Abstract

Title of Dissertation: A framework for integrating Mobile Hosts within the Internet

Pravin Bhagwat, Doctor of Philosophy, 1995

Dissertation directed by: Professor Satish K. Tripathi
Department of Computer Science

Host mobility and wireless access are two emerging design considerations that pose challenging problems at all layers of the networking protocol stack. This dissertation investigates their impact on the design of link, network, and transport layer protocols. At the network layer, we have designed and implemented a new routing architecture that allows the current set of Internet standards to support routing to mobile hosts. At the link and transport layers, we have designed mechanisms to improve throughput over error-prone wireless channels.

At the network layer, the most crucial problem is that of routing. The existing Internet routing mechanisms cannot route packets to hosts whose points of attachment to the network change over time. Exploiting IP's Loose Source Route option, we have designed and implemented a routing scheme which provides location independent network access to TCP/IP compliant mobile hosts. It also allows mobile hosts equipped with multiple network interfaces to dynamically migrate active network sessions from one network interface to another. The proposed scheme only requires the addition of two new entity types, Mobile Routers and Mobile Access Stations. These entities perform all required mobility-aware functions, such as address translation, user tracking and location management. No modifications to existing host or router software are required.

Although MobileIP provides continuous network connectivity to mobile hosts, the effects of host movement and wireless medium characteristics are often visible at the transport layer. We

consider the effect of wireless medium characteristics on the performance of Transmission Control Protocol (TCP) sessions. Unlike wired networks, packets transmitted on wireless channels are often subject to burst errors which cause back to back packet losses. We show that TCP's error-recovery mechanisms perform poorly when packets from a TCP session are subject to burst errors. Unlike other approaches which require modification to TCP, our solution requires enhancements only at the wireless link layer, thus making it applicable to other transport protocols as well. We use a Channel State Dependent Packet (CSDP) scheduler which takes wireless channel characteristics into consideration in making packet dispatching decisions. Our results show that the CSDP technique provides improved throughput, better channel utilization, and fairness among multiple TCP streams.

A framework for integrating Mobile Hosts within the Internet

by

Pravin Bhagwat

Dissertation submitted to the Faculty of the Graduate School
of The University of Maryland in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
1995

Advisory Committee:

Professor Satish K. Tripathi, Chairman/Advisor
Prof. Ashok K. Agrawala
Prof. Mike Franklin
Prof. Richard Gerber
Prof. Joseph JaJa

Dedication

To my parents

Acknowledgements

I would like to express my sincere appreciation to my advisor, Prof. Satish Tripathi, for his experienced guidance and encouragement throughout the course of my study and research at the University of Maryland. He always believed in my abilities, which made me think much harder and challenge myself much more. I am also deeply indebted to Charles Perkins for introducing me to the area of Mobile Networking and providing me crucial support throughout the evolution of this work.

Many thanks should go to the research staff members at the IBM Watson Research Center, and most notably Dr. Peter Hortensius, Dr. Arvind Krishna, and Dr. David Bantz for their assistance. I would like to thank Dr. Yakov Rekhter and Dr. Partha Bhattacharya for their help and guidance.

I wish to express my deep gratitude to Dr. Ashok Agrawala for serving on my proposal and dissertation committees, for helping me in many ways. I thank Dr. Rich Gerber, who was on my proposal and dissertation committees, for his feedback and support. I also thank the other members of my dissertation committee, Dr. Mike Franklin and Dr. Joseph JaJa, for their interest in this dissertation and for their feedback. Thanks are also due to Dr. Christos Faloutsos for serving on my proposal committee.

