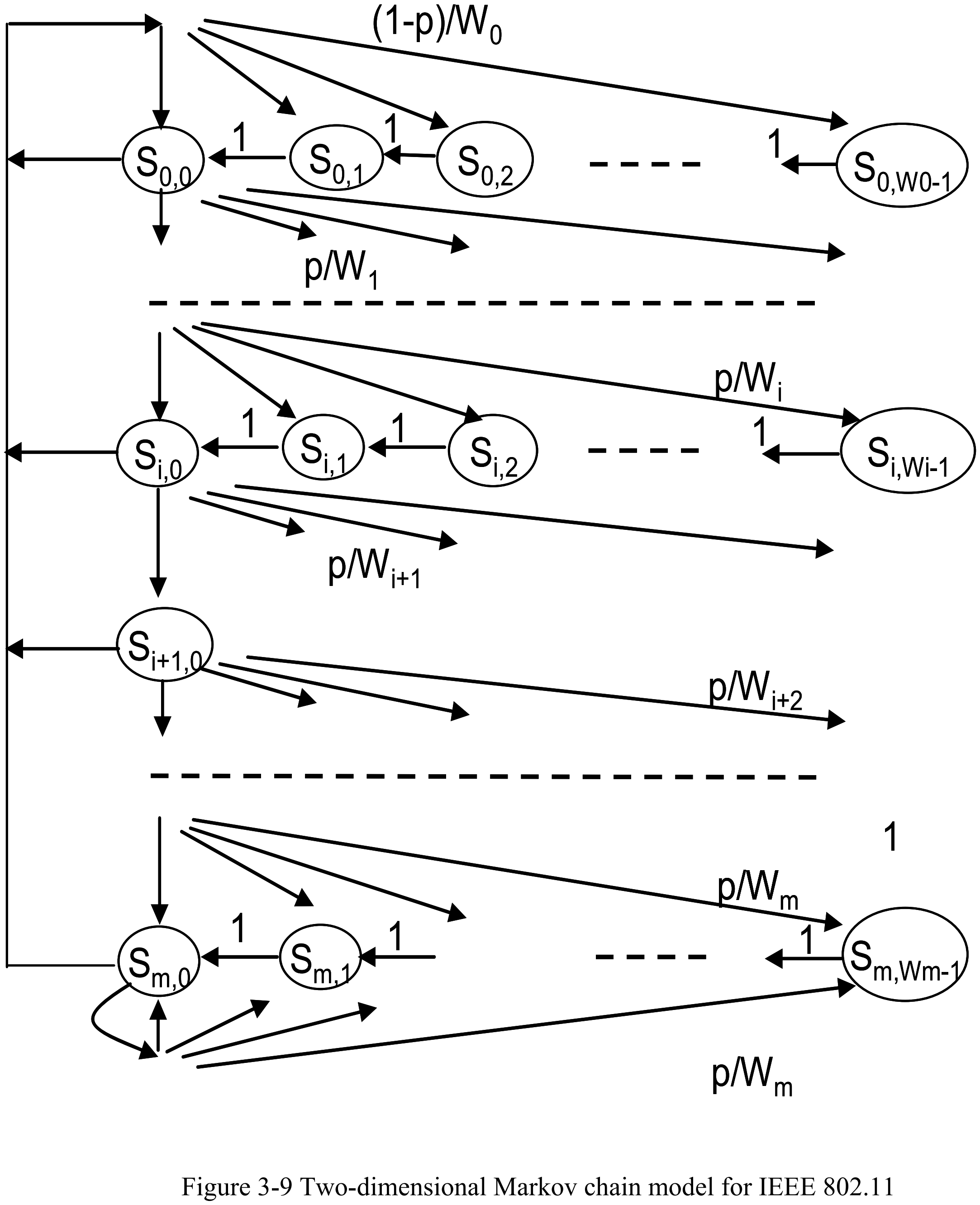


Figure 3-9 Two-dimensional Markov chain model for IEEE 802.11

The simple Markov chain presented in section 3.2 ignores the details of state transition of the change of contention window and therefore, it is not suitable for accurate analysis. Dr.Giusepee Bianchi[23] proposed a two-dimensional Markov chain to demonstrate the IEEE 802.11 MAC behaviors. In addition to define the state transition of back-off stage on the vertical direction, he proposed to add states in the horizontal direction to show the moment when the transmitter has different *CW* counts, as shown in figure 3-9. In transition states on the horizontal axis, the transmitter moves to a left state after every step with probability 1. When it reaches the left-most state, it either finishes the transmission successfully and then comes back to the state in the top-left, or moves to the state below if collision occurs. When it reaches the bottom line, it randomly jumps to one state in the same horizontal line if it fails to finish the transmission before it gives up the transmission and discards the packet, as stated in section 3.2.

In figure 3-9, S represents the states where the transmitter is in currently. p is the probability of collision between transmitters, which is constant and independent. Wm is defined as the backoff stage whose value is $CWmax = 2^m W$.

Using this Markov model, the normalized throughput, S, can be calculated by the formula below. S is defined as the fraction of time that the channel is used to transmit payload bits successfully.

$$S = \frac{E[payload\_\inf o\_transmitted\_in\_a\_slot\_time]}{E[length\_of\_a\_slot\_time]} \quad (3\text{-}1)$$

$$S = \frac{P_s P_{tr} E[P]}{(1-P_{tr})\sigma + P_{tr} P_s T_s + P_{tr}(1-P_s)T_c} \quad (3\text{-}2)$$

The detail meanings of symbols are summarized in table 3-2.

The denominator in the formula 3-1 can also be viewed as the average length of a slot time which includes an empty slot, a successful transmission and a collided transmission. Notice that the average slot length is different from the time slot σ defined in the IEEE 802.11 specifications.

Symbol	Meaning
Ptr	Probability that at least one transmission occurs in a given slot time
Ps	Probability of a successful transmission, conditioned on P_{tr}
Ts	The average time occupied by a successful transmission
Tc	The average time occupied by a collided transmission
σ	The duration of an empty slot time
E[p]	The average packet payload size

Table 3-2 Symbol meaning in formula 3-2

When RTS/CTS access mechanism is used, in a successful transmission, the transmission time is calculated by

$$Ts = RTS + SIFS + \delta + CTS + SIFS + \delta + H + E[P] + SIFS + \delta + ACK + DIFS + \delta \quad (3\text{-}3)$$

Where δ is the transmission delay, H is the header size, $E[P]$ is the expected value of the packet sizes

If a collision occurs during the transmission, because the collision can occur only on the RTS frames, the average occupied time is given by

$$Ts = DIFS + \delta \quad (3\text{-}4)$$

We will extend these formulas to evaluate the channel capacity consumed by packets with different sizes in chapter 5.

3.7 IEEE 802.11n

To further improve the IEEE 802.11 throughput, besides adopting new techniques in PHY layer, IEEE 802.11n also makes some modification to the MAC layer[24, 25]. The new MAC techniques to enhance the performance include:

- Frame aggregation

- Multiple Traffic ID block acknowledgement
- Power Save Multi-poll

Theoretically, the IEEE 802.11n can provide more than 100Mbps transmission rate. However, this does not necessarily mean that the TCP packets can be transmitted in this rate because TCP traffic is also controlled by congestion window. In some scenarios, e.g., in the slow-start TCP stage, there may not be enough number of packets to be bundled together to take advantage of the frame aggregation. And therefore, the benefit provided by the MAC aggregation is lost. Due to this mismatch between MAC layer and transport layer, the actual transmission rate may be much lower than the nominal transmission rate.

Chapter 4

Fast Handover for Multiple-interface Mobile Devices Connecting to a Single Foreign Agent in MIPv4 with MIH Support

4.1 Introduction

In heterogeneous networks, mobility is an important issue. Users expect uninterrupted services moving from one network to another. Traditionally, a mobile client is designed to use a single network technology and is equipped with only one type of network interface. Recently, a growing number of mobile devices are equipped with multiple interfaces that can receive services from different service providers, possibly via different access technologies like Ethernet, WiFi, 3GPP, 3GPP2, WiMax, LTE, etc. However, there is no generic solution for mobility access across different networks so far. Media Independent Handover (MIH, IEEE 802.21 standard) is an ongoing, evolving effort by the IEEE society to address this issue. By working together with a network-layer mobility management protocol, MIH makes it possible to achieve better handover performance than existing mechanisms, in terms of latency and packet loss rate. Currently, Mobile IPv4 (MIPv4) is the dominant mechanism for mobility management and is expected to persist into the future. However, when multiple interfaces of a mobile client are connected to the same FA, MIPv4 and its existing improvements do not perform well when the active network interface fails unexpectedly. In this chapter, we propose a novel mechanism that accomplishes fast handover. Based on this, we then propose a new MIP extension. Compared with the existing mechanism, our experiments show that our algorithm achieves much better handover performance, with MIH support. Our method also allows foreign-agent-bicasting, which can help improve transmission reliability by combining traffic from different links, through the same foreign agent.

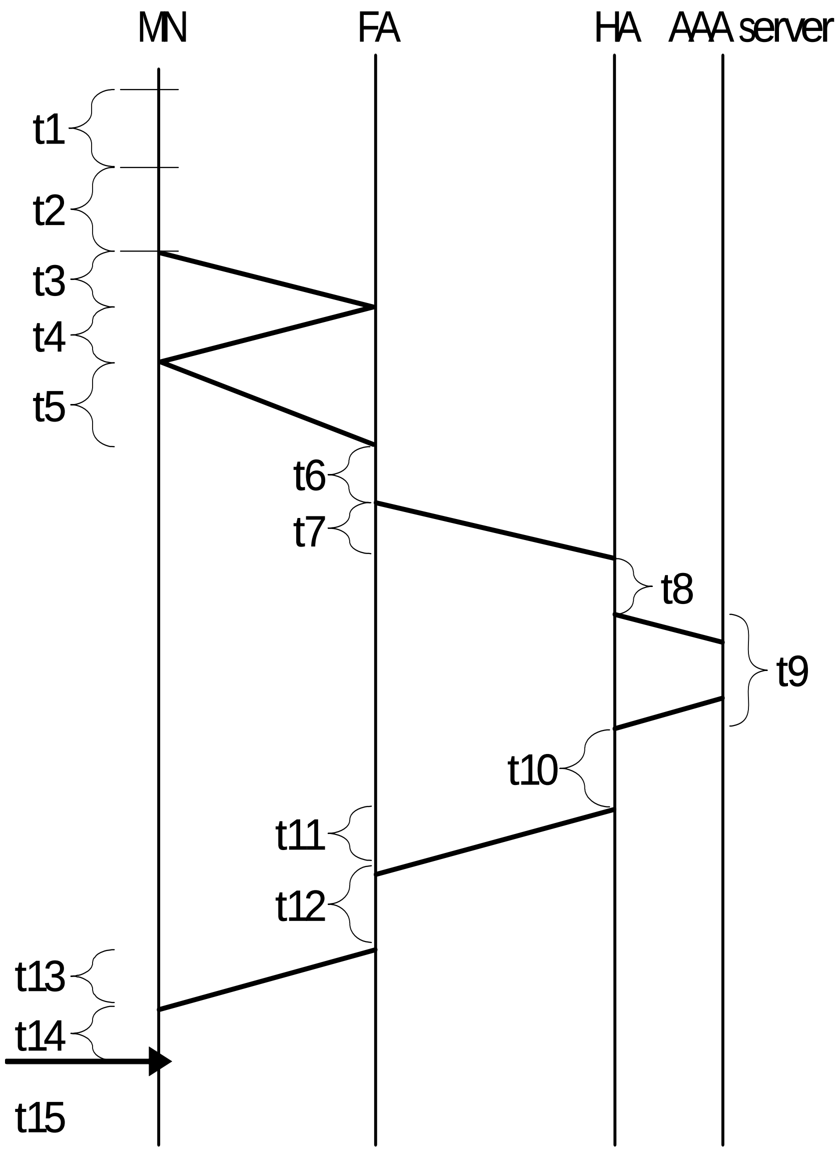


Figure 4-1 Timing diagram for handover using Mobile IP

4.2 Mobile IP timing diagram

Client mobile IPv4 (MIPv4)[26-28] is the most popular L3 mobility management protocol. For simplicity reason, we use MIP or Mobile IP to represent MIPv4 or Mobile IPv4 subsequently. Let's look at a scenario when a mobile client entering a MIP-enabled network searches for a Foreign Agent. Through a registration procedure initiated by the MN, a tunnel is established between the Foreign Agent and the Home Agent. All data from Correspondent Nodes (CN) destined to the MN are intercepted by the HA and forwarded through this tunnel to the current care-of-address (CoA). The FA is then responsible for delivering the data to the MN. The MIP mechanism provides a transparent solution for mobility management, which means that the CN does not need to know where the MN resides (e.g., home or any visited network). When a MN roams from one network to another, a handover must be completed to provide for session continuity. The handover includes L2 and L3 handover.

Figure. 4-1 shows the detailed handover timing diagram for the MIP. The explanations of stages during a handover are given as follows.

- t1: L2 disconnection detection
- t2: L2 connection up (e.g., power up, association)
- t3: MN solicitation
- t4: FA advertisement
- t5: MN registration request
- t6: FA processing and MN-FA authentication

- t7: Registration forwarding
- t8: Home Agent (HA) processing
- t9: Authentication, authorization and accounting
- t10: HA processing (e.g., tunnel setup)
- t11: Registration reply
- t12: FA processing (e.g., visitor table update)
- t13: Registration reply
- t14: MN update (e.g., routing table update)
- t15: Connection resumed

Layer 2 handover occurs in the periods of t1~ t2 while Layer 3 handover takes place in the periods from t3 to t14. The total latency is given by the formula below.

$$L_t = \sum L_i \qquad (4\text{-}1)$$

Where L_t is the total latency, L_i is the latency of each stage, $i \in \{1,2,\ldots, 14\}$.

The overall latency also can be decomposed as detection latency and registration latency, as described by the formula below.

$$L_t = L_d + L_r \qquad (4\text{-}2)$$

Where L_t is the total latency, L_d is the detection latency, and L_r is the registration latency.

Many methods have been proposed to reduce the various types of handover latencies, mostly through switching the order of L2/L3 handover, or reducing the absolute latency value by localizing the L3 registration processing. As an example of the first approach, Low-latency Handover[29] proposes a method called *pre-registration* that allows the MN to register with a new FA through the current FA before attaching to a new network. Hence, the MN can start the L3 handover before the L2 handover is completed. Ideally, the L3 handover latency, shown as *t3 ~ t14* in Figure 4-1, can be removed and therefore a quicker handover is achieved. Regional Registration [18] suggests a hierarchical architecture that alleviates the high latency of MN-HA registration and authentication by adding a regional GFA (Gateway Foreign Agent) to provide a quicker registration locally. All of the abovementioned algorithms are designed for mobiles with a single interface registering with two different FAs. However, for vertical handover at the same FA, these methods are not directly applicable without extensive modifications. Note that different access technologies typically use different authentication mechanisms.

