PERFORMANCE ANALYSIS AND COMPARISON OF TWO WIRELESS WIDE AREA NETWORKS: MOBILE INTERNET PROTOCOL AND CELLULAR DIGITAL PACKET DATA

By

Paraskevas Andrea Polydorou

B.A.Sc., Simon Fraser University

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in the School of Engineering Science

SIMON FRASER UNIVERSITY
December 1996

ISBN 0-612-17059-4

PARTIAL COPYRIGHT LICENSE

Title of Thesis/Project/Extended Essay

"Performance Analysis and Comparison of two Wireless Wide Area Networks: Mobile-IP and CDPD"

Author: _____
(signature)

_____
(name)

November 26, 1996
(date)

APPROVAL

Name: Paraskevas Andrea Polydorou

Degree: Master of Applied Science

Title of thesis: Performance Analysis and Comparison of two Wireless Wide Area Networks: Mobile Internet Protocol and Cellular Digital Packet Data

Examining Committee: Dr. Ash Parameswaran

Associate Professor, Engineering Science, Chairman

Dr. R.H.S. Hardy

Professor, Engineering Science

Senior Supervisor

Dr. Paul Ho

Associate Professor, Engineering Science

Supervisor

Dr. Jacques Vaisey

Associate Professor, Engineering Science

Internal Examiner

Date Approved: Dec 3-96

ABSTRACT

Two of the most important wireless wide area networks are Mobile Internet Protocol (IP), developed by the Mobile IP Working Group of the Internet Engineering Task Force (IETF) and Cellular Digital Packet Data (CDPD), proposed by the CDPD Forum. In spite of their common roots the two protocols have some important differences which affect their performance during some critical phases of wireless communications. Two examples are the handling of the initial registration of the mobile node with the mobile base station and with its home network, and the handling of handoffs.

In this thesis we concentrate our attention on the network layer performance by studying the number and size of the messages exchanged during the critical phases as well as the efficiency of the routing of packets from and towards the mobile node. The draft proposal for route optimization in Mobile IP is also considered and its improved routing performance is compared to the base Mobile IP and CDPD protocols.

The performance of the two protocols as well as three variations: intra-area handoff for CDPD, multiple bindings for Mobile IP and the route optimization proposal for Mobile IP, are measured using the Optimized Network Engineering Tools (OPNET) simulation package. The simulation results are verified using analytical queueing methods. The Transmission Control Protocol (TCP) is used as the transport layer for all protocols and various scenarios of operation of the mobile connection are applied, such as a file

transfer with a constant stream of equally sized packets and a Poisson stream of packets.

CDPD and the base Mobile IP protocol simulations show very little differences in their performance in spite of CDPD's use of slightly shorter registration packets. In addition, the simulations show a significant improvement of throughput and end-to-end delay for the optimized Mobile IP while the mobile node is operating within a single cell. The improvement depends on the extra overhead introduced by the triangular routing of the base protocols. During handoffs, however, optimized Mobile IP is outperformed by both the base Mobile IP and CDPD due to the outdated location caching.

ACKNOWLEDGMENTS

I would like to thank my supervisor, Dr. R.H.S. Hardy, for his guidance and support throughout the duration of my research and my fellow researchers Pamela Lee and Panayiotis Toundas for their cooperation. I would also like to thank Motorola for its financial support and MIL 3, Inc. for providing access to their OPNET simulator.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

CDPD	Cellular Digital Packet Data
CLNP	ConnectionLess Network Protocol
CN	Correspondent Node
DHCP	Dynamic Host Configuration Protocol
EKE	M-ES Key Exchange
ESH	End System Hello
ETE	End To End
FA	Foreign Agent
GRE	Generic Routing Encapsulation
HA	Home Agent
ICMP	Internet Control Message Protocol
IKE	MD-IS Key Exchange
IP	Internet Protocol
ISO	International Standardization Organization

LAN	Local Area Network
MDBS	Mobile Data Base Station
MD-IS	Mobile Data Intermediate System
M-ES	Mobile End System
MHF	Mobile Home Function
MN	Mobile Node
MSF	Mobile Serving Function
OPNET	OPtimized Network Engineering Tools
OSI	Open System Interconnection
RDC	ReDirect Confirm
RDR	ReDirect Request
RF	Radio Frequency
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

CHAPTER 1

INTRODUCTION

This document presents my research project in the area of mobile data communications. Two of the most important wide area protocols were investigated and compared to each other. The project consisted of the analytical derivation of performance measures and the development of simulation models of the two protocols focusing most of the attention to their network layer performance. The two protocols are Mobile IP[1], submitted by the Mobile IP Working Group of the Internet Engineering Task Force (IETF) and Cellular Digital Packet Data (CDPD)[2], proposed by the CDPD Forum.

An introduction to the area of mobile networks will now be presented indicating the deficiencies of existing network protocols when dealing with mobility issues. The two protocols will then be described in some detail. Following in the next two chapters, are the analytical and simulation results and, finally, analysis and conclusions from the results will be attempted.

1.0.1. Increasing demand for mobility

The continuously decreasing size and price of computer components and devices has made it possible for the average user to afford a portable computer which often has comparable performance to the best desktop workstations. The resulting proliferation of portable computers created a new breed of computer users who are continuously on the move but still require to maintain uninterrupted network connectivity either to their workplace LANs or to the Internet. As a result the increasing numbers of users attracted a greater number of companies in this area.

Many operations done previously using dial-up lines and modems can be performed more efficiently using mobile networks. Sales-people, for example, would no longer need a phone line to send their orders to the main office. They can, instead, acquire continuous connectivity with their main computer network through a wireless wide area network such as CDPD. Travelers would not need to subscribe to different Internet service providers in each place they visit, but may access the Internet through the local mobile network provider.

1.0.2. Deficiency of conventional protocols

Mobile nodes are, by definition, not constrained to be stationary but their location changes with time. The mobile node may be found in a moving vehicle or be a hand-held device carried in the workplace. The advantage of mobile nodes over traditional stationary nodes is the freedom to be able to connect to the network from arbitrary places

without requiring messy cabling or manual reconfiguration of the network each time the physical location changes.

This freedom, however, interferes with the existing hierarchical routing algorithms which assume that the node is stationary and attached to its home network. Intermediate routers in conventional networks, such as the Internet, route packets to a known router in the home network of the destination node. The local router is, then, responsible for the delivery of the packet to the local node. A new protocol is, therefore, needed for the correct delivery of packets destined for a mobile node even if the node is attached to a network away from its home network.

In the case of mobile air communications the airlink poses additional security problems. The broadcast nature of RF communication means that anyone can interfere with the communication in the form of either passive eavesdropping or active replay attacks where an attacker may use (replay) a registration message from a previous registration to pose as the mobile node. An authentication and cryptographic scheme is, therefore, very important for this type of communication. Currently, the aim is for a wireless standard which will be at least as secure as the existing wireline networks.

Finally, performance is pivotal to the wide acceptance of mobile networks due in part to the increasing popularity of multimedia applications. Since the most common communications medium for mobile communications is the radio channel, performance suffers from effects of multipath fading and from 'poor' signal areas. Many attempts have been made to discover an efficient protocol which would improve the performance in spite

of these effects by using error correction and/or by minimizing the size and number of messages sent over the airlink as well as optimizing packet delivery to mobile nodes. An additional performance concern is the additional delay during handoffs, the transfer, that is, of the mobile node to a different cell. The extra delay during the registration in the new cell causes higher layers to misinterpret the situation as congestion and activate their congestion avoidance algorithms causing an even greater disruption of service.

1.0.3. Early stages of mobile networks

There have been many attempts to establish standards which would enable mobile nodes using TCP/IP to communicate over a radio link with existing stationary nodes. Three of the major considerations of these attempts were:

- Existing hosts and routers should not be required to make any software modification
- Adequate mobile user authentication and data security should be provided
- Acceptable performance of the mobile network for the intended application should be guaranteed.

Most of the attempts relied on mobile aware routers which use some form of encapsulation/decapsulation, also called tunneling, of the messages to their correct destination. This method does not require any changes to existing intermediate nodes and routers (see Figure 1). The mobile node communicates with the mobile-aware host, which in turn notifies the node's home network about the node's current position. The home

network will then forward any packets destined for the mobile node to this mobile-aware host. A drawback of this method is the longer routing path taken by the packets destined for the mobile node.

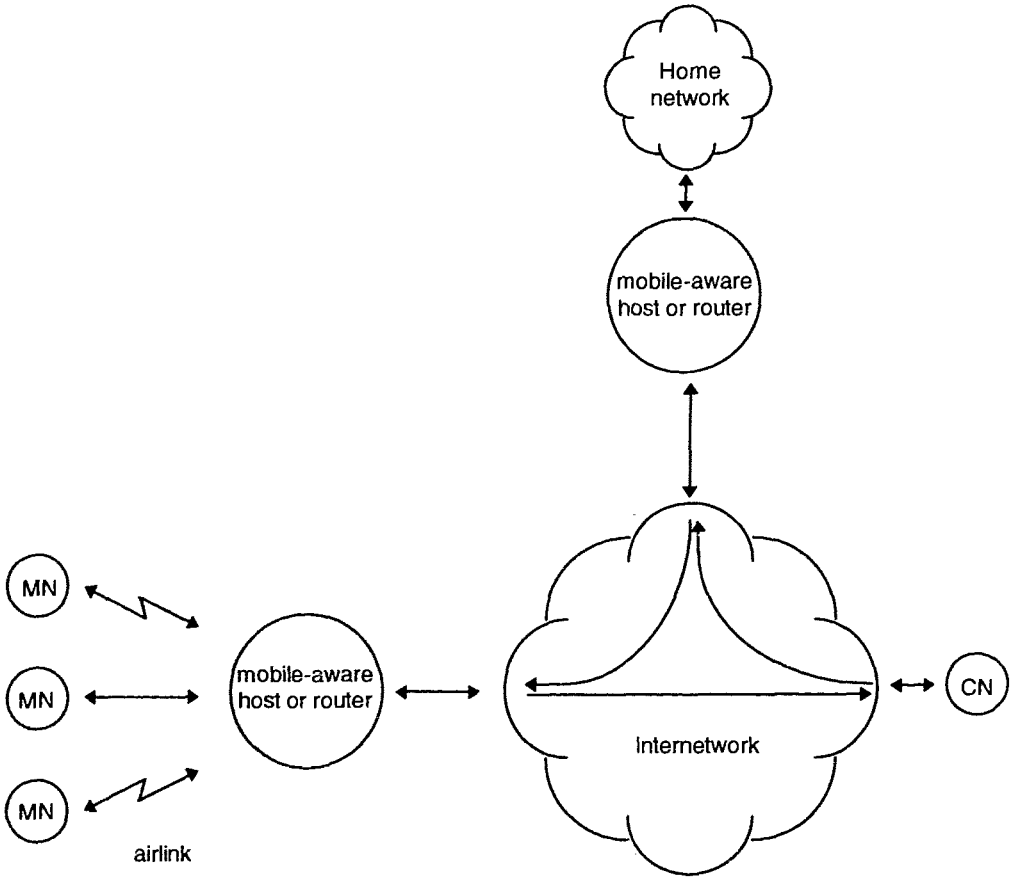


Figure 1. Sample network supporting mobile nodes.

Some of the most important early proposals for IP enhancements that enable support for mobile users were the VIP - Sony Project, the IBM MHP project, the Matsushita MHP project and the Columbia MHP project. These proposals are summarized in [3]. Mobile IP and CDPD derived many of their features from these proposals.

1.1. Introduction to the protocols

1.1.1. Mobile IP

Mobile IP is the submission by the Mobile IP Working Group of the Internet Engineering Task Force (IETF). IETF is the principal standards development body for the Internet. The protocol enables a mobile node to send and receive packets over the Internet using its home address regardless of the point of attachment.

The mobile node must be associated with a home agent which resides in a router in the home network. This home agent is responsible for ensuring that the mobile node will receive any packets destined for it. The mobility association is established with registration when the mobile node either obtains a dynamically assigned temporary address or it establishes a connection with a foreign agent which assigns a care-of address to the mobile node. A dynamically assigned address (or co-located care-of address) is local to the visited network and is obtained by the mobile node either temporarily using a mechanism such as DHCP or it is owned by the mobile node and is used each time it visits the foreign network. The option of using a dynamically assigned address by a mobile node enables the operation of the Mobile IP protocol even in networks unaware of mobility with the mobile node also assuming the role of the foreign agent.

The care-of address, which in most cases is the address of one of the interfaces of the foreign agent, is communicated to the home agent during registration and is used as the end of the 'tunnel' between the foreign and home agents. The term 'tunnel' means that packets destined for the mobile node will be encapsulated with a different IP

header with destination the foreign agent. Thus, the encapsulated packet will be unaware of the fact that it traveled from the home agent to the foreign agent. This form of routing is called triangular routing.

The foreign agent is responsible for routing packets to the mobile node. After the mobile node's registration with its home agent, packets destined for the mobile node will reach it via its home agent and foreign agent, if used, whereas messages originating at the mobile node will travel through an optimal route using conventional routing protocols.

In addition to the standard features of Mobile IP, optimization enhancements have been proposed in a separate document. The enhancements attempt to improve the routing of packets to the mobile node by notifying the correspondent node of the mobile node's position, thus, eliminating triangular routing.

In a separate draft IETF proposes a set of optimization enhancements. The draft proposes the caching of the mobile node's location by the corresponding node (which supports mobility) which will tunnel each packet directly to the mobile node, thus avoiding the triangular routing which extends the travel of each packet. The document also provides for the forwarding of any packets which are in flight during a hand-off to the mobile node thus saving the network from unnecessary TCP retransmissions. The mobile node may request notification of its previous foreign agent when registering with a new foreign agent. In this case, the new foreign agent sends a binding update message to the previous foreign agent which updates its binding cache to point to the new location of the mobile node and forward any messages erroneously routed there.

1.1.2. CDPD

The Cellular Digital Packet Data (CDPD) Network is a 'peer multi-protocol, connection-less network', proposed by the CDPD Forum, a trade association of carriers. equipment suppliers, and application developers.[2] It is based on early IETF Mobile IP work so the two proposals have many similarities, but also some differences. The main idea behind CDPD is that it may share unused channels in existing Advanced Mobile Phone Systems (AMPS) to provide a 19.2 kbps data channel. The CDPD specification describes in detail the two lower network layers.

The terminology used in CDPD follows the ISO - OSI model terminology and is different from the terminology used in Mobile IP. The mobile node is now called a mobile end-system (M-ES), the home and foreign agents are called mobile home and mobile serving functions (MHF and MSF respectively) and reside in a mobile data intermediate system (MD-IS). A mobile data base station (MDBS) is also defined which deals with the airlink communications and acts as a data link layer relay between the M-ES and the serving MD-IS. Two protocols, the Mobile Node Registration Protocol (MNRP) and the Mobile Node Location Protocol (MNLP) are responsible for the registration of the M-ES with its home MD-IS and the proper routing of packets destined for the M-ES.

1.2. Thesis Overview

For this project the performance of the mobile networks CDPD and Mobile IP was investigated. These two protocols are the most important of the mobile network

proposals, with CDPD currently being deployed at major cities in the US and recently in Vancouver by BC-Tel Mobility, and Mobile IP, which is proposed by the principal standards development body for the Internet, has just issued its RFC (Request For Comments) with number 2002.

Although CDPD was derived from an early draft of Mobile IP, it is different in some respects and, therefore, a difference in throughput performance and time delay of packets is expected. Some of the sources of discrepancy are expected to be:

- The size of the registration messages sent over the airlink and how they affect the initial registration and the registration/de-registration process during hand-off.

- Initial key exchange messages in CDPD (for encryption purposes) which are not provided for in Mobile IP.

- Better integration of the network and lower layers in CDPD allowing seamless intra-area handoffs.

- Proposed Mobile IP optimization enhancements which, in effect, eliminate triangular routing.

The project was divided in two parts with each part producing performance results in a different way. Analytical methods produced theoretical measures as well as providing the background for creating the simulation models, and simulations produced more

accurate measures of the performance of the two protocols.

Each part considers various modes of mobile operation. A slow moving (or stationary) mobile node requires a one time connection establishment overhead whereas a fast moving node provides information about the hand-off overhead and its effect on overall performance. A large file transfer grants information about the throughput, whereas small packet transfer (interactive data) provides information about packet delay. Finally, a user requiring continuous connection was contrasted with a user who requires many connections/registrations.

1.3. Contribution of the Thesis

The simulation results indicate the dependency of the network layer mobility handling on other factors, namely the transport layer, the channel error characteristics and the datalink layer's error handling. The effect of errors occurring during handoff is, also, identified indicating how fragile mobile connections are. Finally, a scheme for interoperability of Mobile IP and CDPD is proposed which will allow mobile users to roam seamlessly between the two networks.

CHAPTER 2

BACKGROUND AND PRELIMINARY WORK

In this chapter, we present a detailed description of the two protocols. Some preliminary work which consists of packet flow diagrams indicating the procedures for the operation of each protocol and the size of the messages exchanged during these procedures can be found in Appendix A. Also included in this chapter is analytical work related to the calculation of end to end delay in a packet switched network.

2.1. Mobile IP

The first step in establishing network connectivity by the mobile node is the discovery of a mobility agent using the Internet Control Message Protocol (ICMP)[4]. The mobile node, home agent and foreign agent implement ICMP router discovery [5] with a few mobility extensions. Agents (foreign or home) periodically send agent advertisements which are actually the router advertisements defined in [5] with

additional mobility extensions. Mobile nodes upon receiving the advertisements process them in order to find a suitable mobility agent. If they do not receive advertisements for some time, the mobile nodes may send router solicitations requesting a mobility agent. To accommodate the differences of a wireless link the mobile node may solicit more frequently than what is allowed in the conventional ICMP protocol and may do this for an unlimited number of times.

2.1.1. Registration Process

After discovering a mobility agent, and the agent is not the home agent, the mobile node must use the registration function in order to create an association of its home address with a care-of address. The care-of address is an address local to the network of attachment which will be used as a temporary proxy address for the mobile node. The home agent will "tunnel" any arriving packets destined for the mobile node to this address.

The registration takes place between the mobile node and its home agent. This process may be done in two ways, depending on the method used to obtain the care-of address. If the care-of address is dynamically assigned then a foreign agent is not needed and the registration/deregistration may be done with two messages:

a) registration request from the mobile node to its home agent requesting service

b) registration reply from the home agent granting or denying service to the mobile node.[1]

---

[1] Both registration messages are carried in a User Datagram Protocol [6] message.

If the care-of address is associated with a foreign agent, the foreign agent will act as a relay of the messages described for the previous case. Therefore, the foreign agent does not play an active role in the registration process but simply relays the registration request and reply messages. It keeps, however, a visitor list with the media address, home address, and home agent of the mobile node and the lifetime of the mobility association.

The registration request includes a lifetime field in which the mobile node indicates the desired duration of registration. The home agent may accept the request as is or may grant a shorter lifetime. Authentication of registration messages is also required and each of the mobile node, the home agent and the foreign agent must maintain an internal table with a list of security associations for mobile entities.

To protect against security threats mobile nodes and home agents must, and the foreign agents should, support authentication using the default algorithm, keyed MD5 [6] with a key size of 128 bits which precede and follow the data. There is also the option of supporting other authentication algorithms.

2.1.2. The Mobile Node

A mobile node must be initially configured with a home address and mobility security association for its home agent. It must monitor the link connection for agent advertisements thus detecting a change in its network connectivity but must not attempt to register more often than once per second. For each pending registration it must have the link-layer address of the foreign agent, if applicable, care-of address, registration

identification and the lifetime requested/granted. A mobile node may have more than one simultaneous mobility binding. This option, when exercised, may improve the hand-off performance when the mobile node switches back and forth to a few foreign agents since no new registration will be needed. A Mobile IP network exercising this option is studied in addition to the base Mobile IP.

When the mobile node discovers that it has reached its home network it should attempt to deregister with any foreign agents with which it had an association. In this case, the mobile node will not require any mobility services and will behave as a conventional node.

A mobile node which requires location privacy may create a two-way tunnel to and from its home agent. It will, thus, not only receive tunneled packets but it will also tunnel all outgoing packets to its home agent which would then route these packets to their destination giving the illusion that they emanated from the home network.

2.1.3. The Mobility Agent

The home agent maintains for each authorized mobile node the mobile node's home address and a mobility security association which provides for an authentication algorithm and mode, a secret key and a style of replay protection. During registration it receives the registration request message, validates the request and either accepts or denies registration. For each registered mobile node it maintains a list with the care-of address, registration identification and the lifetime of the association. It may also support

the option of multiple simultaneous mobility bindings.

When the registration process is completed the tunneling of datagrams is done using IP in IP encapsulation [7] and must be supported by the foreign agents, home agents, and the mobile nodes which can accept dynamically assigned care-of addresses. It is not required from a mobile node which will only register through a foreign agent. Minimal [8], as well as Generic Routing Encapsulation (GRE) [9], encapsulations may be used if they are supported by all nodes involved.

During data transfer, the foreign agent decapsulates any tunneled datagrams destined for the mobile node and acts as the gateway for outgoing packets. It may, optionally, maintain security associations with the mobile node and the home agent. The home agent is responsible for encapsulating every packet destined for a mobile node in its list of registered nodes and forwarding it to the care-of address.

2.1.4. Mobility Message Extensions

The Mobile IP proposal requires several mobility message extensions:

- The mobility extension is used to indicate that a router advertisement message is actually an agent advertisement sent by a mobility agent.

- The prefix length extension in agent advertisements indicates the subnet prefix used by the agent for the interface used for the advertisement.

- The key identifier extension is used in the registration requests to indicate that

authentication is performed using a different method than the default. Absence of this extension indicates use of the default message authentication code.

- The Mobile-Home, Mobile-Foreign and Foreign-Home authentication extensions are used to provide the authenticator for each case.

2.2. Optimized Mobile IP

The main purpose of the optimization enhancements is to avoid the triangular routing defined by the base protocol. In order to implement these enhancements, however, more nodes must be aware of mobility and more messages need to be exchanged during the critical periods of operation. The increased number of nodes taking part in the routing produces a synchronization problem which degrades the performance during critical periods such as during a handoff. Since the optimized Mobile IP is a supplement to the base protocol we will describe the operation assuming familiarity with the base Mobile IP.

Communications begin when the mobile node receives an ICMP agent advertisement message from a nearby foreign agent. Using the information provided in the ICMP message the mobile node sends a registration request message to its home agent via the discovered foreign agent. This message includes, in addition to the base Mobile IP fields, an authentication field for the foreign agent. The home agent, then, replies with a registration reply message granting or denying the request. So far the procedure is the same as in the base protocol, except for the extra foreign agent authentication key

exchange.

When the mobile node communicates with a correspondent node implementing the optimized Mobile IP protocol the triangular routing is eliminated using the following procedure: The first packet from the correspondent node to the mobile node travels normally to the home network where it is intercepted by the home agent. The home agent forwards this packet to the mobile node but it also sends a binding update message to the correspondent node. In this message the home agent informs the correspondent node of the binding of the mobile node to its foreign agent, thus, indicating a more optimal route. The correspondent node updates its routing tables and communicates directly with the mobile node through its foreign agent bypassing the home agent.

Handoffs are handled in the following way: When the mobile node realizes that it has moved to a different foreign agent it sends a registration request to its home agent via this new foreign agent. The new foreign agent forwards, as before, the request to the home agent but it also sends a binding update message to the previous foreign agent informing it of the new location of the mobile node. The home agent, upon receiving the registration request, grants or denies the request and sends the registration reply to the mobile node via the new foreign agent. The old foreign agent, upon receiving the binding update message, updates its routing tables and forwards any unsent messages destined for the mobile node to the new foreign agent.

The correspondent node also needs to update its routing tables after a handoff: When a packet from the correspondent node reaches the old foreign agent it triggers a

binding warning message to be sent back to the home agent informing it of the outdated routing information of the correspondent node while the packet is forwarded to the mobile node via the new foreign agent. Upon receiving the binding warning message the home agent sends a binding update message to the correspondent node with the updated location information of the mobile node.

2.3. CDPD

CDPD is a cellular standard defining the two lower layers of the network model. The basic service area of CDPD is the cell which is served by a single Mobile Data Base Station. (A MDBS may support various cells.) The area served by MDBSs which are controlled by the same MD-IS is called the routing area subdomain. A CDPD domain comprises all the MD-ISs served by a single CDPD service provider.

The M-ES is the network user, which may attach at different points in the CDPD network. The applications on the M-ES may use either the Internet IP or the ISO CLNP network protocols.

The MD-IS is the mobile-aware router in the CDPD network. It is responsible for routing messages to and from its home M-ESs, even when they are not attached to the home area, hiding mobility from the conventional network. To perform the routing two mobility routing functions are used:

- Mobile Home Function (MHF): The MHF resides in the home area MD-IS and consists of two services: The Location Directory, which keeps

information about the current location of its M-ESs, and the Redirection Server, which uses encapsulation to forward packets destined for each of its M-ESs that are outside of the home area.

- Mobile Serving Function (MSF): The MSF resides in the serving area MD-IS and consists of two services: The Registration Directory, which keeps information about M-ESs currently in the service area, and the Readdress Server, which decapsulates messages tunneled from the Redirection Server and delivers them to the M-ESs.

The MDBS is responsible for the airlink communication with the M-ES and behaves as a data link layer relay between the M-ES and the serving MD-IS. It is responsible for clearing a channel which is needed for a voice call and notifying each M-ES to move to an unused channel (channel hopping).

After the data link connection establishment, the M-ES must register with the new serving MD-IS by sending the End System Hello (ESH) message. The serving MD-IS sends a Redirect Request (RDR) message to the home MD-IS which replies with a Redirect Confirm (RDC) message and sends a Redirect Flush (RDF) message to the previous serving MD-IS. The serving MD-IS then sends a MD-IS Hello Confirm message to the M-ES. Authentication parameters are also included in the registration messages.

The transfer of a M-ES from one cell to the next, with both cells controlled by the same MD-IS, is called an intra-area transfer and is transparent to the network layer. The

transfer to a cell controlled by a different MD-IS is called an inter-area transfer and re-registration is required.

Data security and authentication are also included in the CDPD specifications. Key exchange is initiated by the MD-IS in order to encrypt all data transferred through the airlink. Only after the successful key exchange can the registration process start. Authentication parameters are exchanged as options of the registration messages.

Datagrams from the M-ES (reverse direction) travel through the serving MD-IS to their destination with conventional routing. Datagrams destined for the M-ES (forward direction) are first routed to the home MD-IS which in turn forwards them, using encapsulation, to the serving MD-IS. The serving MD-IS, then, decapsulates and delivers the datagrams to the M-ES.

2.4. Analytical Performance Measures

Network performance is often difficult to evaluate analytically except for trivial cases. In many cases a realistic estimate of the performance of networks is obtained only through simulation. For this project, an attempt was made to obtain accurate analytical solutions for the throughput performance and average packet delay of the two mobile networks under consideration.

The packet flow models provided the basis for the analytical work. Information about the arrival and service time distributions was obtained from theory.

Simplifying assumptions were made in order to facilitate the calculations. These assumptions were also verified with data obtained from the other two parts of the project providing a more justified result. Key assumption is Kleinrock's Independence Assumption [11] which states that:

> *Each time a message is received at a node within the network, a new length, $\tilde{b}$, for the message is chosen independently from the pdf*
>
> $$p(b) = \mu e^{-\mu b} \qquad b \geq 0$$

The above assumes an exponential message size distribution but may be generalized to any message length distribution with similar results.[12] This assumption was very useful when a conventional network is between the foreign and home agents.

From the packet flow diagrams it can be seen that the network is an open queueing network and its performance may be evaluated using methods found in [11], [12], [13], [14], [15] and [16].

Assume there are M interconnected nodes in the network and the probability of a packet leaving node i to go to node j is r_{ij}. The packet rate entering the source is λ and the service rate in queue i is μ_i. The vector probability p(**n**) is the joint probability of having n_i packets in queue i and p($\mathbf{n}$-$\mathbf{1}_i$) is the probability of a packet leaving from queue i. If the queues in each node behave as M/M/1 queues then we may solve the global balance equation for the vector probability p(**n**)

$$\left[\lambda + \sum_{i=1}^{M} \mu_i\right] p(\mathbf{n}) = \sum_{i=1}^{M} \lambda r_{si} p(\mathbf{n} - \mathbf{1}_i) + \sum_{i=1}^{M} \mu_i r_{id} p(\mathbf{n} + \mathbf{1}_i) + \sum_{i=1}^{M} \sum_{j=1}^{M} \mu_j r_{ji} p(\mathbf{n} + \mathbf{1}_j - \mathbf{1}_i)$$

using

$$\gamma_i = r_{si}\lambda + \sum_{j=1}^{M} r_{ji}\gamma_j \qquad i=1,2,3,...,M$$

where γ_i is the packet arrival rate to queue i, to get

$$\gamma_i p(\mathbf{n} - \mathbf{1}_i) = \mu_i p(\mathbf{n}) \qquad i=1,2,3,...,M$$

We may, then, solve recursively to get the product form solution for the equilibrium state probabilities:

$$p(\mathbf{n}) = \prod_{i=1}^{M} \left[1 - \frac{\gamma_i}{\mu_i}\right]\left[\frac{\gamma_i}{\mu_i}\right]^{n_i}$$

The delay may, thus, be obtained by summing the delays of the packets as they traverse the queues

$$E(T_i) = \frac{1}{\mu_i - \gamma_i}$$

The mobile component is a major factor affecting the performance due to the lower bandwidth and the significantly higher error rate.

Agent discovery and registration/deregistration with the foreign agent was taken into account as well as the extra overheads due to tunneling. Agent discovery and registration/deregistration may be considered as closed network operations with a single packet circulating through the network. This assumption is due to the fact that no other communications take place before the discovery or registration/deregistration is complete and because during the discovery or registration/deregistration when a request packet is sent the sender is placed on hold until the reply is received. Agent discovery, registration/deregistration and data communications are mutually exclusive since each operation must be completed before any other can start, therefore the analysis may again be separated.

The analytical methods can be applied to the simple network shown in Figure 2 where the packets travel through four wireline links and a wireless link of lower capacity. These methods produce results following very closely the average values obtained from simulation. However, in the case of the larger network configuration the analytical results were well below the averages obtained from the simulations. As discussed in [12] the analytical results become unreliable and begin to deviate from the actual values for networks having more than five nodes.