I especially acknowledge the help and support accorded to me by the staff of my department and by many friends and colleagues. Special thanks are due to Debanjan Saha, Ibrahim Matta, Cynthia Davis, Rohit Dubey, Gagan Agrawal, Cengiz Alaettinoğlu, Sedat Akyurek, Sarit Mukherjee, Partho Mishra, Dheeraj Sanghi, Sanjeev Setia, and James da Silva, and also to Dr. Nancy Lindley and Janet Doherty, who have helped me in numerous administrative affairs.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Since the invention of the computer, we have been witnessing major shifts in computing paradigms almost every decade. Mobile Computing is expected to be the next major paradigm in the evolution of computing. The 70s was the time when the concept of time sharing and multiprogramming revolutionized main-frame computer industry. In the 80's we saw distributed computing platforms emerging as a powerful and economic alternative to centralized main-frame systems. The driving force behind this revolution was the microprocessor technology which shrunk the size of the computing device to a point where it could be used as a desktop commodity. Today clusters of desktop workstations interconnected through high speed Local Area Networks constitute the core of network computing environments world-wide.

Technological advances of the 90s (see Figure 1.1) are bringing about two major changes that have the potential to reshape the existing form of network computing into what is more popularly known as Mobile Computing. First, portable computers which are as powerful as some desktop workstations in terms of computing power, memory, display, and disk storage, are beginning to appear. Second, with the availability of wireless network interfaces, users of laptop computers are no longer required to remain confined within the wired LAN premises to get network access. With the confluence of wireless and VLSI technologies, we are poised for yet another computing revolution.

Mobile personal computing devices are becoming ubiquitous as their prices drop and their capabilities increase. With the growing dependence of day to day computing on the distributed information base, providing network attachment to these devices is an essential requirement. Using wireless network interfaces, mobile devices can be connected to the Internet in the same way as desktop machines are connected using ethernet, token-ring or point-to-point links. The major difference, however, is that mobile devices can move while in operation, which means that their point of attachment to the network can change from time to time. From a network's view point, host movement constitutes a change in the network topology. It is natural that mobile users desire uninterrupted access to all networking services even while moving. Unfortunately, neither the Internet protocol suite nor the OSI network architecture can provide this functionality. The assumption that end systems are stationary lies at the very foundation of the Internet and OSI network architectures. This is a serious problem, since it is not possible to deploy a new "mobility-aware" protocol stack in the Internet which already consists of over 80 million hosts. The challenge lies in finding a solution that allows mobile end-systems to function efficiently within the Internet architecture without requiring modifications to the existing infrastructure and host software. In this dissertation we address both, the functional and efficiency aspects of the mobile networking problem.

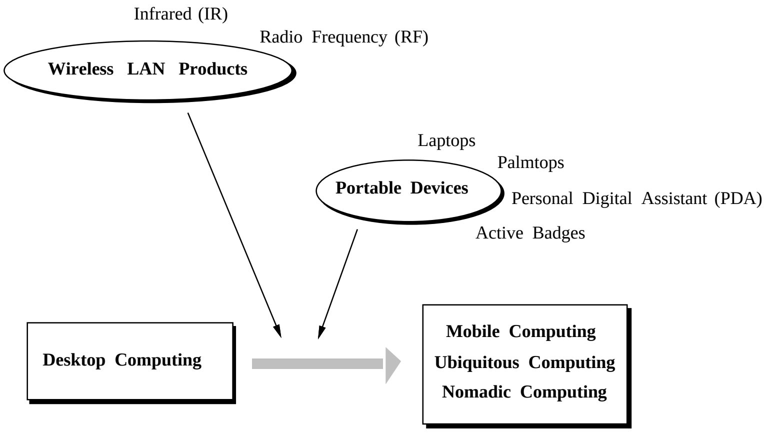


Figure 1.1: Evolution of Mobile Computing

At the network layer, we have identified a set of functions that play a fundamental role in the design of any mobile networking solution. Using these function primitives, we have proposed a network layer architecture that enables smooth integration of mobile end systems within the existing Internet. Our proposed modification to the Internet Protocol (IP), which we refer to as MobileIP, enables mobile hosts to change their network attachment points without disrupting any active network sessions. The key feature of our design is that all required functionalities for processing and managing mobility information is embedded in specialized entities, called Mobile Router and Mobile Access Station. Contemporary solutions either require changes to the existing network architecture [49] or introduce new encapsulation protocols [22, 51] to handle this problem. Our approach, since it exploits existing mechanisms available within IP, is completely transparent to the transport and higher layers and does not require any changes to existing internet hosts and routers.