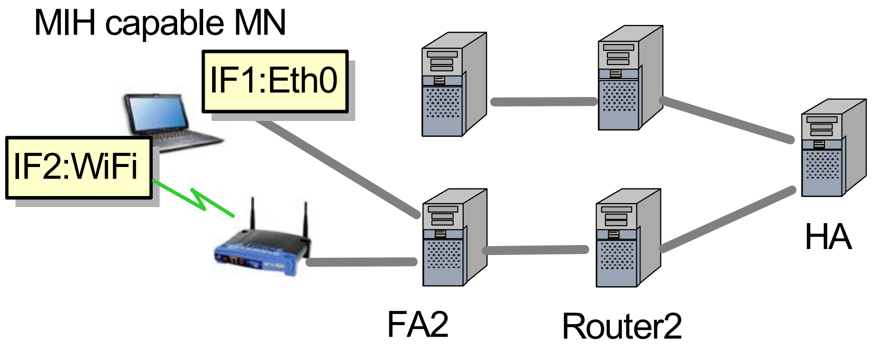


Figure 4-2 A MN connecting to one FA through multiple interfaces

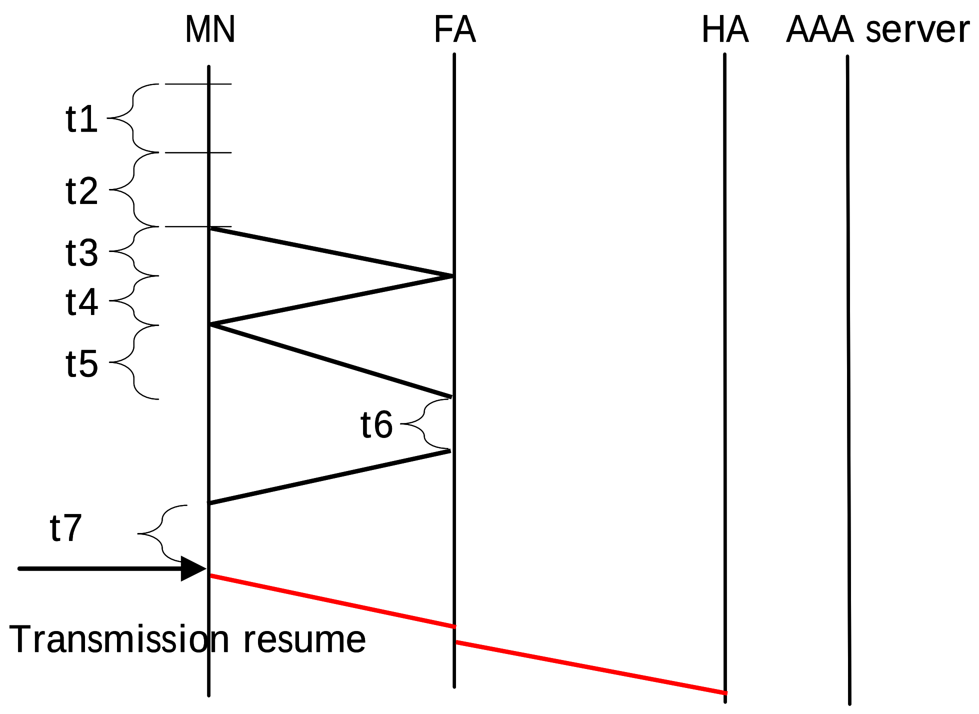


Figure 4-3 Timing diagram for the new method

4.3 Multiple-interface MN connecting to a FA

4.3.1 Existing problem in traditional client MIP

Here we adopt a use case to identify an existing problem in the traditional client MIP. Imagine a scenario where a mobile client with Ethernet and WiFi interfaces connects to HA via the same FA regardless of the interface which is active. Figure 4-2 illustrates this scenario. We assume that MIP service is available and that the same FA is used for both Ethernet and WiFi in the same building. While in the office, the user prefers to use Ethernet, which typically provides better connection quality, e.g., higher security protection, higher transmission rate, than wireless connection. When the user leaves the office, the connection must be transferred to WiFi. Generally, unplugging an Ethernet cable is not pre-detectable and therefore, the L2 disconnection is an unexpected event. If the user is using the client MIP without MIH, he will lose his connection for a relatively long period of time until the MIP detects the disconnection by methods defined in [26], which also has been introduced in chapter 2. For the case where MIH is available, a quicker handover can be achieved because the MIH can notify the MIP that the L2 connection is lost, without the significant latency of MIP move detection. Using IF2 (WiFi interface, see figure 4-2), the MIP client in MN starts a L2, L3 handover process that goes through the *t1~t15* stages in Figure 4-1, and the L3 handover occurs after L2 handover is completed. However, in most cases, the FA-HA latency, which typically is in the order of hundreds of *ms*, dominates the overall latency. If we could eliminate this latency, we will reduce the latency further. One possible solution is to have IF2 ready for handover all the time, whenever the IF1 (Ethernet interface, see figure 4-2) connection is lost, the MN can switch to IF2 after disconnection is detected and therefore remove most stages of the *t2~t14*. Therefore, a faster handover can be achieved. Figure4-3 demonstrates the timing diagram for this improvement.

MN	IP	MAC	Lifetime
MN1	xx.xx.xx.xx	xx:xx:xx:xx	xx
MN2	xx.xx.xx.xx	xx:xx:xx:xx	xx
MN3	xx.xx.xx.xx	xx:xx:xx:xx	xx

Table 4-1 Example of FA visitor table

MN	MN Home address(IP)	CoA (IP)	Lifetime	SPI
MN1	xx.xx.xx.xx	xx:xx:xx:xx	xx	xx
MN2	xx.xx.xx.xx	xx:xx:xx:xx	xx	xx
MN3	xx.xx.xx.xx	xx.xx.xx.xx	xx	xx

Table 4-2 Example of HA registration table

Next, we analyze the standard MIP to see whether it can support multiple active interfaces in one MN.

In the standard MIP, when a new registration is authenticated and completed successfully, The HA sends messages back to FA and MN. After receiving the positive response from the HA, the FA updates its binding record, e.g., visitor table. Table 4-1 shows a simplified visitor table. The new registration record replaces the current one. Specifically, assume that the MN is using IF1 (Eth0) and it sends a new registration request over IF2 (WiFi). As defined in the procedure by the standard MIP, the FA forwards the request to the HA for registration. Table 4-2 illustrates a simplified HA record table for its registered MNs. Because the FA to which MN attaches is not changed, from the HA's perspective, the registration request is no different from the one sent over Eth0, i.e., it looks like a re-registration rather than a move, i.e., FA switching. Therefore, the HA simply follows the standard MIP procedure and only updates the expiration timer. However, from the FA's perspective, the link-layer address, e.g., MAC address, in the binding record (visitor table) would be replaced by the link-layer address of IF2 once the HA reply is received. As a result, the active connection will be transferred to IF2 even if IF1 can still adequately support the connection. To minimize the handover latency, the desired behavior would be to maintain the current FA registration and add a secondary one in preparation for a handover. However, in the current MIP standard, there is no mechanism to support a secondary registered network interface and, more importantly, to control the states of multiple registered connections. Therefore, using the standard MIP, it is impossible to have a backup interface that is ready for handover, when multiple interfaces are connected to a single FA.

4.3.2 Interface simultaneous/backup FA binding and novel MIP extension

MN	IF	IP	MAC	Active	Lifetime	Auth
MN1	IF1	xx.xx.xx.xx	xx:xx:xx:xx	1	xx	1
	IF2		xx:xx:xx:xx	0	xx	0
MN2	IF	xx.xx.xx.xx	xx.xx.xx.xx	1	xx	1
MN3	IF	xx.xx.xx.xx	xx.xx.xx.xx	1	xx	1

Table 4-3 Example of extended FA visitor table

Based on the discussion above, we can see that the re-registration adds more latency to resume the connection after handover. If we can mitigate the latency caused by re-registration, the overall handover latency will be minimized. In order to reduce the latency of handover caused by unpredictable events, it is desired to have a handover-ready backup interface. To that goal, we introduce a concept of *Multiple Interface Simultaneous/backup FA binding*, which allows the FA to maintain records of multiple registered interfaces (from the same MN). Based on this, we propose a new MIP extension. The first step is to allow simultaneous interface binding by expanding the FA visitor table to keep records of all registered interfaces, along with their link-layer addresses and active/backup status. Only active interfaces are involved in forwarding the traffic to the MN. The second step is to define a new MIP extension (Figure 4-4), called *simultaneous FA binding extension*. For clarity reason, we call our new method *SFB-MIP* in the following sections. Note that we use the conventional term of bi-casting even if more than two interfaces are used to send duplicate packets.

4.3.3 Basic flow to set up a simultaneous FA binding

In this section, we show the basic flow to set up a simultaneous FA binding. Assume the MN is connected to a FA using IF1 and it begins to register IF2.

1. *The MN sends a registration request with the new extension over IF2.*
2. *The FA checks the registration and finds this extension. From the MN home IP address, FA knows that the request comes from the same MN as IF1.*
   a. *The FA forwards the registration to the HA for authentication and timer update, after removing the simultaneous FA binding extension.*
   b. *The FA updates the visitor table. Table 4-3 shows an example of simplified table. Some parameters, e.g., HA address, SPI, etc., are not shown.*
3. *The HA sends reply after processing the registration message.*

4. The FA updates its visitor table again based on the reply. If the HA accepts the request, the FA updates the corresponding entry in the visitors list as authenticated (auth). If the HA denies the request, the FA need to remove the IF2 from the visitor table.

5. The FA relays the HA reply to the MN using the interface over which the original request was received (IF2).

6. MN updates its interface record.

7. When MN needs to use IF2, e.g., when MIH detects that IF1 is unplugged and notifies MIP, the MIP sends a registration through IF2 with the "A" bit set and "B" bit unset to inform FA to use IF2 to forward packets. Meanwhile, the MN performs the necessary local update, e.g., Routing Table update. Note that if "A" bit and "B" bit are both set, the FA enters bi-casting state and both IFs are active.

8. Because IF2 is marked as "auth", the FA sets IF2 as active, deactivates IF1, and sends a reply to MN. After this, the MAC of IF2 is used to send packets and the transmission is moved to IF2, eliminating most of the latency in t3~t14. Additionally, the FA forwards the registration (without the extension) to the HA for normal processing (this includes forwarding any HA registration replies to the MN when they are received from the HA). Figure 4-3 demonstrates the time diagram of our new protocol.

```
0                   1                   2
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0
+-+-+-+-+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-
|     Type      |     Length    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Lifetime             |A|B|Reserved
+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-+-++-+-+
```

Type: Identifies this simultaneous FA binding extension.

Length: Indicates the length (in bytes) of the data fields in this extension, not including the Type and Length bytes.

Lifetime: The time in seconds before the binding with the FA is considered expired. A zero value indicates a MN's request for deregistration of the specific interface over which this extension is sent. Note that it is different from the lifetime of MN registered at HA.

A: Active bit. If the "A" bit is set, this link is active and should be used to forward traffic. If it is unset, this link is not ready for transmission.

B: Bi-casting. If this bit is set, the FA uses all active interfaces to forward traffic. If unset, only the interface over which the latest registration is sent will be the active interface.

Figure 4-4 Simultaneous FA binding extension

The basic rules for processing of the "A" and "B" bits are:

(1) "A" bit set, "B" bit set: add this interface to the set of active interfaces

(2) "A" bit set, "B" bit unset: activate this interface and deactivate all other interfaces, just as the step 7 in the basic flow

(3) "A" bit unset: either pre-register an interface that the MN does not want it to be activated at the current moment or deactivate an active interface, in all cases the "B" bit is ignored.

After two interfaces from the same MN are bound to a FA, we say that this FA is in the FA-backup-binding state for that MN.

When a MN activates an interface, it sends the FA a MIP registration with this extension and the "A" bit set through this specific interface. The FA forwards the MIP registration (with this extension removed) to the HA and sends a reply to the MN in the meantime. Since this interface has already been authenticated, the reply is used to notify the MN that it can start to use this interface immediately (i.e., it needs not wait for the MIP registration reply from HA). The MN will retry if no response is received within a short period (depending on the specific technology used by that interface).

For simplicity, the basic flow described above is based on the case with only two interfaces present. It can be easily extended for MNs with more than two interfaces.

Before registering the second interface, MN needs to determine if both interfaces are connected to the same or different FAs. This can be easily determined by checking the agent advertisement or reply to the MN solicitation.

Interface deregistration in FA is achieved when the MN sends a registration request that includes this new extension with a zero-lifetime over the interface that needs to be deregistered. It is also possible that the MN does not send any re-registration messages through the interface and the FA removes the corresponding interface when the corresponding timer expires. Another option arises when the MN sends a de-registration message without the extension attached. In this case, all interfaces are deregistered at the FA and the entire MN-FA context is torn down. Note that this deregistration needs to be authenticated by the HA as the standard MIP requires.

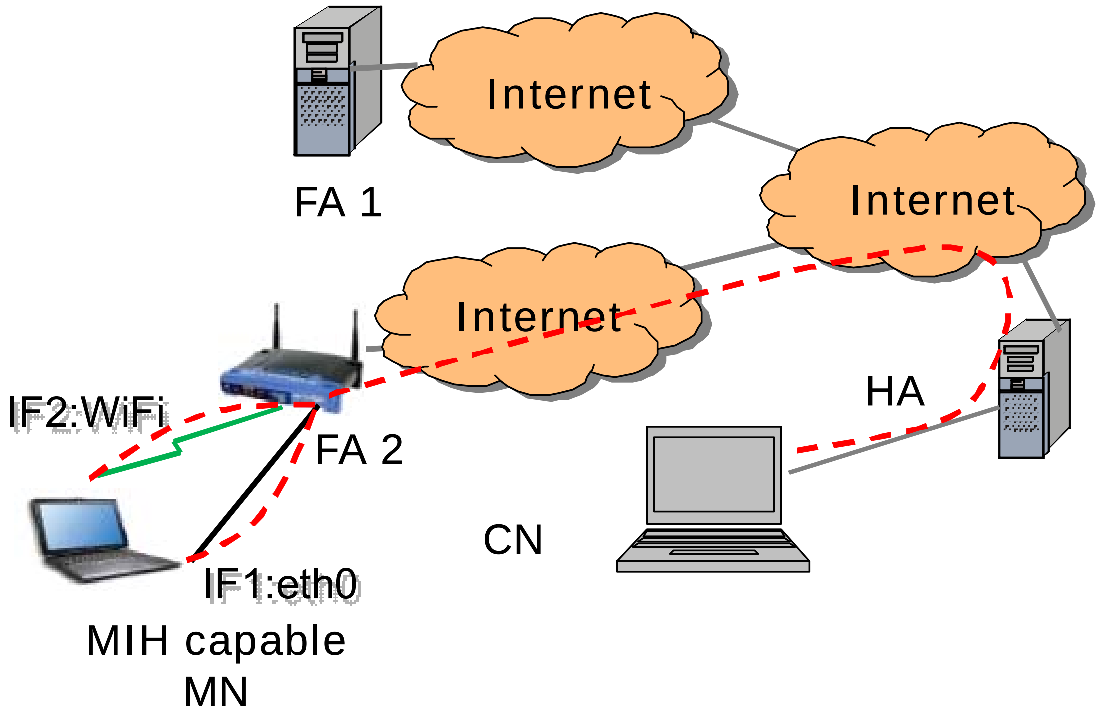


Figure 4-5 Traffic path for FA bi-casting

Sometimes, it is possible to achieve a better transmission performance using FA bi-casting in the presence of high data error rate in interfaces. This is likely to happen in the case when multiple cards are using wireless connections. Bi-casting means that packets are duplicated and then send through different paths. Because our method allows the FA to keep information of all interfaces in the MN, it then can provide FA bi-casting if multiple interfaces are active simultaneously. Then, packets are duplicated, properly framed for each active L2 connection and sent via the corresponding interfaces. We call this *FA bi-casting*, which is unlike the HA bi-casting (See figure 2-5) in that the packets are duplicated in the FA rather than in the HA. Figure 4-5 demonstrates the traffic path for FA-bicasting. As in the HA bi-casting, improved bi-casting techniques like FEC-bicasting are applicable in FA bi-casting.

4.3.4 Security and other issues

In the standard MIP, the MN-FA authentication is optional while the MN-HA authentication is mandatory. However, to achieve fast handover in the case of multiple interfaces connecting to one FA, security association between MN and FA is generally required, as the messages to activate interface(s) have to be authenticated by the FA to achieve desired latency reduction.

The interface Simultaneous/backup FA binding messages are transparent to the HA. To achieve this, the FA removes the extension after it processes the registration request. The extension order should be as defined in [26]. Hence, the authentication between MN and HA will not get out of sync.

4.4 Test-bed design and implementation

To evaluate our proposal and validate its effectivity, we have developed a complete test system which includes hardware test-bed and software. The software contains 3 major parts, (1) MIH component (2) MIH-capable device driver (3) Mobile IP component. Because Linux provides source code which allows us to make any necessary modification, we adopt Linux as our operating system.

4.4.1 Hardware components

The hardware used in our test-bed are

- IBM T20 laptop for mobile terminal, which is equipped with WiFi and Ethernet.
- IMB T20 laptop used as the corresponding node. The traffic generator is running in this device.
- Linux boxes running Foreign Agent and Home Agent. The Linux version is Fedora 2.6.18.
- Linux box used as link simulator.
- Linksys WiFi access point connected to a Linux box that runs the FA
- Cisco routers and switch.