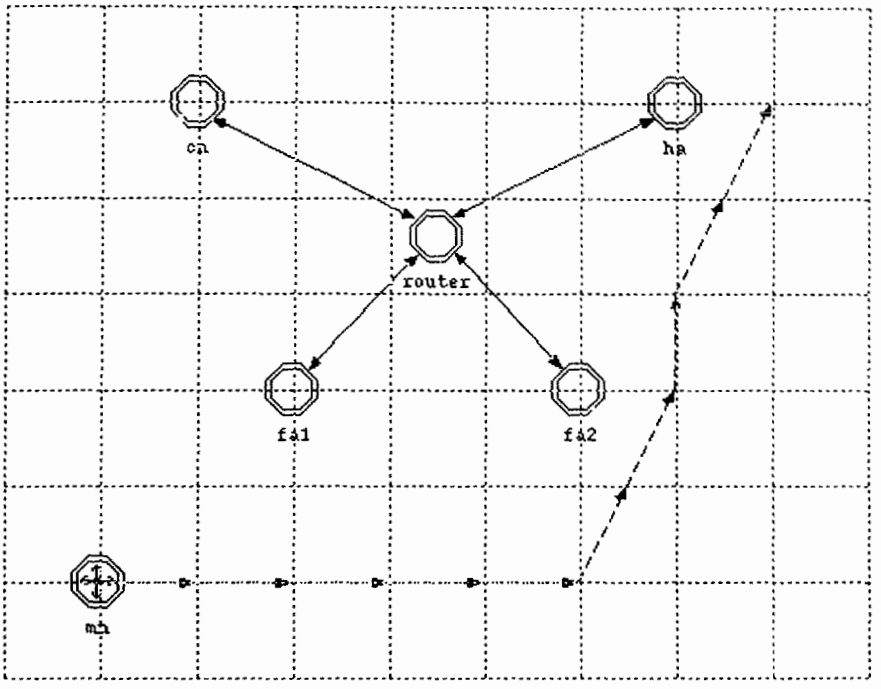


Figure 2. The simple network configuration.

CHAPTER 3

SIMULATION WORK

3.1. Simulation Tools

The OPNET (Optimized Network Engineering Tools) [17] simulation package was used for the simulation part of the project. This package comes with built-in models for TCP/IP and other protocols which eliminated the modeling of the conventional IP component of the simulation. It enabled the concentration of the effort to the modeling of the enhancements proposed by Mobile IP and CDPD.

OPNET's main advantages over other simulation packages are:

- Built-in network models enable us to concentrate on the main subject of the simulation.
- Graphical specification of models give an easier to understand visual representation.

- Hierarchical, object-based modeling breaks down the problem into smaller more manageable segments.

- Friendly user interface creates a more efficient working environment.

The IP layer which comes with OPNET consists of three modules, *ip_encap*, *ip*, and *arp*, as shown in Figure 3. To implement the enhancements of the mobile protocols new modules were added and some of the existing modules were amended in order to properly interface with the new modules. The source code of the amended modules is listed in Appendix D.

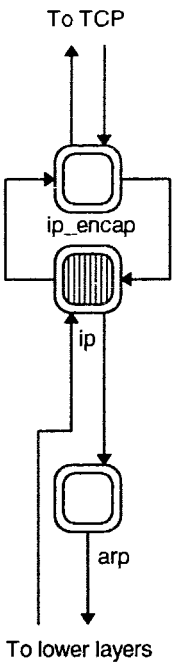


Figure 3. The Node Model of the IP layer taken from the OPNET manual.

3.2. The Mobile IP Model

Two new modules were added to the IP node model provided by OPNET. The IP-in IP module provides encapsulation services and the Mobile IP module performs the registration services. A third module is used to assist for the switching of frequencies and helps for the modeling of handoffs.

3.2.1. The mobility agent

The model of the mobility agent is shown in Figure 4. The wireline interface is

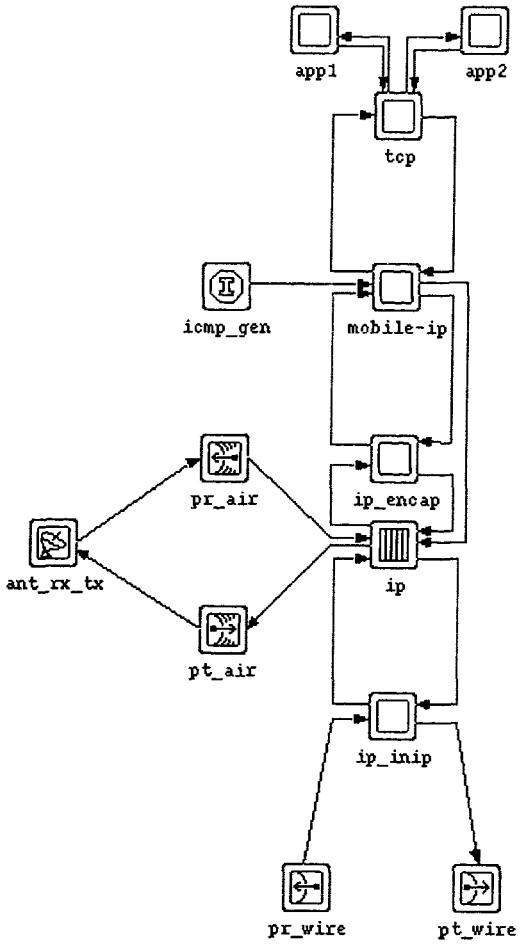


Figure 4. The node model of the mobility agent.

connected to IP through the IP-in IP module (*ipinip*) which encapsulates or decapsulates packets when needed. The wireless interface is connected directly to the IP module (*ip*). For the handling of the registration of the mobile node and the generation of ICMP agent advertisements the Mobile IP module (*mobile IP*) is introduced between IP and TCP. The TCP module (*tcp*) is used as the transport layer to provide reliable, connection oriented service to the applications. The application modules (*app1* and *app2*) handle the inter-arrival and packet size statistics for the simulation.

The process model of the Mobile IP module is shown in Figure 5. The *icmp* state is used to create icmp agent advertisement messages which are sent periodically through the air interface to indicate the availability of the node as a foreign agent. These messages help the mobile node discover this foreign agent and register with its home agent through it. The *xmt* state is used to collect delay information for segments coming from TCP.

The *rcv* state is the most involved in this module. It handles the registration of the mobile node with its home agent and since the mobility agent can be either a foreign or a home agent this state manages both situations. In the case of the foreign agent this state arranges for the registration request message to be forwarded to the home agent and dynamically modifies the routing tables to enable routing to the attached mobile node. In the case of the home agent, this state grants or denies the registration request from the mobile node and also dynamically modifies the routing tables to point to the foreign agent as a proxy to the mobile node.

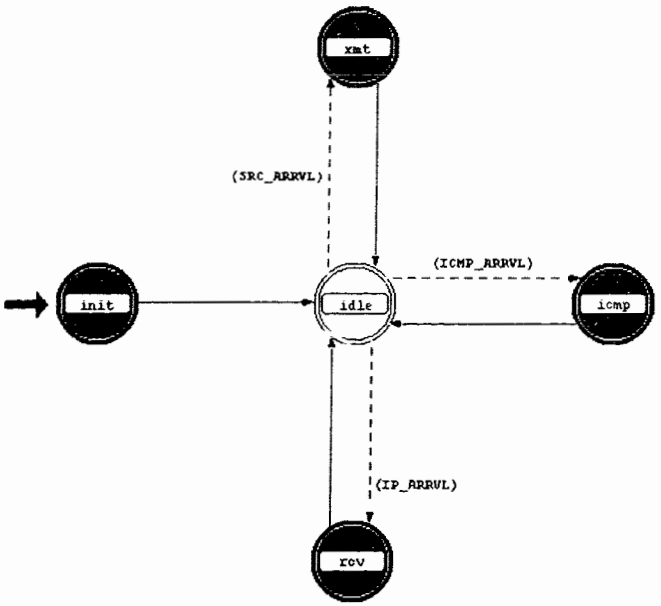


Figure 5. The process model of the Mobile IP module.

The process model for the modified *ip_rte* module is shown in Figure 6. A new state, *tables*, was added to the original module to accommodate the dynamic routing table updates due to mobility. The Mobile IP module sends routing information to this state about the position of the mobile node. The state *svc_compl* was also modified in order to use the dynamic tables.

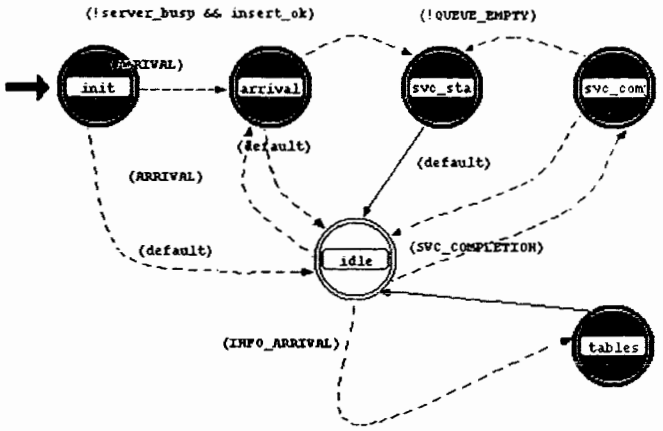


Figure 6. The process model of the modified IP module.

Finally, the state model for the IP-in IP module is shown in Figure 7. This module has two states named *encap* and *decap*. The *decap* state checks if a packet arriving needs decapsulation before passing it to the IP module. The *encap* state, if needed, encapsulates packets arriving from the IP module.

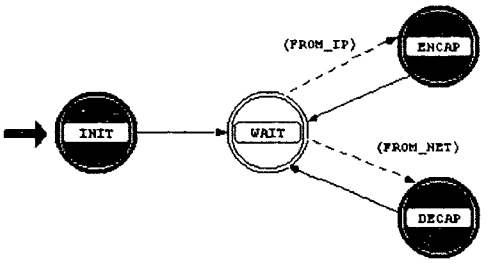


Figure 7. The process model of the IP-in IP module.

3.2.2. The mobile node

The model for the mobile node is shown in Figure 8. This model is simpler than that of the mobility agent, with only one interface, the wireless interface, and does not require an IP-in IP module. However, it has a new module named *freq_ctrl* which manages the frequency of operation of the wireless interface.

The process model for the *freq_ctrl* module is shown in Figure 9. To produce a handoff the mobile node, at predetermined times, sets a number of interrupts. At the interrupt time the *handoff* state arranges for a change of the frequency of operation to match the frequency of the nearest foreign agent.

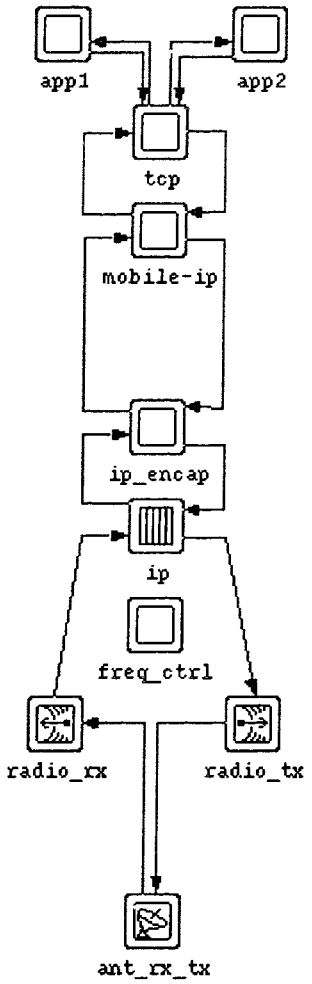


Figure 8. The node model of the mobile node.

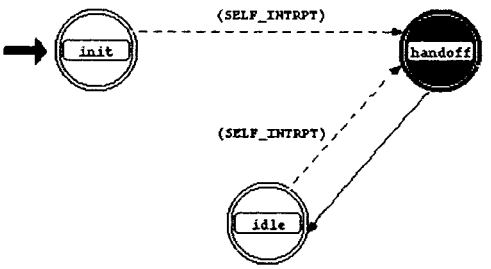


Figure 9. The process model of the frequency control module.

The mobile node has a different Mobile IP module than the mobility agent model and it is shown in Figure 10. The *xmt* state serves the same purpose as

previously but the *rcv* state here is now responsible for initiating the registration once a change is detected. The *timeout* state is used for ensuring that the registration request message is resent if a reply is not received within a certain timeout period.

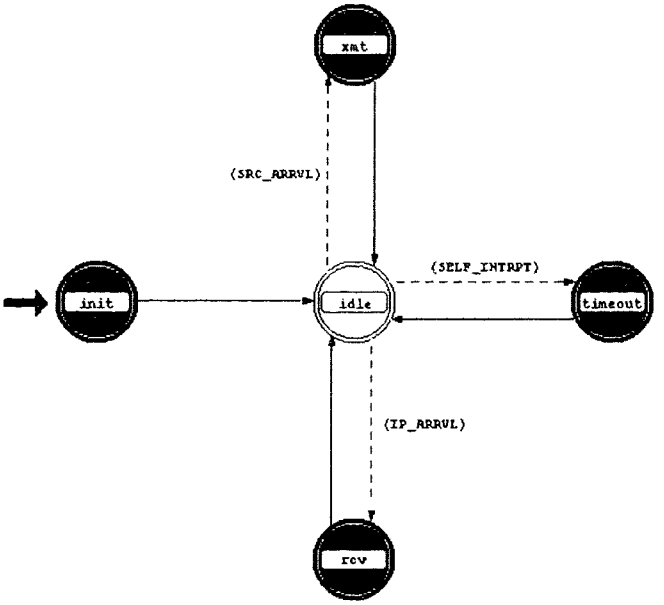


Figure 10. The process model of the Mobile IP module for the mobile node.

3.3. Optimized Mobile IP Model

The optimized Mobile IP protocol has additional functionality and the models for the respective entities are more involved. In addition to the mobile node and mobility agents, the correspondent node is now also aware of mobility and its model has some additional functionality.

3.3.1. The mobility agent

As can be seen from the node model for the mobility agent in Figure 11 the IP module has, now, two way communications with the Mobile IP module. This extra communications path helps to keep the routing tables updated by generating binding warning and binding update messages.

The process model for the Mobile IP module is shown in Figure 12. The *icmp* and *xmt* states are identical to the respective base Mobile IP states. The *info* state handles information received directly from the IP module. This information is an indication of a packet received by the IP module due to outdated routing tables and it triggers transmission of either a binding update or binding warning message.

The *rcv* state again is responsible for managing the registration procedure of the mobile node with its home agent. Furthermore, it is responsible to keep the routing tables of each mobility agent and correspondent node updated by sending and processing binding warning, binding request, binding update and binding acknowledgment messages.

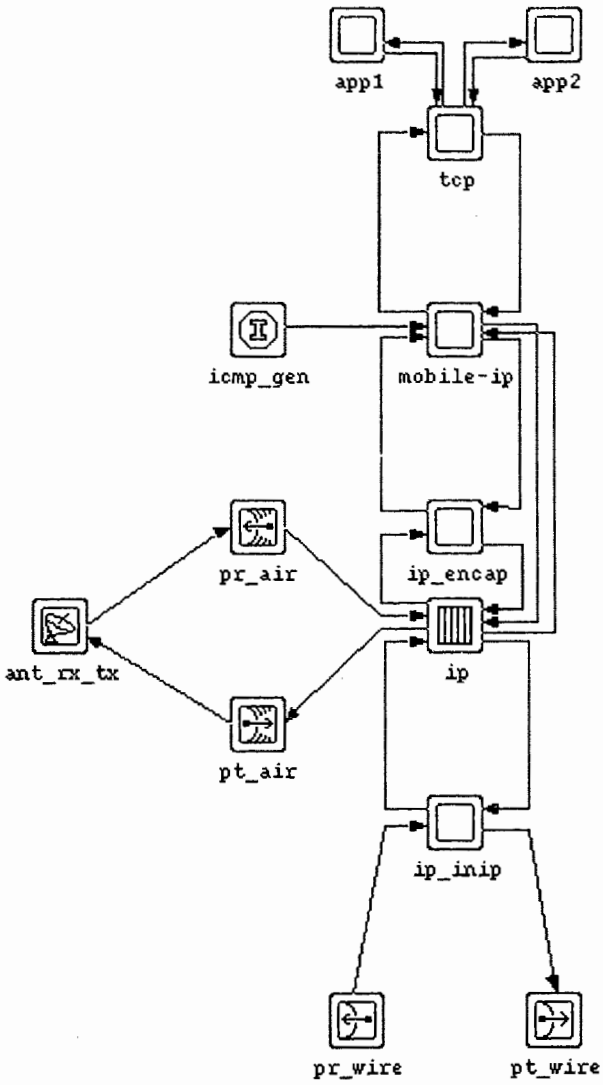


Figure 11. The node model of the mobility agent.

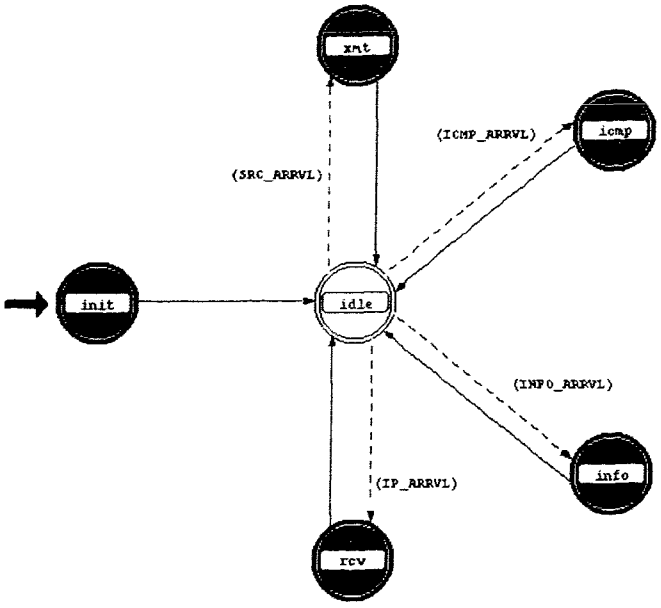


Figure 12. The process model of the Mobile IP module.

The remaining modules perform in a similar manner as the respective modules in the base Mobile IP model.

3.3.2. The mobile node

The mobile node model is almost identical to the base Mobile IP model. The only difference is in the size of the registration messages which now include exchange of authentication keys between the mobile node and its foreign agent. The registration request also includes a request to the new foreign agent to notify the previous foreign agent of its current location.

3.3.3. The correspondent node

The model for the correspondent node is shown in Figure 13. The extra functionality added to this node is found mainly in the Mobile IP module shown in Figure 14. The *rcv* state processes binding update messages sent by the home agent. The routing information, relating the mobile node to a foreign agent, extracted from a binding update message is passed to the IP module to update its routing table.

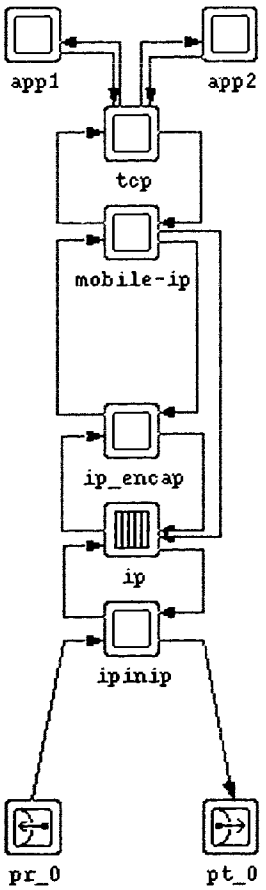


Figure 13. The node model of the correspondent node .

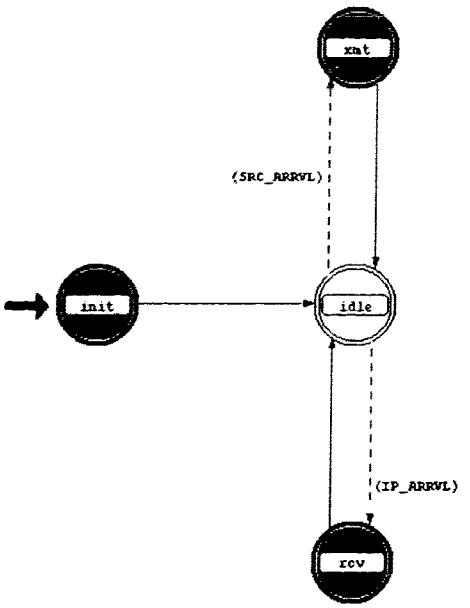


Figure 14. The process model of the Mobile IP module.

3.4. CDPD Model

3.4.1. The MD-IS model

The model for the MD-IS is shown in Figure 15. The wireline interface is very similar to the Mobile IP models and it is connected to IP through the IP-in IP module. The MD-IS does not have a wireless interface but it is connected to an MDBS node through a wireline interface which, in turn, is responsible for the wireless interface. The Mobile IP module found in the previous models is now replaced by a very simple module (*stat*) whose only purpose is statistical data collection. For the communications between the MD-IS and its MDBS we introduced an SNDCP module and the module responsible for registration is called MNRP (*mnrp*). The transport layer used is TCP and the application modules are *app1* and *app2*.

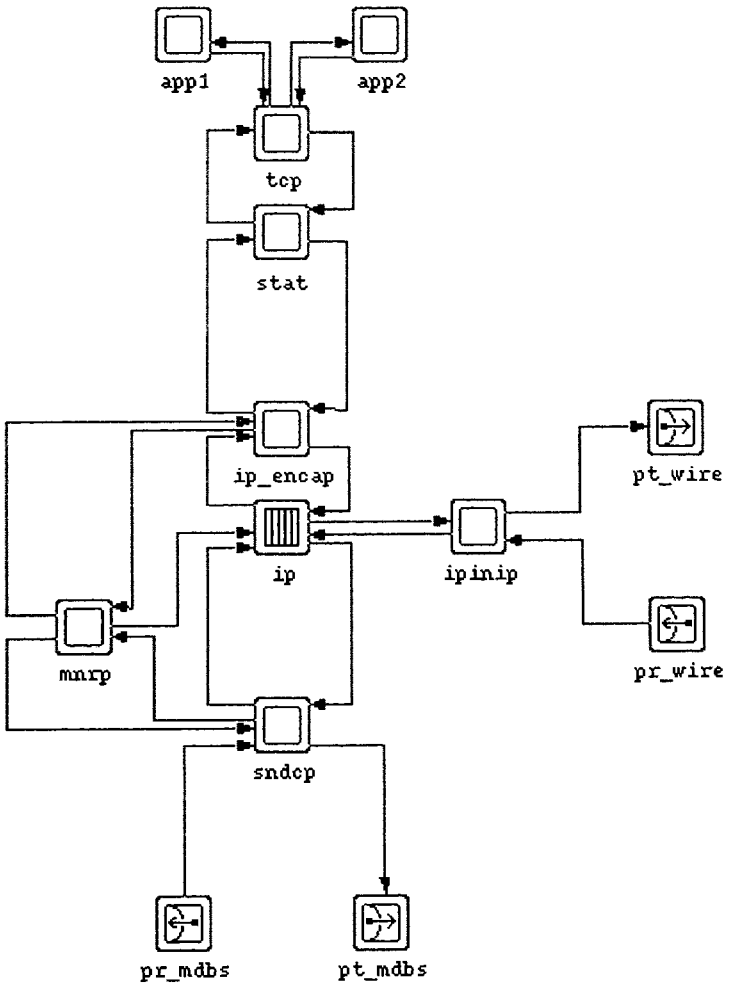


Figure 15. The node model of the MD-IS.

The process model for SNDCP is shown in Figure 16. The purpose of SNDCP (Subnetwork Dependent Convergence Protocol) is to ensure compatibility of the IP layer with the lower CDPD layers. SNDCP is, therefore, a passive module forwarding the packets or registration messages in the correct format to their destination.

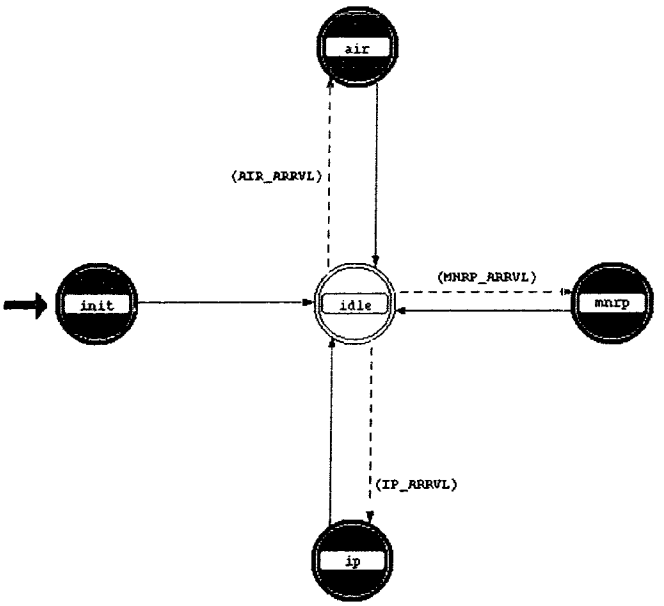


Figure 16. The process model of the SNDCP module.

The module MNRP, shown in Figure 17, is performing the most difficult tasks of the registration and hosts the Mobile Home and Mobile Foreign Functions. State *reg* is entered when an End System Hello arrives from the M-ES through the SNDCP module. This state, then, prepares a Redirect Request message and sends it to the home MD-IS. It, therefore, represents part of the Mobile Foreign Function.

State *ip* is entered when a registration message arrives from the network. If the received message is Redirect Request then the Mobile Home Function prepares and sends a Redirect Confirm message to the sender MD-IS node. If the received message is a Redirect Confirm message the Mobile Foreign Function processes the message and sends an Intermediate System Confirm message to the M-ES through the SNDCP module and the MDBS node.

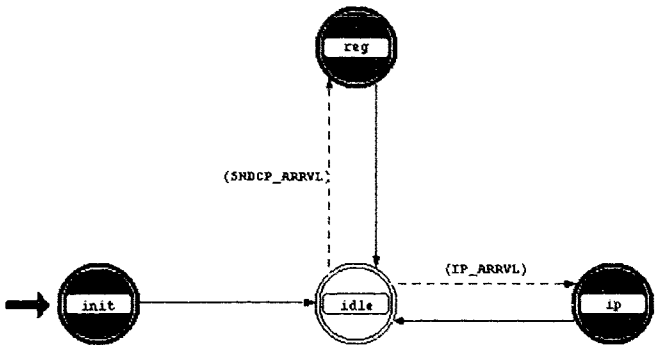


Figure 17. The process model of the MNRP module.

3.4.2. The MDBS Model

The model for the MDBS is shown in Figure 18. The MDBS is acting as a layer two relay between the MD-IS and the M-ES and is managing the wireless interface. There is only one important module in this model, the Radio Resource Management (*rrm*) module which simply relays the messages passing through it.

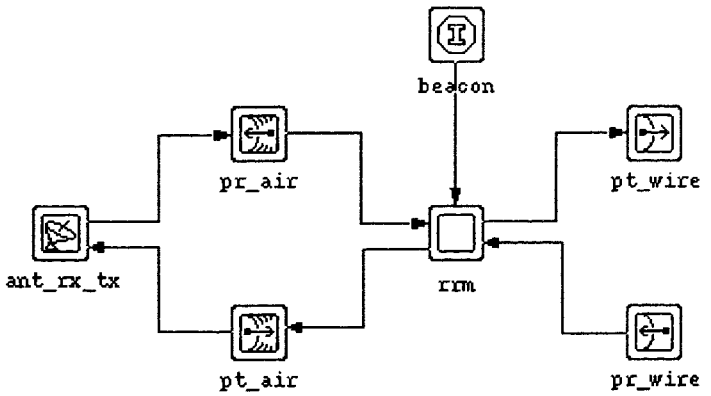


Figure 18. The node model of the MDBS.

3.4.3. The M-ES model

The model for the Mobile End System is shown in Figure 19. This model appears simpler than the model of the MD-IS in an analogous manner that the mobile node model is simpler than the mobility agent model. The model includes the frequency control module which is identical to the module found in the Mobile IP model. The only difference is found in the MNRP (*mnrp*) module shown in Figure 20.

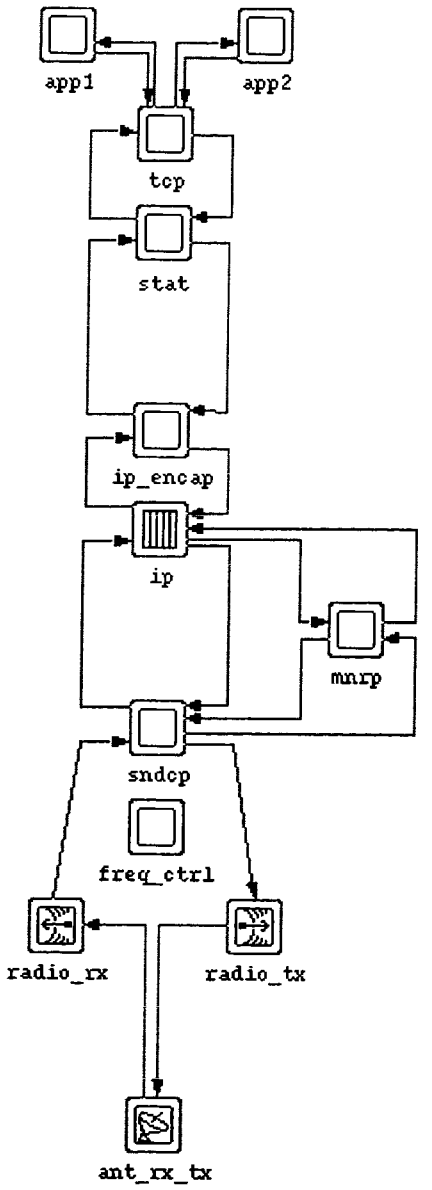


Figure 19. The node model of the M-ES.

If a beacon message is received indicating a change of cells the *reg* state is entered. This state prepares an End System Hello message and sends it to the MD-IS serving this cell and waits for an Intermediate System Confirm message indicating the completion of the registration with its home MD-IS. After sending the End System Hello message this state also sets a timeout interrupt to ensure that the registration message is not lost. The interrupt is handled by the state *interrupt* and if there is no reply the message is resent.

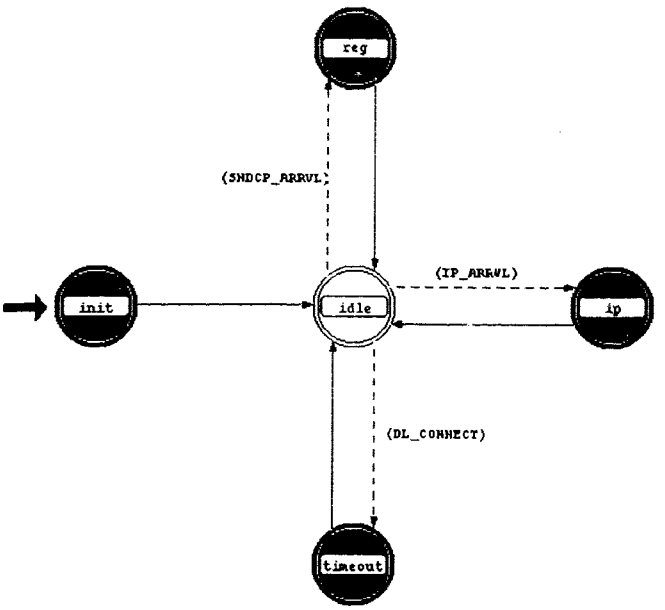


Figure 20. The process model of the MNRP module.