MobileIP only provides a mechanism for packet exchange among mobile and stationary end-systems. The efficiency of transport layer sessions among stationary and mobile hosts depends crucially on the error characteristics of the wireless medium and the Medium Access Control (MAC) protocol employed at the wireless link layer. Unlike wired networks, packets transmitted on wireless channels are often subject to burst errors which cause back to back packet losses. This phenomenon results in severe performance problems; the end-to-end transport protocols incorrectly interpret these losses as signs of network congestion and react by throttling their transmission rate. Existing solutions for this problem [15, 5, 55, 2] require modifications to the transport layer protocol software of all stationary and mobile hosts, which makes their deployment difficult in practice. We have demonstrated that this problem can be alleviated by changing the packet dispatcher at the wireless link layer. The new class of packet dispatchers, which we call CSDP schedulers, explicitly take wireless channel characteristics into consideration in making packet dispatching decisions. By employing a CSDP scheduler at the wireless link layer, significant performance improvement can be achieved in typical wireless LAN configurations. Since our solution requires modification only at the wireless link layer, it allows existing transport protocols and applications to run unmodified over wireless channels.

In this dissertation, we focus only on the link, network, and transport layer issues that pertain to the design of mobile networking systems. Our experience, however, has led us to believe that the realm of the problems involved is not limited to these layers [11]. In fact, the domain of related problems spans the entire protocol stack (see Figure 1.2). The main reason is that the Internet and OSI protocols were standardized prior to the technological revolution that shaped today's computing environment. Naturally, a majority of assumptions made during that period no longer hold today. Below, we briefly describe how the mobile computing environment affects design choices for network protocols. Starting from the physical layer, we traverse up the Internet protocol stack[1] pointing out deficiencies and need for new services at each layer.

1.1 Protocol Stack and Host Mobility

1.1.1 Physical Layer

It is at the physical layer where two directly communicating entities interface with each other. This layer is responsible for transporting bits from one end of the communication channel to another. The design issues at the physical layer deal with hardware, mechanical and electrical interfaces, and the characteristics of the physical transmission medium. Various physical media can be used for the actual transmission, such as twisted pair, coaxial cable, fiber channel, wireless, etc. Transmission mediums widely differ in term of their propagation characteristics, channel capacity, and error rate.

[1]Unlike OSI stack which has 7-layers, the internet protocol stack has only 5-layers; the physical, link, network, transport and application. Since the Internet architecture is the primary focus of our investigation, we limit our discussion to the internet protocol stack.