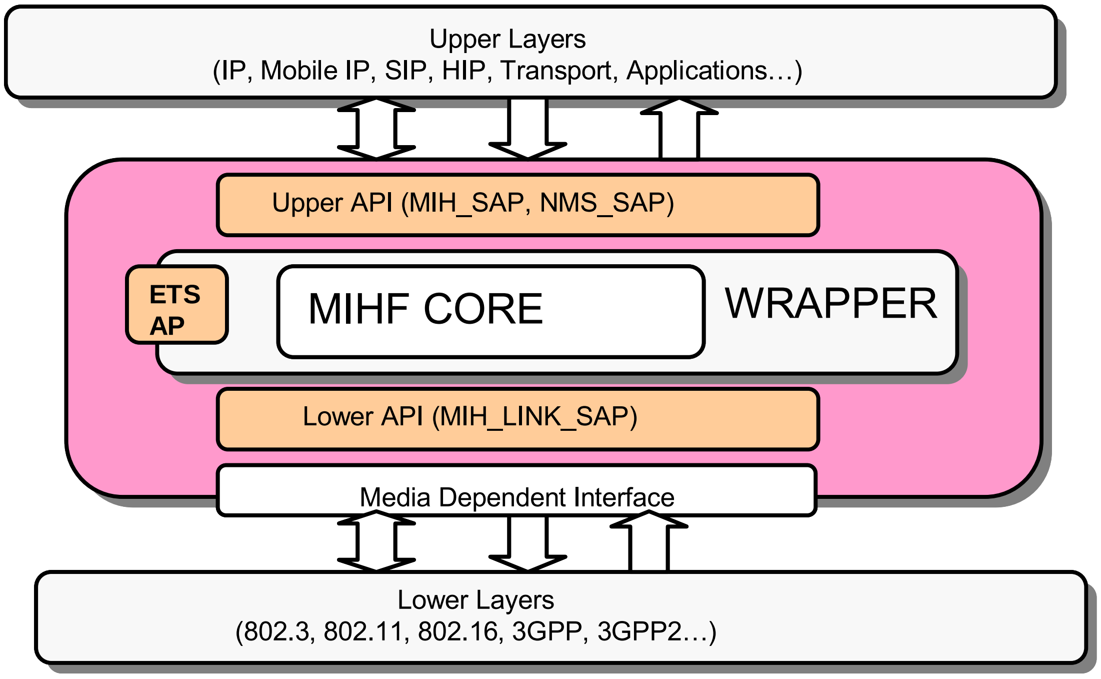

Upper Layers
(IP, Mobile IP, SIP, HIP, Transport, Applications…)
Upper API (MIH_SAP, NMS_SAP)
ETS AP
MIHF CORE
WRAPPER
Lower API (MIH_LINK_SAP)
Media Dependent Interface
Lower Layers
(802.3, 802.11, 802.16, 3GPP, 3GPP2…)

Figure 4-6 MIH software architecture

4.4.2 Software components

4.4.2.1 MIH software components

For our research, we have developed an implementation of the IEEE 802.21 (MIH) draft. The high-level software architecture is shown in Figure 4-6. The key MIH functions (MIHF CORE) are wrapped in a wrapper which isolates the implementation details of MIH functions to other software components. This approach allows us to change the MIHF without affecting the whole system. This is very important because the MIH standard (draft) is in the evolution phase and it may be modified very frequently. On top of the wrapper, an upper API is provided to connect to the upper layer. Below the wrapper, the lower API is also provided for communication with the lower layers.

4.4.2.2 MIH-capable device driver

In MIH, the lower layer should detect the availability of different access networks. This may include that a new card is inserted or unplugged, or a connection to a new access network becomes available. The off-the-shelf products do not provide this type of ability. Therefore, we need to modify WiFi driver and Ethernet driver [30] to send messages to upper layer when such types of events occur.

Most of the off-the-shelf WiFi cards do not come with source code. The MadWiFi project [31, 32] comes to rescue. This project is carried out by a team of volunteer developers working on Linux drivers for WiFi. It provides MadWiFi driver which is one of the most advanced WiFi driver for Linux, with source code available online. Based on the compatibility shown on their website, we select a WiFi card made by Airnet. For Ethernet, we use the internal Ethernet which comes with the IBM T20 laptop. The source code is available in the Linux source tree.

4.4.2.3 Mobile IP software

The Mobile IP software includes 3 components, MN, FA, HA which implement the MIPv4 standard and our proposed MIP extension.

4.4.2.4 Integration of software components

All software components discussed above run as separate programs which are independent to each other. To integrate them into one system, we use inter-process communication technique and script language[33-36].

4.4.2.5 Test software

To measure the handover latency, we have developed a traffic generator which can keep sending UDP packets from one end point to another in a connection, with packet sequence number in every packet.

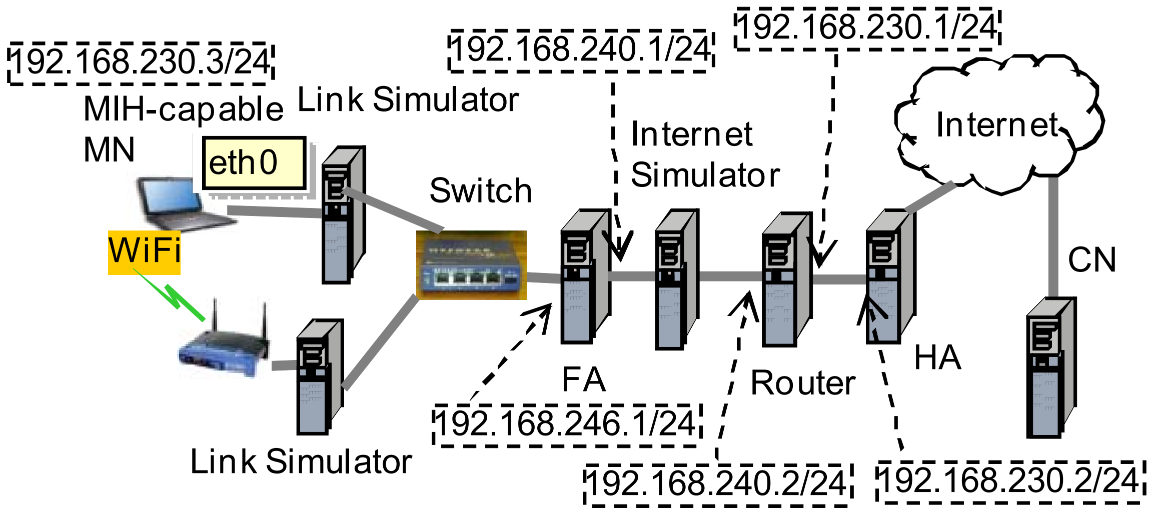


Figure 4-7 Test bed topology

4.4.3 Test-bed topology

Figure 4-7 illustrates the basic topology of our test-bed. The interface (WiFi & Ethernet) drivers in the MN have been modified to provide MIH capabilities and faster link status detection. When a network card is plugged into or unplugged from the laptop, the Linux kernel will detect this event and send messages to upper layer immediately. In order to integrate the modified drivers, the Linux kernel has been re-compiled. The simulators like Internet simulator in the test-bed can mimic a real network environment by introducing impairments, such as traffic delay, packet loss, etc. We can easily simulator the FA-HA latency by changing the settings in the Internet simulator.

4.4.4 Supplementary tools

To check the network status, troubleshoot network, and monitor data traffic, we adopted the *wireshark* [37] and *tcpdump*[38] packet analyzer. *Wireshark* and *tcpdump* allow us to intercept and display network traffic being transmitted. *Tcpdump* is based on command-line and it is very easy to export the result to a text file for analysis. Wireshark is very similar to *tcpdump* but it has a graphic interface and provides many more processing options like information sorting and filtering.

To emulate the end-to-end network characteristics under different network conditions in the test bed, we have multiple options. One is using network emulator like *NIST Net*[39], another one is to use the *qdisc* tool provided by Linux. *NIST Net* can emulate many network conditions and allows users to add delay, drop selected packets, add jitter, etc.

qdisc is a Linux component for traffic control which is a scheduler for output interface. It can be used to control the output traffic and therefore, it can be used to emulate some network characteristics like adding delay to the traffic, dropping a percentage of packets, etc. We use these two techniques in our research.

Other useful tools include (1) computer network administration utility, *ping,* (2) arbitrary TCP/UDP connection/listens, *nc*, etc.

4.5 Performance evaluation

To our best knowledge, our method is the first proposal to solve the type of problem that we have identified. The only other method we can use to compare is the standard MIP. Therefore, in our evaluation, we only compare SFB-MIP with the MIP in necessary cases.

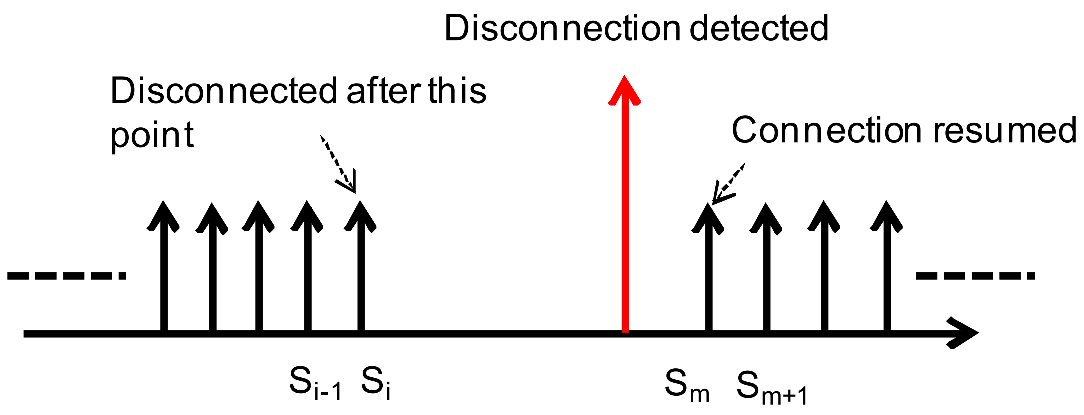


Figure 4-8 Latency measurement methodology

4.5.1 Latency measurement methodology

To measure the latency accurately, we placed the sender of the traffic generator in the CN and a receiver in the MN. The traffic generator keeps sending packets with sequence number to allow monitoring the connection and calculating the latency. The receiver receives and keeps records of the arrival time and sequence number of arrival packets. When the Ethernet cable is unplugged, packets start to be dropped. When the connection is resumed, packets go through again and the receiver records them. The discontinuity of the sequence numbers in the received packets indicates the handover, see figure 4-8. The handover latency can be obtained by simply calculating the difference between the arrival times of the last packet before the handover and the first packet after the handover, as indicated by the formula below.

$$L_h = T_{sm} - T_{si} \qquad (4\text{-}3)$$

Where L_h is the handover latency, T_{sm} is the arrival time of the first packet after handover, T_{si} is the arrival time of the last packet before handover.

Meanwhile, in MN, the time when the disconnection is detected is also recorded. This can be done in the device driver in the kernel space, or in the user space when this event is passed up from the kernel space. Because the message needs to be passed from kernel space to the user space, these two moments would be slightly different. For simplicity reason, we include this minor latency caused by message processing in the detection latency and adopt the second method, i.e., the starting of registration is the moment when the MIP receives the detection message. The detection latency can be calculated by subtracting the arrival time of the last packet before handover from the time when disconnection is detected, as shown in the formula below.

$$L_d = T_d - T_{si} \tag{4-4}$$

Where L_d is the detection latency, T_d is the time when disconnection is detected, T_{si} is the arrival time of the last packet before handover.

The time interval between the moments when disconnection is detected and the connection is resumed is defined as registration latency.

$$L_r = T_{sm} - T_d \tag{4-5}$$

Where L_r is the registration latency, T_d is the time when disconnection is detected, T_{sm} is the moment when the connection is resumed.

4.5.2 Detection latency, registration latency and handover lantency in SFB-MIP

Round of experiment	1	2	3	4	5	6	7	8	9	10	avg
Handover latency(ms)	216	176	242	171	203	208	232	232	171	174	202.5
Detection latency(ms)	198	156	231	156	184	185	212	202	138	149	181.1
Registration latency(ms)	18	20	11	15	19	23	20	30	33	25	21.4

Table 4-4 Handover latency, detection latency and handover latency of SFB-MIP
FA-HA latency=400ms, with MIH support

To evaluate the performance of MIP and SFB-MIP, we conduct experiments to measure the latency with and without MIH support.

When an unexpected cable unplug event occurs, first, the mobile terminal needs to detect the disconnection. After that, it starts the handover procedure to finish the re-registration. After the re-registration is finished, the connection resumes. Therefore, the overall handover latency includes detection latency and registration latency. This can be clearly seen in figure 4-1.

In the experiments, we configure our test-bed to set the FA-HA latency to different values. We also setup MIH support. First, we measure the detection latency and handover latency of SFB-MIP. Hundreds of rounds of experiments are conducted. The experiments under conditions of different FA-HA latencies show similar results. In figure 4-9 and table 4-4, we show a set of typical results from a series of experiments, under the condition of 400ms FA-HA latency.

As seen, the handover latency is independent of the FA-HA latency but strongly depends on the detection latency. The registration latency, which is on top of the detection latency, is only around 20ms and the average detection latency is around 180ms. The roughly 200ms of overall handover latency implies that the disconnection is not perceivable to a

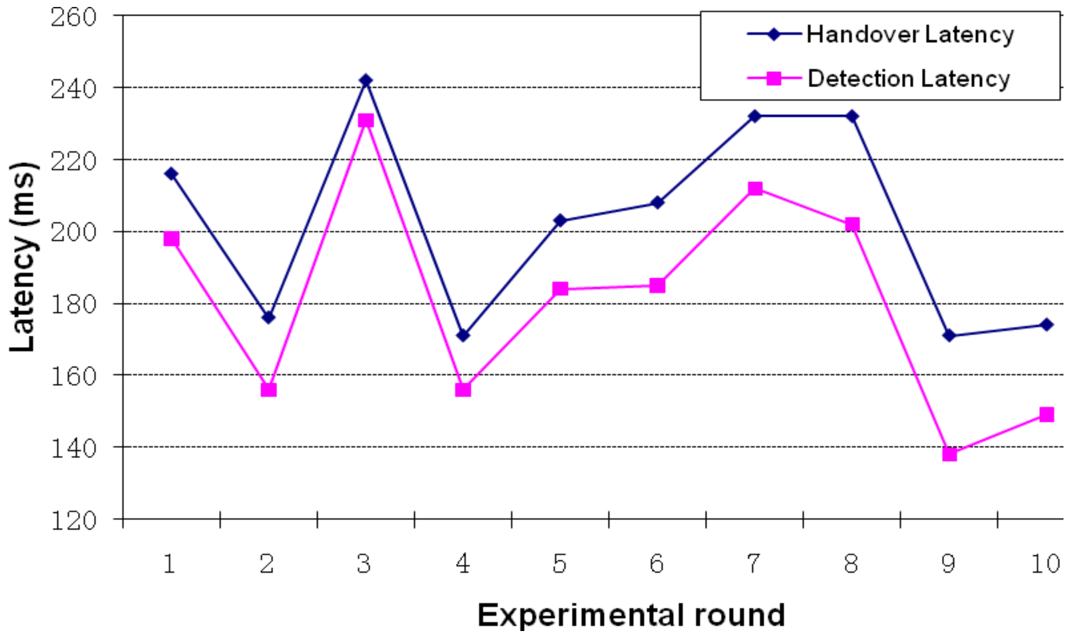


Figure 4-9 Detection and Handover latency of SFB-MIP, FA-HA latency=400ms, with MIH support

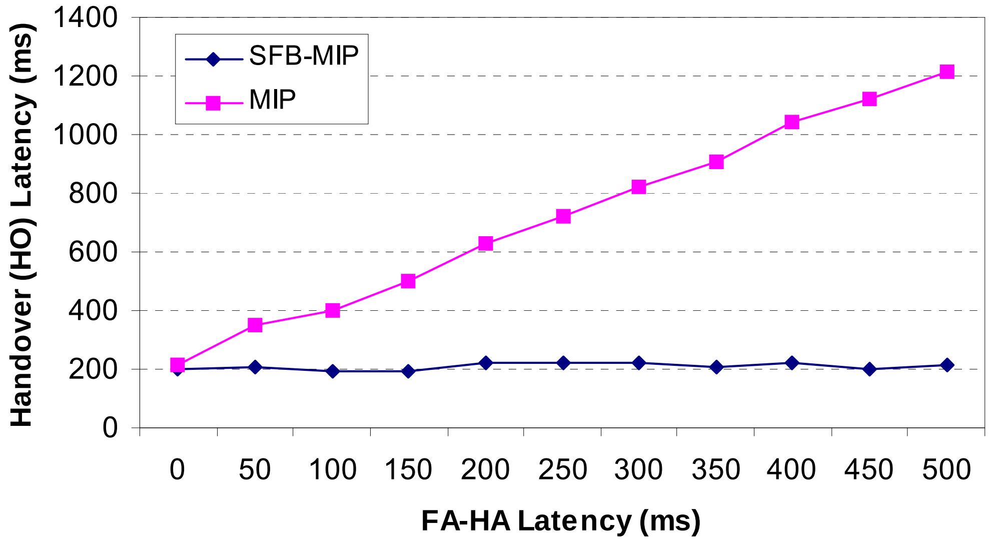


Figure 4-10 Latency comparison with MIH support

VoIP user when he unplugs the cable. This will significantly improve user experience. We also notice that there is latency variation in different experiments. This is because the operating system schedules different tasks at discrete time intervals. The disconnection cannot be detected immediately (the variation appears great only due to the vertical axis zoom-in). Due to the fact that we are using the general Linux kernel which is not optimized for network interface card detection, to further reduce the handover latency, it is desirable to design a customized kernel which gives the higher priority to process the events of plugging/unplugging network interface cards[40].

4.5.3 Handover latency under different FA-HA latency conditions

In this section, we compare the SFB-MIP to MIP under conditions of different FA-HA latencies. The FA-HA latencies are set as {0,50,100,150, …, 500} *ms* in the Internet simulator. Notice that this is one-way latency. The round-trip latency would be doubled.