3.5. The Simulation Networks

Two different network configurations were used for the simulations. The simple configuration was shown in Figure 2. It consists of the mobile node (*mn*), two foreign agents (*fa1* and *fa2*), a home agent (*ha*) and a correspondent node (*cn*) interconnected with a router node (*router*). The mobile node roams through two foreign agent and,

finally, reaches its home network.

The second network configuration is shown in Figure 21. It has more intermediate routers so the mobility effects are not as clearly obtainable as in the simpler network configuration but a more realistic view of the overall internetwork is obtained.

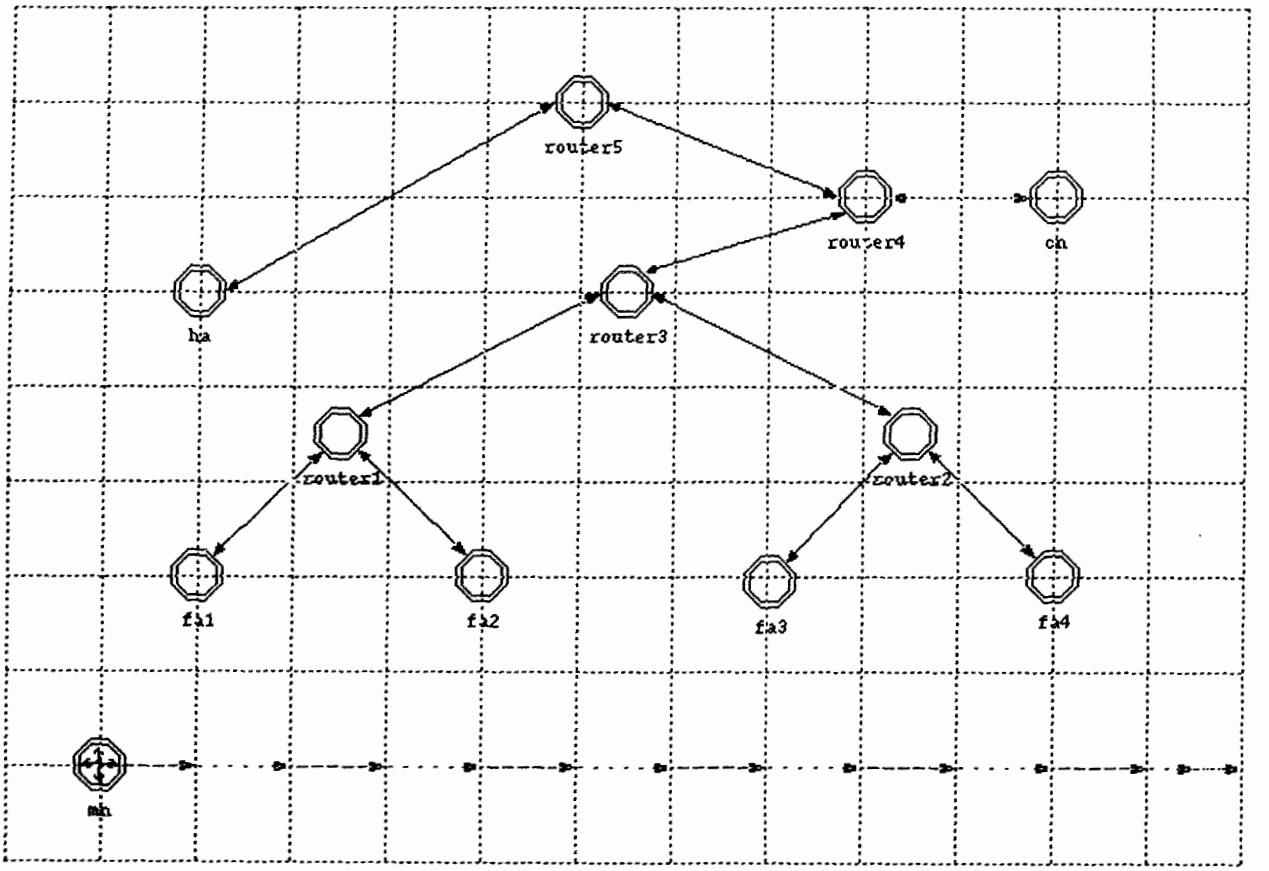


Figure 21. The larger network configuration.

CHAPTER 4

RESULTS

The network model used for most of the simulations was shown in Figure 2. The mobile node begins its operation in a foreign network and moves towards home passing through a second foreign network. There is, therefore, handoff between two foreign agents and between a foreign agent and the home agent. The correspondent node is connected to the internetwork through a router.

The larger network model, shown in Figure 21, was also used for a simulation round of the four protocols. During the duration of this simulation the mobile node traverses through the four different foreign agents or serving MD-IS's.

For all simulations the links joining the nodes in the wireline networks had a 56,000 bps speed and the airlink speed was 12,000 bps. The 56,000 bps value is common to wide area links and the 12,000 bps was chosen for the airlink to account for the signaling, synchronization, error correction and other administration overheads.

The constant or deterministic interarrival times and packet sizes were chosen for the first part of the simulations because they indicate more clearly the effects of the handoff for each protocol. In this case, the interarrival time was chosen to be 0.27 seconds and the packet size send by the application was chosen to be 1024 bits. The size of 1024 bits was selected because it is the average packet length in the ARPA network [18] and the interarrival time of 0.27 seconds was selected because as the minimum interarrival time which does not cause the network connection to break down during the handoffs, since according to the simulation results, the end to end delay of the packets is 0.27 seconds.

The size of the packet leaving the IP layer was increased by the addition of 160 and 184 bits of the TCP and IP headers respectively. There is generally no queueing in this case except when a packet needs to wait while the beacon packet is transmitted. This very small queueing delay may be seen as the regular pattern of dots slightly above the average ETE delay line in figures 22-25.

Poisson arrivals and exponential packet size distributions were also used for a different set of simulations. In this case the mean interarrival time was set to 0.4 seconds and the mean packet size was set to 1024 bits. The packet size was again selected because of its similarity to the average packet size in the ARPANET and the interarrival time was selected to produce close to maximum throughput but without producing excessive delays or connection break down in any protocol. This size is increased by 160 and 192 bits by the TCP and IP layers respectively.

The large network, shown in Figure 21, was also simulated with each of the 4 variations of the protocols. The Poisson/exponential distributions were used with interarrival times of 0.6 seconds and mean packet size left at 1024 bits. The interarrival time was increased from the value of 0.4 seconds in order to operate close to maximum throughput in this network configuration with the additional queueing and transmission delays.

The cases above were investigated assuming an error-free link. Simulation runs were also performed using a calculated error rate resulting after Reed-Solomon error correction is performed. Using analytical methods and simulation the error interarrival rate was shown to be approximated by a Poisson distribution.

Finally, several parameters were varied and their effect on the performance of the protocols was investigated. Specifically, the following parameters were varied:

- Minimum RTO (retransmit timeout) parameter of TCP
- Packet size for both constant and Poisson simulations
- Interarrival time for the Poisson simulation
- Handoff occurrence with respect to the beacon arrival
- Error rate

4.1. Constant interarrival times and packet sizes

4.1.1. TCP segment end-to-end (ETE) delay

The end-to-end delay of the TCP segments is shown in figures 22-25. The handoffs for the Mobile IP protocols occur at the same time as for the inter-area handoffs of CDPD. In addition, for CDPD the home MD-IS and the second MD-IS were modified to combine the functionality of the MDBS in the same module. This eliminates the extra delay due to the link between the MD-IS and its MDBS and enables a fairer comparison of Mobile IP and CDPD. Figures 46-49 in Appendix B show the results obtained when CDPD uses separate MD-IS and MDBS nodes for each subnetwork.

Of special interest in the following graphs is the activity during the handoff and the ETE delay at the steady state, i.e. when the mobile node is operating within a single cell.

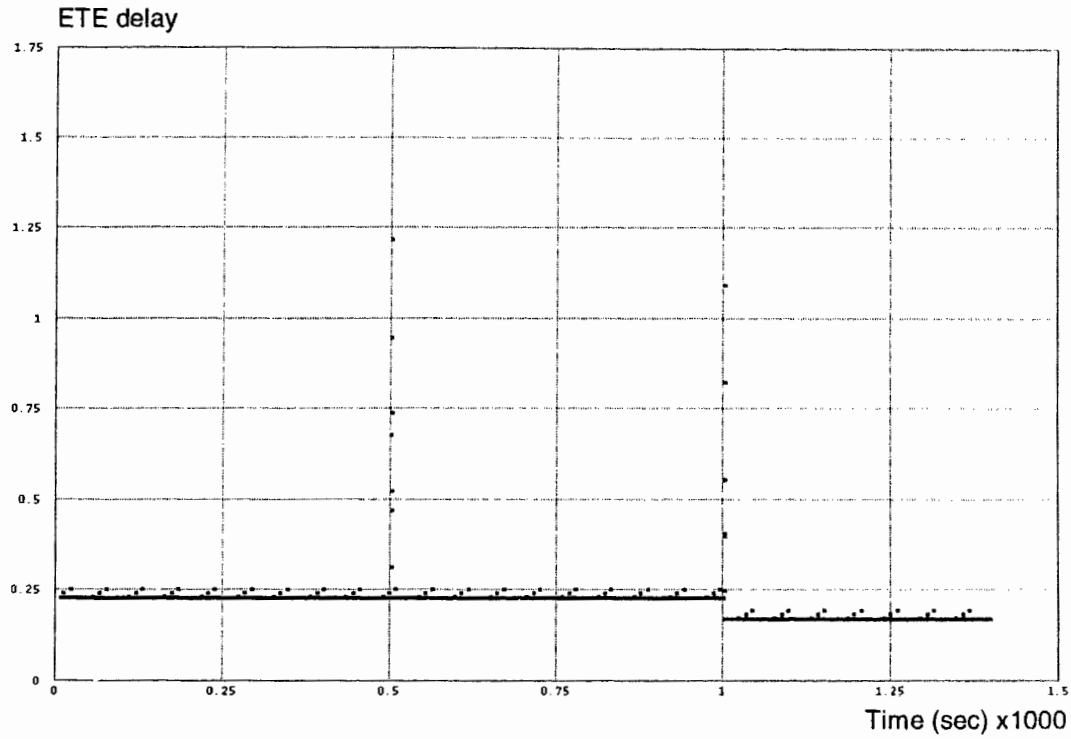


Figure 22. Base Mobile IP ETE delay.

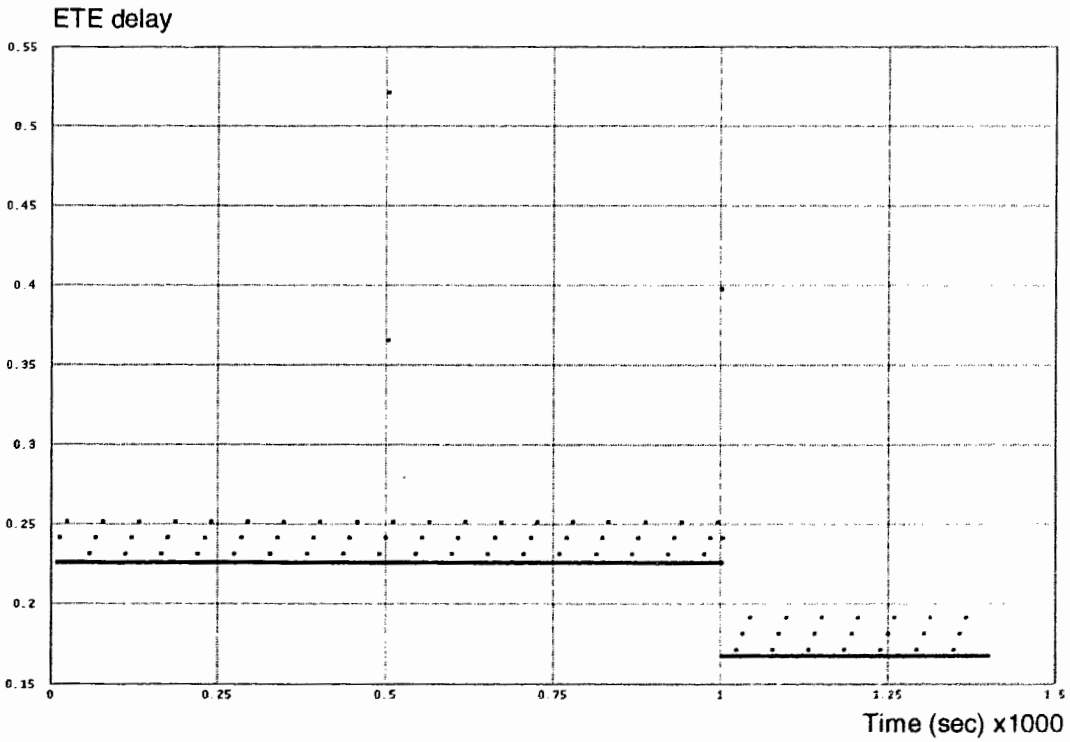


Figure 23. Base Mobile IP with simultaneous bindings ETE delay

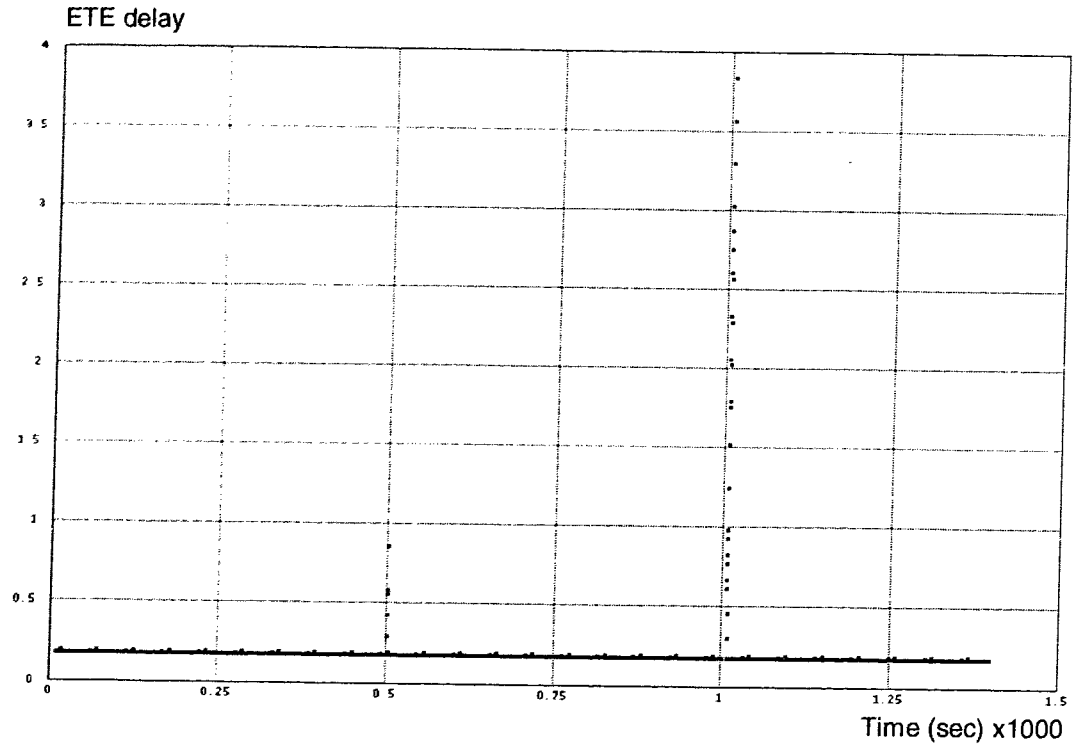


Figure 24. Optimized Mobile IP ETE delay

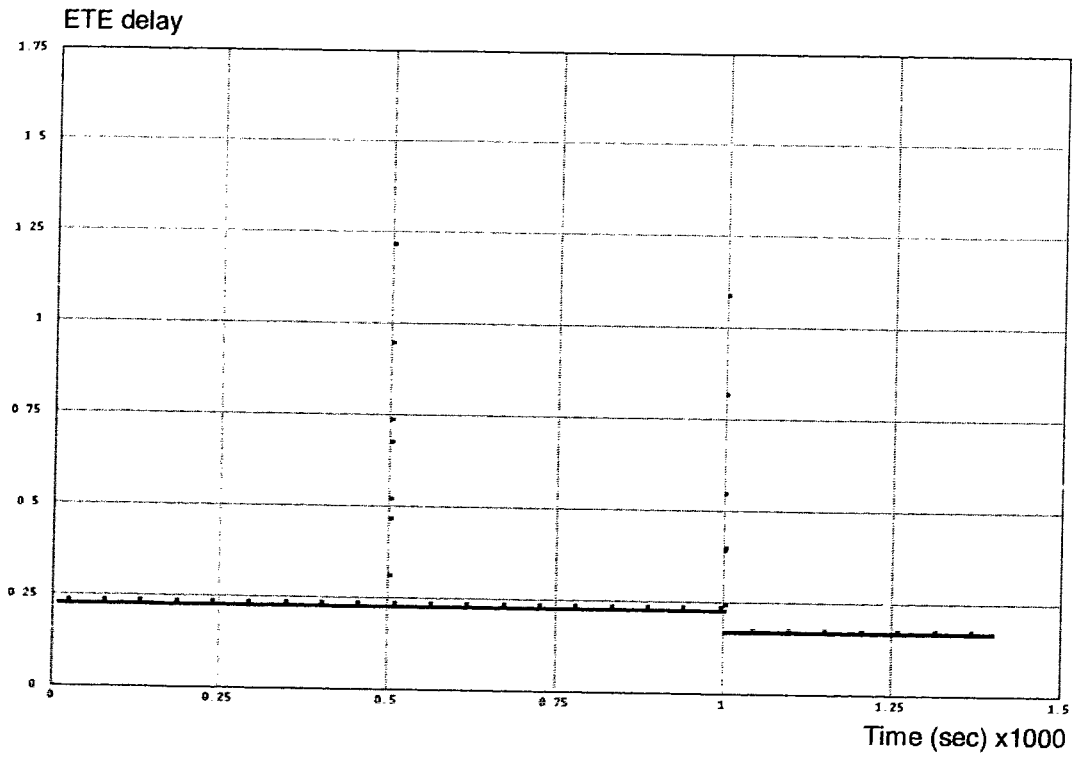


Figure 25. CDPD ETE delay

4.1.2. Registration packet delay

The delay of the registration packets for figures 22-25 is shown in Table 1. This data presents the overhead introduced by each protocol during handoff and hence provides some indication of their handoff performance .

Table 1. Registration delays.

		Network delay	Airlink delay	Total delay
Base Mobile IP		0.0293	0.0634	0.0927
Mobile IP - Multiple Bindings		0.0293	0.0634	0.0927
CDPD	REG_REQ	0.0210	0.0227	0.0437
	REG_REPLY	0.0159	0.0234	0.0393

4.1.3. Comments on the results

From the plots we can conclude that the main benefit of the optimized Mobile IP protocol is the reduced ETE delay under normal operation. The larger registration messages, as indicated in Table 1, appear to increase the packet delay during handoff. The provision for forwarding packets in transit during handoff does not appear to have any effect in the handoff performance.

The Mobile IP with multiple bindings showed significant improvement in handoff performance over the base Mobile IP. In spite of the uninterrupted communications of the mobile node with the network through either the old or the new foreign agent the handoff

still introduces extra delay to some packets. This is mainly due to the extra queueing delay of the registration messages passing through the airlink.

The CDPD intra-area handoff passes unnoticed to the higher layers, as expected, since the registration message is very small and arrives at the MD-IS in only 5.5 ms. This message only travels from the M-ES to the serving MD-IS.

In figures 22 and 25 the ETE delay during handoff of CDPD is identical to the ETE delay of base Mobile IP. In figures 46 and 49 of Appendix B, however, the CDPD handoff appears to add greater delay to the TCP segments than the Mobile IP handoff. This is due to the additional link between the MD-IS and the MDBS in the case of CDPD and, although this is a very high speed link it introduces additional transmission and processing delay to each packet.

Under steady state all the protocols appear to be similar except in the case of the optimized Mobile IP. In this case, the optimal routing reduces the number of links traversed by the packets and improves the ETE delay significantly. All the protocols, including the optimized Mobile IP, show similar performance when the mobile node/M-ES is at home since no mobility functions are required.

4.2. Poisson arrivals and exponential packet size distribution

The interarrival time for the common simulation runs has a mean of 0.4 seconds and the packet size has a mean of 1024 bits. The handoffs occurred at the same times as for

the simulations in section 4.1 and the results are shown in figures 50-57 in Appendix B.

It is more difficult to arrive to conclusions about the handoff handling from the Poisson/exponential simulations due to the randomness of events and the interaction of the TCP traffic with the beacon messages. After examining the statistics, however, we verify the conclusions from the previous section about the similarity of the two base protocols Mobile IP and CDPD.

We can also apply the analytical formula from section 2.4 in order to verify the correctness of the average ETE delay obtained from the simulations. Even though the analytical methods assume Poisson traffic and the traffic entering TCP is indeed Poisson the TCP layer slightly changes the packet length characteristics with the addition of the headers.

Using the equation for the average delay in each node, $E(T_i) = \frac{1}{\mu_i - \gamma_i}$, we calculate the average ETE delay to be 0.271 seconds which is very close to the mean value of the delay in the simulations. The analytical solution is slightly lower than the simulation result but this deviation can be explained by considering the larger delays during handoffs.

4.3. Large Network Configuration

The network shown in Figure 21 was used for the simulations in this section. The mobile node performs three handoffs with foreign agents.

The data shown in figures 58-61 in Appendix B indicate even greater improvement of the optimized Mobile IP protocol over all the other protocols in spite of the greater delay during the third handoff at time 1001.9 seconds.

4.4. Performance under error conditions

The distributions of fade duration and the inter-fade intervals for a Rayleigh fading channel were obtained from [19] and [20]. A sample signal was generated using the distributions in [20] and then passed through a Reed-Solomon error correcting simulator. The simulator scanned the 378 bit frames for consecutive errors and indicated the words in error. Appendix C has details on the error calculations. It was found that the distribution of the words in error could be approximated by a Poisson distribution. The Poisson distribution was then used in OPNET for the simulation of error conditions during the simulation.

4.4.1. ETE delay for two error rates

Two error rates, 0.1 and 0.05 frames in error per second, were used to obtain the data in figures 26 and 27 respectively.

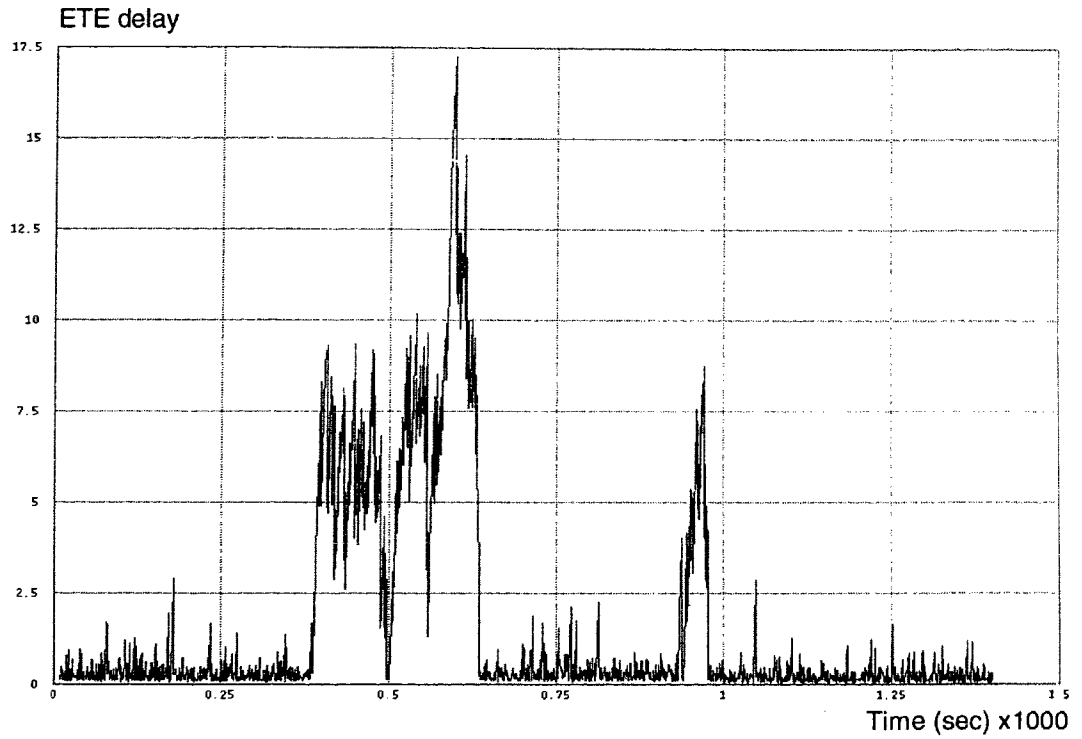


Figure 26. Base Mobile IP ETE delay with average error rate 0.1 sec^{-1}.

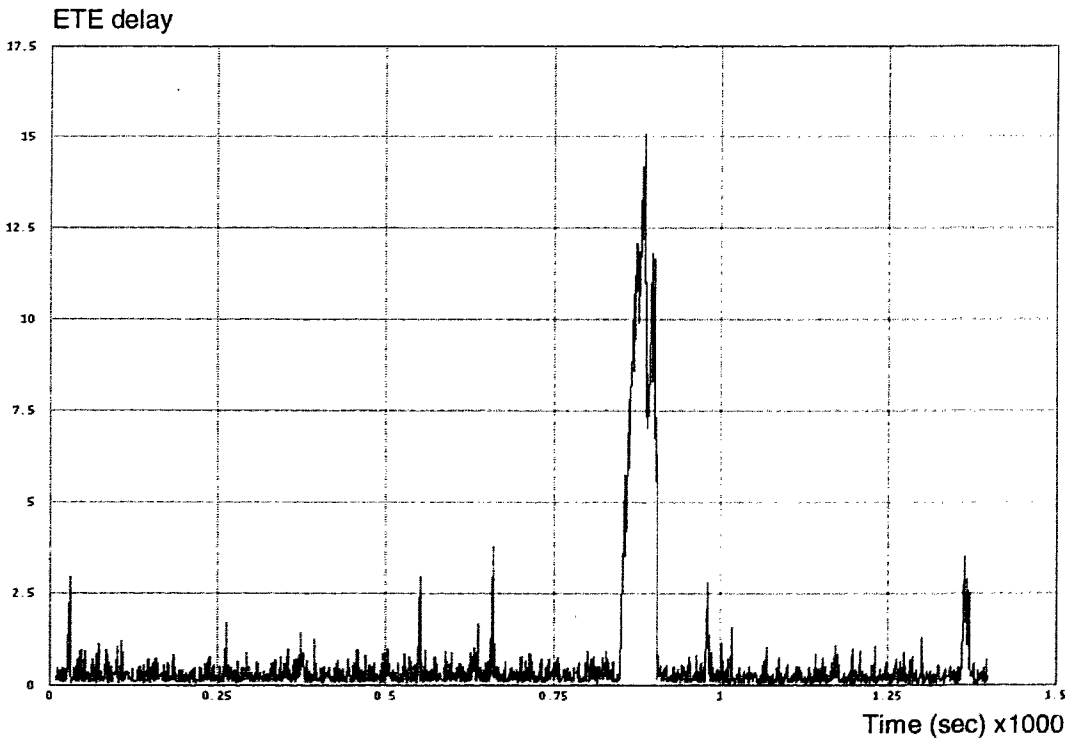


Figure 27. Base Mobile IP ETE delay with average error rate 0.05 sec^{-1}.

4.4.2. Variation of error parameters

The error interarrival was varied from 2 to 80 seconds and the average ETE delay is shown in Figure 28. The ETE delay as function of time for error interarrival rates of 36 and 54 seconds is shown in figures 29 and 30. These two rates were chosen in order to investigate the discrepancy shown in Figure 28.

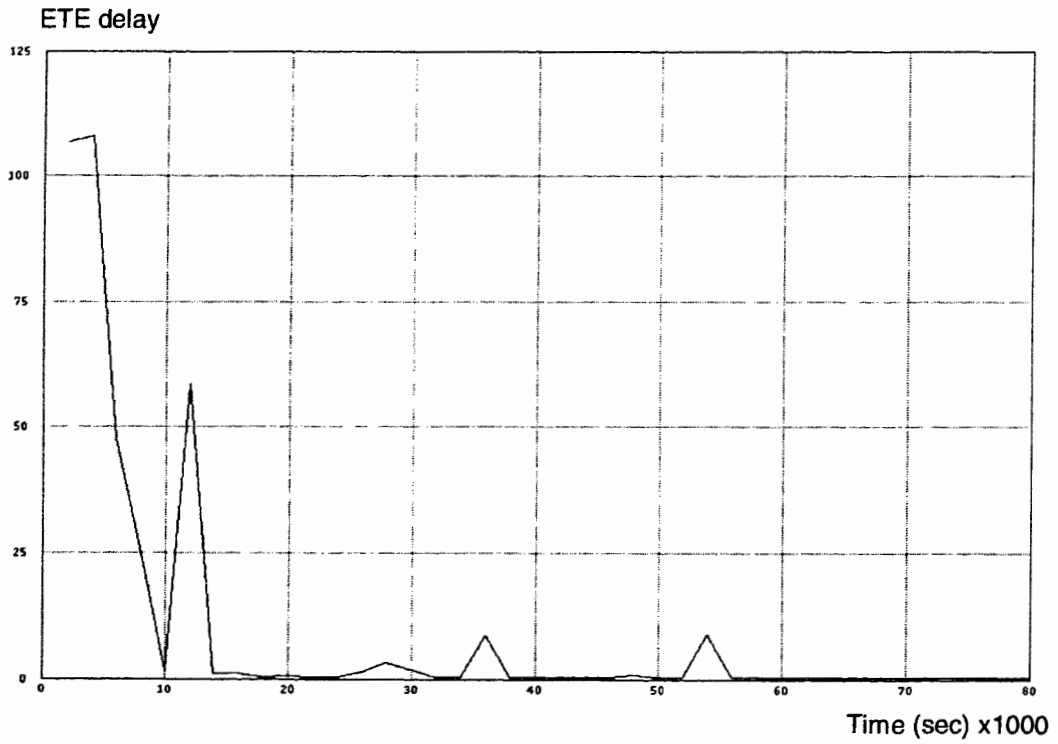


Figure 28. Base Mobile IP average ETE delay with average error rates ranging between 0.0125 and 0.5 sec^{-1}.