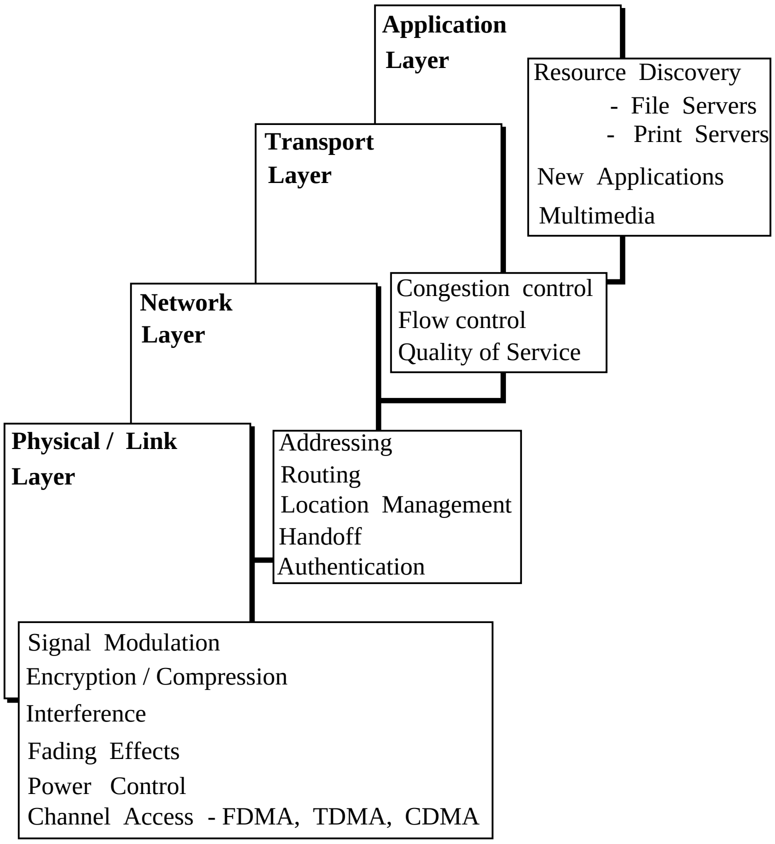


Figure 1.2: Mobility and the Network Protocol Stack

At the physical layer, channel characteristics vary with the location of the user and, because of mobility, also vary in time. A mobile radio link is hindered by a number of propagation mechanisms, namely, multipath scattering from objects near the mobile transmitter, shadowing by dominant obstacles, and attenuation mechanisms on the propagation path between transmitter and receiver. The use of a wireless channel for data communication is not a new development. In fact, the wireless technology has been in use over satellite, microwave, and cordless channels for over half a century. What is new about wireless is its application for providing local and wide area digital Personal Communication Service (PCS). Physical layer design considerations for operation in wireless PCS bands are quite different from those used in traditional wireless applications. What is now required is a low-power technology that has the appropriate compromises for voice and moderate-rate data to small, light-weight, economical, pocket-size personal computing devices that can be used for tens of hours without replacing batteries.

1.1.2 Link Layer

The link layer is responsible for reliably transmitting blocks of information over the raw physical medium. This is accomplished by having the sender break the input data up into data frames, transmit the frames sequentially, and process the acknowledgement frames sent back by the receiver. Since transmitted data could be lost due to noise present on the channel, the link layer provides mechanisms to protect data frames through integrity checks.

The link layer consists of two sub-layers: Media Access Control(MAC) and Logical Link Control (LLC). The MAC sublayer directly interfaces with the physical layer and is responsible for framing and checksum. For multiple access mediums, MAC protocols also arbitrate access to the shared medium. The LLC layer is primarily responsible for multiplexing different network layer protocols on the same link and providing recovery from lost frames.

Wireless medium is different in many ways to wired media and differences give rise to many unique problems and solutions. Some important differences are:

- The strength of a radio signal decreases as the square of the distance from the transmitter (in some systems the decrease is with the fourth power of the distance). Since transmit power level is usually high, stations, while transmitting, cannot monitor the channel to detect if a collision is happening. As a result, ethernet-like CSMA/CD cannot be used as a MAC protocol in wireless medium. Also, since transmitters are mobile, token based access protocols cannot be used either.

- Within the frequency band allocated for PCS, radio waves reflect off solid objects, giving rise to what is known as multi-path Raleigh fading problem. When two signal components arrive after traveling different distances they add together in the receiver. If difference in the length of the paths they traveled is an odd multiple of half the wavelength of the carrier signal, then they will cancel each other out. The result is that signal strength varies with distance from the source, and it also depends on the relative location of the receiver. If the receiver is mobile, rapid variations in signal strength are usually detected.