Figure 4-10 shows the experimental results. The data from 10 rounds of experiments are averaged. The blue line shows the handover latency of SFB-MIP. It slightly vibrates against 200ms. It proves that the handover latency is not affected by FA-HA latency. The pink line represents the handover latency for the MIP. It increases linearly with the increasing of latency between the FA and HA. We re-write formula 4-2 as follows.

$$L_t = L_d + L_{fa\text{-}ha} + L_{proc} \qquad (4\text{-}6)$$

Where L_t is the overall latency, L_d is the detection latency, $L_{fa\text{-}ha}$ is the FA-HA latency, L_{proc} is the latency that caused by other processes.

We can use the formula 4-6 to explain our experimental results. For instance, when the FA-HA latency is 200ms, for MIP, the overall latency is 620ms which can be decomposed as 220ms + 400ms, where 220ms is $L_d + L_{proc}$ and 400ms is the $L_{fa\text{-}ha}$ (200ms * 2). For SFB-MIP, because the backup interface does not need to be authenticated by the HA, and therefore, the term of $L_{fa\text{-}ha}$ is 0 and only the term $L_d + L_{proc}$ is left, which is around 200ms.

The experimental results are as expected and validate our algorithm.

4.5.4 Packet losses comparison

Data loss of traffic stream during handover is directly dependent on the handover latency and the data rate. Table 4-5 compares the data loss and packet loss of SFB-MIP and MIP

	Observed Data loss	Observed Packet loss
SFB-MIP	256~448 bytes	4~7 packets
MIP	576~960 bytes	9~15 packets

Table 4-5 Data-Loss for voice data, stream rate=16Kbps, packet Size = 64 bytes, FA-HA latency=100 ms

for voice data stream. As shown in the table, the number of lost data and packets using SFB-MIP are only half of that using MIP, when the FA-HA latency is 100ms, which typically is a small value. We can reasonably expect that the performance gain would be higher in the cases when the FA-HA latency is higher. This results validate that SFB-MIP achieves much better performance compared to MIP with a common FA-HA delay..

4.5.5 Latency affected by MIH

Finally, we show that MIH support is important for SFB-MIP to work. As introduced in chapter 2, two basic methods are defined in [26] for move detection in MIP. The first method is to setup a timer based on the agent-advertised lifetime. Failure to receive a new advertisement when the timer expires implies a disconnection. The second method uses new network prefixes. When a MN receives an advertisement with a new network prefix, it concludes that it has moved. However, when multiple interfaces of a MN connect to a single FA, this method is unsuitable for move detection because this MN is still in the same subnet and will receive control message from the same FA. The first method is still applicable. In our experiments, we set the lifetime in agent advertisement to 90 seconds and the interval between advertisements to 30 seconds. Those parameters are typical selections after considering the channel capacity consumed by the control message.

Table 4-6 shows a typical experimental result. The average move detection time is 68 seconds which is fairly large. Such a big latency is impractical in many scenarios. Compared to the typically handover process latency of 20*ms*, this huge move detection latency would dominate the total latency and eliminate the benefit achieved by the SFB-MIP. To have a quicker detection, we can set the interval time between the advertisements to a much smaller number. However, the advertisement message will consume more channel capacity, which is not desirable. Moreover, even if we set the advertisement interval time to 1 second, the move detection latency is still unacceptable. Therefore, to allow our algorithm to achieve meaningful handover improvement, MIH support or similar mechanism is needed.

HO latency (sec)	1	2	3	4	avg
Without MIH	83	71	46	72	68

Table 4-6 Handover (HO) latency without MIH support

4.6 Issues discussion

4.6.1 Simultaneous binding in MIP and SFB-MIP

Simultaneous binding (or registration) as defined in client MIP [26] has quite a different meaning from that in SFB-MIP. In MIP, simultaneous binding means that a MN registers at a HA using two or more FAs. In SFB-MIP, it means that multiple interfaces simultaneously bind to the same FA. We also call it a backup binding because one or more interfaces are ready for use in case of an unexpected disconnection. In cases where handover occurs unexpectedly, our method yields a great improvement in handover latency. The benefits might be reduced when the handover can be predicted in advance, giving the secondary interface enough time to perform full registration. Even then, one can still take advantage of the bi-casting feature.

4.6.2 IP addresses of different network interface cards

The mobile terminal carries multiple network interface cards. How to assign IP address to those cards is an issue. Generally, different active cards represent different connection points and therefore, distinct IP address should be assigned to each card. Otherwise, there would be IP address confliction. However, in MIPv4, from the perspective of an end-to-end connection, the mobile terminal is simply one end-point of a connection. If the IP address is changed after the handover, the connection before the handover will inevitably be disconnected after the handover. The current session will be interrupted. This is not desirable. Therefore, we need to assign the same home IP address to all cards.

4.6.3 Necessity of simultaneous FA binding

SFB-MIP achieves a much faster handover when an unexpected L2 handover occurs. In the example in section 4.3 above, when the user unplugs the Ethernet cable (IF1), after detecting this event with the help from MIH support, the MN sends a message to FA to activate the WiFi (IF2) without the latency of *t2* and *t5~t14* in figure 4-1, but sending a simultaneous FA binding extension is still needed, as shown in figure 4-3.

4.6.4 The number of re-registration messages

The re-registration with FA and HA can be combined in one message and therefore, the number of re-registration requests to HA will not necessarily increase because they can be sent over the bound interfaces in turn, combining with the re-registration. The identification field defined in [26] suffices to distinguish the reply for each registration request.

4.6.5 Power consumption

Power consumption is a important factor needs to be taken in to account in desinging ptotocol for mobile devices. In SFB-MIP, althogh multiple network interface cards are ready for handover, binding multiple interfaces to an FA doe not imply that all registered interfaces need to stay powered all the time. The bound but inactive interfaces can go into a power saving mode, and they just need to wake up periodically to refresh the binding timer at the FA. Typically, the refresh duty is in the order of second, therefore, the extra power consumption caused by the backup interfaces would be negligible.

4.6.6 Experiment in other networks

Our performance evaluation is based on WiFi and Ethernet. It can be easily to extend our evaluation to include EvDO or WiMax. For EvDO, to set up a connection, a procedure of dial-up to access server must be finished, which typically takes seconds. This means that the EvDO L2 handover latency is significantly longer than that of WiFi. We have tested the connection set-up time using the EvDO commercial service provided by Verizon Inc. and verified that it may take up to 10 seconds. Therefore, we can reasonably to expect SFB-MIP to provide greater performance improvement in the case when EvDO interfaces are involved.

Chapter 5

TCP and New Methods for Better Performance in Multi-hop Wireless Networks

The TCP (Transport Control Protocol) [41-43] is the most widely deployed protocol for reliable data transfer over Internet and it is proved to be robust in dynamically changing network conditions. To reuse the huge amount of research and industrial experience, and also to provide compliance to the existing network, it is desirable to extend TCP to multi-hop wireless networks. However, TCP, which is originally designed for wired networks, where buffer-overflow is the key reason for packet loss, suffers significant performance deterioration in multi-hop wireless network where the key reason for packet loss is different. The wireless network factors that affect TCP performance include medium access contention complicated by hidden/exposed terminal problems, much higher transmission errors caused by random channel fluctuations and errors, etc.[44]. Many researchers have made efforts to improve TCP performance under this completely different situation and have achieved significant progress. In this chapter, our analysis and simulations show that for high speed connections the much smaller TCP ACK packets consume comparable channel resource as the much longer data packets, which implies that a key step in improving the TCP performance is to maximally reduce the number of control packets. Based on this, we propose a new delay ACK algorithm which can enhance TCP performance significantly, up to 203% improvement over regular TCP in long hop wireless networks and 35% throughput gain in cross topology. We also extensively evaluate the performance over long-hop networks which are omitted in the prior work.

The rest of this chapter is organized as follows. Section 5.1 briefly reviews the TCP and its behavior in wireless network. Section 5.2 introduces existing effort to improve TCP performance. Section 5.3 gives a detail analysis of the optimal congestion window and provides a better result than that in previous works. Section 5.4 analyses the negative brought by TCP ACK traffic. Based on the analyses results, we propose two methods to improve TCP performance. Section 5.5 presents our basic idea and evaluation of the first method which is used in a grid wireless network. Section 5.6 presents a more generic approach. Evaluations are also given. Section 5.7 discusses future work.

5.1 TCP in wireless network

5.1.1 Brief review of TCP

The TCP is a connection based transport protocol and has been widely used for reliable data transfer. In TCP protocol, the receiver needs to notify the sender for every packet it has received. Table 5-1 shows the TCP ACK generation criteria. Based on the acknowledge packets sent by the receiver, the sender can detect any lost packets, which are then retransmitted until all packets are received by the receiver. TCP also provides congestion control mechanism to adjust the traffic transmission rate based on the connection condition. The control parameters like sequence number, window size, are carried in the TCP packet header, whose format is shown in Table 5-2. Typically, the size of the TCP header is 40 bytes. However, the TCP also provides an option field which can contain more control messages in case it is necessary.

Event	TCP Receiver Action
Arrival of in-order segment. All segments up to the expected sequence number have been acknowledge	1. Immediately send ACK when segment received, or 2. Delayed ACK. The receiver waits up to 500ms for the next coming segment to send an ACK. It would also send an ACK if it fails to receives the second segments during that interval.
Arrival of in-order segment. Some other in-order segments are waiting for ACK	Immediately send a cumulative ACK
Arrival of out-of-order segment. Gap detected	Immediately send duplicate ACK to indicate the expected sequence number
Arrival of segment that fills in the gap partially or completely	Immediately send ACK

Table 5-1 TCP ACK Generation Criteria

Source Port	Destination Port
Sequence Number	
Acknowledge Number	
Header size and control bits	Window size
Checksum	Urgent Point
Option Field	

Table 5-2 TCP Header Format

Researchers have proposed numerous modifications to improve TCP performance under different circumstances, e.g., TCP NewReno [45, 46], TCP Selective Acknowledge

Options (SACK) [46], TCP Vegas[47], etc. TCP NewReno is the most popular one and we will use it in our evaluations. When we mention (regular) TCP in subsequent sections, we mean TCP NewReno.

5.1.2 TCP behavior in multi-hop wireless network

TCP is originally designed for wired network. At the beginning, the sender sends only one packet, and then double the number of packets in flight after it receives every acknowledgement, until the packets in flight is higher than the connection capacity and some packets start to be dropped. In wired network, the main reason for packet lost is mainly due to the buffer overflow in the bottleneck routers. However, in multi-hop wireless network, the main reason for packet drop is the link layer contention[44]. As we have introduced in Chapter 3, the sender would only try retransmission 7 times by default if it does not receive acknowledge from the receiver. In the case of using CTS/RTS, the sender would drop the head-of-line packet after it tries the CTS seven times. [44] shows simulation results to verify that the chance of buffer overflow should be very limited and the main reason for packet drop is due to the link-layer contention.

5.2 Existing work for TCP improvement in wireless multi-hop network

TCP performance over wireless network has been an active research topic and has been investigated from different perspectives. Most early works focused on mixed system where the last hop is the wireless connection. Subsequently, researchers investigated TCP performance in ad-hoc wireless networks and tried to allow the sender to identify the reasons for packet loss, and then adopt different strategies to tackle the problem of packet drop. A typical example is the ELFN (Explicit Link Failure Notification) like method proposed by Holland and Vaidya [48]. This method attempts to distinguish the packet loss due to wireless routing error from congestion loss and avoids the unnecessary shrinking of congestion window. However, in multi-hop wireless network, large congestion window is the key reason for channel under-utilization [44]. Therefore, such a strategy does not offer much benefit.

Fu et al. [44] show that the main factor affecting the TCP performance in multi-hop wireless networks is the link-level contention, rather than buffer overflow, and there exists an optimal *cwnd* size. They introduce Link Random Early Drop (LRED) scheme which, by randomly dropping a packet, indirectly informs the sender to slow down its traffic injection. However, the loss is discovered based on duplicate ACKs, which are generated when subsequent packets reach the receiver. It can take a long time until the sender discovers the packet loss in long hop scenarios. This is a reactive method. Additionally, they also propose adaptive pacing which intends to evenly distribute the traffic over the nodes on the connection path to reduce contention. Some researchers proposed proactive methods to limit the injected traffic load by setting a low *cwnd* limit, intending to match the optimal *cwnd* size. K. Chen et al. [49] use the number of hops to determine the optimal *cwnd* window.

Another way to reduce contention is to reduce the number of control packets. In TCP, two types of packets are transmitted: the actual data, fulfilling the transmission goal, and ACK packets, providing control services. Reducing the number of transmitted packets results in less contention and shorter contention periods for channel access. E. Altman and T. Jimenez[50] studied TCP performance based on different Delay ACK. Yuki et al. [51] proposed a technique to combine TCP DATA and ACK packets into a single packet. Oliveira and Braun [52] proposed TCP-DAA (dynamic adaptive acknowledgement) that tries to combine the idea of restricting *cwnd* to its optimal/suboptimal value while reducing the number of ACK packets. They fix the *cwnd* size at 4 packets, therefore the receiver can only set the delay window limit up to 4. However, having a fixed *cwnd* limit number may restrict applicability of their method. Moreover, it makes it impossible to further reduce the number of ACK packets.

Y. Chen et al. [53] removed the *cwnd* restriction, and proposed a method called TCP-DCA which selects different delay window limit based on the number of hops. They use a low-pass filter to process the inter-arrival time of consecutively arriving data packets and then setup the receiver's ACK-delay timer. In our work, we change the strategy to generate ACK packets. Instead of limiting the delay window to a fixed small number, we attempt to minimize the number of ACK packets, which is inspired by the fact that an ACK packet consumes channel bandwidth comparable to a data packet. Our analysis is provided in the subsequent sections.

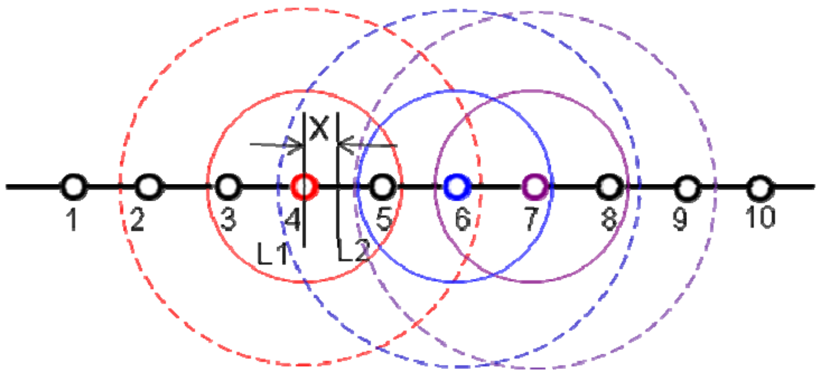


Figure5-1 Typical chain topology

The solid circle is the valid transmission range. The dotted circle denotes the interference range.

5.3 Optimal congestion window analysis

In this section, we first review existing results of optimal congestion window in multi-hop wireless network, and then conduct more accurate analysis to correct the conclusion from other researchers and give a better result.

In IEEE 802.11 wireless network, the interference range of a transmitting node is bigger than its transmission range. This introduces the hidden node problem. Therefore, in a multi-hop wireless network as shown in figure 5-1, although the node 4 and node 6 cannot hear each other, if they both send packets to node 5 simultaneously, the signals will collide and the transmission will fail. To alleviate this problem, IEEE 802.11 introduces the RTS/CTS mechanism. In addition, every node backs off for an exponentially increasing random interval after overhearing other nodes' transmission.

Figure 5-1 shows a typical chain topology in which any two adjacent nodes reside 200 meters apart. The transmission range is about 250m and the interference range is about 550m. Based on this topology, previous work [44] claimed that TCP achieves the optimal throughput when *cwnd* equals *n*/4, where *n* is the number of hops. The reasoning is given as follows.

1) Nodes which are 3-hop away can send packets concurrently, e.g., when node 4 sends packets to node 3, node 7 can sends packets to node 8 but cannot send packets to node 6, which is within 3-hop from node 4. Because the signals sent by node 4 would interfere with the signal sent to node 6 from node7.
2) On the MAC layer, every packet follows the procedure of data_sending/ACK_receiving (see figure 5.2), which is similar to the stop-and-wait protocol on the transport layer. For example, when node 3 sends packets to node 4, node 4 also sends packets back to node3. Combining with item (1) described above, when node 3 communicates with node 4, node 7 can only communicate concurrently with node 8, instead of node6. Therefore, only nodes which are 4-hop away can transmit concurrently. Based on this, other researchers concluded that the maximum number of transmitting packet in the medium cannot exceed *n/4*. Because the total number of packets in flight cannot bigger than *n/4*, the congestion window of TCP should not be bigger than *n/4*, to achieve the optimal throughput.

The significance of this reasoning is that it points out that the TCP congestion window in multi-hop wireless network actually is restricted by the space reusing. However, this reasoning neglects the capture effect which affects the transmission range significantly. This causes the wrong conclusion of the optimal TCP congestion window. We provide a more accurate analysis below.