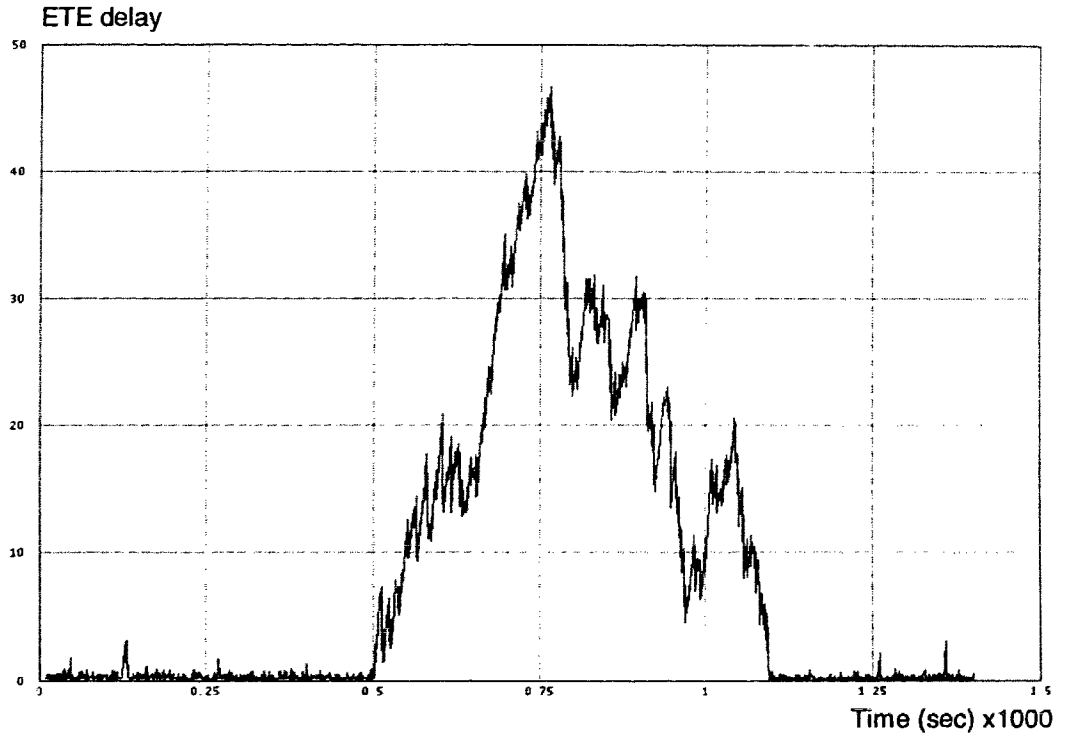


Figure 29. Base Mobile IP ETE delay with average error rate 0.0278 sec^{-1}.

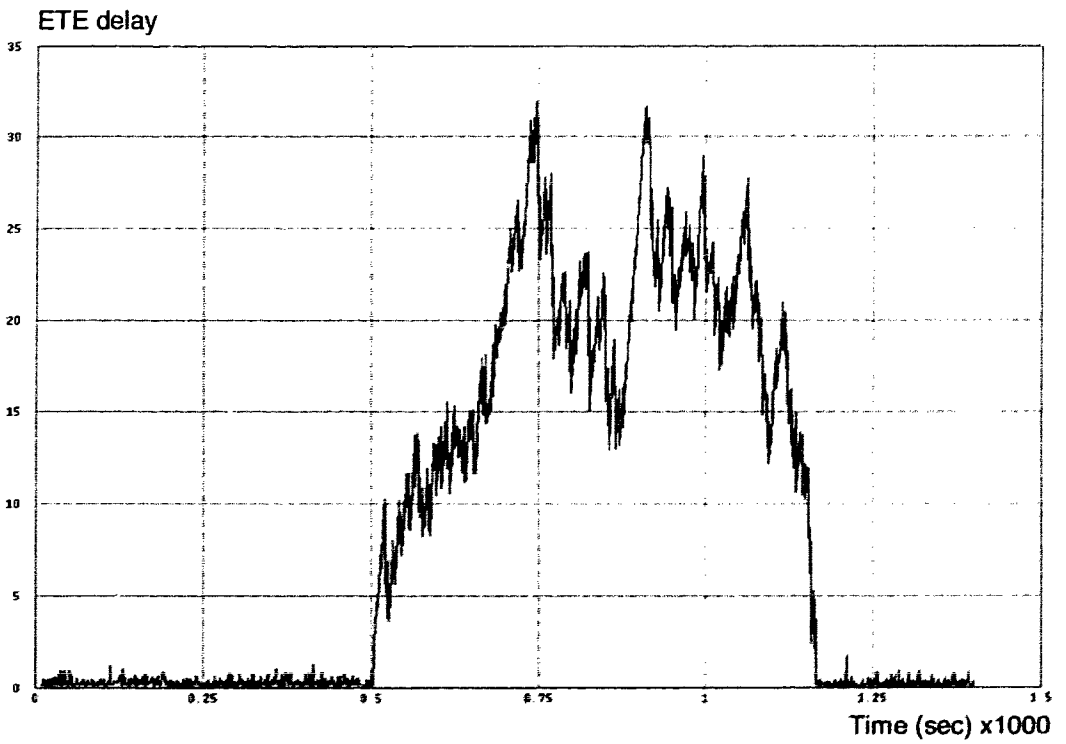


Figure 30. Base Mobile IP ETE delay with average error rate 0.0185 sec^{-1}.

4.4.3. Comments on the results

From the data shown in figures 28-30 we can conclude that with an error interarrival rate of 0.05 sec^{-1} or slower the average ETE delay is 'bearable' at about 0.4 seconds but it may reach an unstable state at certain rates. With error interarrival rates slower than 0.025 sec^{-1} the delay is 'acceptable' and approaches the average ETE delay of 0.3 seconds.

4.5. Variation of TCP and traffic parameters

In most cases, selecting the appropriate TCP parameters is the key to the correct and efficient operation of the mobile protocols. By selecting slightly different parameters the connections may break down or become too slow to use. In Figure 31 and figures 62-64 in Appendix B we present the ETE delay under varying TCP parameters.

The traffic is another factor affecting the operation of the protocols. Heavy traffic would create a congested link and cause the connection to timeout and possibly cause the connection to break down. Figures 32-35 and figures 65-68 in Appendix B show the ETE delays under varying traffic parameters. Related to the traffic parameters is the handoff occurrence with respect to the next beacon message arrival. Figure 36 and Figure 69 show the ETE delay response to the various handoff times.

4.5.1. Variation of the minimum Retransmission Timeout (RTO) parameter

When the minimum RTO was varied the results showed an increase in the maximum ETE delay as the minimum value of the RTO is increased. This is

caused by the longer period it takes for TCP to realize that a handoff has occurred as the timeout value is increased. The average ETE delay however appears to reach a minimum value not at the lowest value of minimum RTO but at time 1 second as shown in Figure 64. This is explained by the trade-off between faster discovery of handoff and a larger value of the congestion window (*cwnd*) parameter. A larger congestion window increases throughput by allowing more packets to be sent without acknowledgment.

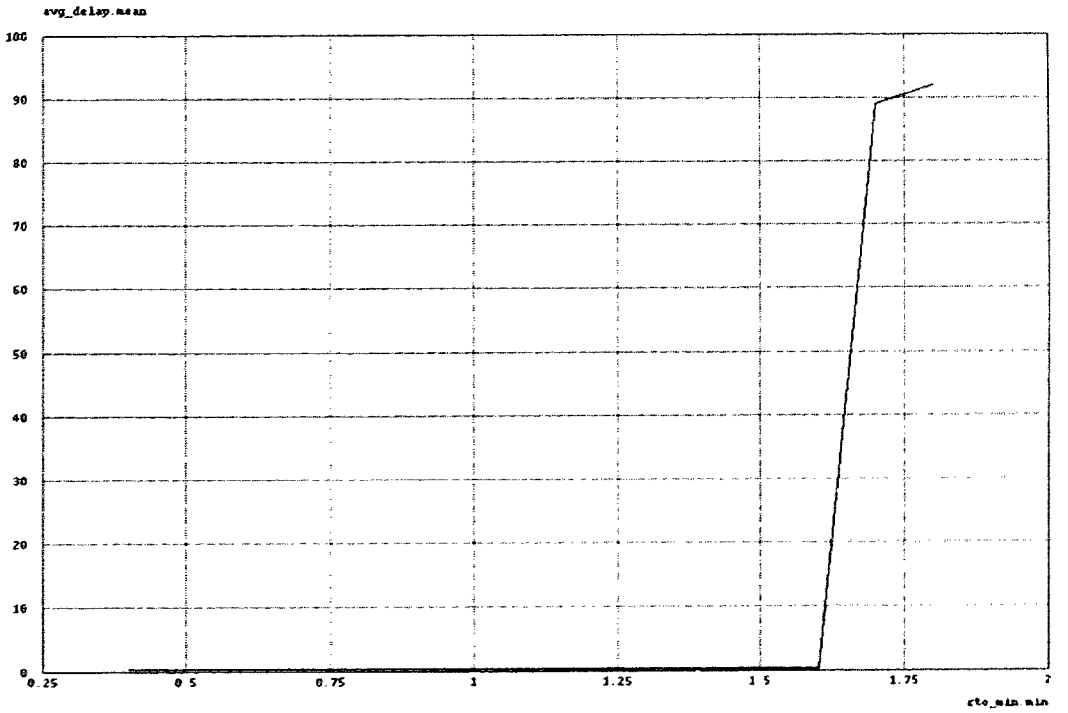


Figure 31. Base Mobile IP average ETE delay with minimum RTO values ranging between 0.4 and 1.8.

4.5.2. Variation of packet size

The packet size plays a direct role in the ETE delay. The larger packets take longer to reach to their destination and they have higher probability of being discarded as having too many errors. The plots of the ETE delay with varying packet sizes are shown in figures 32-33 and figures 65-66 in Appendix B for the Poisson distributions.

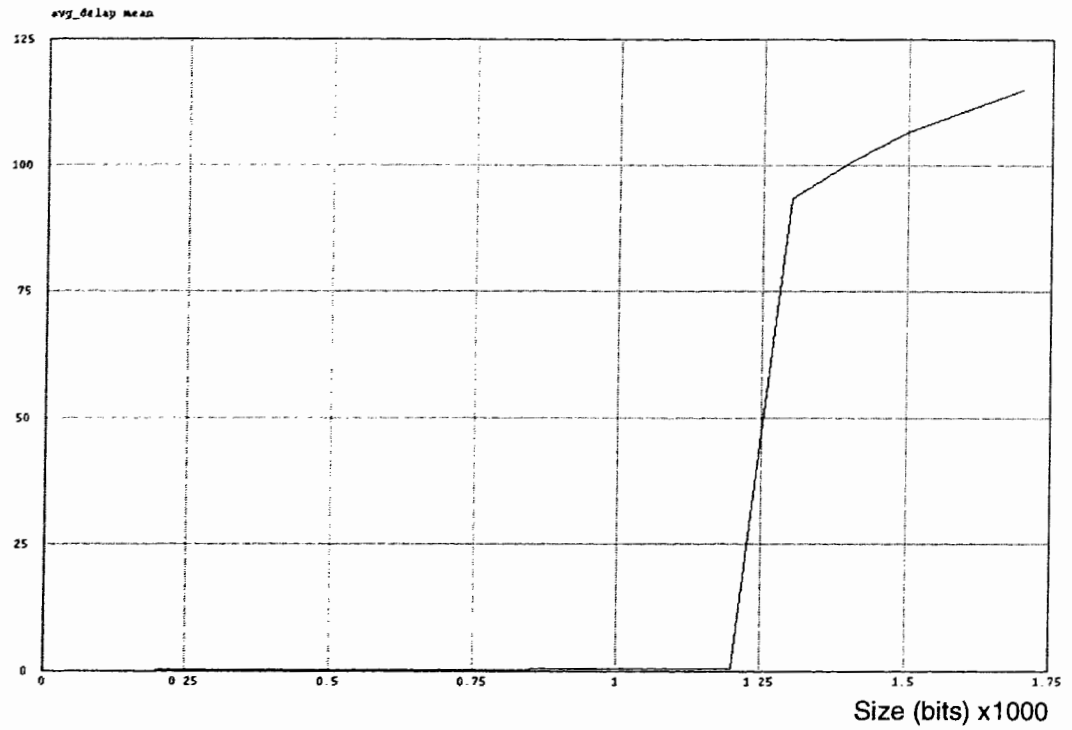


Figure 32. Base Mobile IP average ETE delay (Poisson) with packet sizes ranging between 200 and 1700 bits.

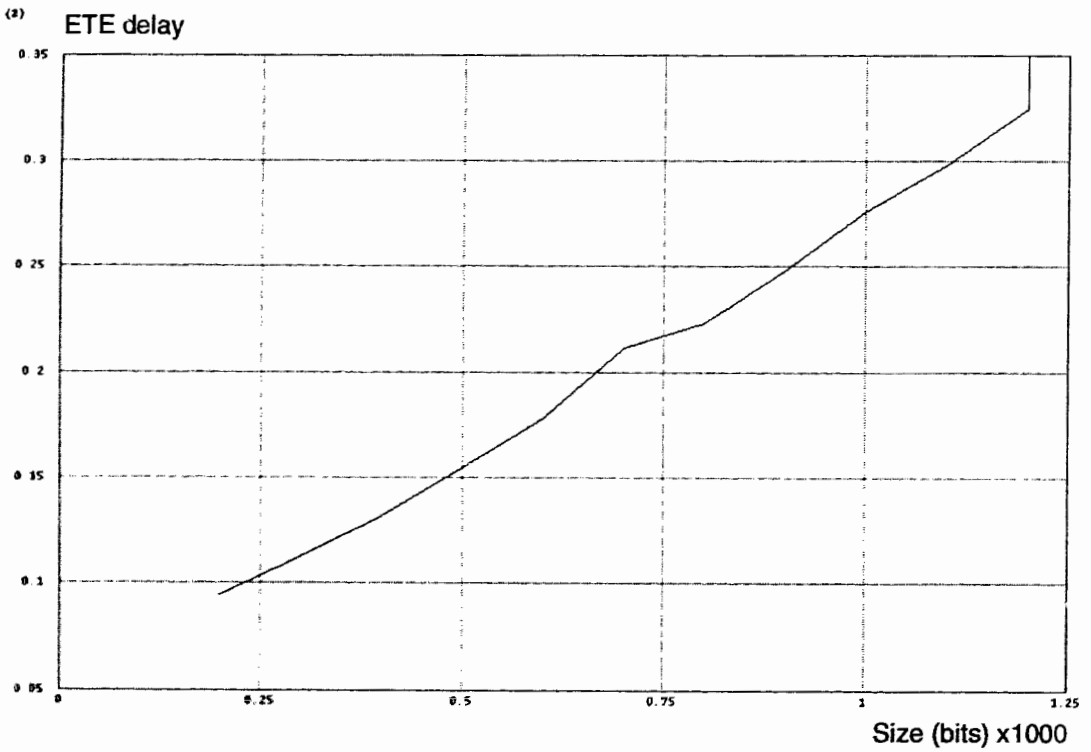


Figure 33. Base Mobile IP average ETE delay (Poisson) with packet sizes ranging between 200 and 1200 bits.

4.5.3. Variation of packet interarrivals

The packet interarrival interval play a direct role in the ETE delay similar to the packet size. The shorter interarrival times increase congestion until connections begin breaking down. The plots of the ETE delay for various average interarrival times (Poisson distribution) are shown in figures 34-35 and figures 67-68 in Appendix B.

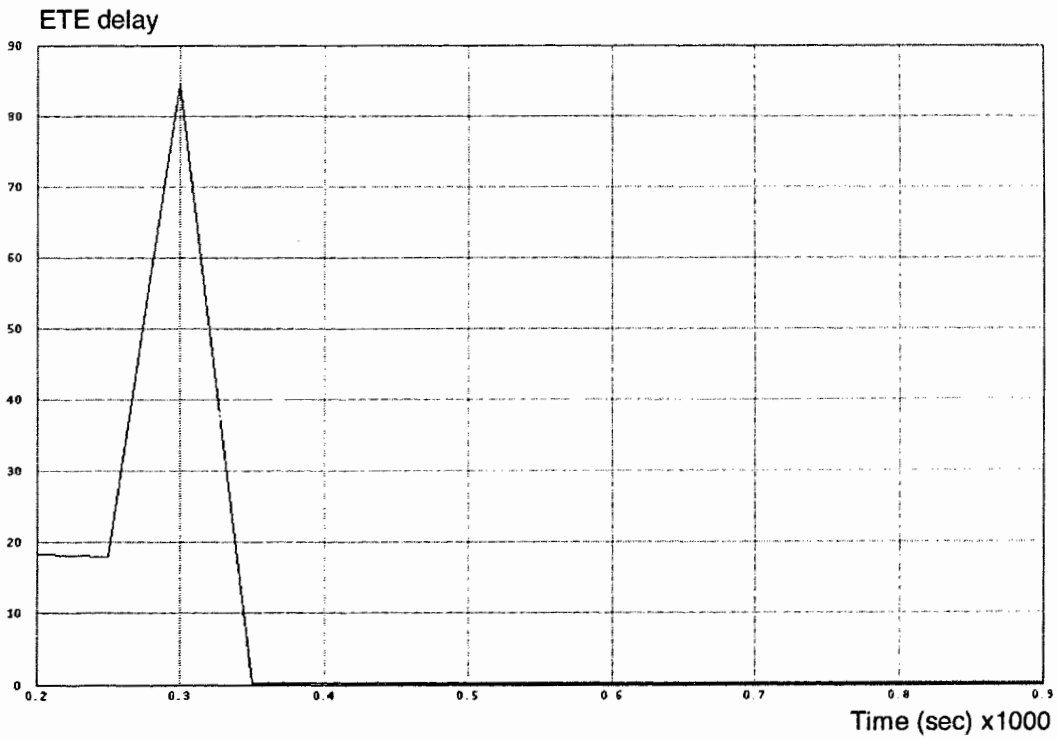


Figure 34. Base Mobile IP average ETE delay with average interarrival times ranging between 0.2 and 0.9 seconds.

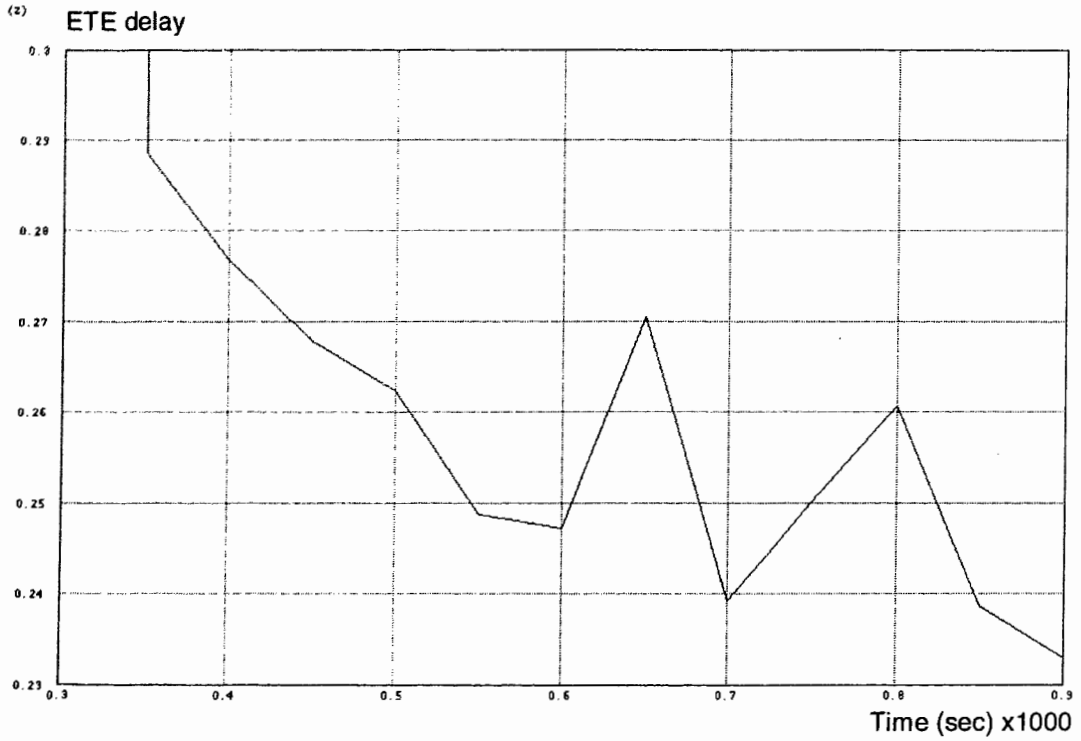


Figure 35. Base Mobile IP average ETE delay with average interarrival times ranging between 0.35 and 0.9 seconds.

4.5.4. Handoff Time

Finally, the handoff time was varied with respect to the next beacon arrival. Figure 36 and Figure 69 in Appendix B show the maximum and average ETE delay for the range of handoff times. As can be clearly seen from Figure 36 and Figure 69 in Appendix B the period between the handoff occurrence and the arrival of the first beacon/ICMP message also affects the performance of the protocol.

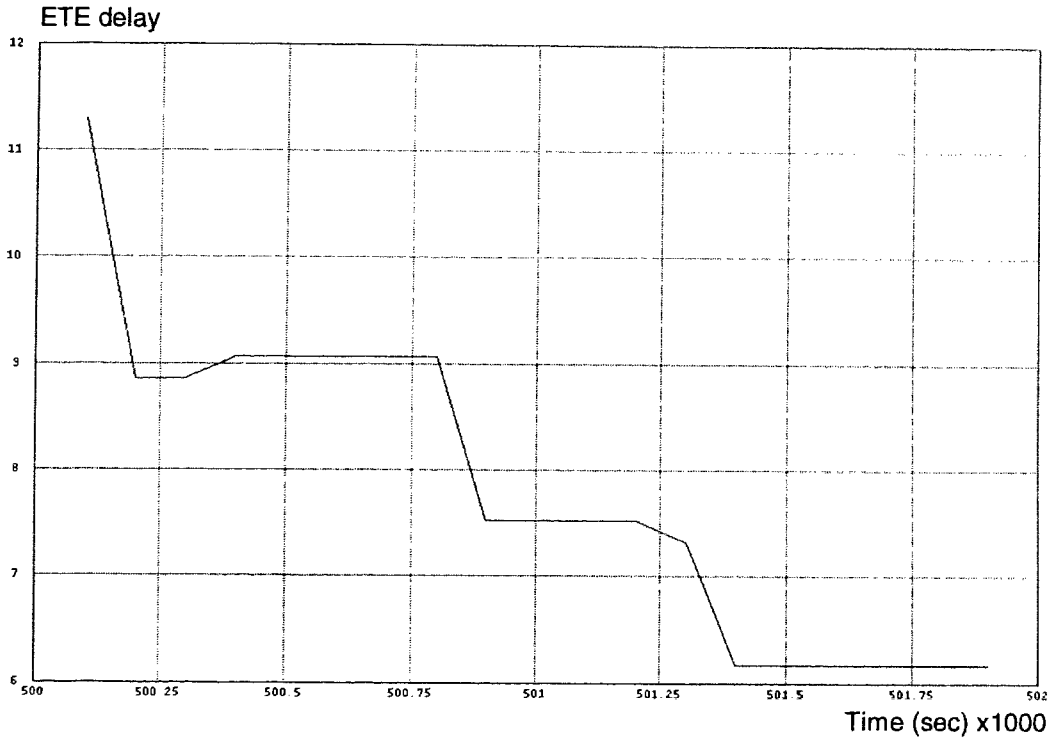


Figure 36. Base Mobile IP maximum ETE delay with handoff times ranging between 0.1 and 1.6 seconds before the beacon arrival.

CHAPTER 5

CONCLUSIONS

5.1. Protocol Comparison

The base Mobile IP and CDPD operating under similar conditions perform almost identically. To discover the differences we need to investigate special situations such as the intra-area cell transfer for CDPD and simultaneous bindings case or the optimization extensions for Mobile IP.

5.1.1. Handoff Handling

Both CDPD and the base Mobile IP behave similarly during handoff. This conclusion is derived from the deterministic case where the delay due to the registration with the new foreign agent or serving MD-IS is not blurred by random variations of packet arrivals. As seen in figures 22 and 25 the ETE delay at the handoff times is identical for both of the base protocols.

The intra-area handoff of CDPD in Figure 25 and Figure 49 in Appendix B is not distinguishable in the plot and passes unnoticed by TCP. The very fast intra-

area handoff can be explained by observing that it is handled entirely by the lower layers and thus the overhead is very small. This clearly indicates the greater strength of CDPD when the M-ES is operating locally in the serving area of a single provider with mostly intra-area handoffs. Intra-area handoffs will indeed be the most common operations.

The simulation of Mobile IP with simultaneous bindings shows the benefits of the mobile node having simultaneous connections with at least two foreign agents. Of course to achieve maximum gains the foreign agents' serving areas must overlap. In this situation the mobile node may continue to receive and transmit data through one of its registered foreign agents while performing registration with the other thus providing uninterrupted connectivity to the higher layers.

Finally, the optimized Mobile IP protocol does not appear to provide any benefits in the case of handoffs as can be clearly seen in figures 24, 48 and 52. On the contrary it introduces additional delay during those periods because of its larger registration messages. The provision for the forwarding of old messages from the old foreign agent to the current location of the mobile node does not reduce the handoff delay.

5.1.2. Steady State (no handoff) Performance

The clear strongest performer in the steady state (no handoff) case, that is the operation within a single cell, is the optimized Mobile IP protocol. The ETE delay of the packets is significantly reduced by eliminating the triangular routing used in all other protocols. The correspondent node, however, is expected to be mobile aware and

implement the optimized Mobile IP protocol, a condition which will be very rare in the near future. All the other protocols perform similarly since this operation is not dependent on mobility handling other than the redirection of incoming traffic from the home network to the current location of the mobile node.

5.1.3. Conclusion

The mobility handling of all the protocols performs adequately but in most cases the performance is dominated by factors out of the control of these protocols. The TCP parameters are a major factor in the performance and there are already suggestions on the adjustment of TCP to take mobility into account. ([21], [22])

The error conditions of the mobile channel are also a major factor of performance. In spite of the provision for error correction in many situations errors cannot be corrected and they are passed on to the higher layers.

To sum up, the mobile networks' performance is more dependent upon other factors such as the TCP layer's parameters and the channel's error conditions than on Mobile IP's and CDPD's mobility handling. In cases of frequent handoff, for example due to a small cell configuration, it may also be concluded that the simultaneous bindings option for Mobile IP and the provision for fast intra-area handoffs for CDPD offer significant improvement over the base protocols.

5.2. A Related Topic of Interest: CDPD and Mobile IP Interoperability

CDPD is already available in many cities in North America and Mobile IP has recently issued its RFC. Judging from the major Telecommunications companies behind CDPD and the interest shown by companies such as FTP Software Inc. on Mobile IP, we may expect these two protocols to be implemented widely in the near future. If our prediction is correct, an interesting problem would then be to provide seamless operation for users roaming between these two types of mobile networks.

The two protocols have implemented their mobility handling in different layers. TCP with Mobile IP can, therefore, be used as the higher layers common to both networks and the mobile user will need two separate modems/network cards to attach to each network. A few amendments and additional software will be necessary for the proper operation of such a hybrid system.

Assuming that the mobile node/M-ES will be a notebook computer running Windows 95, the lower two layers will have to conform to the Network Driver Interface Specification (NDIS). This conformance will enable multiple modems/network cards to work with multiple transports[2]. NDIS drivers will therefore be needed to be placed between the hardware and the NDIS interface. The link layer software for the Mobile IP

[2] In the Microsoft terminology TCP/IP, IPX/SPX and NETBEUI are called transports.

network will also need to be NDIS compliant. TCP and Mobile IP will, then, be used on top of NDIS as the transport.

A daemon software, an application similar to the dialer in other forms of network connectivity will be needed in order to monitor the link activity and decide on which underlying network will be used. This daemon will cooperate with the Mobile IP software to provide seamless roaming to TCP and to the user.

The hybrid protocol will require additional functionality on the part of Mobile IP due to the absence of a foreign agent. The mobile node will, thus, need the extra functionality of a mobility agent in order to register with its home agent using the IP address provided by CDPD as its co-located address. In addition, the daemon monitoring the links will have to be written.

5.2.1 Operation Overview

We assume that the user will usually use his[3] office Mobile IP network but will also require connectivity on the road. The user will, therefore, use the valid Internet address given by his office administrator as his permanent address.

When the user leaves the coverage area of his office Mobile IP network and enters a CDPD coverage area the daemon application will sense this change in network attachment and it will begin the registration procedure with the CDPD provider. When

---

[3] In every occurrence of 'he' or 'his' please read 'he/she' and 'his/her'.

the registration is complete the daemon application will use the IP address assigned by the CDPD provider as its co-located care-of address. Using the new IP address the Mobile IP will, then, be used to register the Mobile Node directly with its Home Agent.

Intra-area handoffs within the CDPD network will not require re-registration by Mobile IP since the co-located IP address will remain unchanged, whereas inter-area handoffs will require use of the Mobile IP registration process.

When the user returns to his office Mobile IP network the daemon application may sign off from the CDPD network and invoke the Mobile IP procedure for returning home. The user may instead choose to remain connected to the CDPD network entering a power saving mode thus reducing the overhead of registering again the next time he leaves the office.

5.3. Further work

A study of the channel error characteristics and its interaction with the handoff handling by the mobile protocols would be an interesting project. From the results shown in figures 26-30 it appears that the timing of error arrivals plays an important role in the handoff performance.

There are already various Mobile IP implementations in the public domain, mainly for the free UNIX clone, Linux. An interesting project would be the implementation of a (minimum) three node mobile network with a Home Agent and a Foreign Agent running one of the public domain mobility agent software and the implementation of Mobile IP for

Windov - 95. This testbed in combination with a CDPD subscription may be used to study the interoperability suggestion presented in section 5.2.

REFERENCES

1. Perkins, Charles. "IP mobility support", Internet Draft -- work in progress, July 1995.

2. CDPD Forum, "*Cellular Digital Packet Data System Specification*", Release 1.0, July 19, 1993.

3. Vuong, S.T., et. al., "*Issues in Internetworking Wireless Data Networks for Mobile Computing*", Proc. IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing, May 1995.

4. Postel, J. "*Internet Control Message Protocol - DARPA Internet Program Protocol Specification*", RFC 792, USC/Information Sciences Institute, September 1981.