Over the last few years, spread spectrum techniques, such as DSSS-CDMA[2] and FH-CDMA[3] have been developed. CDMA techniques effectively combat the specific characteristics of the fading PCS radio channels. These techniques attempt to shield higher layer protocols from the effects of the wireless channel. Yet, in contrast to wired channels with errors randomly distributed in time, burst errors are experienced in mobile channels. A typical mobile channel allows relatively

[2] Direct Sequence Spread Spectrum - Code Division Multiple Access

[3] Frequency Hopping - Code Division Multiple Access

reliable communication during certain periods, interrupted by other periods of particularly poor communication known as fades. In chapter 5, we explore how higher layer protocols react to these burst errors, and propose mechanisms to alleviate some of the problems that arise.

1.1.3 Network Layer

An internetwork is a large heterogeneous collection of networks operating on a variety of different hardware platforms and using widely different sets of protocols. The network layer provides mechanisms which enable higher layers to view the internetwork through uniform abstractions. The mechanisms include naming and addressing schemes for network entities, and a method for routing packets from source to destination nodes. Network layer abstractions heavily draw upon the assumption that end-systems are stationary. When hosts become mobile, the current addressing and routing methods are rendered completely ineffective.

The network access point of a host may change from time to time as the user moves from one network to another. Networks must be capable of tracking the location of mobile hosts, and be able to route packets to mobile hosts at all time. This is a new requirement which is not supported at any layer in the traditional protocol stack. In chapter 2 we argue that this function is best supported at the network layer. In the same chapter we also describe our solution for gracefully integrating the new function at the network layer without compromising utility and power of network layer abstractions.

1.1.4 Transport Layer

The basic function of the transport layer is to accept data from the application layer, split it into packets and then pass it to the network layer in a controlled manner. Optionally, it also supports functions to perform data integrity checks and mechanism to recover from lost segments. The function of multiplexing multiple virtual sessions on a single network access point also belongs to this layer.

The transport layer is a true end-to-end layer in the sense that two transport entities located at each end of a virtual connection directly communicate with each other without intervention from any intermediate layer or agent. Ideally, effects of wireless access and host mobility should not be visible at the transport layer at all. It is true that transport protocols operate unmodified in the presence of mobile systems, but it is only at a cost of significant throughput degradation. New mechanisms are needed so that active transport layer connections do not break as mobile hosts switch network access points. Wireless links coupled with host motion violate some of the basic design assumptions around which existing transport layer protocols were architected. For example, losses in wireless links are mistakenly interpreted by transport protocols as the signs of congestion, and the corrective action taken by the transport protocol reduces the efficiency of the wireless link.

There are two approaches for addressing this problem. One is to modify transport layer functions for operation over wireless channels [15, 5]. Another approach, which we propose, makes the link layer more robust so as to shield higher layers from observing the effects of the wireless channel. A detailed analysis of this problem will follow in chapter 5.

1.1.5 Application Layer

The impact of mobility on the application layer can be assessed in the form of answers to the following two questions:

- Do existing distributed applications perform well when used on mobile platforms?
- Is the client-server paradigm the right choice for developing new mobile applications?

It is not surprising that answers to both questions is negative. Distributed applications depend heavily on resources that are retrieved via the network. Portable computers do not normally have access to the same rich communication capabilities that are available to desktop machines. Portable computers frequently operate without network connections, or are at times connected via low-bandwidth, high-cost link that must be used sparingly. Existing applications which depend on network support typically 'hang' when network connectivity is disrupted [10]. Examples of such applications include NFS [52], X-windows [54] and numerous other client-server applications. Clearly, new operating system primitives are required which provide support for disconnected and autonomous operation in face of intermittent network connectivity.

1.2 Contribution

The issues summarized above capture the essence of current research activity in the area of future generation mobile data communication systems. New constraints posed by host mobility require a careful re-evaluation of services provided at all layers of the communication protocol stack. This dissertation investigates the impact of wireless access and host mobility on the design of link, network, and transport layer protocols. At the network layer, we have designed and implemented a new routing architecture that allows the current set of Internet standards to support routing to mobile hosts. At the link and transport layers, we have designed mechanisms to improve throughput over lossy wireless channels.