Capture effect states that when a receiver receives signals from two different senders, the stronger signal could completely suppress the weaker one and therefore can be correctly decoded by the receiver, if the difference between these two signals exceeds a threshold (typically 10 *dB*).

There are multiple available models to demonstrate the radio transmission characteristics, e.g., two-ray ground reflection model, free space model[54]. The former one gives more accurate prediction at a long distance than the later one and therefore, we use it to conduct the analysis.

In the two-ray ground reflection model, the received power, P_r, at distance d is expressed by

$$P_r(d) = \frac{P_t G_t G_r h_t^2 h_r^2}{d^4 L} \tag{5-1}$$

where P_t is transmit power, G_t and G_r are transmit and receive antenna's gain, respectively, h_t and h_r are the heights of the transmit and receive antenna, respectively [54]. L is a constant. Assuming P_1 and P_2 are the signal energies received from two nodes, the following formula can be used to determine whether the capture effect occurs.

$$10 * (\lg(P_1) - \lg(P_2)) > 10 \qquad (5\text{-}2)$$

We can use this formula to calculate the location where the difference of signal energy level is 10 *dB*.

In figure 5-1, we assume node 3 and node 6 are sending packets. *X* is the distance from node 4. *200+x* is the distance from node 3 while *400-x* is the distance from node 6. We calculate the point where the difference of signal energy level is 10 *dB* as follows.

$$((400 - x) / (200 + x))^4 = 10$$

$$x = 15.82 \text{ (m)}$$

The above calculation shows that in the case when node 6 is sending packets, node 4 can correctly decodes the packets sent from nodes 3, although node 4 is in the interference range of node 6. In other words, the transmission between node 6 and 7 would not affect the transmission between node 3 and node 4. This means that nodes that are 3-hop away can transmit simultaneously and reuse the space. Therefore, the optimal *cwnd* size should be *n*/3 at least, instead of *n*/4 as stated in [44, 52].

Besides, there is a slight possibility that node 3 can transmit to node 4 while node 6 is transmitting to node 5. For example, nodes 4 and 5 send RTS at the same time to nodes 3 and 6, respectively. Because of the capture effect, the CTS from node 4 to 3 and from node 6 to 5, respectively, can be correctly decoded and the transmission can succeed. But in most cases, the transmission between nodes 3 and 4 would interfere with that between nodes 5 and 6 (for example, when node 4 is transmitting, node 5 cannot receive), which means that the optimal *cwnd* must be less than *n*/2.

Combined with the analysis above, we conclude that the optimal *cwnd* should be between *n*/3 and *n*/2, slightly higher than *n*/3. Therefore, in a 7-hop chain wireless network, the optimal cwnd should be in the range of 2.1 and 3.5, and in a 10-hop network, the optimal *cwnd* should be in the range of 3.3 and 5. These results can be validated by the evaluation results in [44, 52], where the optimal *cwnd* is 3 and 3~4, respectively, instead of 2 and 2.5 which calculated by *n/4*.

5.4 Impact of ACK packets on channel utilization

As introduced in chapter 3, a packet transmitted in IEEE 802.11 wireless network contains payload and overhead. The overhead includes MAC overhead and PHY overhead, where MAC overhead includes RTS/CTS/ACK/DIFS/SIFS while PHY overhead includes preamble, PHY header and transmission delay. Different from the scenario in wired network, these overheads have significant impact on channel utilization. In this subsection, we discuss the relationship of channel utilization and packet size.

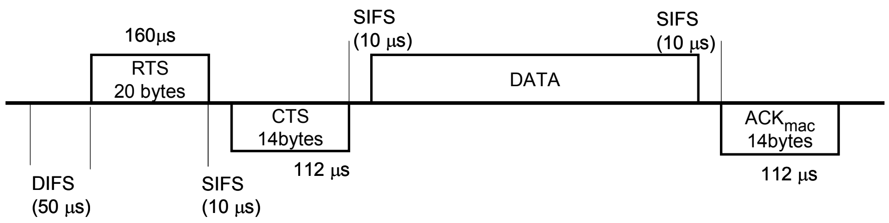


Figure 5-2 Timing diagram for packet transmission with RTS/CTS

A typical TCP ACK packet is 40 bytes long which include TCP header (20 Bytes, see Table 5-2), and IP header (20 Bytes). For simplicity, we ignore the minimum size requirement of Ethernet packet. We also note that the TCP ACK is different from the MAC ACK. Figure 5-2 shows the transmission timing diagram that include RTS/CTS.

5.4.1 Intuitive observation

For every packet, the length of overhead in MAC and PHY are constant while the length of payload could be different. We can image one extreme case when the transmission rate is infinite fast. In such a case, the transmission time of payload is zero, and therefore, the time cost to transmit a short packet and a long packet would be the same. See figure 5-3 for the timing diagram. In other words, the short packet and the long packet consume the same channel capacity. Although we cannot achieve infinite fast transmission, this imagination gives us a hint that the channel capacity consumed by different packets do not proportional to their length. To have a more solid idea, we provide an intuitive calculation.

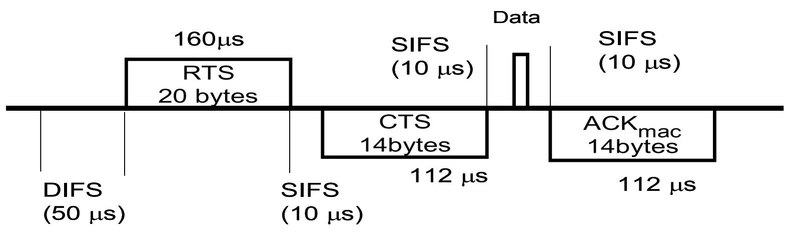


Figure 5-3 Timing diagram for packet transmission with RTS/CTS, infinite transmission rate

For simplicity, we ignore PHY overheads and some MAC overheads, e.g., PHY header, DIFS, etc., which would support our conclusion even stronger, see figure 5-2 for the overhead considered. Assuming the basic rate is 1Mbps (used for RTS/CTS/ACK$_{mac}$) and the data transmission rate is 11 Mbps, the overhead transmission time is 3SIFS+RTS+CTS+ACK$_{mac}$ = 414μs. After including this overhead, the transmission times for 40-byte ACK$_{tcp}$ packets and 1040-byte data packets are 443μs and 1170μs, respectively. They are on the same order of magnitude. Moreover, because of the exponential backoff mechanism, the ACK$_{tcp}$ packets have a greater impact than proportional to their transmission time. When the transmission rate becomes higher, the transmission time for data shrinks and therefore the overhead brought by RTS/CTS is greater.

5.4.2 Mathematical analysis

To clarify this conclusion, we use the two-dimensional Markov chain model[23] to provide more accurate analysis. This model allows us to analyze the affect caused by the backoff time that is ignored in the previous discussion.

We first extend the formula 3-2 to calculate the normalized throughput of packets using different transmission rate.

$$S = \frac{P_s P_{tr} L / R}{(1 - P_{tr})\sigma + P_{tr} P_s T_s + P_{tr}(1 - P_s) T_c} \tag{5-3}$$

Where *L* is the length of packet, in unit of bit, *R* represents the transmission rate, in unit of *Mbps*, the unit of T_s, T_c are *usec*.

We can view the denominator in the formula as the slot time, defined as T_{slot}, and view the numerator as the effective transmission time, defined as T_{eff}. Therefore, we can rewrite the formula as follows.

$$S = \frac{T_{eff}}{T_{slot}} \tag{5-4}$$

Based on formula 5-4, we can calculate the number of transmitted packet in a unit time using the formula below.

$$C = \frac{T_{eff}}{T_{slot}} * \frac{1}{T_{packet}} \tag{5-5}$$

Where T_{packet} is the transmission time for one packet, which can be calculated by

$$T_{packet} = \frac{Number\_of\_bits\_in\_a\_packet}{Transmission\_rate} \tag{5-6}$$

The unit of transmission rate is *Mbps* and the unit of T_{packet} is *usec*.

Based on formula 5-5 and 5-6, we deduce the formula below to calculate the ratio of transmitted packets of different sizes in a given time.

$$\rho = \frac{C1}{C2} = \frac{T1_{eff}}{T1_{slot} * T1_{packet}} * \frac{T2_{slot} * T2_{packet}}{T2_{eff}} \tag{5-7}$$

Using this formula, we compute the ratio of transmitted regular data packet and ACK packet. The results are shown in the table 5-3.

	Transmission rate = 1M bps	Transmission rate = 11M bps
Ratio	5.22	1.58

Table 5-3 Transmission ratio of data packet and ACK packet

From table 5-3, we can see that the transmission ratio of data packet and ACK packet is only 1.58 when the transmission rate is 11M bps, which implies that in a given time period which can transmit a 1000-byte packet, only 1.58 ACK packets can be transmitted. In other word, the channel capacity consumed by 40-byte packet and 1000-byte packet is comparable.

Num of hops	5	6	7	8	9	10	11	12
Num of pkts (1040 byte)	4768	4212	3496	3529	2282	2359	1967	1947
Num of pkts (40 byte)	5732	5184	4470	4255	2311	2698	2762	2969
Ratio	1.20	1.23	1.28	1.21	1.01	1.14	1.40	1.53

Table 5-4 Achieved number of transmitted packets vs. packet size, transmission rate 11Mbps

5.4.3 Validation by Simulation

In order to validate our analysis result, we carried out simulation to measure the number of actual transmitted packets in the chain topology, using TCP [42]. We compare two cases whose packet sizes are 1040 bytes and 40 bytes (the size of an ACK_{tcp}), respectively. Table 5-4 shows the simulation result. We can see that the ratio is in the range of 1.01 ~ 1.53, which matches our mathematical analysis very well. Based on this, it is clear that although

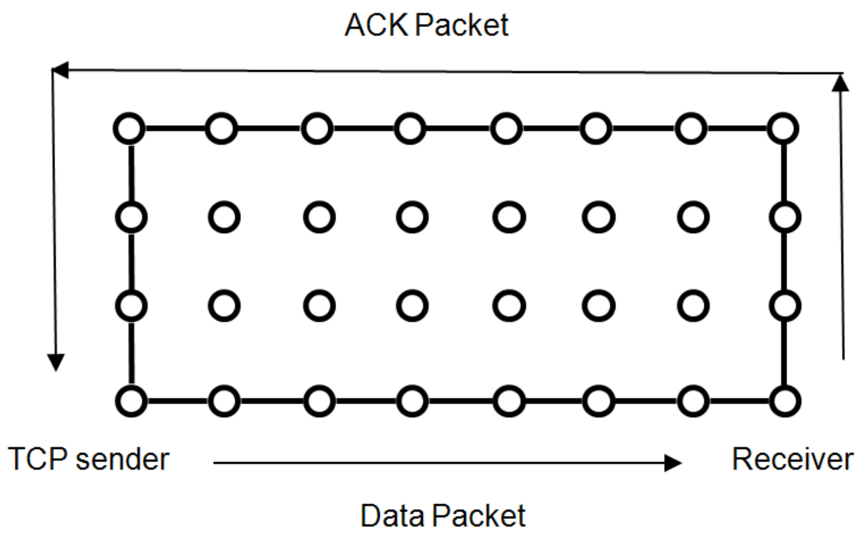


Figure5-4 Grid wireless network and TCP flow

the 40-byte packet is only 1/26 of the size of a 1040-byte one, the number of transmitted packets only increases by 53% in the given time period, in the best case. It verifies that channel capacity consumed by TCP ACK packet is compatible to that of much longer data packets.

This fact points to a promising direction to enhance the TCP performance over multi-hop network by lowering the number of transmitted ACKs even more. Motivated by this finding, we propose novel algorithms.

5.5 Improving TCP throughput in grid-topology wireless network

5.5.1 Basic idea

The fact that the TCP-ACK packets consume compatible channel capacity as the data packets suggests that the TCP performance may be improved if we can alleviate the competition between data and ACK packets, especially, if we can remove the ACK traffic from the path of data packets. Different network deployment would allow us to use this strategy in different ways. In the scenario of a big static grid wireless network, we propose to use different route for these two types of traffic, which can be achieved by static routing. Figure 5-4 shows a typical deployment. In such a static network, by assigning deferent routes to data packet and ACK packet flow, the interference between data and ACK will be alleviated and therefore, the performance can be improved.

5.5.2 Performance evaluation

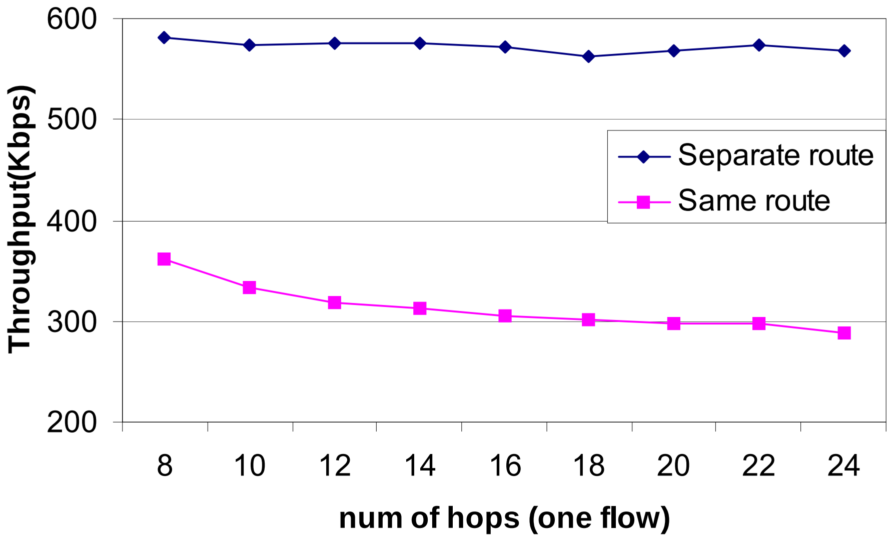


Figure5-5 Throughput comparison between the cases when data and ACK packets take separate route vs. the same

We use the ns-2.30 simulator [55] in our evaluations. The distance between adjacent nodes is set to 200 meters. Most of parameters use the NS2 default value, e.g., packet size is 1000 bytes. We have chosen FTP as the traffic generator, simulation time is 100 seconds, and channel bandwidth is 11 Mbps. We use static protocol to configure separated routes for data packets and ACK packet, as indicated in figure 5-4.

Figure 5-5 shows the simulation results. It can be seen that our simple strategy achieves 60% ~ 100% throughput improvement in cases of different hops in the x-axis.

The advantage of this strategy is that it is very simple and the existing standard TCP does not need any modification. However, the restriction of its applicability is also considerable. For example, the performance may be deteriorated when some nodes in the TCP ACK path suffer from heavy load caused by other traffics and therefore, some globally scheduling may be needed. In the next section, we propose a more generic method.

5.6 A more generic method for better TCP performance

As we have discussed above, two main factors affect the TCP performance in multi-hop wireless network. One is the injected traffic load to the network and another is the control traffic. Based on our conclusion of the optimal *cwnd*, we have a straightforward method to improve the throughput by fixing the *cwnd* to the optimal value. However, setting a fixed *cwnd* has its restriction. First, in a chain wireless topology, the total number of nodes must be known in advanced. Second, although the optimal congestion window can provide theoretical guidance, it is calculated based on a simple case, i.e., chain wireless network. In a complex wireless network like a grid network, the interference between nodes is much more complicated and it is very difficult to figure out the optimal congestion window. To preset a optimal congestion window would be impossible. Therefore, we focus on improving TCP performance by minimizing the number of ACK packets, without setting a pre-fixed congestion window.