5. Deering, S. "*Router Discovery*", RFC 1256, September 1991.

6. Rivest, R. "*The MD5 Message-Digest Algorithm*", RFC 1321, April 1992.

7. Perkins, C. "*IP Encapsulations within IP*", Internet Draft -- work in progress, July 1995.

8. "*Minimal Encapsulation within IP*", Internet Draft -- work in progress, July 1995.

9. Hanks, S. et al. "*Generic Routing Encapsulation GRE*", RFC 1701, October 1994.

10. Postel, J. *"User Datagram Protocol"*, RFC 768, August 1980.

11. Kleinrock, L., *"Queuing Systems, Volume II: Computer Applications"*, John Wiley & Sons, Inc., 1976.

12. Schwartz, M. *"Telecommunication Networks: Protocols, Modeling and Analysis"*, Addison-Wesley Publishing Company, 1988.

13. Kleinrock, L., *"Communication Nets: Stochastic Message Flow and Delay"*, McGraw-Hill, New York, 1964.

14. Wong, J.W., Lam, S.S. *"Queuing Network Models of Packet Switching Networks, Part 1: Open Networks"*, Performance Evaluation, Volume 2, 1982.

15. Robertazzi, T. G., *"Computer Networks and Systems: Queuing Theory and Performance Evaluation"*, Springer-Verlag New York Inc., 1990

16. Kleinrock, L., *"Queuing Systems, Volume 1: Theory"*, John Wiley & Sons, Inc., 1975.

17. *"OPNET manual"*, Mil 3, Inc., 1994

18. Rubin, I., *"Communication Networks: Message Path Delays"*, IEEE Transactions on Information Theory, Vol. IT-20, No. 6, November 1974

19. Arnold, H.W., Bodtmann, W.F., *"Fade-Duration Statistics of a Rayleigh-Distributed Wave"*, IEEE Transactions on Communications, Vol. COM-30, No. 3, March 1982.

20. Arnold, H.W., Bodtmann, W.F., *"Interfade Interval Statistics of a Rayleigh-Distributed Wave"*, IEEE Transactions on Communications, Vol. COM-31, No. 9, September 1983.

21. Cáceres, R., Iftode, L., "*Improving the performance of Reliable Transport Protocols in Mobile Computing Environments*", IEEE Journal on Selected Areas in Communications, Vol. 13, No. 5, June 1995.

22. Lee, Pamela, "*A Mobile-Aware Transmission Control Protocol (TCP) for Wireless Communications*", M.A.Sc. Thesis, Simon Fraser University, December 1996.

23. Kunzinger, Charles A., "*Network Layer Mobility: Comparison of CDPD and Mobile IP*", IBM Corp. 1995.

APPENDIX A

PACKET FLOW MODELS

A.1. Base Mobile IP

During router discovery there are at most two ICMP [4] messages with the mobility extensions.

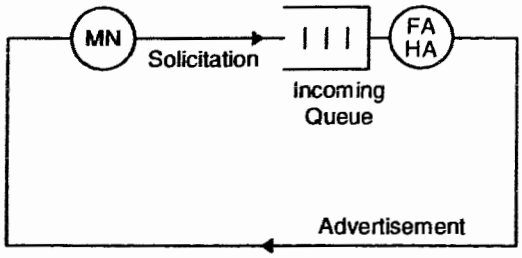


Figure 37. Router discovery using ICMP discovery extensions

The size of the two messages is:

Advertisement: IP header + ICMP + Mobility extension[4] = 5 + 4 + 3

= 12 (32-bit words)

---

[4] The mobility extension informs the mobile node that the agent advertisement message is sent by a mobility agent.

Solicitation: IP header + ICMP = 5 + 2 = 7 (32-bit words)

Registration is achieved with two messages: a registration request from the mobile node and a registration reply from the home agent.

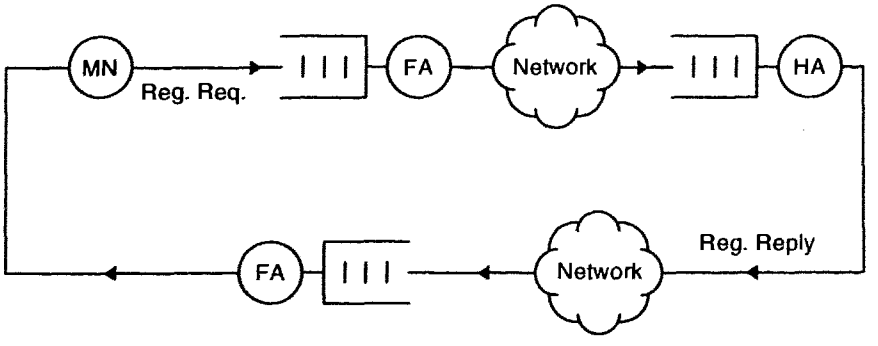


Figure 38. Registration with home agent

The size of the registration request and reply messages is:

Request: IP header + UDP[5] header + mobile IP fields + key identifier extension + mobile-home authentication extension

= 5 + 2 + 6 + 1 + 8 = 22 (32-bit words)

Reply: = IP header + UDP header + mobile IP fields + mobile-home authentication extension

= 5 + 2 + 5 + 8 = 20 (32-bit words)

[5] UDP [16] is a connectionless transport protocol and it may be used instead of TCP if an acknowledgment is not needed.

When the mobile node reaches home it sends a deregistration message and the home agent responds with a registration reply.

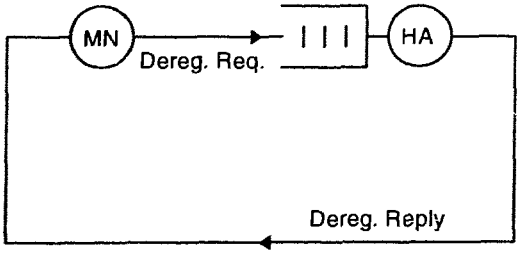


Figure 39. Deregistration with foreign agents

The size of the deregistration request and reply messages is equal to that of the corresponding registration messages.

When at home the mobile node behaves as a conventional stationary node.

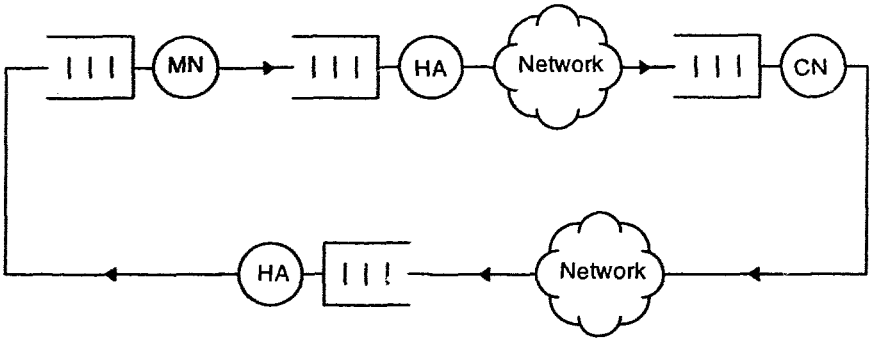
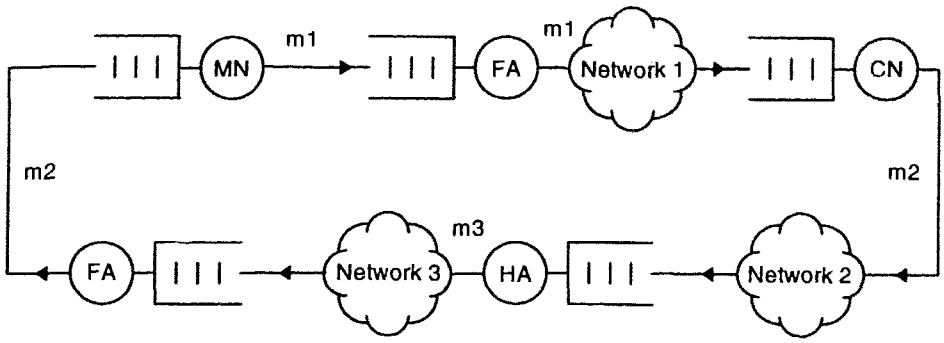


Figure 40. Data exchange. Mobile node - Stationary node communications (mobile node at home)

The size of the data messages' header size is therefore equal to the IP header.

When the mobile node is away from home only inbound messages need tunneling.



m1: message from MN to CN
m2: message from CN to MN
m3: message m2 encapsulated and sent to FA

Figure 41. Data exchange. Mobile node - Stationary node communications (mobile node away)

The length of the header of an inbound data packet is equal to two IP headers between the home agent and the foreign agent. In the other cases it is equal to one IP header.

Message m1, m2: IP header = 5 (32-bit words)

Message m3: 2 x IP header = 2 x 5 = 10 (32-bit words)

A.2. Optimized Mobile IP

When the mobile node moves to an area serviced by a different foreign agent it registers with its home agent through this new foreign agent and also notifies the previous foreign agent about its relocation. As shown in Figure 42 the mobile node sends the

registration request to the new foreign agent which processes it forwards it to the home agent. The new foreign agent also creates a binding update message and sends it to the previous foreign agent notifying it of the mobile node's current location. This message will enable the previous foreign agent to forward any misdirected messages for the mobile node to their correct destination. The home agent and the previous foreign agent acknowledge the messages with a registration reply and binding acknowledgment respectively.

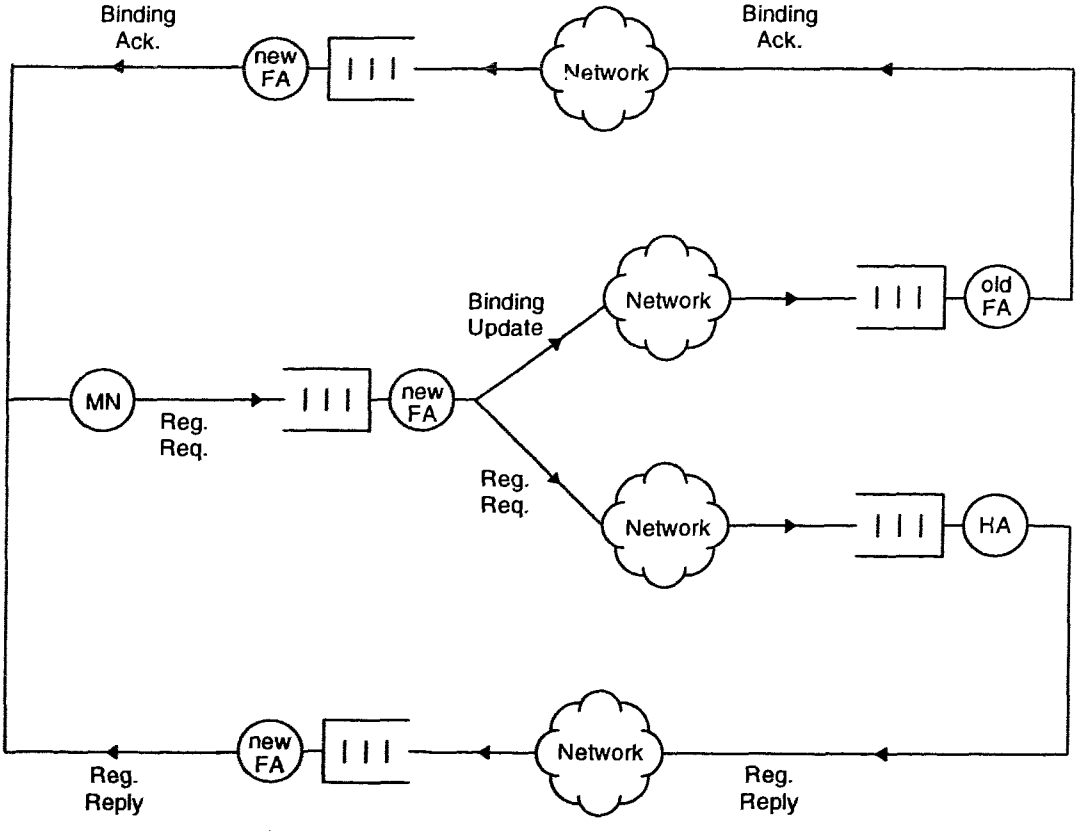


Figure 42. Handoff.

The size of the messages

Reg. Req.: IP header + UDP header + mobile IP fields + key identifier extension + mobile-home authentication extension + previous foreign agent notification extension

= 5 + 2 + 6 + 1 + 8 + 5 = 27 (32-bit words)

Reg. Rep.: = IP header + UDP header + mobile IP fields + mobile-home authentication extension

= 5 + 2 + 5 + 8 = 20 (32-bit words)

Binding Update: IP header + UDP header + binding update + route optimization authentication extension

= 5 + 2 + 5 + 3 = 15 (32-bit words)

Binding Acknowledge: IP header + UDP header + binding acknowledge

= 5 + 2 + 4 = 11 (32-bit words)

When a mobile node moves to a new foreign agent the correspondent node initially sends packets to the old foreign agent. The old foreign agent sends a Binding Warning message to the home agent which in turn sends a Binding Update message to the correspondent node.

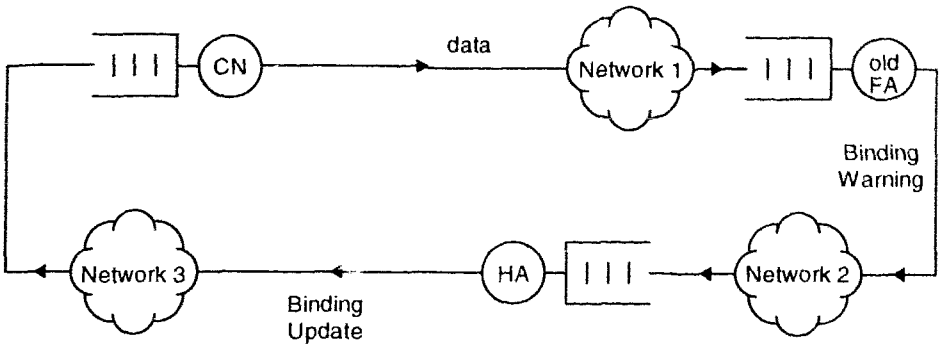


Figure 43. Routing Information Update after a Handoff.

The size of the messages

Binding Warning: IP header + UDP header + binding warning

= 5 + 2 + 3 = 10 (32-bit words)

The Binding Request message is used when a node needs to confirm or update its routing information. Its size is:

Binding Request: IP header + UDP header + binding request

= 5 + 2 + 4 = 11 (32-bit words)

A.3. CDPD

Key exchange procedures are required before the exchange of registration messages. The keys are used for the encryption of all subnetwork layer packets. The serving MD-IS sends the MD-IS Key Exchange (IKE) message to initiate change of secret keys and the M-ES replies with the M-ES Key Exchange (EKE) message.

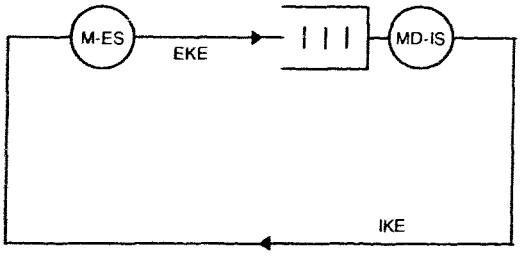


Figure 44. M-ES — MD-IS key exchange

The size of the messages are:

IKE : SNDCP header + IKE = 2 +99 = 101 (octets)

EKE : SNDCP header + EKE = 2 +34 = 36 (octets)

Registration is achieved with four messages: M-ES Hello from the M-ES to the serving MD-IS, MD-IS Redirect Request from the serving MD-IS to the home MD-IS, MD-IS Redirect Confirm from the home MD-IS to the serving MD-IS, and MD-IS Hello Confirm from the serving MD-IS to the M-ES.

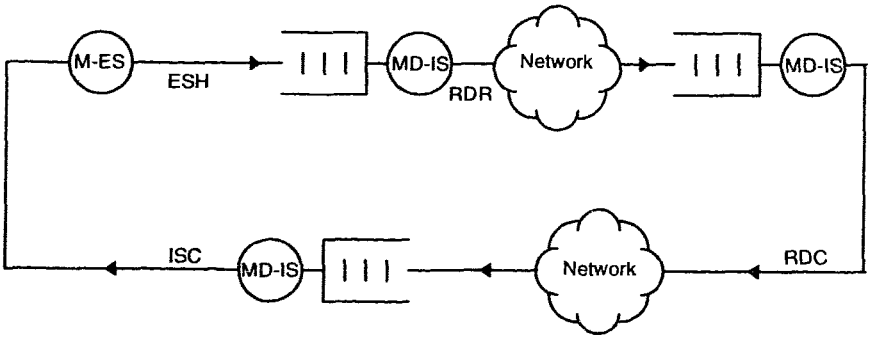


Figure 45. M-ES registration

The size of the messages are:

ESH : SNDCP header + ESH = 2 +19 + 13 = 34 (octets)

RDR : CLNP header + RDR = 25 + 26 + 13 = 64 (octets)

RDC : CLNP header + RDC = 25 + 10 + 11 = 46 (octets)

ISC : SNDCP header + ISC = 2 + 18 + 4 + 11 = 35 (octets)

The diagrams for data transfer are very similar to the Mobile IP diagrams so they are omitted.

APPENDIX B

ADDITIONAL SIMULATION RESULTS

The plots in figures 46-49 show the ETE delay of the protocols with a handoff for the Mobile IP protocols at 351.9 seconds and 901.9 seconds for the CDPD protocol an intra-area handoff at time 301.9 and two inter-area handoffs at 501.9 seconds and 1,001.9 seconds. The CDPD protocol for this simulation uses separate nodes for the MDBS and the MD-IS connected with a 560,000 bps line.

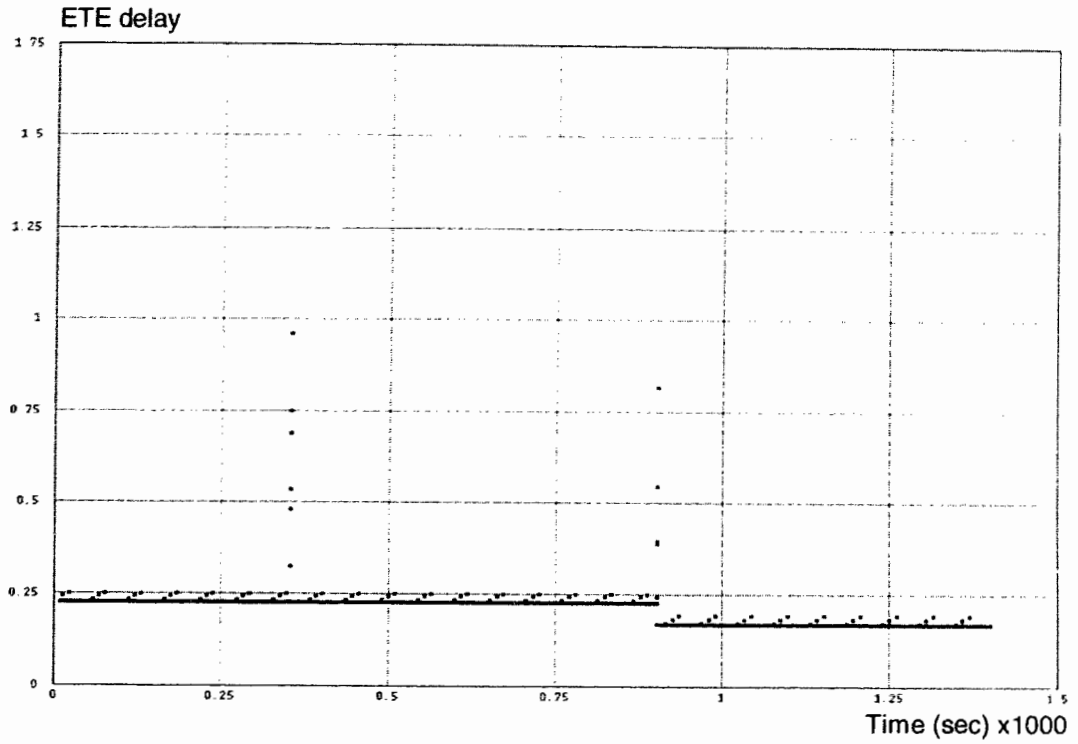


Figure 46. Base Mobile IP ETE delay.

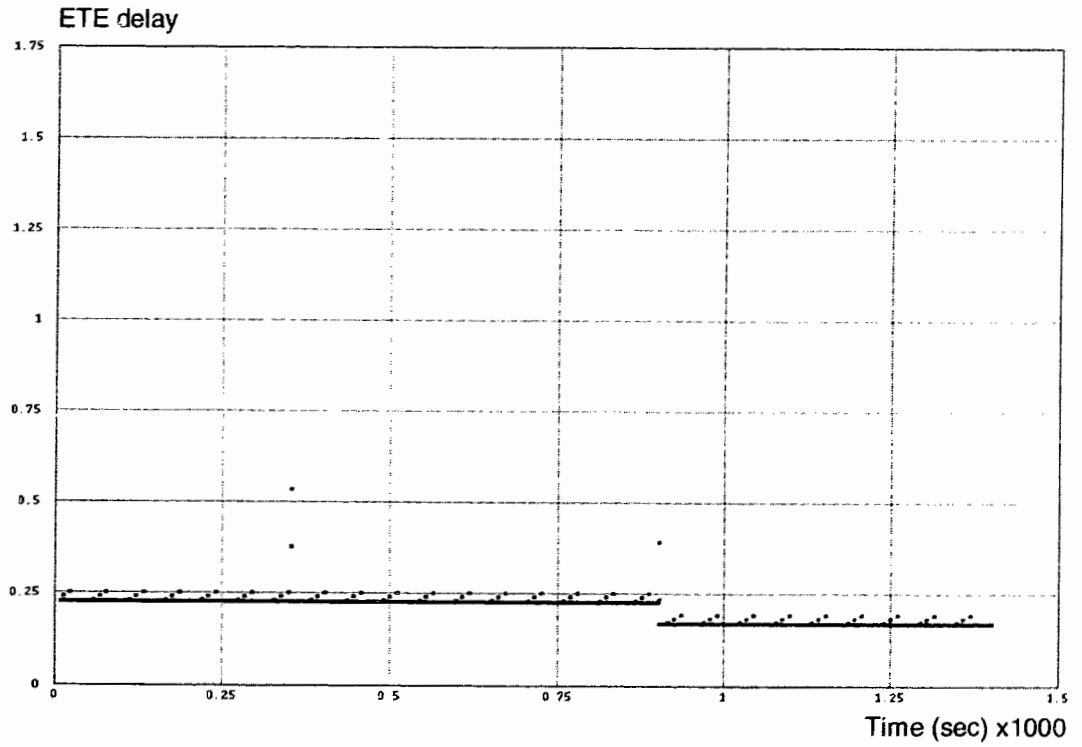


Figure 47. Base Mobile IP with simultaneous bindings ETE delay

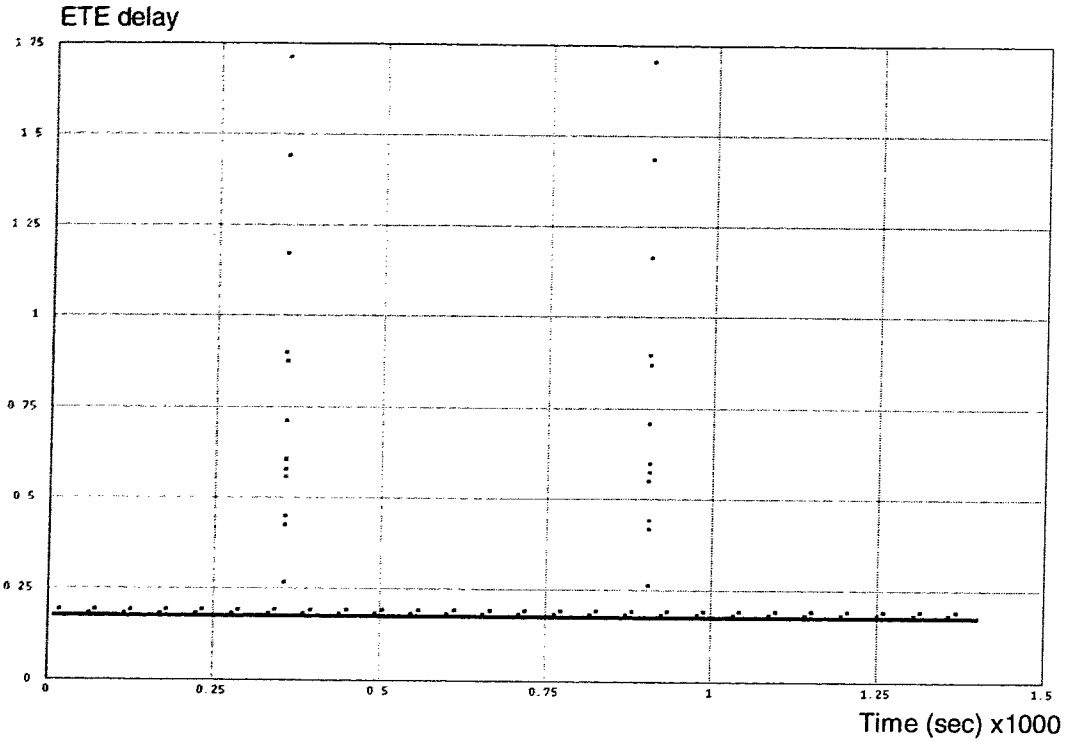


Figure 48. Optimized Mobile IP ETE delay.

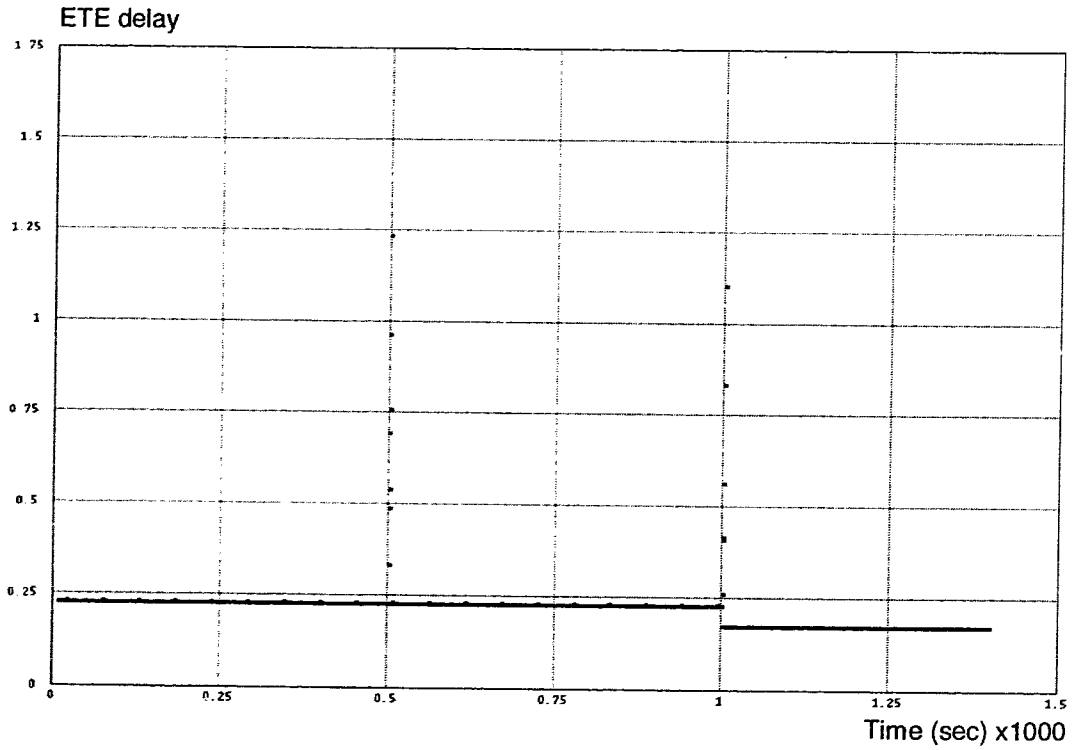


Figure 49. CDPD ETE delay.

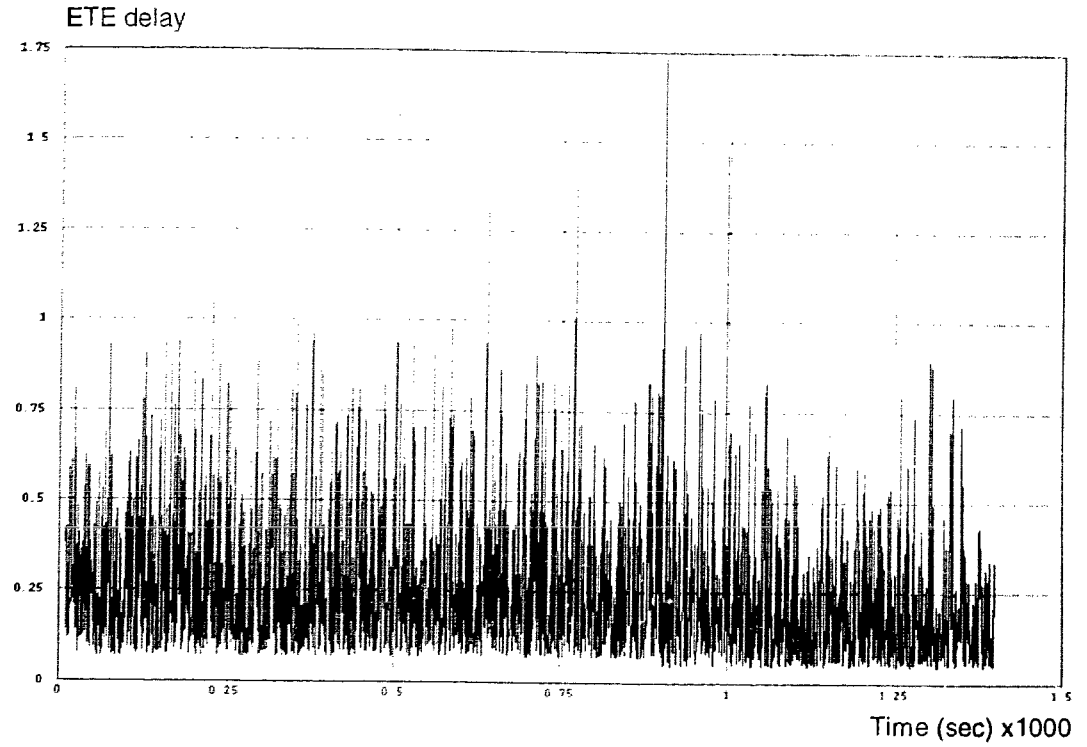


Figure 50. Base Mobile IP ETE delay.