Host mobility introduces several new addressing and routing problems at the network layer. The Internet routing system routes a datagram to a host based on the network number contained in the host's Internet address. If a host changes its point of attachment and moves to a new network, IP datagrams destined for it can no longer be delivered correctly. Our solution allows mobile hosts to retain their addresses regardless of their point of attachment to the network. This is achieved by employing a two tier addressing scheme for mobile hosts. The first component of the address, the topologically significant part, reflects the mobile's point of attachment to the network and facilitates the routing process. The second static component, which is the home address of the mobile host, serves as an end-host identifier. Mechanisms are designed so that only the static part of the address is visible to the end-to-end transport layer while only the topologically significant part of the address is exposed to the routing system. Since the transport layer is provided with a location independent abstraction of the mobile end-system, existing networking applications run unmodified on mobile hosts.

The proposed routing scheme has been implemented using IP's Loose Source Route (LSR) option in the AIX and Mach kernels. The reference implementation provides mobile hosts, connected over 1Mb/s infrared wireless link, full Internet connectivity even as mobile hosts move between networks. It also allows mobile hosts equipped with multiple network interfaces to dynamically migrate active sessions from one network interface to another. The proposed scheme only requires the addition of two new type of entities, Mobile Routers and Mobile Access Stations, which perform all required mobility-aware functions, such as address translation, user tracking, and location management. No modifications to existing host or router software are required. Unlike other Mobile-IP proposals that are encapsulation based, this approach provides optimal routes for all TCP sessions, has less overhead, and is more suitable for deployment in the next generation IP protocol. The scheme is fully distributed, scalable, and tolerant of component failures.

Routing protocols only provide mechanisms for packet exchange between mobile and stationary end-systems. Performance of these data exchange sessions depends crucially on the characteristics of the wireless medium and the MAC protocol employed at the wireless link layer. We have investigated performance implications of various MAC layer packet scheduling policies on the dynamics of TCP sessions. Burst losses on the wireless channel interact with the error recovery mechanisms employed at the transport layer in quite unexpected ways. This effect is more pronounced when a single wireless channel is shared purely on demand among multiple competing stations. Our results indicate that the FIFO packet dispatching mechanism, commonly used in wireless LAN cards, leads to poor utilization of the wireless medium. We have shown that by employing a Channel State Dependent Packet (CSDP) scheduler in the wireless LAN device driver, significant improvement in wireless link utilization can be achieved. CSDP can be employed in conjunction with any wireless link layer protocol to improve throughput of transport layer sessions over multiaccess wireless channels.

1.3 Organization

This dissertation is outlined as follows. First we present a discussion of the mobility problem from the Internet routing perspective, pointing out what makes this problem difficult to solve in practice. In chapter 2, we propose a general network layer architecture that supports routing to mobile end-systems and show that our proposed architecture captures the essence of all contemporary MobileIP proposals. In chapter 3, we describe the design and implementation our MobileIP protocol. In chapter 4, we show that our proposed MobileIP scheme lends an elegant solution to the problem of switching from one communication medium to another. In chapter 5, we turn our focus to some performance problems related to wireless internetworking. We point out deficiencies in existing link and transport protocols and outline our proposed solution approach. Finally, in chapter 6, we summarize our findings and outline a direction for future research.

Chapter 2

Mobile Networking Architecture

Mobile end-systems frequently change their point of attachment to the network. In such an environment, in order for mobile devices to run without disruption, a universal networking infrastructure is needed. In addition, a common networking protocol is required which can support network-wide mobility. Mobile devices also need to communicate with the existing pool of information servers and file servers, which means that internetworking solutions for connecting stationary and mobile systems are also required. Unfortunately, the Internet Protocol (IP), which forms the fabric of the current world-wide data communication network, falls short of meeting this demand. The current Internet suite of protocols (TCP/IP) were designed under the assumption that end-systems are stationary. If during an active network session one end of the connection moves, the network session breaks. Naturally, all networking services layered on top of TCP/IP are also disrupted when end-systems become mobile. There are two approaches for solving this problem. One is to completely redesign internetworking protocols with the specific goal of supporting mobile end systems. The other approach is to provide additional services at the network layer in a backward compatible manner which make mobile internetworking possible. The first approach, though an interesting possibility from a research viewpoint, is infeasible since it would require radical changes to the currently deployed networking infrastructure. It is the latter approach that is the focus of our investigation.