In previous work, researchers tried to adjust the ACK-delay timeout period and delay window size dynamically based on inter-arrival time, or to identify an optimal delay window limit. TCP-DAA [52] allows the delay window in the range of 2~4, which means every fourth in-order packet, at most, generates a cumulative ACK packet. TCP-DCA [53] set the maximum delay window limit at 3 or 5 when the total number of hops is greater than 3. All of these algorithms have small delay window limit. When a delay window limit is small, it will be generally reached sooner than the ACK delay timeout, which means that an ACK is mostly generated before the timer expires. To further reduce the number of ACK packets, our strategy is to make the ACK-delay timeout to be the main generator of ACKs, provided that the channel condition is favorable and allows error-free transmission. Therefore, we set up a large delay window (in this dissertation it is set at 25, based on our extensive simulation results). Due to this, ACK-delay timeout becomes the dominant generator of ACKs. This is unlike the existing approaches, which use a small delay-window limit and reaching this limit is the main ACK generator. Chen et al. [53] point out that a higher delay window limit does not necessarily result in higher throughput. However, in our method, because the ACK-delay timeout generates many more ACKs

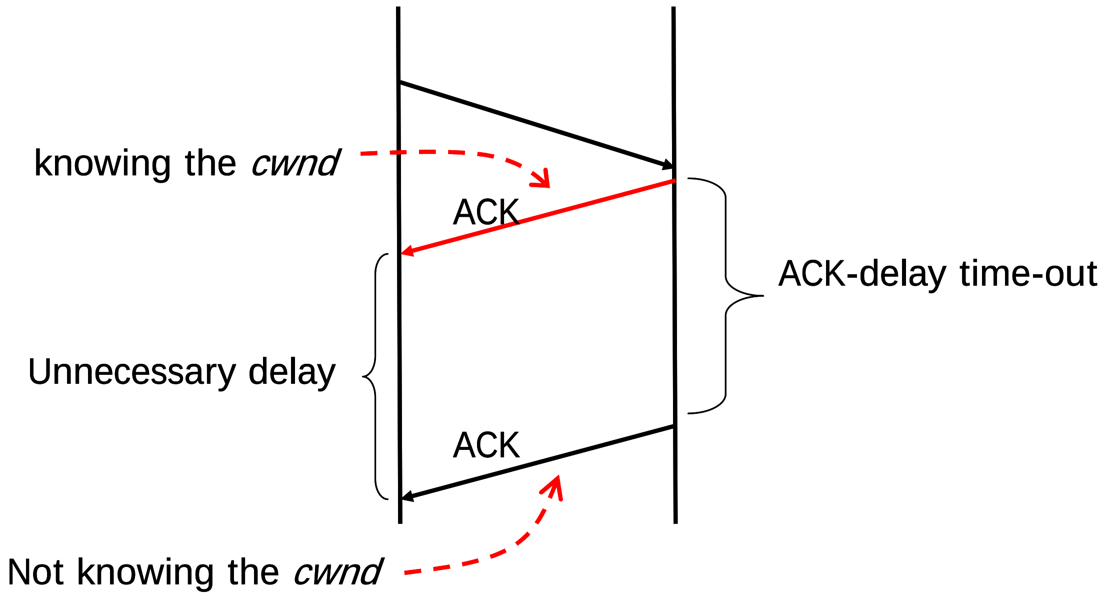


Figure 5-6 Under-utilization of the channel without knowledge of the *cwnd*

than reaching the delay window limit, we can achieve better performance with large delay window limits. In this dissertation, we set the timeout timer as 200 ms, based on extensive experimentation and some consideration of application scenarios.

When the congestion window is small (less than the delay window), the above mechanisms cannot respond quickly and introduce unnecessary delay. We use a simple example to show the under-utilization of channel without knowledge of the *cwnd* (See figure 5-6). In the case shown, when the congestion window is 1, the sender sends one packet and then waits for the ACK. Because no more packets are coming, the receiver only waits until ACK-delay timeout to send the ACK packet. This causes the under-utilization of the channel. To avoid this problem, we adopt a solution similar as in [53]. The sender puts the current value of *cwnd* into the TCP option field (Table 5-1) in packet header to inform the receiver of the current *cwnd*. When the receiver receives the data packet, it extracts the current *cwnd* from the TCP option field and compare the number of received-but-unacknowledged packets. If they are equal, the receiver knows that the sender is waiting for an ACK and sends one immediately. The difference is that we do not need to count the path length, as required in [53].

In order to use the current *cwnd* as one of the trigger conditions for ACK generating, the TCP-sender needs to include the *cwnd* information in the transmitted data. There are at least two approaches to achieve this requirement. The first one is that the sender can put this current *cwnd* into the TCP header option field, which is introduced above . The second method is to use the advertised window field in the TCP header (table 5-2), which is possible because the numerous TCP works mainly in the one-direction mode. Both methods need to change the default TCP/IP stack in the sender and receiver. However, the intermediate nodes do not need to be aware of these modifications. For a two-way TCP, based on our analysis above, the best way to improve performance is to piggyback the ACK in the data packet which removes most of the overhead brought by separate ACK packets.

For lost/out-of-order, gap-filling packets, we use the same mechanism as in the TCP[41, 45], which means that the receiver generates ACK immediately when such an event *occurs*.

Table 5-5 shows all conditions that would trigger an ACK packet.

```
If  Lost/out-of-order, gap-filling packet detected then
        Generate ACK packet
        Return
If  ACK-delay timeout detected then
        Generate ACK packet
        Return
If  number of unacknowledged in-order packets reaches the cwnd then
        Generate ACK packet
        Return
If  number of unacknowledged in-order packets reaches the delay window then
        Generate ACK packet
        Return
```

Table 5-5 Procedure to generate ACK packets in the receiver.

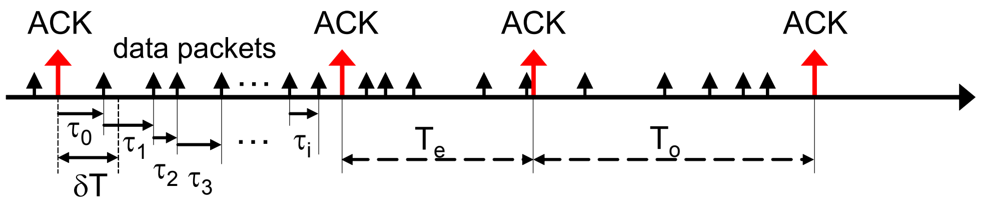


Figure5-7 ACK generating and the number of arrival packets

.

Figure 5-7 shows the relationship between the number of packets that generate one ACK and the interval between ACKs.

T_o is the ACK-delay timeout period, T_e is the time between two ACKs when a loss/out-of-order/gap-filling packet is detected, and τ_i is the time interval between two successive data packets, which varies following the channel condition.

Let T be the time interval between two ACKs generated. M is the number of data packets received within T. $\alpha(t)$ is the number of packets that arrive in a small time interval δT. Therefore, the total number of packets that arrive in T is given by

$$M = \int_0^T \alpha\ (t)dt \tag{5-8}$$

Since the time interval between two data packets received, τ_i, is a random variable which depends on the channel condition, $\alpha(t)$ is also a random variable. Thus, M is a random number decided by the channel condition. When the channel condition is good, $\alpha(t)$ will be larger and M will increase. When the channel condition deteriorates due to contention or error, the $\alpha(t)$ will be smaller and therefore, the M will get smaller. In other words, the number of packets which generates an ACK will change based on the channel condition dynamically. If the M can be much bigger than the small fixed *cwnd* in existing works, we can expect a performance gain.

Our extensive simulation results show that our new strategy improves TCP performance significantly by reducing more unnecessary ACK packets.

5.6.1 Performance evaluation

This section presents the performance results of our algorithm. For the sake of clarity, we name our new algorithm TCP-TDA because we use the ACK-delay timeout as a trigger in a different way from the existing works.

In our evaluation, we mainly focus on comparing against TCP-DCA, TCP-DA and the TCP. We do not include TCP-DAA because it is only designed for short chains; also TCP-DCA is reported to work better than TCP-DAA [52]. TCP Selective Acknowledge Options (SACK) [46] may look like a promising algorithm for multi-hop network because it tries to return more information about missing packets to the sender. However, SACK does not necessarily reduce the number of ACKs. And also based on results reported in [53], it does not significantly outperform TCP-DA and shows worse results than TCP-DAA. Therefore, we do not consider this mechanism in our evaluation. Based on a similar rationale, we also leave out TCP Vegas[47].

5.6.1.1 Simulation scenario

For fair comparison, we evaluate our algorithms using two typical network topologies as in the previous research works[44, 53]. The first one is a chain topology and the second one is a grid network. The chain topology is typically used on the sensor network which has sensors lined up in a geographical area. One possible example is a bridge health monitoring sensor network [56] where sensors are lined up along the bridge. The grid network is a more complicated scenario and can be used in sensor network monitoring a two dimensional area. We adopt the ns-2.30 simulator [55] in our evaluations. In these typical topologies, we place each node residing 200 meters apart from its neighbors. Most of parameters use the NS2 default values, e.g., packet size is 1000 bytes. We have chosen FTP as the traffic generator, AODV as the routing protocol, simulation time is 100 seconds, and channel bandwidth is 11 Mbps. The ns2 simulator is modified to make sure that capture effect has been taken into account. All data points are obtained by averaging the results of 4 or 5 simulations with different random seeds.

5.6.1.2 Throughput in chain topology

5.6.1.2.1 Throughput comparison

In this section, we evaluate our algorithm over a wide range of situations in a chain topology which is shown in figure 5-1. In this evaluation, the TCP sender resides on the first node and the receiver locates on the last one. The evaluation includes 1 to 6 TCP flows, from 1 to 29 hops.

We compare our results with the TCP, TCP-DA and TCP-DCA. Because the throughput is much higher in very short chains compared to long chains, for the sake of clarity we present simulations in two figures: one for small hop counts (1~3) and the other of larger hop counts (4~29), under conditions of different number of TCP flows.

Because experiments using other numbers of TCP flows show similar results, we only show the results for the cases of 1, 2, and 6 TCP flows.

Figure 5-8, table 5-6 show the throughput where hop counts are 1 to 3 in the case of one TCP flow. We can see that our algorithm has almost the same performance as TCP-DCA. The difference is around 1%. This is as expected because these two methods basically have the same factor, i.e., reaching the *cwnd*, that affects the TCP ACK generating in such scenarios. Both algorithms outperform TCP/TCP-DA significantly. The improvement is 26%, 24%, 41% over the TCP-DA and 55%, 53%, 72% over the TCP. Similar results hold for other number of flows. The detailed numbers are shown in table 5-7 and table 5-8.

Number of hops	1-hop	2-hop	3-hop
TCP-DA	26%	24%	41%
Regular TCP	55%	53%	72%

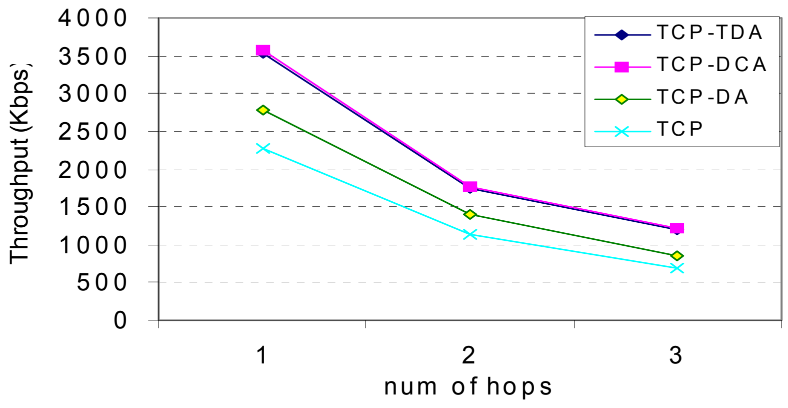
Table 5-6 Performance gain over TCP-DA/Regular TCP for one-flow, hop1 ~ hop3

Number of hops	1-hop	2-hop	3-hop
TCP-DA	26%	24%	31%
Regular TCP	55%	53%	65%

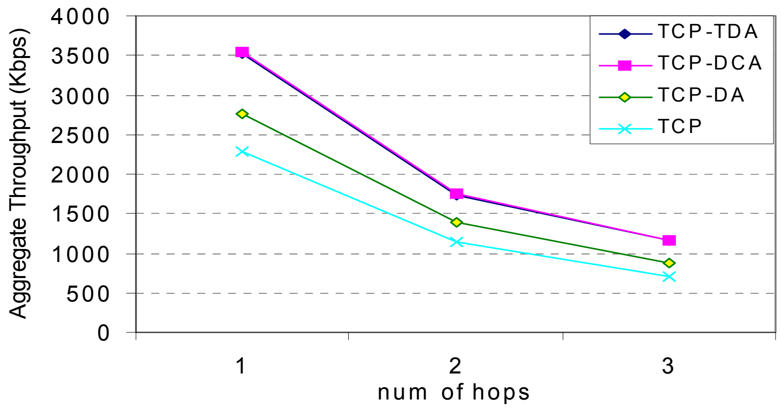
Table 5-7 Performance gain over TCP-DA/Regular TCP for two-flow, hop1 ~ hop3

Number of hops	1-hop	2-hop	3-hop
TCP-DA	24%	22%	14%
Regular TCP	48%	47%	36%

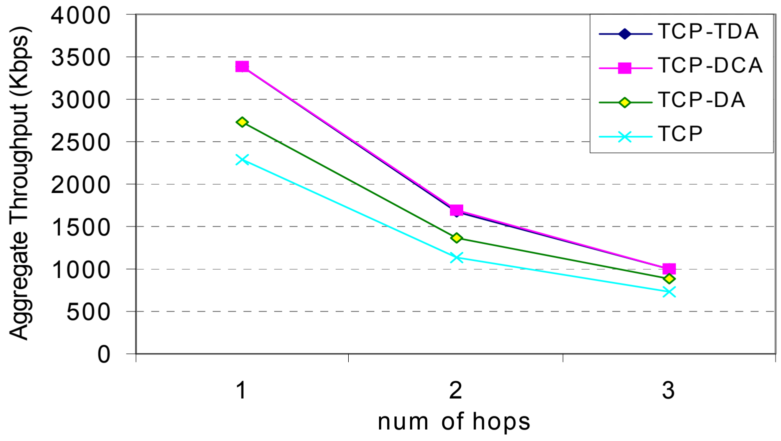
Table 5-8 Performance gain over TCP-DA/Regular TCP for six-flow, hop1 ~ hop3



One flow

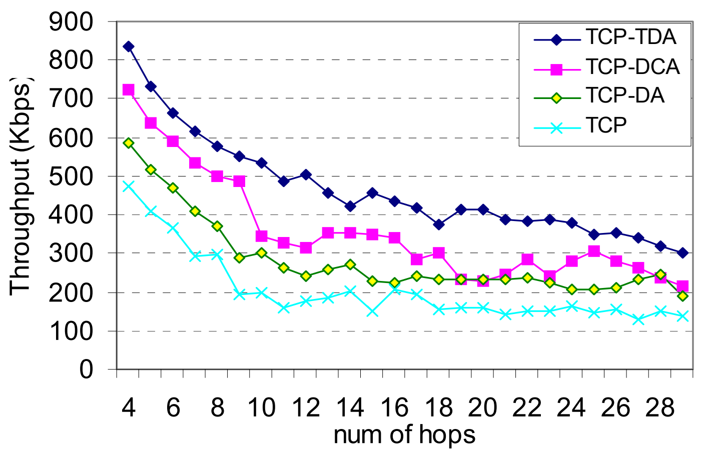


Two flows

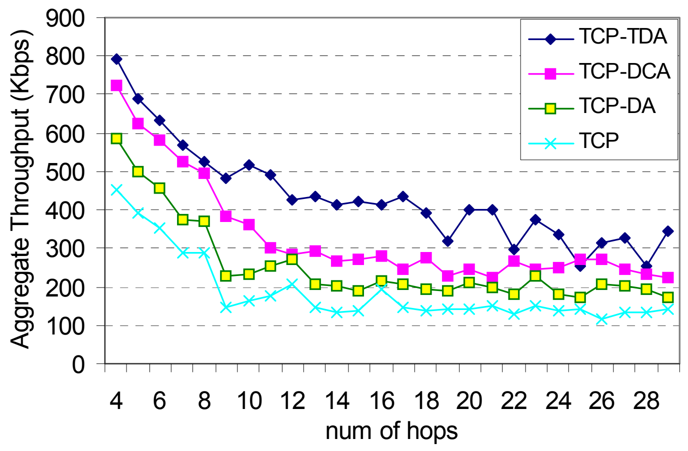


Six flows

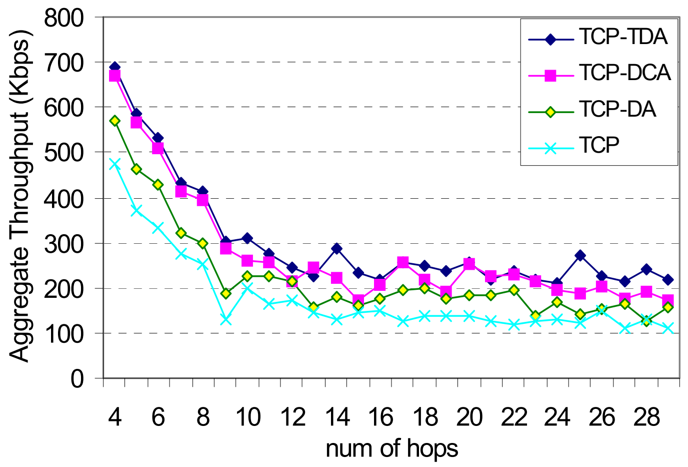
Figure5-8 TCP throughput for one TCP flow, 1~3 hops



(a) One flow



(b) Two flows



(c) Six flows

Figure5-9 TCP throughput for one TCP flow, 4~29 hops

Table 5-9 and figure 5-9 show the performance gain in the case of hop 4 ~hop 29. In the table, the first row is the number of hops, the second row is the performance gain over TCP-DCA, the third row is over the TCP-DA, and the fourth row is over the TCP.