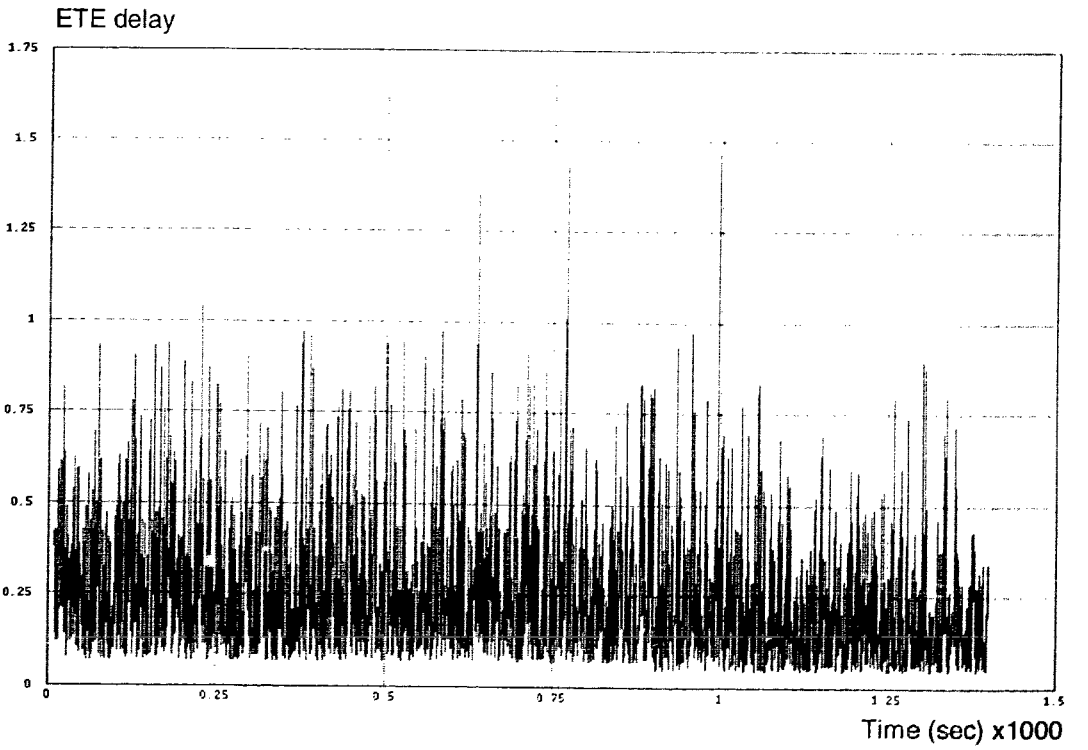


Figure 51. Base Mobile IP with simultaneous bindings.

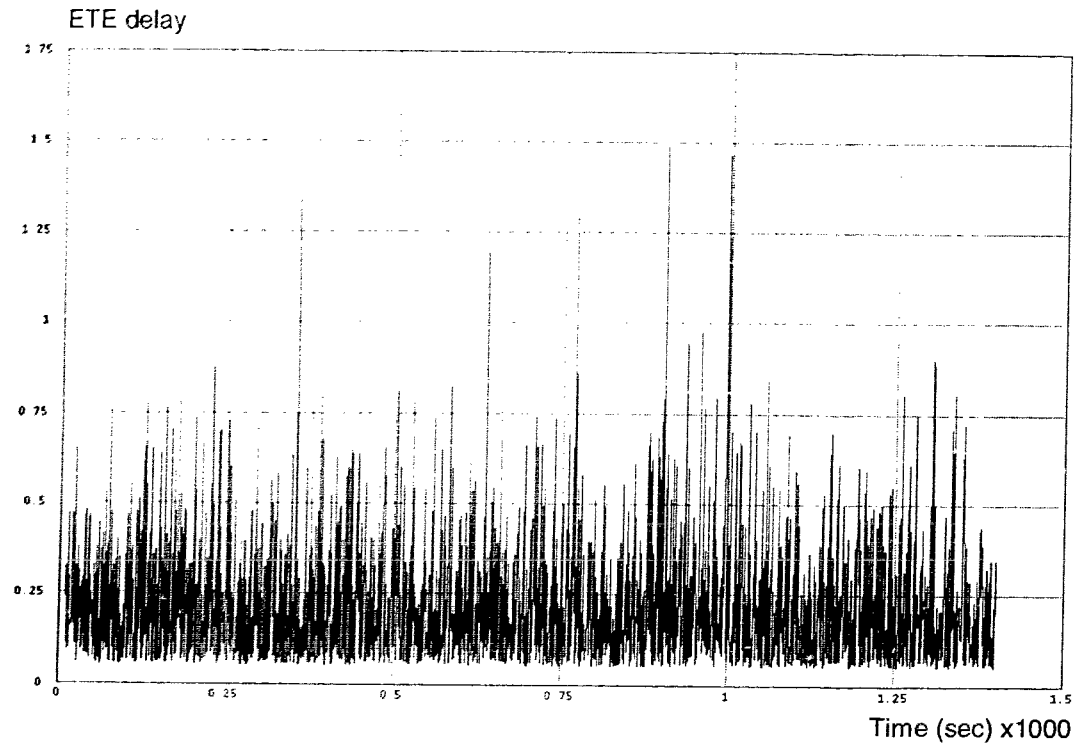


Figure 52. Optimized Mobile IP.

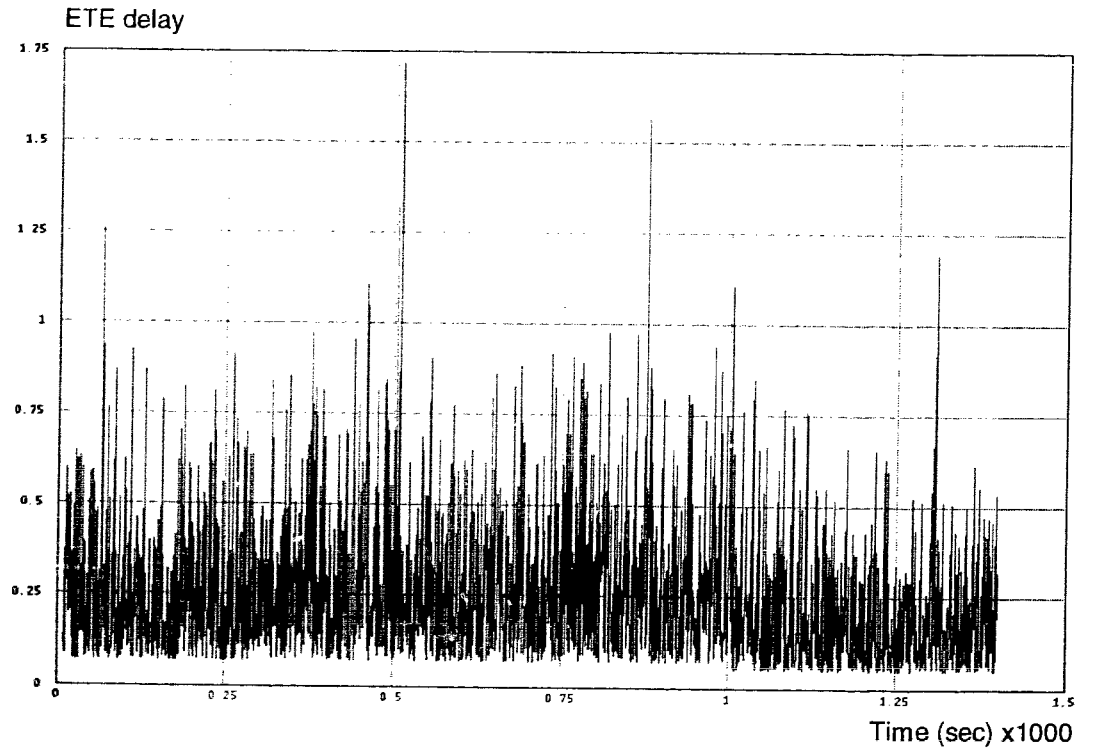
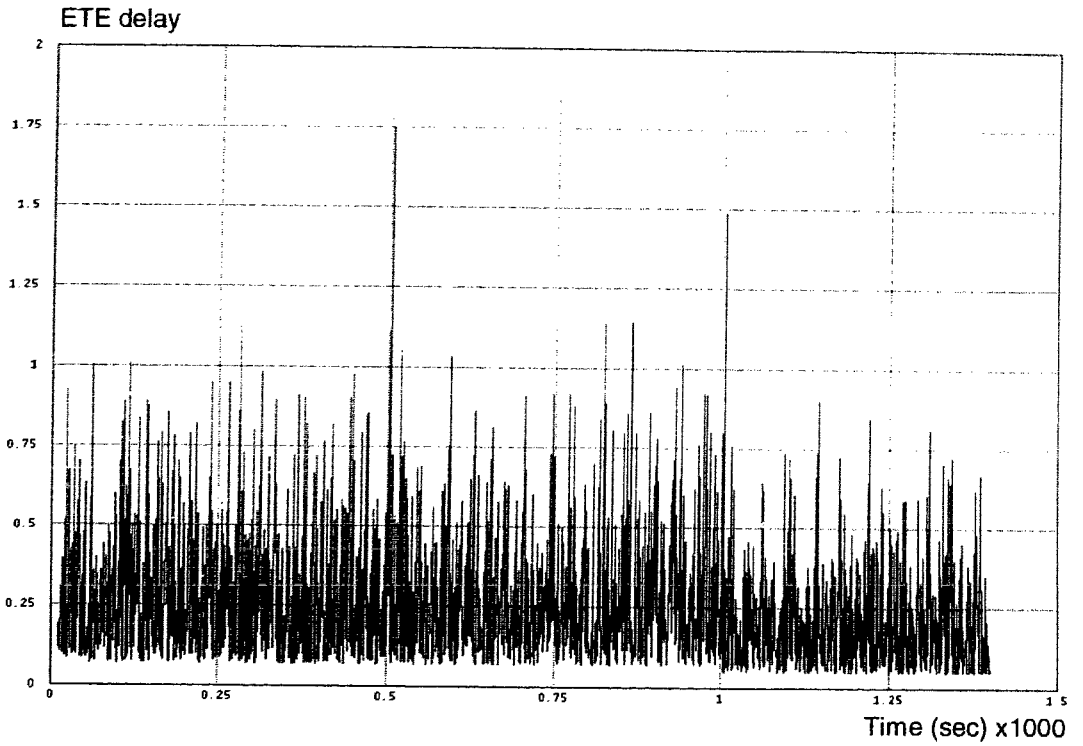
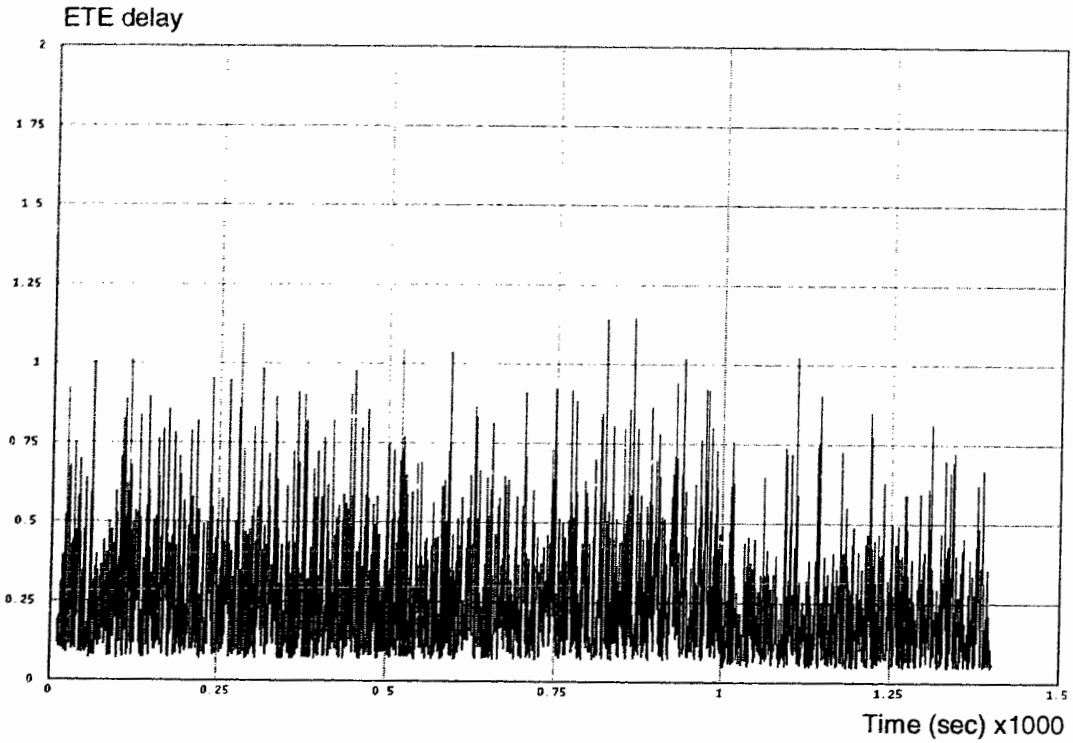


Figure 53. CDPD ETE delay.



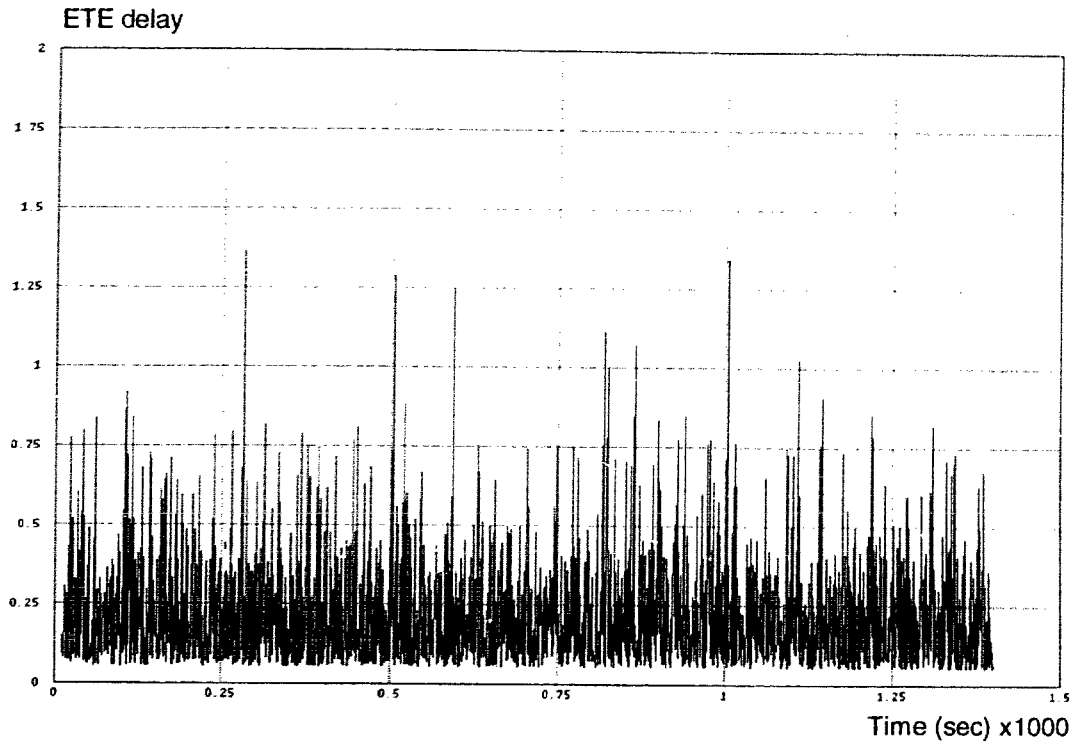
abscissa,	min:	0
	max:	1400
ordinate,	min:	0.0468727325066993
	max:	1.74546815736556
sample mean:		0.276256922229819
variance:		0.0389852629858901
standard deviation:		0.197446861170012

Figure 54. Base Mobile IP ETE delay with statistical information.



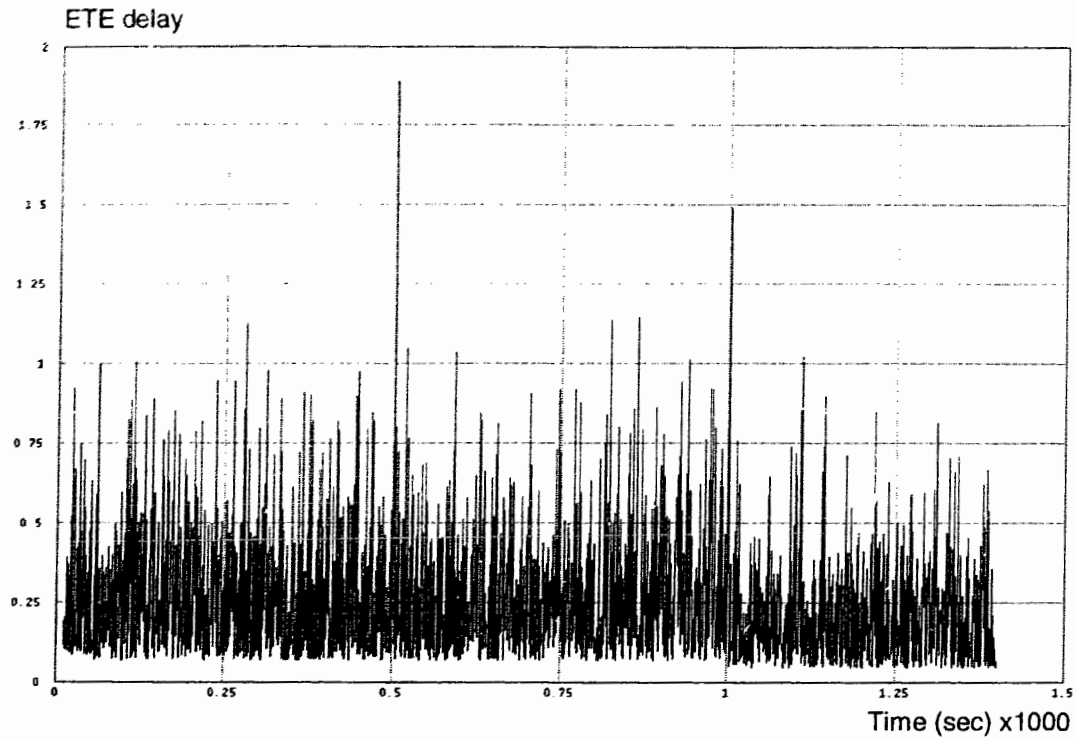
abscissa,	min:	0
	max:	1400
ordinate,	min:	0.0468727325064719
	max:	1.14345096295972
sample mean:		0.274670571702896
variance:		0.0371896796114251
standard deviation:		0.19284625900293

Figure 55. Base Mobile IP with multiple bindings ETE delay with statistical information.



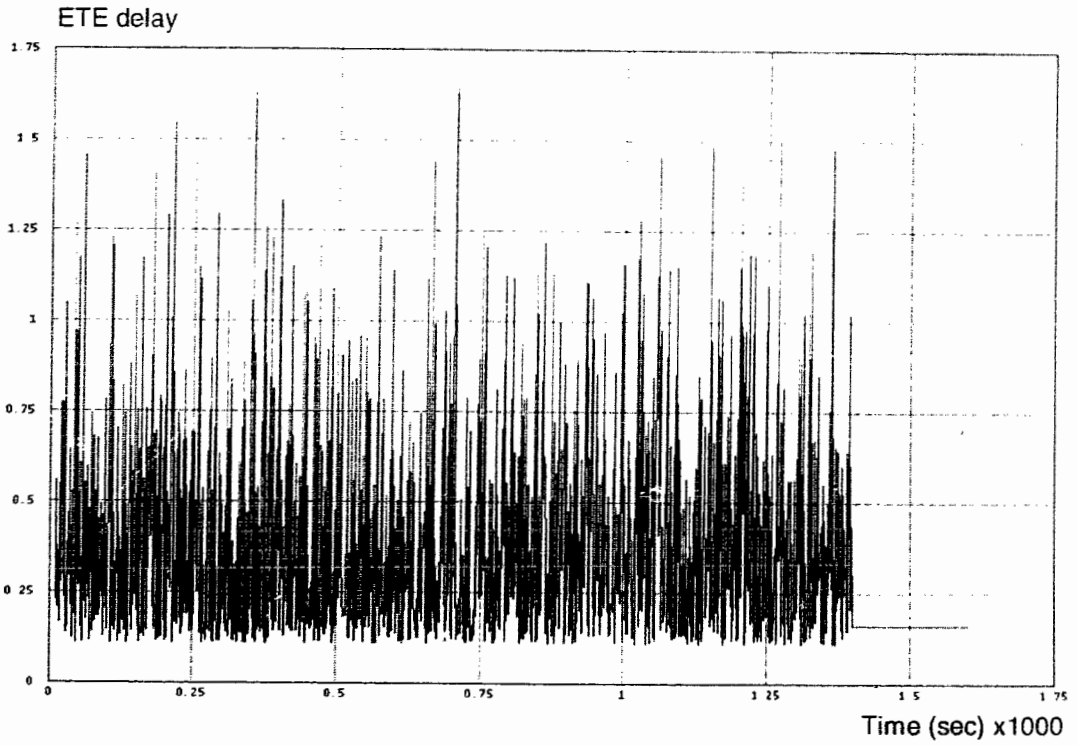
abscissa, min:	0
max:	1400
ordinate, min:	0.0535870186049578
max:	1.36181530736565
sample mean:	0.234281453912594
variance:	0.0307186018422472
standard deviation:	0.175267229801373

Figure 56. Optimized Mobile IP ETE delay with statistical information.



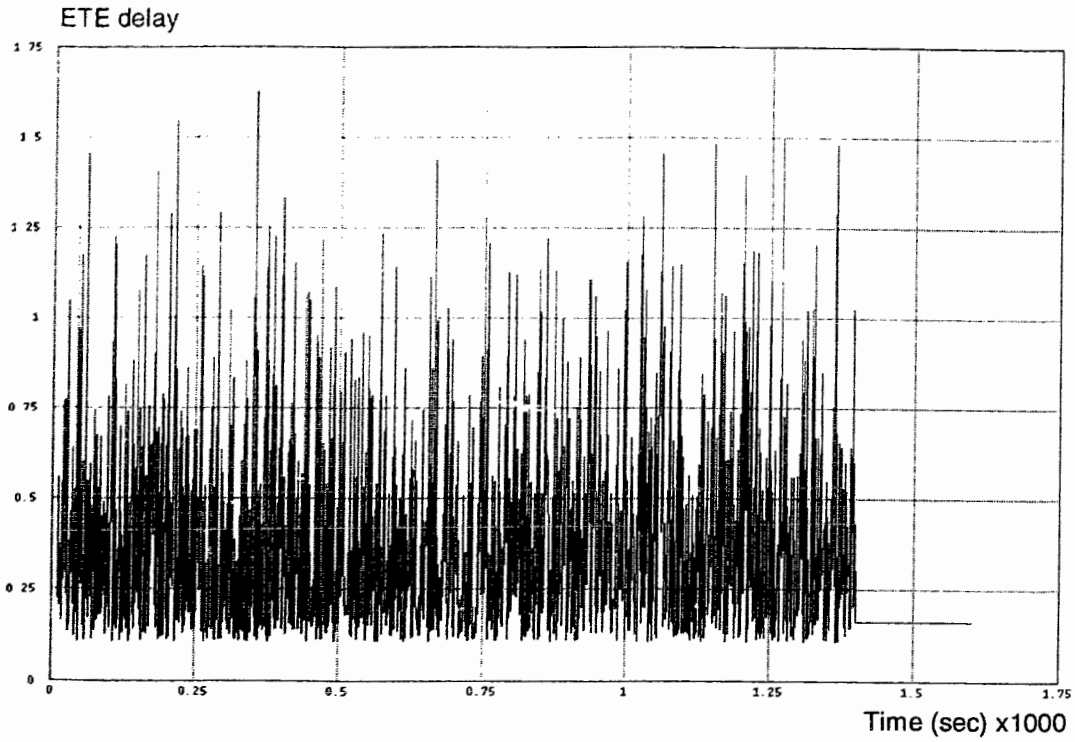
abscissa,	min:	0
	max:	1400
ordinate,	min:	0.0468779627713047
	max:	1.88765414858989
sample mean:		0.276241253514846
variance:		0.0396730285379251
standard deviation:		0.199180894008248

Figure 57. CDPD ETE delay with statistical information.



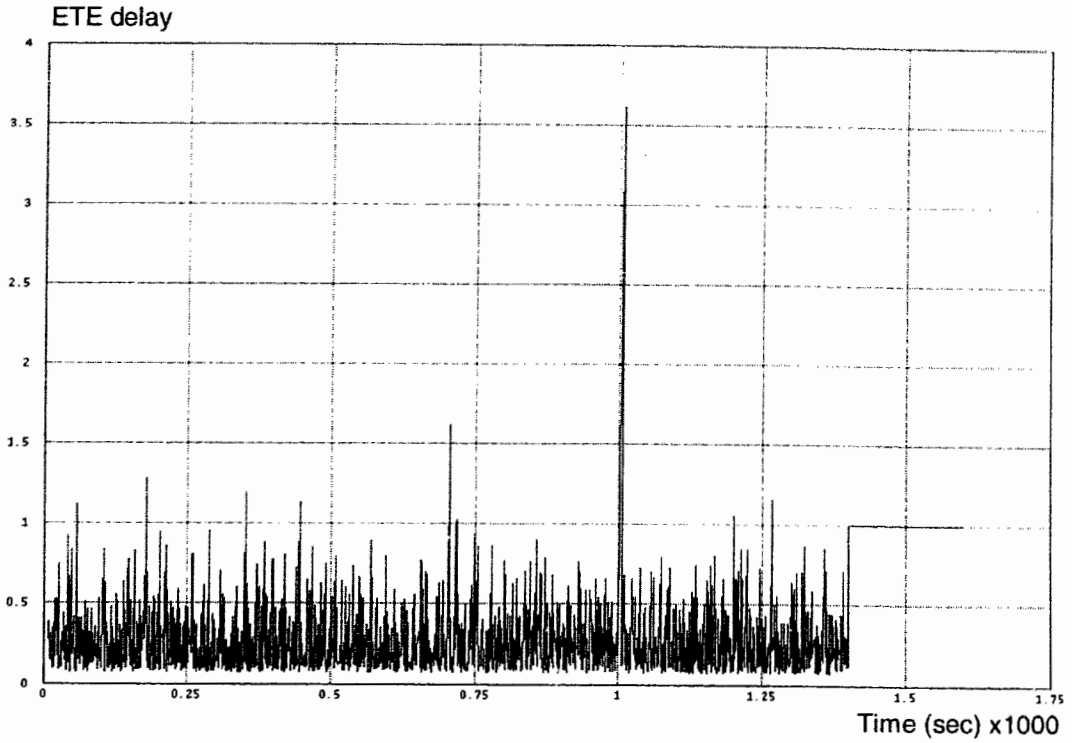
abscissa, min:	0
max:	1600
ordinate, min:	0.108461935571825
max:	1.64368500511216
sample mean:	0.420780956215193
variance:	0.0762411720157028
standard deviation:	0.276118040004095

Figure 58. Base Mobile IP ETE delay with statistical information for the large network.



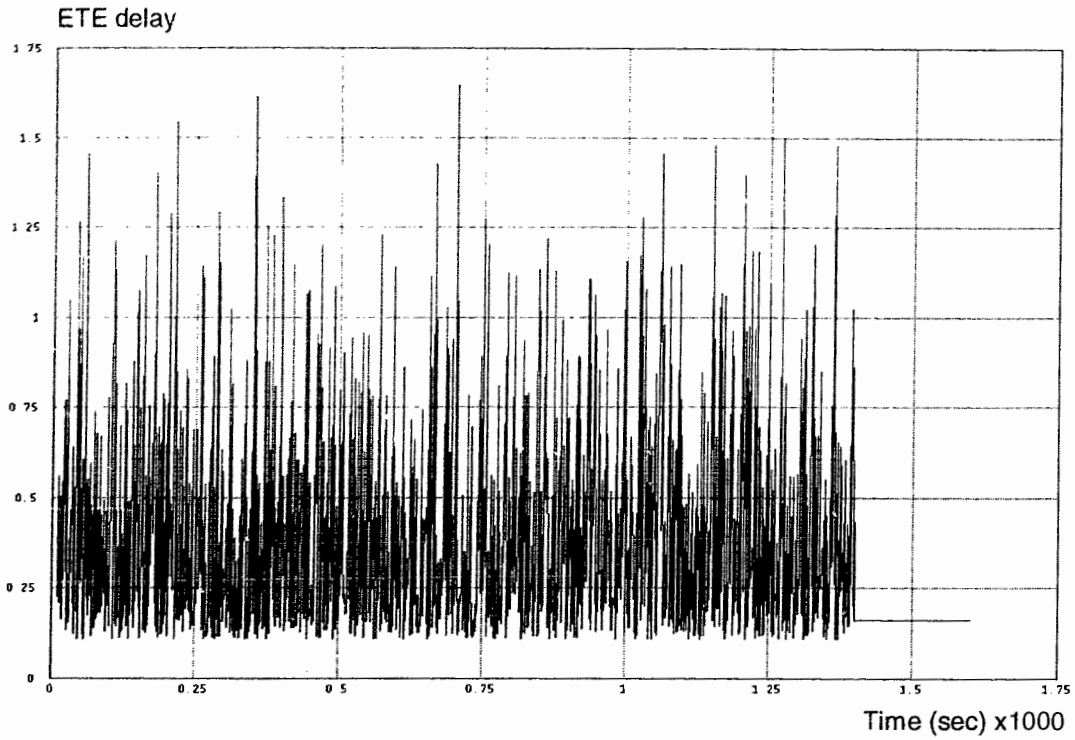
abscissa,	min:	0
	max:	1600
ordinate,	min:	0.108461935571825
	max:	1.62619590684477
sample mean:		0.420304941777977
variance:		0.0756042063858006
standard deviation:		0.274962190829577

Figure 59. Base Mobile IP with multiple bindings ETE delay with statistical information for the large network.



abscissa,	min:	0
	max:	1600
ordinate,	min:	0.0753190786664959
	max:	3.61432606065046
sample mean:		0.288946600411
variance:		0.0540906768545454
standard deviation:		0.232574024462203

Figure 60. Optimized Mobile IP ETE delay with statistical information for the large network.



abscissa, min:	0
max:	1600
ordinate, min:	0.108461956175631
max:	1.64368493452889
sample mean:	0.420229251099338
variance:	0.0760471460317133
standard deviation:	0.275766470100542

Figure 61. CDPD ETE delay with statistical information for the large network.