To ensure inter-operability with the existing infrastructure, the handling of mobility should be completely transparent to the protocols and applications running on stationary hosts. In other words, from a stationary end-system's perspective, a mobile host should appear like any other stationary host connected to the Internet. This means the same naming and addressing conventions, those originally developed for stationary hosts, must apply to mobile hosts. In addition, any changes in a mobile's network attachment point should be completely hidden from the protocols and applications running on stationary hosts.

In this chapter we explore various network layer concepts that pertain to the design of mobile networking systems. We show that mobility is essentially an *address translation* problem and is best resolved at the network layer. We have identified the fundamental services that must be supported at the network layer to carry out the task of address translation. Using these service primitives as building blocks, we propose a network layer architecture which enables smooth integration of mobile end systems within the existing Internet. The architecture is modularized into well-defined logical components. In this chapter, our objective is not to propose *a specific scheme* for supporting mobility, rather it is to highlight and analyze the essential aspects of supporting mobile end-systems and to better understand the trade-off between various design alternatives.

2.1 Internet Naming and Addressing

The Internet is a large collection of networks which share the same address space and inter-operate using a common sets of protocols, such as TCP/IP [41, 42]. A fundamental concept of the Internet architecture is that each host[1] has a unique network address, by which it is reachable from other hosts in the network. Data are carried in the form of packets which contain source and destination addresses. To communicate with another host, a source only need to know the address of the destination. It is the responsibility of the internet routing system to carry packets from a source to a destination node.

Internet routers maintain a view of network topology in the form of routing tables. These tables are consulted when making packet routing decisions. The process of routing involves inspecting the the destination address contained in the packet and, based on the contents of the routing table, determining the next-hop router to which packet should be relayed. Each router along the path from a source to a destination node repeats this process until the packet is finally delivered to the destination host.

If host addresses are treated as *flat identifiers*, routers will be required to maintain routing information on a per-host basis. Obviously, this is not feasible, given the large number of hosts that are connected to the Internet. A natural solution is to impose a hierarchy on the address structure. The purpose of hierarchical addressing scheme is to allow *aggregation* of routing information; higher layers in the hierarchy (e.g., routers) need only concern themselves with the portion of the address that is relevant at that layer. Hierarchical addressing is essential if the routing architecture is to be scalable. The Internet, for example, deploys a two-level hierarchical addressing scheme.

[1] In the Internet jargon, host means an end-system connected to the Internet

2.1.1 Internet Addressing

Each host in the Internet is assigned a unique 32-bit internet address (also known as an IP address) which consists of two parts: network-id and host-id. The boundary between the network-id and the rest of the address is a fixed location determined by the leading bits of an address (as shown in Figure 2.1). IP addresses are commonly represented using dotted notation where each octet is represented as a decimal number and dots are used as octet separators.

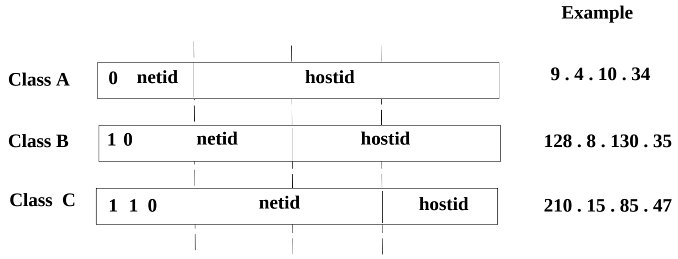


Figure 2.1: IP address structure

Under the current Internet addressing scheme, routers only need to maintain network topology information at the granularity of individual networks. This means only the network part of the destination address is used in making routing decision. Though hierarchical addressing makes routing simple and manageable, as a natural consequence, it puts certain restriction on the address usage. A hierarchical address can only be used within the domain of its definition. For example, an Internet address is only meaningful so long as the host using it remains connected to that network denoted by the network-id part of the address. When the host moves to a new network, it must be allocated a new address which is derived from the address space of the new network. In order for the Internet routing to work:

A mobile host must be allocated a new address when it moves.