4	5	6	7	8	9	10	11	12	13	14	15	16
15%	14%	12%	15%	16%	13%	54%	47%	60%	29%	20%	31%	28%
42%	42%	41%	50%	56%	92%	77%	86%	110%	77%	56%	100%	95%
75%	80%	81%	111%	93%	185%	170%	204%	189%	148%	110%	203%	112%

17	18	19	20	21	22	23	24	25	26	27	28	29
46%	25%	78%	81%	59%	34%	62%	36%	15%	27%	29%	36%	40%
72%	62%	78%	77%	65%	61%	72%	83%	69%	69%	47%	31%	60%
116%	141%	158%	163%	171%	151%	156%	132%	141%	130%	167%	11%	116%

Table 5-9 Performance gain over TCP-DA/Regular TCP for one-flow, hop 4~ hop 29

The performance gain of TCP-TDA over TCP-DCA is in the range of 12% ~ 80%, on average 35%. In most cases, the performance gain is more than 60% over TCP-DA, up to 110%, on average 68%, and 100% over TCP, up to 204%, on average 139%. We notice that the performance gain is higher for most cases in larger hop counts. One reason for this phenomenon is that the TCP sender emits a burst of packets after receiving a cumulative acknowledge and incurs heavy contention in the first few hops. When the chain is longer and the packets spread out the chain, this burst effect is alleviated [4].

For other numbers of flows, the performance gains are also significant. In the case of 2-flow, the average gain is 36%, 77%, 87% over TCP-DCA, TCP-DA, TCP, respectively. In 6-flows case, the gain is 11%, 38%, 76%, respectively.

5.6.1.2.2 New architecture for higher TCP throughput

We notice that although the TCP performance gain of TCP-TDA over TCP-DA/TCP is still significant, that over TCP-DCA decreased significantly. In the 6-flow case, TCP-TDA works only slightly better. In a few cases, the TCP-DCA works even slightly better than TCP-TDA. Our results also show that TCP-DA/TCP-DCA/TCP-TDA's performance deteriorate for larger number of flows (figure 5-10/5-11/5-12). The reason for this is that different TCP flows interfere with each other. This fact suggests that to achieve higher overall TCP throughput, the number of flows should be kept low, especially with large number of hops. This sheds light on architecture design for such scenarios, e.g., the application should be designed to share TCP connections, instead of setting up a lot of separate TCP connections. Based on this understanding, we propose to add a MUX/DEMUX layer between the application layer and TCP layer, as shown in figure 5-13. For simplicity reason, we only show one way TCP. Data stream of the same types from the application should be aggregated into one stream and then transmitted by one TCP connection, when it is possible. For example, assume a wireless terminal collects environmental data like the temperature, air pressure, etc., around its location, this node should not send those data by different TCP connections separately. Instead, it should aggregate all those data into one stream and only use one TCP connection. How to design this embedded layer may vary based on the specific characteristics of a given application, and it could be an interesting research topic for future work.

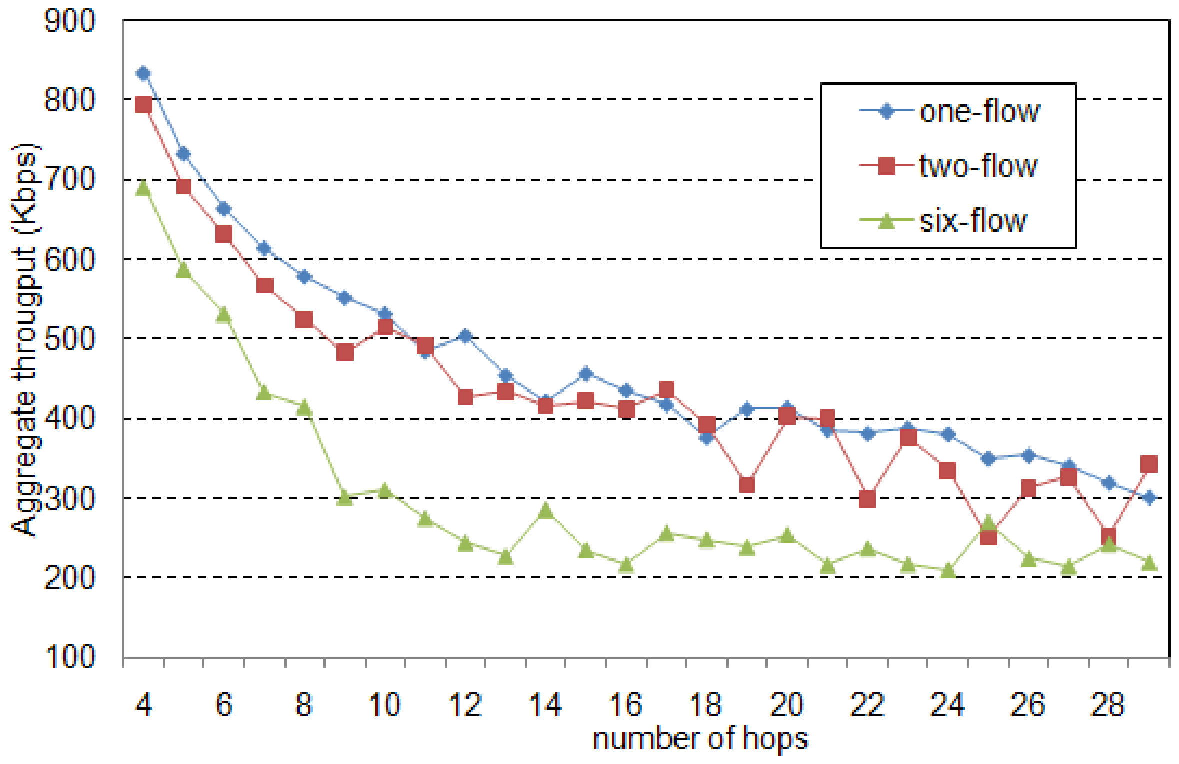


Figure 5-10 TCP-TDA throughput comparison for different number of flows

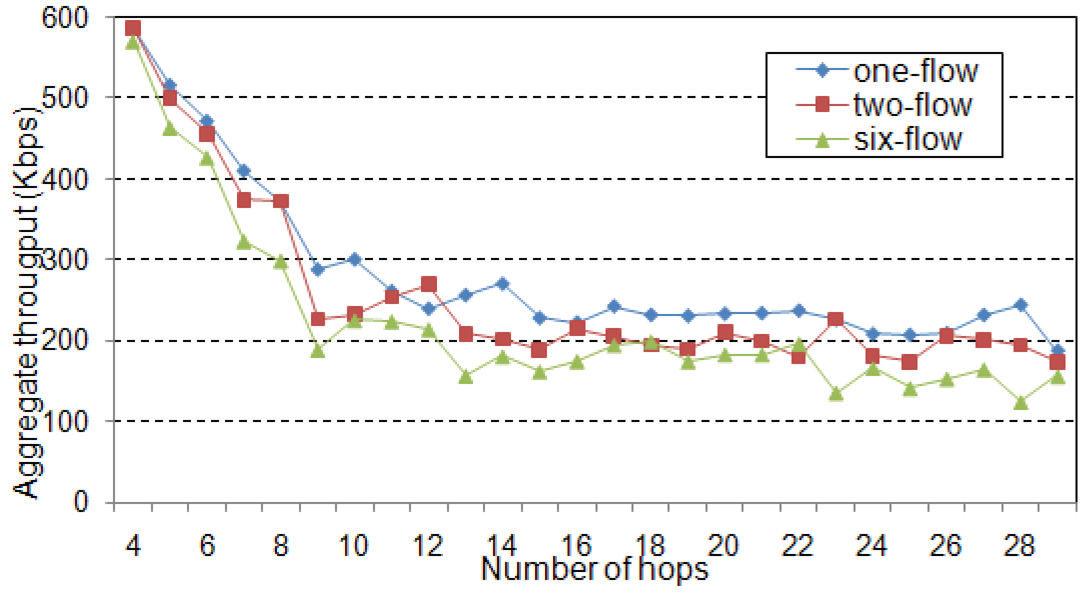


Figure 5-11 TCP-DA throughput comparison for different number of flows

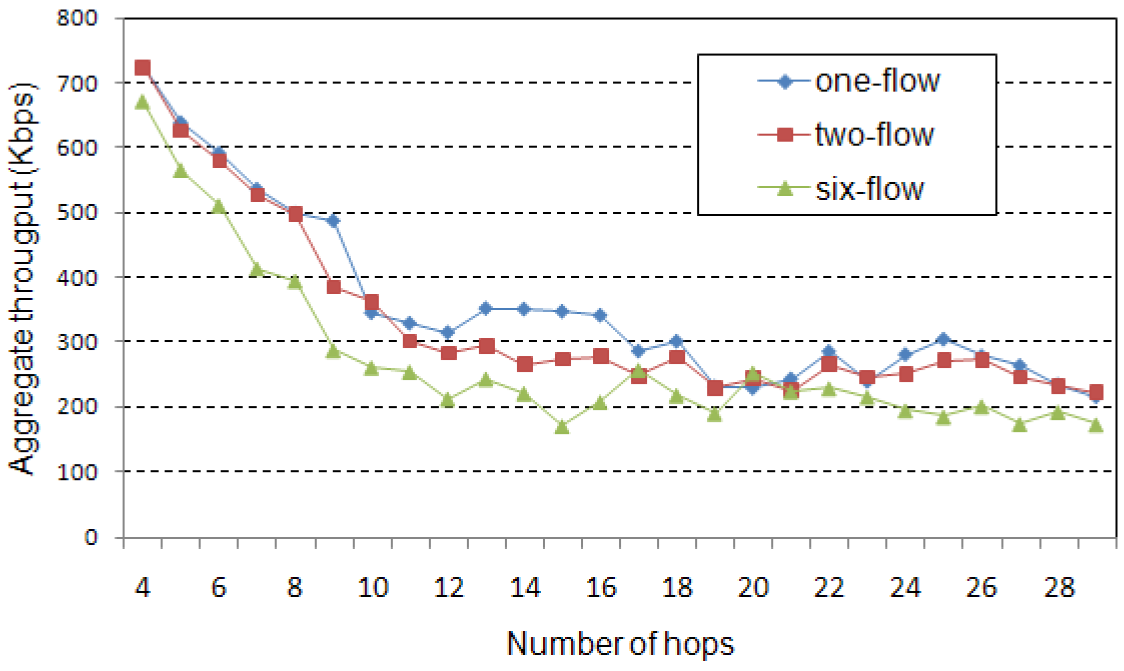


Figure 5-12 TCP-DCA throughput comparison for different number of flows

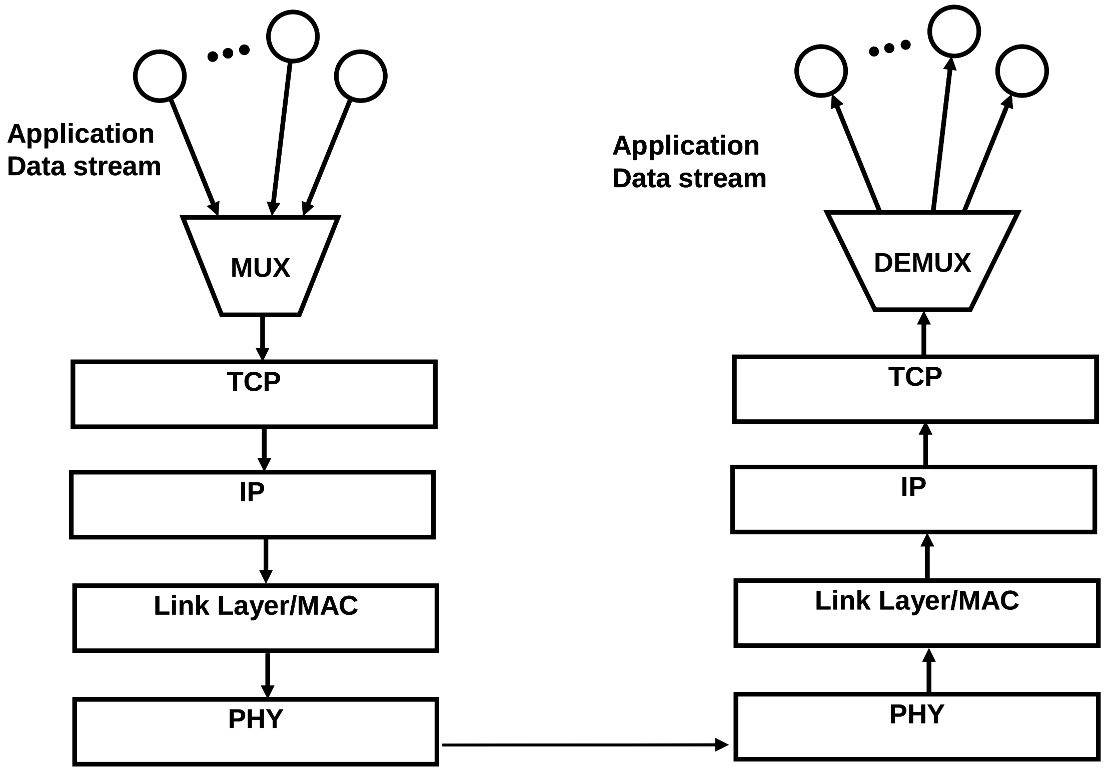


Figure5-13 Data Aggregation on top of TCP

5.6.1.2.3 The relationship between the number of ACKs and packets sent

Our proposed new method is based on the idea to minimize the number of acknowledge packets. We measure the number of data packets and the acknowledge packets using the simulation to validate this relationship.

Table 5-10 shows a typical simulation result of number of ACKs and data packets sent in the case of 13-hop chain and one TCP connection. For every 100 packets sent, the generated numbers of ACKs are 12, 35, 53 for TCP-TDA, TCP-DCA, and TCP-DA, respectively. This confirms that our TCP-TDA generates much less ACKs than other

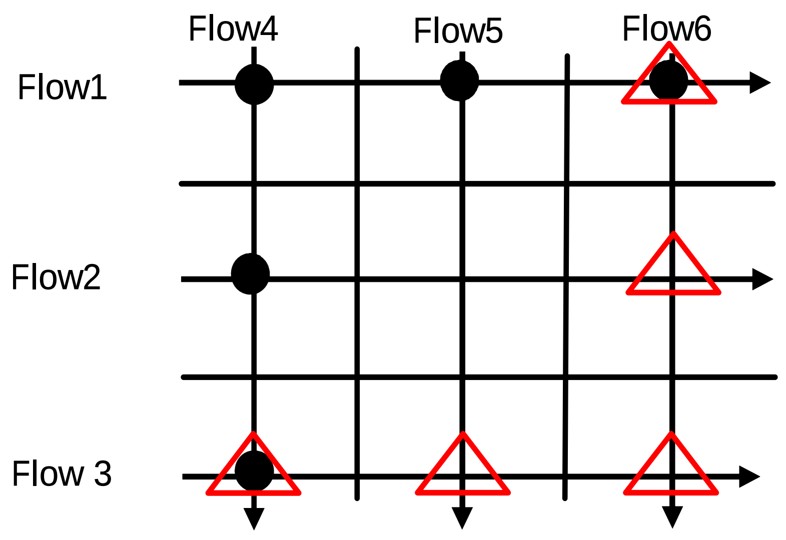
	Num of ACKs sent	Num of packets sent	Ratio	Throughput (Kbps)
TCP-TDA	693	5644	0.12	478
TCP-DCA	1509	4321	0.35	368
TCP-DA	1542	2887	0.53	242

Table 5-10 Number of ACKs and data packets for a 13-hop chain and 1 flow

algorithms. In the 6-flow case, we observed that, in many scenarios, the ratio for TCP-TDA increases to more than 0.3 and accordingly, the ratio for TCP-DCA increases to more than 0.4, which means that the number of generated ACK increases significantly and therefore the performance gain deteriorates, compared with the cases of fewer flows (figure 10/11/12). This confirms that reduction in the number of ACKs does help improve the TCP performance. Because other factors, e.g., packet retransmission, also affect the TCP behavior, throughput does not increase linearly with the number of reduced ACKs.

We notice that the ratio for TCP-DA is not 0.5 exactly although every two data packets are supposed to return one ACK packet. This is because some packets are dropped and need to be retransmitted.

We can also see an interesting phenomenon in figure 5-9. The throughput does not decrease monotonously with the increasing hop count, for all four algorithms. This may appear surprising because the more hops, the greater the interference, implying a lower throughput. This is because the TCP throughput is affected by the optimal amount of injected traffic. When the number of hops increases, the number of dropped packets may increased and generate more duplicate ACKs, which results in sooner *cwnd* shrinking. This, in turn, brings the injected traffic closer to the optimal value and a better throughput is achieved.