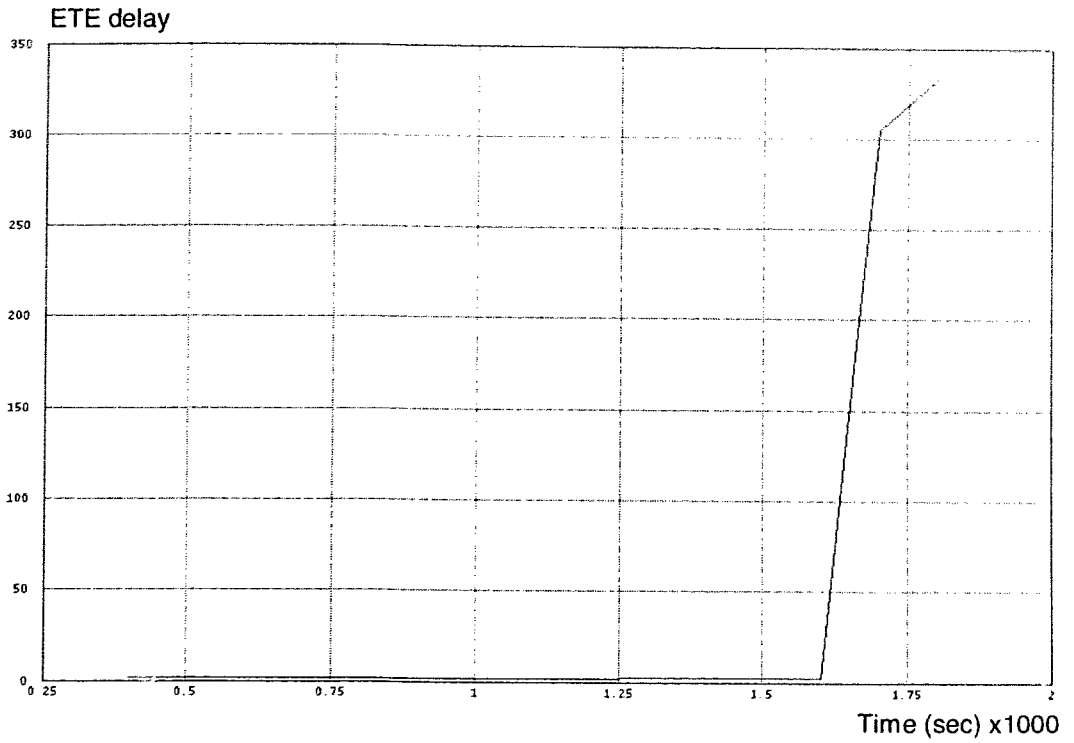


Figure 62. Base Mobile IP maximum ETE delay with minimum RTO values ranging between 0.4 and 1.8.

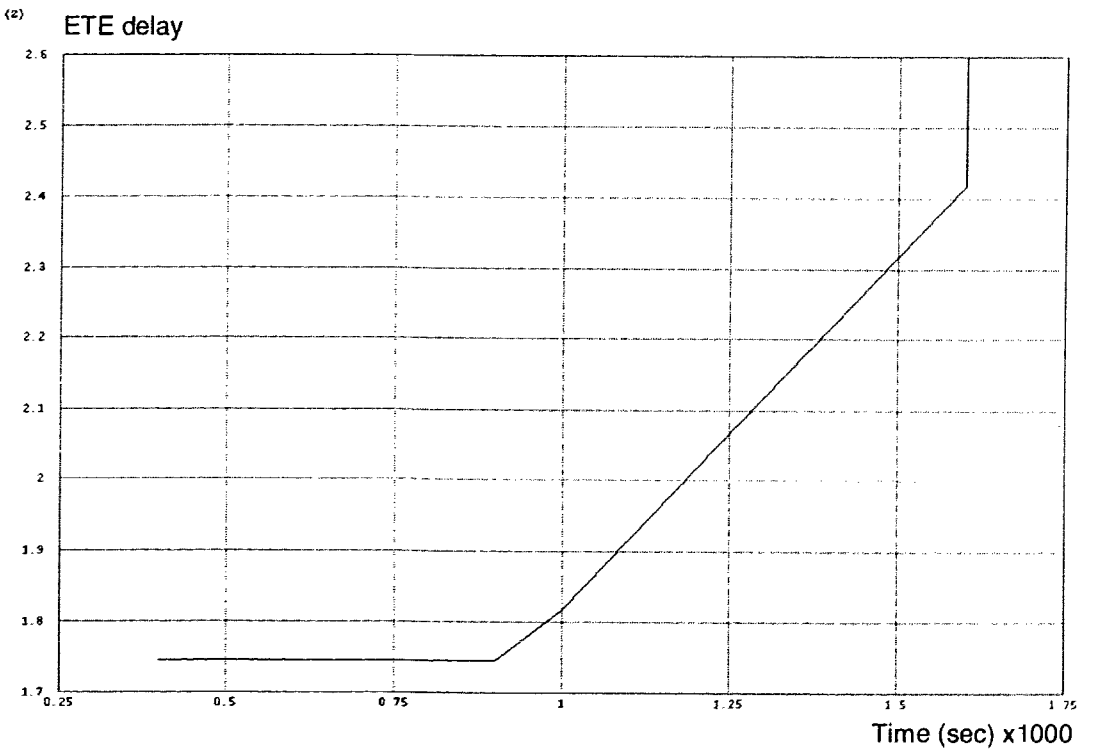


Figure 63. Base Mobile IP maximum ETE delay with minimum RTO values ranging between 0.4 and 1.6.

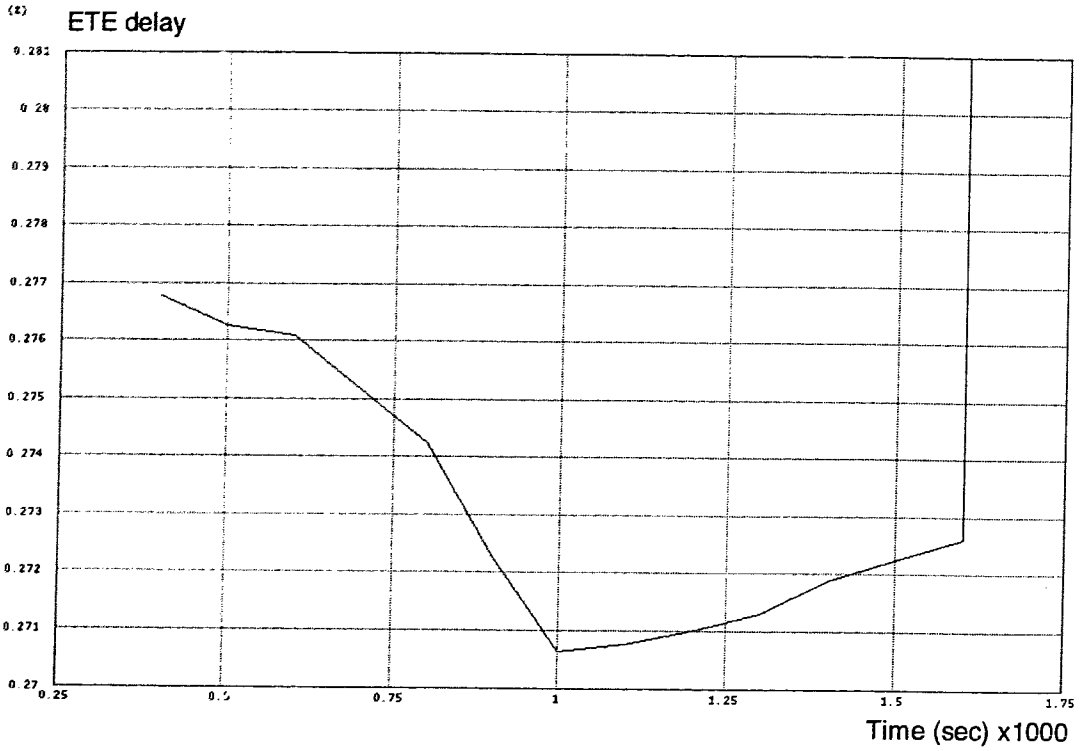


Figure 64. Base Mobile IP average ETE delay with minimum RTO values ranging between 0.4 and 1.6.

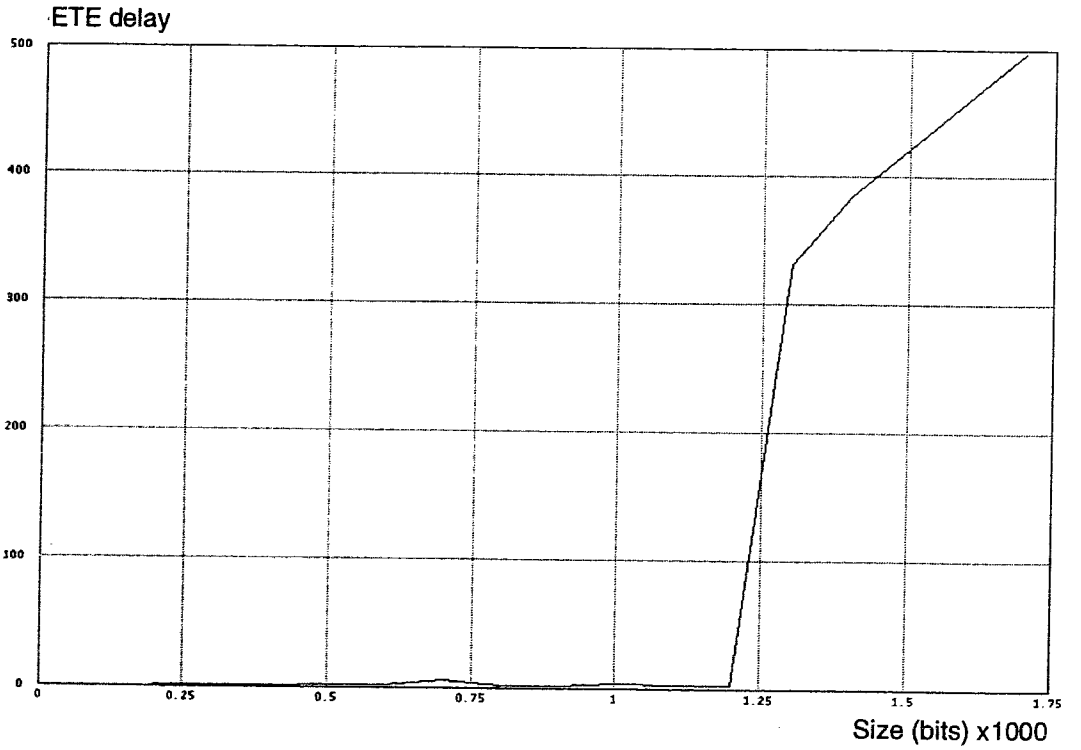


Figure 65. Base Mobile IP maximum ETE delay (Poisson) with packet sizes ranging between 200 and 1700 bits.

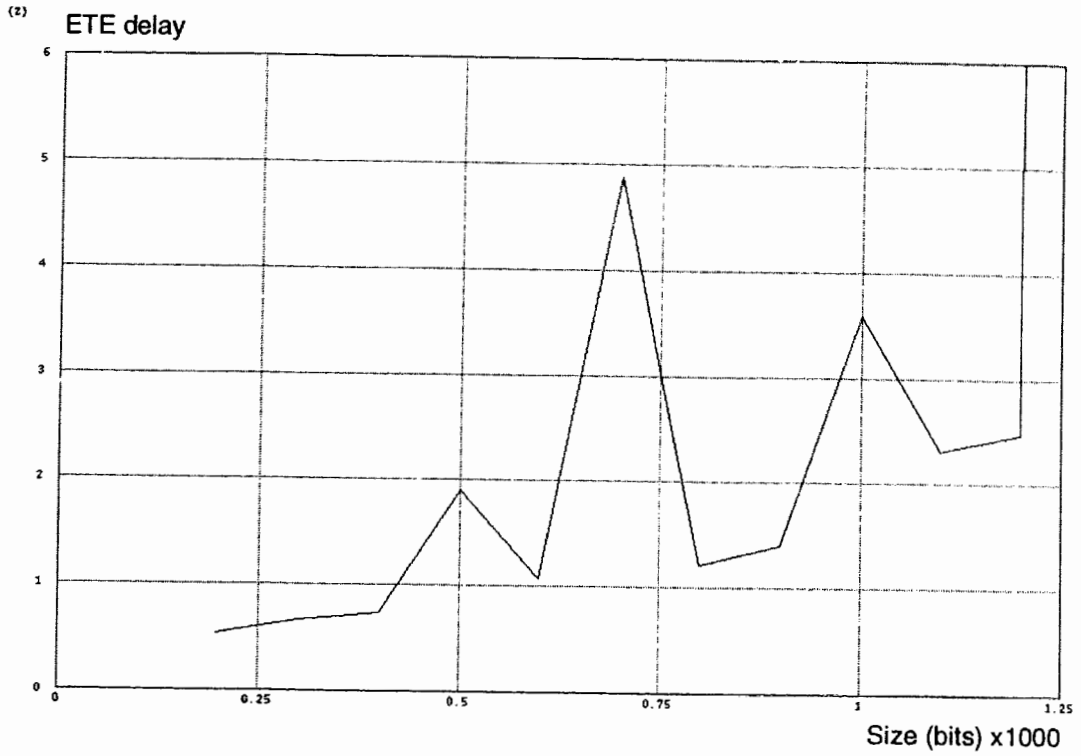


Figure 66. Base Mobile IP maximum ETE delay (Poisson) with packet sizes ranging between 200 and 1200 bits.

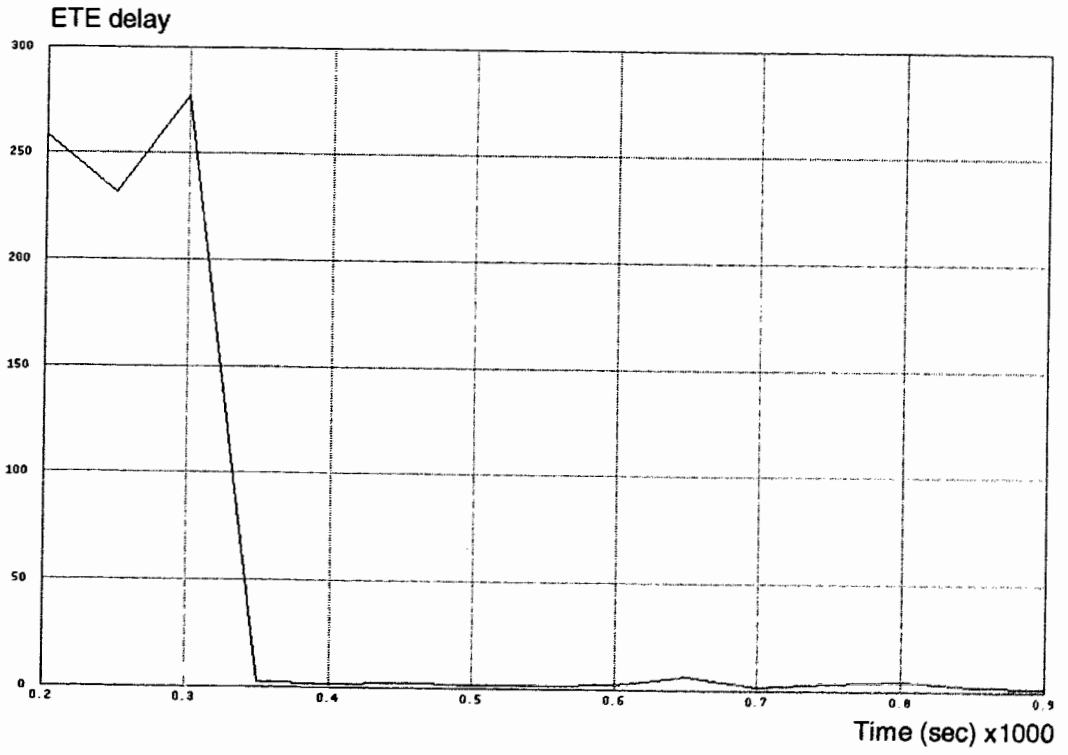


Figure 67. Base Mobile IP maximum ETE delay with average interarrival times ranging between 0.2 and 0.9 seconds.

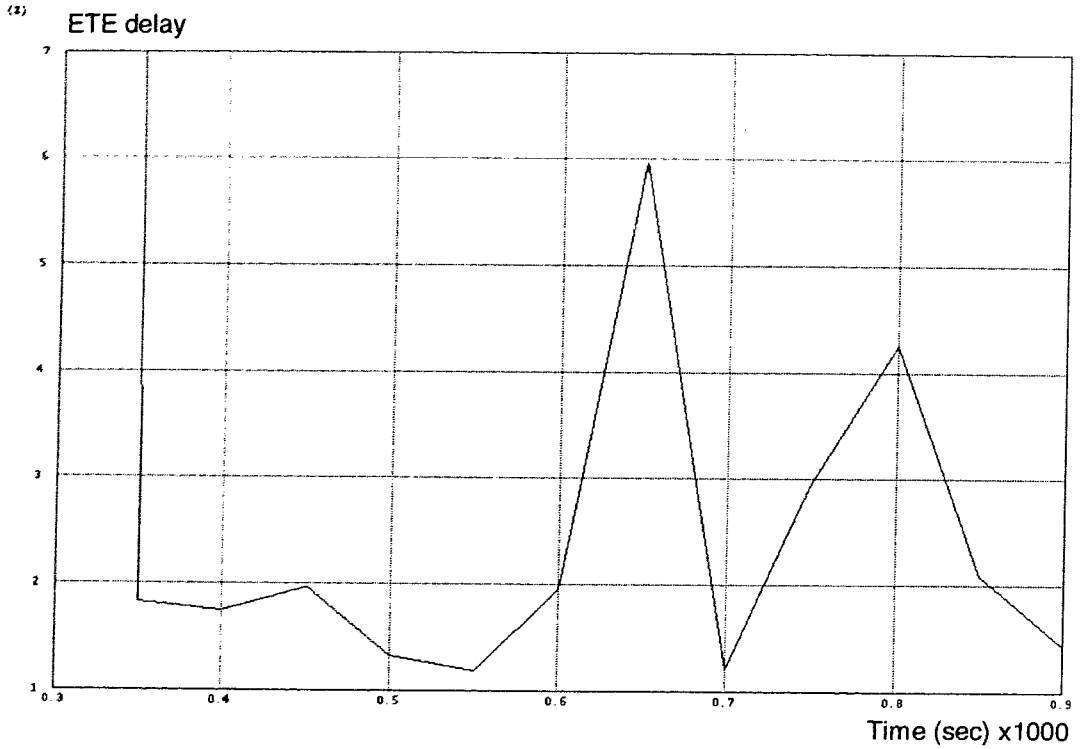


Figure 68. Base Mobile IP maximum ETE delay with average interarrival times ranging between 0.35 and 0.9 seconds.

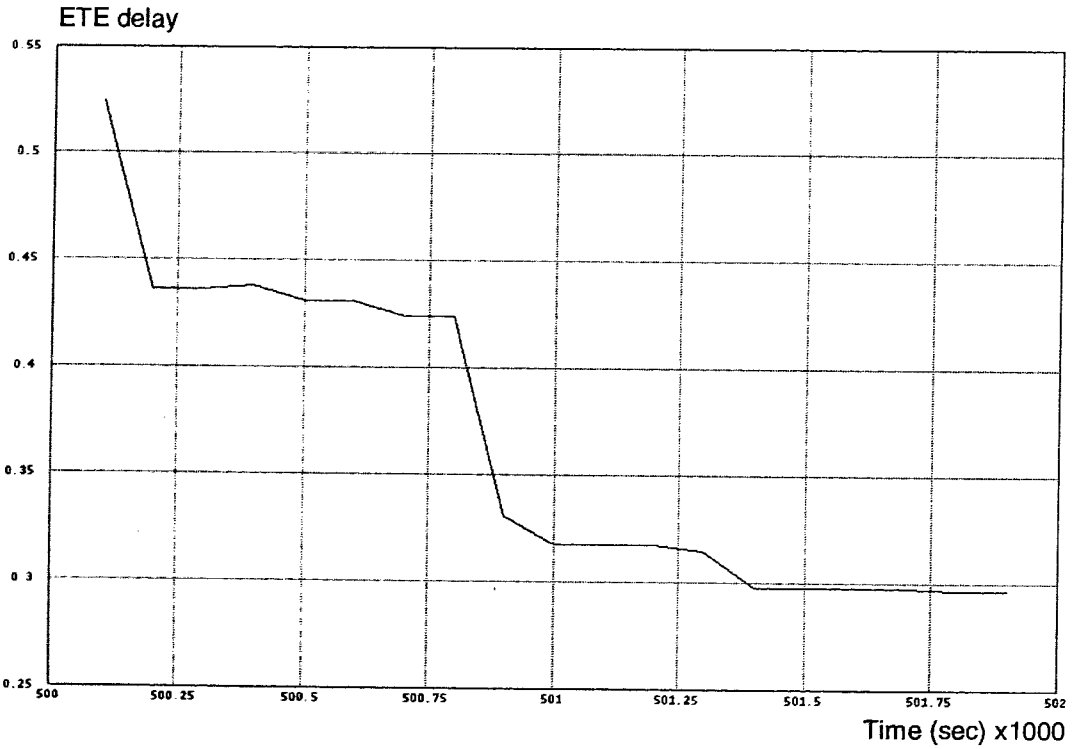


Figure 69. Base Mobile IP maximum ETE delay with handoff times ranging between 0.1 and 1.6 seconds before the beacon arrival.

APPENDIX C

ERROR CALCULATIONS

According to [20] the pdf of the duration of the interfade intervals for fades 20dB below the median is given by:

$$N(x) = 0.2087 \cdot \exp(-0.2087 \cdot x)$$

The mean duration of fades below 20dB is given in [19] to be 10^{-3} seconds

We used these values in a Matlab simulation to create a sample signal with successive periods of fades and non-fades. The signal was divided into 378 bit error control blocks which were checked for errors. If the number of errors exceeded a certain number, 25 in this case, the block was marked as bad. The Matlab program is shown in Figure 70.

From the Matlab simulation it was found that the distribution of the interarrival time of errors which could not be corrected by Reed Solomon error correction could be approximated by a Poisson distribution with mean and variance of 3.11 seconds.

```
% Number of fades
N=5000;
u=rand(N,1);

% Length of fades
e=19;

% Generate exponential distribution
for i=1:N
  ifd(i,1)=round(-19200/6.27*log(u(i,1)));
  err(i,1)=e;
end

WORD=378;
errors=0;
prev_block=1;
count=0;

% Concatenate fades and interfades
% Divide data into error correction blocks
for i=1:N
  count=count+ifd(i,1);
  offset=rem(count,WORD)+1;
  block=fix(count/WORD)+1;
  if block>prev_block,
    bad(prev_block,1)=errors;
    errors=0;
  end
  free=WORD-offset;
  if free>err(i,1),
    errors=errors+err(i,1);
    count=count+err(i,1);
    prev_block=block;
  else,
    errors=free;
    bad(block,1)=errors;
    errors=err(i,1)-free;
    count=count+err(i,1);
    prev_block=block+1;
  end
end

% Discover uncorrectable blocks
sym=1;
j=1;
for i=1:size(bad,1)
  if bad(i,1)>25,
    packet_error(i,1)=1;
    iet(j,1)=(i-sym)*0.0197;
    sym=i+1;
    j=j+1;
  else,
    packet_error(i,1)=0;
  end
end
```

Figure 70. Reed Solomon simulator.

APPENDIX D

SIMULATION SOURCE CODE

In this appendix we are providing the source code for the amended modules described in chapter 3.

...
...

Process Model Attributes

attribute	value	type	default value

Header Block

```
/* packet stream definitions */
#define IP_IN_STRM      0
#define SRC_IN_STRM     2
#define ICMP_IN_STRM    1
#define IP_OUT_STRM     0
#define REG_OUT_STRM    1
#define SRC_OUT_STRM    2

#define MIP_PORT        423
#define SRC_PORT        1
#define UDP_NUMBER      17
#define TCP_NUMBER      6
#define ICMP_NUMBER     1
#define ROUTER_AD       9
#define MAX_SUPPORT     5
#define ATTACHED_REQ    1
#define REGISTERED      2
#define ATTACHED_IND    3

#define ETE_STAT        0
#define PKSIZE_STAT     1

/* transition macros */
#define SRC_ARRVL (op_intrpt_type() == OPC_INTRPT_STRM && \
                   op_intrpt_strm() == SRC_IN_STRM)

#define IP_ARRVL  (op_intrpt_type() == OPC_INTRPT_STRM && \
                   op_intrpt_strm() == IP_IN_STRM)

#define ICMP_ARRVL (op_intrpt_type() == OPC_INTRPT_STRM && \
                    op_intrpt_strm() == ICMP_IN_STRM)
```

State Variable Block

```
Gshandle    \reg_gsh, \reg_air_gsh;
Ici*        \ip_ici_ptr, \tcp_ici_ptr;
int         \src_net, \src_node;
int         \reg_req;
Packet*     \reg_pkptr;

int         \reg_mn_net[MAX_SUPPORT], \reg_mn_node[MAX_SUPPORT];
int         \reg_fa_net[MAX_SUPPORT], \reg_fa_node[MAX_SUPPORT];

int         \att_mn_net[MAX_SUPPORT], \att_mn_node[MAX_SUPPORT];
int         \att_ha_net[MAX_SUPPORT], \att_ha_node[MAX_SUPPORT];
int         \att_status[MAX_SUPPORT];
```

...
...

Temporary Variable Block

```
Packet*     pkptr;
Packet*     info_pkptr;
double      ete_delay, reg_delay, reg_air_delay;
Ici*        ici_ptr=OPC_NIL;
Objid       ip_objid;
int         protocol;
char        error_string[512];
int         type;
int         flags;
int         code;
int         dest_port;
int         i, j;
int         care_of_net, care_of_node;
Objid       parent_id;
int         mn_net, mn_node;
int         handoff;
int         from_net, from_node;
int         dest_net, dest_node;
int         ha_net, ha_node,
```

Function Block

forced state init

attribute	*value*	*type*	*default value*
name	init	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs init

```
/* ete_gsh = op_stat_global_reg ("ete_delay"); */
reg_gsh = op_stat_global_reg ("reg_delay");
reg_air_gsh = op_stat_global_reg ("reg_air_delay");

ip_ici_ptr = op_ici_create ("ip_encap_req");
tcp_ici_ptr = op_ici_create ("ip_encap_ind");

parent_id = op_topo_parent (op_id_self());
ip_objid = op_topo_child (parent_id, OPC_OBJTYPE_QUEUE, 0);
op_ima_obj_attr_get (ip_objid, "net_num0", &src_net);
op_ima_obj_attr_get (ip_objid, "node_num0", &src_node);

reg_req = 0;

for (i=0; i<MAX_SUPPORT; i++)
    {
    att_mn_net[i] = -1;
    reg_mn_net[i] = -1;
    att_status[i] = -1;
    }
```

...
...

transition init -> idle			
attribute	*value*	*type*	*default value*
name	tr_0	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

forced state xmt			
attribute	*value*	*type*	*default value*
name	xmt	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs xmt

```
pkptr = op_pk_get (SRC_IN_STRM);

ici_ptr = op_intrpt_ici();
op_ici_attr_get (ici_ptr, "dest_net", &dest_net);
op_ici_attr_get (ici_ptr, "dest_node", &dest_node);

op_ici_attr_set (ip_ici_ptr, "dest_net", dest_net);
op_ici_attr_set (ip_ici_ptr, "dest_node", dest_node);
op_ici_attr_set (ip_ici_ptr, "protocol", TCP_NUMBER);
op_ici_install (ip_ici_ptr);

op_pk_stamp (pkptr);
op_stat_local_write (PKSIZE_STAT, op_pk_total_size_get (pkptr));

op_pk_send (pkptr, IP_OUT_STRM);
```

transition xmt -> idle			
attribute	*value*	*type*	*default value*
name	tr_7	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

unforced state idle			
attribute	*value*	*type*	*default value*
name	idle	string	st
enter execs	(empty)	textlist	(empty)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

...
...

transition idle -> xmt			
attribute	*value*	*type*	*default value*
name	tr_6	string	tr
condition	SRC_ARRVL	string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

transition idle -> rcv			
attribute	*value*	*type*	*default value*
name	tr_10	string	tr
condition	IP_ARRVL	string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

transition idle -> icmp			
attribute	*value*	*type*	*default value*
name	tr_20	string	tr
condition	ICMP_ARRVL	string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

forced state rcv			
attribute	*value*	*type*	*default value*
name	rcv	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs rcv

```
pkptr = op_pk_get (IP_IN_STRM);

ici_ptr = op_intrpt_ici();
if (ici_ptr == OPC_NIL)
    {
    sprintf (error_string, "simulation aborted; error in object (%d)",
        op_id_self());
    op_sim_end (error_string, "amip_tcp_mn: required ICI not received", " ", " ");
    }

op_ici_attr_get (ici_ptr, "protocol", &protocol);
op_ici_attr_get (ici_ptr, "src_net", &from_net);
op_ici_attr_get (ici_ptr, "src_node", &from_node);

if (protocol == TCP_NUMBER)
    {
    ete_delay = op_sim_time () - op_pk_stamp_time_get (pkptr);
    op_stat_local_write (ETE_STAT, ete_delay);

    op_ici_attr_set (tcp_ici_ptr, "src_net", from_net);
```

```
        op_ici_attr_set (tcp_ici_ptr, "src_node", from_node);
        op_ici_install (tcp_ici_ptr);
        op_pk_send (pkptr, SRC_OUT_STRM);
        }
else if (protocol == UDP_NUMBER)
        {
        op_pk_nfd_get (pkptr, "dest_port", &dest_port);
        op_pk_nfd_get (pkptr, "care_of_net", &care_of_net);
        op_pk_nfd_get (pkptr, "care_of_node", &care_of_node);
        op_pk_nfd_get (pkptr, "home_agent_net", &ha_net);
        op_pk_nfd_get (pkptr, "home_agent_node", &ha_node);
        op_pk_nfd_get (pkptr, "type", &type);

        /* MN came home */
        if ((care_of_net==ha_net) && (care_of_node==ha_node))
                {
                op_pk_nfd_get (pkptr, "home_addr_net", &mn_net);
                op_pk_nfd_get (pkptr, "home_addr_node", &mn_node);

                reg_air_delay = op_sim_time() - op_pk_stamp_time_get(pkptr);
                op_stat_global_write (reg_air_gsh, reg_air_delay);
                op_pk_destroy (pkptr);

                i = 0;
                while ((i<MAX_SUPPORT) && ((reg_mn_net[i] != mn_net) || (reg_mn_node[i] != mn_node)))
                        i++;
                if (i<MAX_SUPPORT)
                        {
                        reg_mn_net[i] = -1;
                        reg_mn_node[i] = -1;
                        info_pkptr = op_pk_create_fmt ("reg_info");
                        op_pk_nfd_set (info_pkptr, "type", REGISTERED);
                        op_pk_nfd_set (info_pkptr, "mn_net", -1);
                        op_pk_nfd_set (info_pkptr, "mn_node", -1);
                        op_pk_nfd_set (info_pkptr, "agent_net", -1);
                        op_pk_nfd_set (info_pkptr, "agent_node", -1);
                        op_pk_nfd_set (info_pkptr, "index", i);
                        op_pk_send (info_pkptr, REG_OUT_STRM);

                        pkptr = op_pk_create_fmt ("mip_reg");
                        op_pk_nfd_set (pkptr, "type", 3);
                        op_pk_nfd_set (pkptr, "flags", 1);
                        op_pk_nfd_set (pkptr, "home_addr_net", mn_net);
                        op_pk_nfd_set (pkptr, "home_addr_node", mn_node);
                        op_pk_nfd_set (pkptr, "home_agent_net", src_net);
                        op_pk_nfd_set (pkptr, "home_agent_node", src_node);
                        op_pk_nfd_set (pkptr, "care_of_net", -1);
                        op_pk_nfd_set (pkptr, "care_of_node", -1);
                        op_pk_nfd_set (pkptr, "src_port", MIP_PORT);
                        op_pk_nfd_set (pkptr, "dest_port", SRC_PORT);

                        op_ici_attr_set (ip_ici_ptr, "dest_net", mn_net);
                        op_ici_attr_set (ip_ici_ptr, "dest_node", mn_node);
                        op_ici_attr_set (ip_ici_ptr, "protocol", UDP_NUMBER);
                        op_ici_install (ip_ici_ptr);

                        op_pk_stamp (pkptr);

                        op_pk_send (pkptr, IP_OUT_STRM);
```

```
                reg_fa_net[i] = -1;
                reg_fa_node[i] = -1;
                }
        }
    /* Foreign Agent? */
    else if ((care_of_net == src_net) && (care_of_node == src_node))
        {
        /* From MN */
        if (type == 1)
            {
            i = 0;
            while ((att_mn_net[i] != -1) && (i < MAX_SUPPORT) && ((att_mn_net[i] != from_net) || (att_mn_node[i] != from_
                i++;
            if (i < MAX_SUPPORT)
                {
                op_pk_nfd_get (pkptr, "home_addr_net", &att_mn_net[i]);
                op_pk_nfd_get (pkptr, "home_addr_node", &att_mn_node[i]);
                op_pk_nfd_get (pkptr, "home_agent_net", &att_ha_net[i]);
                op_pk_nfd_get (pkptr, "home_agent_node", &att_ha_node[i]);
                op_ici_attr_set (ip_ici_ptr, "dest_net", att_ha_net[i]);
                op_ici_attr_set (ip_ici_ptr, "dest_node", att_ha_node[i]);
                op_ici_attr_set (ip_ici_ptr, "protocol", UDP_NUMBER);
                op_ici_install (ip_ici_ptr);

                reg_air_delay = op_sim_time() - op_pk_stamp_time_get(pkptr);
                op_stat_global_write (reg_air_gsh, reg_air_delay);
                op_pk_stamp (pkptr);

                op_pk_send (pkptr, IP_OUT_STRM);
                info_pkptr = op_pk_create_fmt ("reg_info");
                op_pk_nfd_set (info_pkptr, "type", ATTACHED_REQ);
                op_pk_nfd_set (info_pkptr, "mn_net", att_mn_net[i]);
                op_pk_nfd_set (info_pkptr, "mn_node", att_mn_node[i]);
                op_pk_nfd_set (info_pkptr, "agent_net", att_ha_net[i]);
                op_pk_nfd_set (info_pkptr, "agent_node", att_ha_node[i]);
                op_pk_nfd_set (info_pkptr, "index", i);
                op_pk_send (info_pkptr, REG_OUT_STRM);
                att_status[i] = 1;
                }
            else
                op_pk_destroy (pkptr);
            }
        /* From HA */
        else
            {
            op_pk_nfd_get (pkptr, "home_addr_net", &mn_net);
            op_pk_nfd_get (pkptr, "home_addr_node", &mn_node);
            op_pk_nfd_get (pkptr, "flags", &flags);

            j = 0;
            while ((j<MAX_SUPPORT) && ((mn_net!=att_mn_net[j]) || (mn_node!=att_mn_node[j])))
                    j++;
            if ((j<MAX_SUPPORT) && (flags==0))
                {
                att_status[j] = 2;

                op_ici_attr_set (ip_ici_ptr, "dest_net", mn_net);
                op_ici_attr_set (ip_ici_ptr, "dest_node", mn_node);
```

```
                    op_ici_attr_set (ip_ici_ptr, "protocol", UDP_NUMBER);
                    op_ici_install (ip_ici_ptr);

                    reg_delay = op_sim_time() - op_pk_stamp_time_get(pkptr);
                    op_stat_global_write (reg_gsh, reg_delay);
                    op_pk_stamp (pkptr);

                    op_pk_send (pkptr, IP_OUT_STRM);
                    }
               else if (j<MAX_SUPPORT)
                    {
                    att_mn_net[j] = -1;
                    att_mn_node[j] = -1;
                    att_ha_net[j] = -1;
                    att_ha_node[j] = -1;
                    info_pkptr = op_pk_create_fmt ("reg_info");
                    op_pk_nfd_set (info_pkptr, "type", ATTACHED_REQ);
                    op_pk_nfd_set (info_pkptr, "mn_net", att_mn_net[j]);
                    op_pk_nfd_set (info_pkptr, "mn_node", att_mn_node[j]);
                    op_pk_nfd_set (info_pkptr, "agent_net", att_ha_net[j]);
                    op_pk_nfd_set (info_pkptr, "agent_node", att_ha_node[j]);
                    op_pk_nfd_set (info_pkptr, "index", j);
                    op_pk_send (info_pkptr, REG_OUT_STRM);
                    }
               }
          }
     /* Home Agent */
     else
          {
          reg_delay = op_sim_time() - op_pk_stamp_time_get(pkptr);
          op_stat_global_write(reg_gsh, reg_delay);
          op_pk_nfd_get (pkptr, "home_addr_net", &mn_net);
          op_pk_nfd_get (pkptr, "home_addr_node", &mn_node);
          op_pk_destroy (pkptr);
          i = 0;
          while ((reg_mn_net[i] != -1) && (i<MAX_SUPPORT) && ((reg_mn_net[i] != mn_net) || (reg_mn_node[i] != mn_node)))
               i++;
          if (i<MAX_SUPPORT)
               {
               reg_mn_net[i] = mn_net;
               reg_mn_node[i] = mn_node;
               reg_fa_net[i] = care_of_net;
               reg_fa_node[i] = care_of_node;
               info_pkptr = op_pk_create_fmt ("reg_info");
               op_pk_nfd_set (info_pkptr, "type", REGISTERED);
               op_pk_nfd_set (info_pkptr, "mn_net", reg_mn_net[i]);
               op_pk_nfd_set (info_pkptr, "mn_node", reg_mn_node[i]);
               op_pk_nfd_set (info_pkptr, "agent_net", reg_fa_net[i]);
               op_pk_nfd_set (info_pkptr, "agent_node", reg_fa_node[i]);
               op_pk_nfd_set (info_pkptr, "index", i);
               op_pk_send (info_pkptr, REG_OUT_STRM);

               pkptr = op_pk_create_fmt ("mip_reg");
               op_pk_nfd_set (pkptr, "type", 3);
               op_pk_nfd_set (pkptr, "flags", 0);
               op_pk_nfd_set (pkptr, "home_addr_net", reg_mn_net[i]);
               op_pk_nfd_set (pkptr, "home_addr_node", reg_mn_node[i]);
               op_pk_nfd_set (pkptr, "home_agent_net", src_net);
               op_pk_nfd_set (pkptr, "home_agent_node", src_node);
```

...
...

```
                op_pk_nfd_set (pkptr, "care_of_net", reg_fa_net[i]);
                op_pk_nfd_set (pkptr, "care_of_node", reg_fa_node[i]);
200             op_pk_nfd_set (pkptr, "src_port", MIP_PORT);
                op_pk_nfd_set (pkptr, "dest_port", SRC_PORT);

                op_ici_attr_set (ip_ici_ptr, "dest_net", reg_fa_net[i]);
                op_ici_attr_set (ip_ici_ptr, "dest_node", reg_fa_node[i]);
205             op_ici_attr_set (ip_ici_ptr, "protocol", UDP_NUMBER);
                op_ici_install (ip_ici_ptr);

                op_pk_stamp (pkptr);

210             op_pk_send (pkptr, IP_OUT_STRM);
                }
            }
        }
    else
215     {
        sprintf (error_string, "simulation aborted; error in object (%d)",
            op_id_self());
        op_sim_end (error_string, "amip_tcp_agent: unexpected protocol number", " ", " ");
      }
```

transition rcv -> idle			
attribute	*value*	*type*	*default value*
name	tr_11	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

forced state icmp			
attribute	*value*	*type*	*default value*
name	icmp	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs icmp

```
    pkptr = op_pk_get (ICMP_IN_STRM);
    op_pk_destroy (pkptr);

    pkptr = op_pk_create_fmt ("icmp");
5
    /* ICMP Router Advertisement Fields */
    op_pk_nfd_set (pkptr, "router_net", src_net);
    op_pk_nfd_set (pkptr, "router_node", src_node);

10  op_ici_attr_set (ip_ici_ptr, "dest_net", src_net);
    op_ici_attr_set (ip_ici_ptr, "dest_net", 0);
    op_ici_attr_set (ip_ici_ptr, "protocol", ICMP_NUMBER);
```

...
...

```
op_ici_install (ip_ici_ptr);
op_pk_send (pkptr, IP_OUT_STRM);
```

transition **icmp -> idle**			
attribute	*value*	*type*	*default value*
name	tr_21	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

...
...

Process Model Attributes

attribute	*value*	*type*	*default value*
ha_net	promoted	integer	1
ha_node	promoted	integer	1
dl_conn_time	promoted	double	0.1

Header Block

```
/* packet stream definitions */
#define IP_IN_STRM      0
#define SRC_IN_STRM     1
#define IP_OUT_STRM     0
#define TCP_OUT_STRM    1

#define MIP_PORT        423
#define SRC_PORT        1
#define UDP_NUMBER      17
#define TCP_NUMBER      6
#define ICMP_NUMBER     1

#define ETE_STAT        0
#define PKSIZE_STAT     1

/* transition macros */
#define SRC_ARRVL (op_intrpt_type() == OPC_INTRPT_STRM && \
        op_intrpt_strm() == SRC_IN_STRM)

#define IP_ARRVL  (op_intrpt_type() == OPC_INTRPT_STRM && \
        op_intrpt_strm() == IP_IN_STRM)

#define SELF_INTRPT (op_intrpt_type() == OPC_INTRPT_SELF)
```

State Variable Block

```
Gshandle    \etc_gsh, \reg_gsh, \reg_air_gsh;
Ici*        \ip_ici_ptr, \tcp_ici_ptr;
int         \src_net, \src_node;
int         \ha_net, \ha_node;
int         \fa_net, \fa_node;
int         \reg_wait;
int         \reg_flag;
double      \reg_timeout;
double      \dl_conn_time;
int         \bak_fa_net, \bak_fa_node;
```

Temporary Variable Block

```
Packet*     pkptr;
double      ete_delay, reg_delay, reg_air_delay;
Ici*        ici_ptr=OPC_NIL;
Objid       ip_objid;
int         protocol;
char        error_string[512];
int         code;
int         dest_net, dest_node;
int         prev_fa_net, prev_fa_node;
```

...
...

```
10  Objid   parent_id;
    int     from_net, from_node;
```

forced state init			
attribute	*value*	*type*	*default value*
name	init	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs init

```
    /* ete_gsh = op_stat_local_reg ("ete_delay"); */
    reg_gsh = op_stat_global_reg ("reg_delay");
    reg_air_gsh = op_stat_global_reg ("reg_air_delay");

 5  ip_ici_ptr = op_ici_create ("ip_encap_req");
    tcp_ici_ptr = op_ici_create ("ip_encap_ind");

    op_ima_obj_attr_get (op_id_self(), "ha_net", &ha_net);
    op_ima_obj_attr_get (op_id_self(), "ha_node", &ha_node);
10  op_ima_obj_attr_get (op_id_self(), "dl_conn_time", &dl_conn_time);

    /* Get ip module's id to get our address */
    parent_id = op_topo_parent (op_id_self());
    ip_objid = op_topo_child (parent_id, OPC_OBJTYPE_QUEUE, 0);
15  op_ima_obj_attr_get (ip_objid, "net_num0", &src_net);
    op_ima_obj_attr_get (ip_objid, "node_num0", &src_node);

    reg_wait = 0;
    reg_flag = 0;
20  fa_net = -1;
    fa_node = -1;
    reg_timeout = 1;
```

transition init -> idle			
attribute	*value*	*type*	*default value*
name	tr_0	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

forced state xmt			
attribute	*value*	*type*	*default value*
name	xmt	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

...
...

enter execs **xmt**

```
pkptr = op_pk_get (SRC_IN_STRM);

ici_ptr = op_intrpt_ici();
op_ici_attr_get (ici_ptr, "dest_net", &dest_net);
op_ici_attr_get (ici_ptr, "dest_node", &dest_node);

if ((reg_flag) && (reg_wait==0))
    {
    op_ici_attr_set (ip_ici_ptr, "dest_net", dest_net);
    op_ici_attr_set (ip_ici_ptr, "dest_node", dest_node);
    op_ici_attr_set (ip_ici_ptr, "protocol", TCP_NUMBER);
    op_ici_install (ip_ici_ptr);

    op_pk_stamp (pkptr);
    op_stat_local_write (PKSIZE_STAT, op_pk_total_size_get (pkptr));

    op_pk_send (pkptr, IP_OUT_STRM);
    }
else
    {
    op_pk_destroy (pkptr);
    }
```

transition **xmt -> idle**

attribute	*value*	*type*	*default value*
name	tr_7	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

unforced state **idle**

attribute	*value*	*type*	*default value*
name	idle	string	st
enter execs	(empty)	textlist	(empty)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

transition **idle -> xmt**

attribute	*value*	*type*	*default value*
name	tr_6	string	tr
condition	SRC_ARRVL	string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

transition **idle -> rcv**

attribute	*value*	*type*	*default value*
name	tr_10	string	tr
condition	IP_ARRVL	string	

...
...

executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

transition idle -> timeout			
attribute	*value*	*type*	*default value*
name	tr_22	string	tr
condition	SELF_INTRPT	string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

forced state rcv			
attribute	*value*	*type*	*default value*
name	rcv	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs rcv

```
pkptr = op_pk_get (IP_IN_STRM);

ici_ptr = op_intrpt_ici();
if (ici_ptr == OPC_NIL)
    {
    sprintf (error_string, "simulation aborted; error in object (%d)",
        op_id_self());
    op_sim_end (error_string, "amip_tcp_mn: required ICI not received", " ", " ");
    }

op_ici_attr_get (ici_ptr, "protocol", &protocol);
if (protocol == TCP_NUMBER)
    {
    if (reg_flag)
        {
        ete_delay = op_sim_time() - op_pk_stamp_time_get(pkptr);
        op_stat_local_write (ETE_STAT, ete_delay);
        op_ici_attr_get (ici_ptr, "src_net", &from_net);
        op_ici_attr_get (ici_ptr, "src_node", &from_node);
        op_ici_attr_set (tcp_ici_ptr, "src_net", from_net);
        op_ici_attr_set (tcp_ici_ptr, "src_node", from_node);
        op_ici_install (tcp_ici_ptr);
        op_pk_send (pkptr, TCP_OUT_STRM);
        }
    else
        {
        op_pk_destroy (pkptr);
        }
    }
else if (protocol == ICMP_NUMBER)
    {
    prev_fa_net = fa_net;
    prev_fa_node = fa_node;
    op_pk_nfd_get (pkptr, "router_net", &fa_net);
```

...
...

```
35      op_pk_nfd_get (pkptr, "router_node", &fa_node);
        op_pk_destroy (pkptr);

        if ((!reg_flag) || (prev_fa_net != fa_net) || (prev_fa_node != fa_node))
            {
40          reg_wait = 9;
            op_intrpt_schedule_self (op_sim_time() + dl_conn_time, 0);
            }
        }
    else if ((protocol == UDP_NUMBER) && (reg_wait==1))
45      {
        reg_wait = 0;
        op_pk_nfd_get (pkptr, "flags", &code);
        if ((code == 0) || (code == 1))
            reg_flag = 1;
50      reg_air_delay = op_sim_time() - op_pk_stamp_time_get(pkptr);
        op_stat_global_write(reg_air_gsh, reg_air_delay);
        op_pk_destroy (pkptr);
        }
    else
55      {
        sprintf (error_string, "simulation aborted; error in object (%d)",
            op_id_self());
        op_sim_end (error_string, "amip_tcp_mn: unexpected protocol number", " ", " ");
      }
```

transition rcv -> idle

attribute	*value*	*type*	*default value*
name	tr_11	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

forced state timeout

attribute	*value*	*type*	*default value*
name	timeout	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs timeout

```
    if (reg_wait == 9)
        {
        reg_flag = 0;
        pkptr = op_pk_create_fmt ("mip_reg");
5
        /* Mobile-IP fields */
        op_pk_nfd_set (pkptr, "home_addr_net", src_net);
        op_pk_nfd_set (pkptr, "home_addr_node", src_node);
        op_pk_nfd_set (pkptr, "home_agent_net", ha_net);
10      op_pk_nfd_set (pkptr, "home_agent_node", ha_node);
        op_pk_nfd_set (pkptr, "care_of_net", fa_net);
```

...
...

```
        op_pk_nfd_set (pkptr "care_of_node", fa_node);
        if ((fa_net==ha_net) && (fa_node==ha_node))
            op_pk_nfd_set (pkptr, "life", 0);

        /* UDP fields */
        op_pk_nfd_set (pkptr, "src_port", SRC_PORT);
        op_pk_nfd_set (pkptr, "dest_port", MIP_PORT);

        /* ICI pointer for IP */
        op_ici_attr_set (ip_ici_ptr, "dest_net", fa_net);
        op_ici_attr_set (ip_ici_ptr, "dest_node", fa_node);
        op_ici_attr_set (ip_ici_ptr, "protocol", UDP_NUMBER);
        op_ici_install (ip_ici_ptr);

        op_pk_stamp (pkptr);

        op_pk_send (pkptr, IP_OUT_STRM);

        /* Error in registration packet */
        bak_fa_net = fa_net;
        bak_fa_node = fa_node;
        op_intrpt_schedule_self (op_sim_time() + reg_timeout, 0);
        reg_wait = 1;
        }
    else if (reg_flag == 0)
        {
        reg_flag = 0;
        pkptr = op_pk_create_fmt ("mip_reg");

        /* Mobile-IP fields */
        op_pk_nfd_set (pkptr, "home_addr_net", src_net);
        op_pk_nfd_set (pkptr, "home_addr_node", src_node);
        op_pk_nfd_set (pkptr, "home_agent_net", ha_net);
        op_pk_nfd_set (pkptr, "home_agent_node", ha_node);
        op_pk_nfd_set (pkptr, "care_of_net", bak_fa_net);
        op_pk_nfd_set (pkptr, "care_of_node", bak_fa_node);
        if ((bak_fa_net==ha_net) && (bak_fa_node==ha_node))
            op_pk_nfd_set (pkptr, "life", 0);

        /* UDP fields */
        op_pk_nfd_set (pkptr, "src_port", SRC_PORT);
        op_pk_nfd_set (pkptr, "dest_port", MIP_PORT);

        /* ICI pointer for IP */
        op_ici_attr_set (ip_ici_ptr, "dest_net", bak_fa_net);
        op_ici_attr_set (ip_ici_ptr, "dest_node", bak_fa_node);
        op_ici_attr_set (ip_ici_ptr, "protocol", UDP_NUMBER);
        op_ici_install (ip_ici_ptr);

        op_pk_stamp (pkptr);

        op_pk_send (pkptr, IP_OUT_STRM);

        op_intrpt_schedule_self (op_sim_time() + reg_timeout, 0);
        reg_wait = 1;
        }
```

...
...

transition **timeout -> idle**			
attribute	*value*	*type*	*default value*
name	tr_23	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

...
...

Header Block

```
#include "ip_frag_sup.h"

#define INSTREAM_IP      0
#define OUTSTREAM_IP     1
#define INSTREAM_NET     1
#define OUTSTREAM_NET    0

#define FROM_IP      op_intrpt_strm () == INSTREAM_IP
#define FROM_NET     op_intrpt_strm () == INSTREAM_NET
```

State Variable Block

```
int         \src_net0, \src_node0;
int         \src_net1, \src_node1;
int         \dgram_id;
Ici*        \ipinip_iciptr;
```

Temporary Variable Block

```
Packet*     ipinip_pkptr;
Packet*     pkptr;
int         dest_net;
int         dest_node;
int         ipinip;
Objid       ip_objid;
Ici*        iciptr;
int         data_len;
int         ipinip_src_net;
int         ipinip_src_node;
char        error_string [512];
int         protocol;
```

forced state ENCAP

attribute	*value*	*type*	*default value*
name	ENCAP	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs ENCAP

```
/* Obtain a packet arriving from the IP. */
pkptr = op_pk_get (INSTREAM_IP);

/* Get ICI pointer to find out if ipinip is needed */
iciptr = op_intrpt_ici ();
if (iciptr == OPC_NIL)
    {
    sprintf (error_string, "simulation aborted; error in object (%d)",
        op_id_self ());
    op_sim_end (error_string, "ip_inip: required ICI not received", "", "");
    }
op_ici_attr_get (iciptr, "ipinip", &ipinip);

if (ipinip)
```

...
...

```
15      {
        /* Obtain the length of the encapsulated packet (in bytes) */
        /* (Note: up to 7 bits may be unmodeled for packets which do */
        /* not contain an integral number of bytes. */
        data_len = op_pk_total_size_get (pkptr) / 8;
20
        /* Create an IP packet and encapsulate the */
        /* new arrival within its data field. */
        /* The data field has a modeled size of zero */
        /* in order to allow breaking of the packet into */
25      /* pre-determined arbitrarily small sizes. */
        /* The bulk size attribute of the IP packet will */
        /* instead be used to model the size of the */
        /* encapsulated data. */
        ipinip_pkptr = op_pk_create_fmt ("ip_dgram");
30      op_pk_nfd_set (ipinip_pkptr, "data", pkptr);

        /* Set the bulk size of the IP packet to model the */
        /* space occupied by the encapsulated data. */
        op_pk_bulk_size_set (ipinip_pkptr, data_len * 8);
35
        op_ici_attr_get (iciptr, "dest_net", &dest_net);
        op_ici_attr_get (iciptr, "dest_node", &dest_node);
        op_pk_nfd_set (ipinip_pkptr, "dest_net", dest_net);
        op_pk_nfd_set (ipinip_pkptr, "dest_node", dest_node);
40
        /* Set the source address */
        op_pk_nfd_set (ipinip_pkptr, "src_net", src_net0);
        op_pk_nfd_set (ipinip_pkptr, "src_node", src_node0);

45      /* set the length attributes of the IP packet. */
        op_pk_nfd_set (ipinip_pkptr, "orig_len", data_len);
        op_pk_nfd_set (ipinip_pkptr, "frag_len", data_len);

        /* Indicate that the packet is not yet fragmented. */
50      op_pk_nfd_set (ipinip_pkptr, "frag", 0);

        /* Assign a unique identity to this packet (unique among */
        /* packets injected into the net by this host. */
        op_pk_nfd_set (ipinip_pkptr, "ident", dgram_id++);
55
        /* Set the protocol to IPINIP */
        op_pk_nfd_set (ipinip_pkptr, "protocol", 4);

        /* Forward the ipinip packet to the net */
60      op_pk_send (ipinip_pkptr, OUTSTREAM_NET);
    }
  else
    {
    /* Forward the packet to the net */
65  op_pk_send (pkptr, OUTSTREAM_NET);
    }
```

transition **ENCAP -> WAIT**			
attribute	*value*	*type*	*default value*
name	tr_31	string	tr

...
...

condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

forced state **INIT**

attribute	*value*	*type*	*default value*
name	INIT	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **INIT**

```
/* Obtain the object id of the IP module */
/* It is expected to be directly connected to */
/* this module's only output. */
ip_objid = op_topo_assoc (op_id_self(), OPC_TOPO_ASSOC_OUT,
                          OPC_OBJMTYPE_MODULE, OUTSTREAM_IP);

/* Use the object id to get the address */
op_ima_obj_attr_get (ip_objid, "net_num0", &src_net0);
op_ima_obj_attr_get (ip_objid, "node_num0", &src_node0);
op_ima_obj_attr_get (ip_objid, "net_num1", &src_net1);
op_ima_obj_attr_get (ip_objid, "node_num1", &src_node1);

/* Create an Ici for communication with the higher layer. */
ipinip_iciptr = op_ici_create ("ip_inip_ind");
```

transition **INIT -> WAIT**

attribute	*value*	*type*	*default value*
name	tr_26	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

unforced state **WAIT**

attribute	*value*	*type*	*default value*
name	WAIT	string	st
enter execs	(empty)	textlist	(empty)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

transition **WAIT -> ENCAP**

attribute	*value*	*type*	*default value*
name	tr_30	string	tr
condition	FROM_IP	string	
executive		string	

color	RGB333	color	RGB333
drawing style	spline	toggle	spline

transition WAIT -> DECAP			
attribute	*value*	*type*	*default value*
name	tr_34	string	tr
condition	FROM_NET	string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

forced state DECAP			
attribute	*value*	*type*	*default value*
name	DECAP	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs DECAP

```
    /* Obtain a packet arriving from the network */
    ipinip_pkptr = op_pk_get (INSTREAM_NET);

    /* Check if IPINIP & Destination */
 5  op_pk_nfd_get (ipinip_pkptr, "protocol", &protocol);
    op_pk_nfd_get (ipinip_pkptr, "dest_net", &dest_net);
    op_pk_nfd_get (ipinip_pkptr, "dest_node", &dest_node);

    if ((protocol == 4) && (((dest_net == src_net0) && (dest_node ==src_node0)) ||
10      ((dest_net == src_net1) && (dest_node == src_node1))))
        {
        /* Decapsulate the IP packet */
        op_pk_nfd_get (ipinip_pkptr, "data", &pkptr);
      op_pk_nfd_get (ipinip_pkptr, "src_net", &ipinip_src_net);
15    op_pk_nfd_get (ipinip_pkptr, "src_node", &ipinip_src_node);

        /* Discard the outer IP packet. */
        op_pk_destroy (ipinip_pkptr);

20      /* Forward the data to the ip module */
        op_ici_attr_set (ipinip_iciptr, "ipinip", 1);
        op_ici_attr_set (ipinip_iciptr, "src_net", ipinip_src_net);
        op_ici_attr_set (ipinip_iciptr, "src_node", ipinip_src_node);
        op_ici_install (ipinip_iciptr);
25      op_pk_send (pkptr, OUTSTREAM_IP);
      }
    else
        {
        op_pk_send (ipinip_pkptr, OUTSTREAM_IP);
30      }
```

...
...

transition DECAP -> WAIT			
attribute	*value*	*type*	*default value*
name	tr_35	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

...
...

Process Model Attributes

attribute	*value*	*type*	*default value*
handoff1	promoted	double	296 (sec)
handoff2	promoted	double	650 (sec)
handoff3	promoted	double	950 (sec)
freq0	promoted	double	30 (MHz)
freq1	promoted	double	40 (MHz)
freq2	promoted	double	35 (MHz)
freq3	promoted	double	45 (MHz)

Header Block

```
#define SELF_INTRPT (op_intrpt_type() == OPC_INTRPT_SELF)
```

State Variable Block

```
Objid   \tx_radioch_id, \rx_radioch_id;
double  \freq[4];
double  \handoff1, \handoff2, \handoff3;
int     \hop;
```

Temporary Variable Block

```
Objid     node_id, radio_tx_id, radio_rx_id;
Compcode  comp_code;
double    freq0, freq1, freq2, freq3;
```

unforced state init

attribute	*value*	*type*	*default value*
name	init	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

enter execs init

```
op_ima_obj_attr_get(op_id_self(), "handoff1", &handoff1);
op_ima_obj_attr_get(op_id_self(), "handoff2", &handoff2);
op_ima_obj_attr_get(op_id_self(), "handoff3", &handoff3);
op_ima_obj_attr_get(op_id_self(), "freq0", &freq0);
op_ima_obj_attr_get(op_id_self(), "freq1", &freq1);
op_ima_obj_attr_get(op_id_self(), "freq2", &freq2);
op_ima_obj_attr_get(op_id_self(), "freq3", &freq3);


node_id = op_id_parent (op_id_self ());
radio_tx_id = op_id_from_name (node_id, OPC_OBJTYPE_RATX, "radio_tx");
radio_rx_id = op_id_from_name (node_id, OPC_OBJTYPE_RARX, "radio_rx");
tx_radioch_id = op_topo_child (radio_tx_id, OPC_OBJTYPE_RATXCH, 0);
rx_radioch_id = op_topo_child (radio_rx_id, OPC_OBJTYPE_RARXCH, 0);
op_intrpt_schedule_self (op_sim_time() + handoff1, 0);

freq[0] = freq0;
freq[1] = freq1;
```

...
...

```
    freq[2] = freq2;
20  freq[3] = freq3;
    hop = 0;
    op_stat_local_write (0, handoff1);
```

transition **init -> handoff**			
attribute	*value*	*type*	*default value*
name	tr_2	string	tr
condition	SELF_INTRPT	string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

forced state **handoff**			
attribute	*value*	*type*	*default value*
name	handoff	string	st
enter execs	(See below.)	textlist	(See below.)
exit execs	(empty)	textlist	(empty)
status	forced	toggle	unforced

enter execs **handoff**

```
    hop ++;
    if (hop == 1)
        {
        op_intrpt_schedule_self (op_sim_time() + handoff2, 0);
5       }
    else if (hop == 2)
        {
        op_intrpt_schedule_self (op_sim_time() + handoff3, 0);
        }
10
    comp_code = op_ima_obj_attr_set (tx_radioch_id, "min frequency", freq[hop]);
    if (OPC_COMPCODE_FAILURE == comp_code)
        op_sim_end ("set radio transmit frequency failed", "", "", "");
    comp_code = op_ima_obj_attr_set (rx_radioch_id, "min frequency", freq[hop]+1);
15  if (OPC_COMPCODE_FAILURE == comp_code)
        op_sim_end ("set radio receive frequency failed", "", "", "");
```

transition **handoff -> idle**			
attribute	*value*	*type*	*default value*
name	tr_5	string	tr
condition		string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline

...
...

unforced state **idle**			
attribute	*value*	*type*	*default value*
name	idle	string	st
enter execs	(empty)	textlist	(empty)
exit execs	(empty)	textlist	(empty)
status	unforced	toggle	unforced

transition **idle -> handoff**			
attribute	*value*	*type*	*default value*
name	tr_4	string	tr
condition	SELF_INTRPT	string	
executive		string	
color	RGB333	color	RGB333
drawing style	spline	toggle	spline