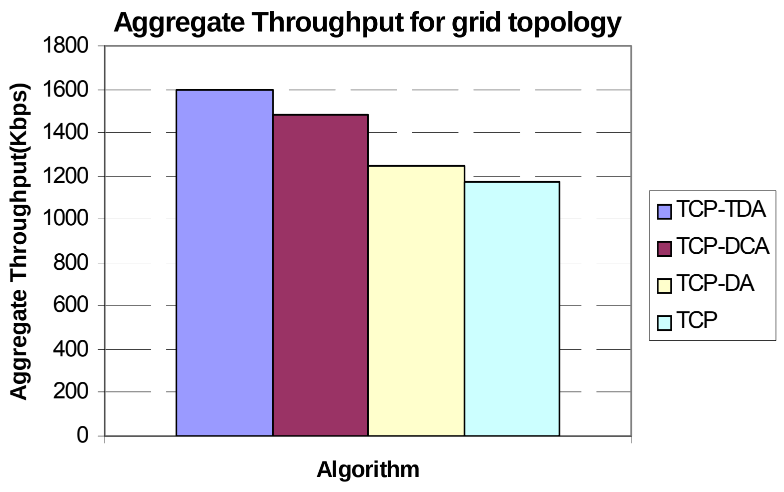


Figure5-14 Throughput for the grid topology

5.6.1.3 Throughput in cross topology

We also evaluate the TCP throughput in a more complicated grid topology shown in figure 5-14. The filled circles depict the sending nodes, the triangles indicate the receivers, and the arrow lines point the transmission direction. The result shown is the average over 5 runs with different seeds.

We compare TCP-TDA with TCP-DCA, TCP-DA and regular TCP. In this critical scenario, the interference and contention has greater impact than in the chain topology. Therefore, the performance gain of our algorithm is degraded significantly, and shows approximately 8%, 25%, 35% improvement over the TCP-DCA, TCP-DA and the regular TCP, respectively. Note that in our simulation scenario, the TCP-DCA outperforms the TCP-DA by around 18%, just match the results reported in [53]. We believe TCP-TDA will be further improved if combined with other techniques.

5.7 Summary

In this chapter, we analyzed the optimal TCP congestion window (*cwnd*) in multi-hop wireless network and provided a more accurate result than existing works. We concluded that the optimal *cwnd* should be between *n/3~n/2*, instead of *n/4* which claimed by other researchers. This result is validated by the data obtained by existing works.

Using mathematical analysis and simulation, we also proved that the much shorter TCP acknowledge packets consume comparative channel capacity with the much longer data packets. This is due to the overhead in MAC layer and PHY layer. Based on this understanding, we first proposed to separate the route for data packet and acknowledge packet in a grid wireless network. Our simulation results validated the effectiveness of this method. Due to the lack of generality of this approach, we further propose a more generic approach to improve TCP throughput by further minimizing the number of TCP Acknowledge. Our method achieves up to 203% throughput gain over regular TCP in long chain topology and 35% throughput gain in cross topology.

Our simulation results revealed a fact that the TCP performance degrades with the increasing of number of TCP flows. Based on this fact, to improve TCP performance, we proposed to reduce the number of TCP connection by introducing a sub-layer of MUX/DEMUX between the applications and the TCP layer, to aggregate multiple data stream into one. Although how to implement the MUX/DEMUX layer depends on specific applications or software platform, our results suggest that when we design a multi-hop wireless network, we should take this factor into account.

Chapter 6

Conclusions and Future Work

Modern network technologies have emerged and evolved following their distinct paths, from wired network to wireless network. How to improve network performance to achieve better interoperation, higher throughput, etc. has been on-going research topic in the past decades. In this dissertation, we have answered some challenges in heterogeneous network and in multi-hop wireless network to provide more reliable and faster services. In this dissertation, how to integrate the emerging IEEE 802.21, the Media Independent Handover, and Mobile IP to provide faster handover in heterogeneous network has been explored by theoretical analysis and experiments using our test-bed. Based on detailed analysis and simulation results, we also proposed new approaches to improve TCP throughput in multi-hop wireless network. This chapter summarizes the main contributions of our research work. The overall research efforts and the related conclusions are introduced first. Then, several suggestions for future research directions are identified.

6.1 Main contributions

Different wireless access technologies have been being prosperous recently and have been adopted by different service providers. In a given location, one technology may dominate other technologies. To allow roaming users to take advantage of this feature, more and more mobile terminals are equipped with multiple interfaces which are based on different wireless technologies, e.g., WiFi, Ethernet, 3G, LTE, etc. How to provide fast handover when the user moves from one network to another has been a big challenge. Currently, Mobile IPv4 (MIP) is the most popular protocol used to provide un-interrupted IP connection when a terminal changes the access network. We first identified the weakness of the Mobile IPv4 protocol in the case that multiple interfaces of a mobile terminal are connected to a single Foreign Agent. After analyzing the procedure of the standard mobile IPv4 protocol, we proposed a novel mechanism to achieve fast handover with MIH support. Based on this, we further proposed a MIP extension. To our best knowledge, this is the first proposal to address this type of problem. To validate our new approach, we have designed and deployed an experimental test-bed. We have also developed a whole set of software to be used for validation, from kernel space to user space in Linux. We run extensive experiments in the test-bed to confirm the effectiveness of our effort. Our experimental results validated that our novel method achieves much better performance during handover, compared with existing mechanism. Regardless of the FA-HA latency, the handover can be finished in around 200ms which is not perceptible even during a VoIP session, when an unexpected disconnection occurs. Using this technique will greatly improve the customer experience for roaming in heterogeneous network.

For TCP performance improvement in multi-hop wireless network, we first focused our work on studying the characteristics of TCP behaviors in multi-hop wireless network and the effect brought by different packet size in wireless network which is based on IEEE 802.11. TCP performance in contention-based multi-hop wireless networks is shaped by factors which are different from the wired network case. First, the maximum throughput for a given topology and flow pattern is reached at an optimal TCP congestion window (*cwnd*). Existing works do not take into account the capture effect which affects the transmission range significantly. We provided an improved analysis of this effect and a better result has been obtained, comparing with existing conclusion given by other researchers. This result provides a better understanding of the TCP behavior in multi-hop wireless network. Second, due to the specific characteristics of MAC and PHY overhead, the control traffic uses higher proportion of the channel bandwidth than in the wired network. We used mathematical analysis and simulation to show that the much smaller TCP ACK packets consume channel resource comparable to the much longer data packets, over high-speed connections. Inspired by these understanding, we proposed methods to improve TCP performance. Based on the first result, we can calculate an optimal congestion window and use it to control the TCP behavior. The second one suggests us to reduce the interference between data packet traffic and the TCP ACK traffic by minimizing the number of TCP ACK packets. Therefore, we first proposed to separate the data traffic and TCP ACK traffic in a grid wireless network using static routing protocol. To alleviate the negative effect of TCP ACK traffic, another way is to minimize the number of TCP ACK packet. As a result, we proposed an improved ACK-delay approach which has been shown to improve TCP performance significantly. This method is more generic than fixing the TCP congestion window to the theoretical value. Our simulation results validated the effectiveness of our new approach. Our research results also revealed that in order to get a better TCP throughput in multi-hop wireless network, it is desirable to minimize the number of TCP connections. Based on this finding, we proposed an architecture to aggregate the data to reduce the number of TCP connections, and therefore, to achieve higher TCP throughput.

6.2 Future work

As we have shown in previous chapters, the PHY and MAC layer overhead are inevitable for any packets. To provide higher throughput, one straightforward consideration is to mitigate the negative of the overhead. Following this idea, the IEEE 802.11 committee has proposed a new IEEE 802.11n. Besides the improvement in the PHY layer, significant changes in MAC have also been made. Briefly speaking, the ideas are (a) aggregate the packets to form a longer packet (b) use shorter inter-frame space (c) use a block acknowledge packet for multiple MAC packets to replace the acknowledge packet for a single MAC packet. All these methods can provide higher throughput theoretically. However, these improvements come with some restrictions and that incurs the mismatch between MAC layer and TCP. Therefore, for TCP, those possible improvements in MAC and PHY may be lost. For example, for a TCP connection, the packets that are allowed to inject into the connection are controlled by the congestion window. TCP connections use slow start technology to detect the connection capability. The original congestion window always starts from one, which means that in some time periods, we cannot take advantage of the benefits provided by the packet aggregation. Therefore, the actual throughput would be less than the MAC layer can provide. The situation becomes worse in the multi-hop scenario because all nodes are forced to transmit only few data packets in one MAC packet. How to eliminate the mismatch between MAC layer and TCP layer is a promising direction to take advantage of the MAC capacity provided by the new standard.

References

[1] IEEE, "IEEE 802.3 ETHERNET WORKING GROUP," http://www.ieee802.org/3/.

[2] IEEE802.11, "IEEE std 802.11: Wireless LAN medium access control and physical layer specifications," 1999.

[3] M. Gast, *802.11 Wireless Networks: The Definitive Guide, Second Edition*: O'Reilly Media, 2005.

[4] 3GPP, "3GPP Specification," http://www.3gpp.org/Specification-Groups.

[5] 3GPP2, "3GPP2 Specifications," http://www.3gpp2.org/public_html/specs/index.cfm.

[6] IEEE, "IEEE 802.16," http://standards.ieee.org/getieee802/download/802.16-2004.pdf.

[7] IEEE802.16e, "IEEE 802.16e," http://standards.ieee.org/getieee802/download/802.16e-2005.pdf.

[8] E. Dahlman, S. Parkvall, J. Sköld, and P. Beming, *3G Evolution - HSPA and LTE for Mobile Broadband*, 2nd edition ed: Academic Press, 2008.

[9] S. Sesia, I. Toufik, and M. Baker, *LTE - The UMTS Long Term Evolution - From Theory to Practice*: John Wiley & Sons, 2009.

[10] F. Khan, "LTE for 4G Mobile Broadband - Air Interface Technologies and Performance," 2009.

[11] M. Ergen, *Mobile Broadband - Including WiMAX and LTE*: Cambridge University Press, 2009.

[12] I. 802.21, "Media Independent handover (MIH), IEEE 802.21/D8.0," Dec 2007.

[13] C. Spurgeon, *Ethernet: The Definitive Guide*: O'Reilly Media, 2000.

[14] IEEE, "IEEE 802.3," http://www.ieee802.org/3/.

[15] 3GPP, http://www.3gpp2.org/.

[16] Samsung, "Samsung SCH-a790 Cellular Phone," http://www.samsung.com/us.

[17] Blackberry, "RIM 8820 smartphone," http://na.blackberry.com/.

[18] E. Fogelstroem, A. Jonsson, and C. Perkins, "“Mobile IPv4 Regional Registration”, RFC 4857," June 2007.

[19] H. Matsuoka, T. Yoshimura, and T. Ohya, "A robust method for soft IP handover," *IEEE Internet Computing*, pp. 18-24, 2003.

[20] R. Koodli and C. Perkins, "Mobile IPv4 Fast Handovers," http://tools.ietf.org/html/rfc4988, October 2007.

[21] S. Das, "TeleMIP: Telecommunication-Enhanced Mobile IP Architecture for Fast Intradomain Mobility," in *IEEE Personal Communications*, vol. 7(4), 2000, pp. 50-58.

[22] IEEE, ""IEEE std 802.11b(R2003): Wireless LAN medium access control and physical layer specifications," ", June,2003.

[23] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," *IEEE Journal on Selected Areas in Communications*, vol. 18, March 2000.

[24] E. PERAHIA and R. STACEY, *Next Generation Wireless LANs, Throughput, Robustness, and Reliability in 802.11n*: Cambridge University Press.

[25] IEEE, "802.11n-2009: Amendment 5: Enhancements for Higher Throughput," http://www.ieee802.org/, 2009, .

[26] E. Perkins, "“Mobility Support for IPv4”, RFC 3344," August 2002.

[27] J. F. Kurose and K. W. Ross, *Computer Networking: A top-down approach featuring the Internet (3rd Edition)*: Pearson Education, Inc., 2005.

[28] L. L. Peterson and B. S. Davie, *Computer Networks: A system Approach (3rd Edition)*: Morgan Kaufmann Publishers, 2003.

[29] K. El Malki and E. Athonet, "Low-Latency Handoffs in Mobile IPv4, RFC 4881," 2007.

[30] J. Corbet, A. Rubini, and G. Kroah-Hartman, *Linux Device Drivers, Third Edition*: O'Reilly Media, Inc., 2005.

[31] MadWiFi, "MadWiFi," http://madwifi-project.org/.

[32] MadWiFi, "MadWiFi," http://www.madwifi.org/.

[33] W. R. Stevens, *UNIX network Programming (2nd edition). Networking APIs:Sockets and XTI*, vol. 1: Prentice Hall, 1998.

[34] T. F. Herbert, *The Linux TCP/IP STACK: Networking for Embedded Systems*: Charles River Media, INC, 2005.

[35] M. Mitchell, J. Oldham, and A. Samuel, *Advanced Linux Programming*: New Riders Publishing, 2001.

[36] K. A. Robbins and S. Robbins, *Pratical UNIX programming: a guide to concurrency, communication, and multithreading*: Prentice-Hall, Inc, 1996.

[37] Wireshark, "Wireshark," http://www.wireshark.org/.

[38] tcpdump, "tcpdump," http://www.tcpdump.org/.

[39] NIST_Net, "NIST Net," http://snad.ncsl.nist.gov/itg/nistnet/.

[40] A. Tanenbaum and A. Woodhull, *Operating Systems: design and implementation 2nd Edition*: Prentice Hall, 1997.

[41] RFC793, "Transmission Control Protocol," 1981.

[42] M. Allman, V. Paxson, and E. Blanton, "TCP Congestion Control," in *http://tools.ietf.org/html/rfc5681*, September, 2009.

[43] IETF, "RFC1122 - Requirements for Internet Hosts - Communication Layers," http://www.faqs.org/rfcs/rfc1122.html, 1989.

[44] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla, "The impact of multihop wireless channel on TCP throughput and loss," presented at Proc. IEEE INFOCOM '03, 2003.

[45] S. Floyd, T. Henderson, and A. Gurtov, " The NewReno Modification to TCP's Fast Recovery Algorithm," http://tools.ietf.org/html/rfc3782, April 2004.

[46] S. Floyd and T. Henderson, "RFC 2582," http://www.ietf.org/rfc/rfc2582.txt.

[47] L. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: New Techniques for Congestion Detection and Avoidance," presented at SIGCOMM, 1994.

[48] G. Holland and N. H. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," presented at Proc. ACM MobiCom '99, Seattle, Aug 1999.

[49] C. Kai, X. Yuan, and K. Nahrstedt, "On setting TCP's congestion window limit in mobile ad hoc networks," presented at IEEE International Communications Conference (ICC '03), 2003.

[50] E. Altman and T. Jimenez, "Improving TCP over multihop networks using delayed ACK," presented at extended abstract, MADNET, Sophia-Antipolis, March 2003.

[51] T. Yuki, T. Yamamoto, M. Sugano, M. Murata, H. Miyahara, and T. Hatauchi., "Performance improvement of TCP over an ad hoc network by combining of data and ACK packets," in *IEICE Transactions on Communications*, 2004.

[52] R. De Oliveira and T. Braun, "A dynamic adaptive acknowledgment strategy for TCP over multihop wireless networks," Miami, FL, United States, 2005.

[53] C. Jiwei, L. Yeng Zhong, M. Gerla, and M. Y. Sanadidi, "TCP with Delayed Ack for Wireless Networks," presented at Broadband Communications, Networks and Systems, 2006. BROADNETS 2006. 3rd International Conference on, 2006.

[54] T. S. Rappaport., *Wireless communications, principles and practice.* , Prentice Hall, 1996.

[55] NS2, "NS2 network simulator," http://www.isi.edu/nsnam/ns/.

[56] K. C. Lu, Y. Wang, J. P. Lynch, C. H. Loh, Y. J. Chen, P. Y. Lin, and Z. K. Lee, "Ambient Vibration Study of Gi-Lu Cable-stay Bridge: Application of Wireless Sensing Units.," presented at Proceedings of SPIE, 2006.

Curriculum Vitae

Beizhong Chen

9/1987 – 6/1991	Department of Radio & Information System, Wuhan University, Wuhan, Hubei Province, P. R. China.
	Bachelor of Science in Radio Electronics, July 1991.
	Graduated with honor
8/1991 – 10/1998	Engineer and R&D Group Leader (from 1995), Shougang NEC, Beijing, P.R.China
9/1999 – 12/2001	Rutgers, The State University of New Jersey, Department of Electrical and Computer Engineering, New Brunswick, New Jersey.
	Master of Science in Electrical and Computer Engineering, January 2001.
03/2001 – 03/2002	Researcher (Intern) in home networking, Panasonic Info & Networking Technologies Laboratory, Princeton, NJ
07/2006 – 03/2008	Researcher (Intern) in wireless communications, Bell Labs, Alcatel-lucent, Murray Hill, NJ
1/2002 – 5/2010	Rutgers, The State University of New Jersey, Department of Electrical and Computer Engineering, New Brunswick, New Jersey.
	Doctor of Philosophy in Electrical and Computer Engineering, October 2010.
6/2010 – present	Senior Software Engineer in Atheros Communications, Inc.