2.1.2 Naming

A related concept for identifying hosts in the network is *Name*. Names are user defined aliases (strings of characters) which are used to denote hosts. For example, *ballast* is the *name* of the file-server in our department, and its address is `128.8.128.88`. An important distinction between names and addresses is that addresses are protocol specific (e.g., an IP address, CLNP[1] address, IPX[50] address, XNS address), but names are not. Names provide a way for applications to make reference to network entities without having to know anything about the underlying network protocol in use. This is useful, since users find names easier to use and remember than cumbersome network addresses.

Though applications refer to end systems by names, when packets are transported through the network they must contain addresses of destination nodes. This is because routers do not understand names, they can only interpret addresses. A translation mechanism, therefore, is required for mapping host names to addresses. To accommodate a large, rapidly expanding set of names, a decentralized naming mechanism called the Domain Name System (DNS) was deployed in the Internet. DNS stores name to address mappings in a distributed data structure. Finding the address of the host is essentially a directory lookup operation (see Figure 2.2). When two hosts on the Internet need to communicate with each other, the source node performs a DNS lookup to obtain the destination node's address and then initiates a connection setup procedure. During connection setup, each end of the connection learns about the address of the other end. So long as the connection is active, no additional DNS lookups are performed, since name to address binding is assumed to be static and is not expected to change during a connection lifetime.

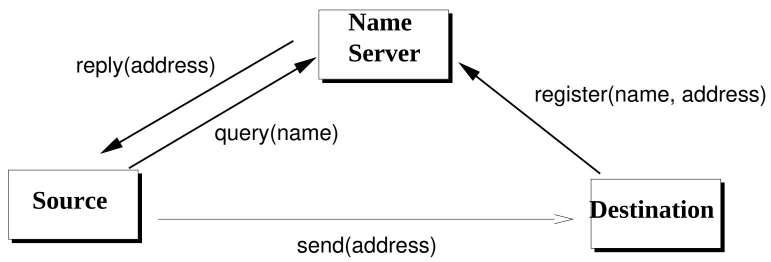


Figure 2.2: DNS based Name to Address resolution

2.2 The Mobility Problem

To illustrate why host mobility poses problem at the network later, it is important to emphasize the distinction between the concepts of *name* and *address*.

- Name: is a location independent identifier of a host. E.g. 'mimsy' is the name of the mail-server in our department.
- Address: indicates the location of a given host. E.g. mimsy's address 128.8.128.8 indicates that it is connected to network 128.8.128

Names remain fixed regardless of where a host is located. An address on the other hand reflects a host's point of attachment to the network. For hosts that remain static throughout their lifetime, both names and addresses can be used interchangeably. For a mobile host, however, an address cannot be used as a unique identifier, since it must change with the location of the host. Name is the only location independent identification mechanism that can be used at the network layer to make references to mobile hosts.

2.2.1 Mobility Problem: Directory Service View

In networks where hosts are static, name to address bindings never change. Host mobility makes this binding a function of time. Therefore, network layer mechanisms are required for resolving names into addresses and tracking the location of hosts as they move. The Domain Name System (DNS), which provides name to address translation service in the Internet today, should be enhanced to meet the additional demands. However, this task is made difficult by many hurdles:

- The DNS has no provision to handle dynamic updates. This is because it was originally designed to provide name lookup service for stationary hosts only.
- The DNS design attempts to optimize the *access* cost, and not the *update* cost. Server replication and client caching provides significant performance gains for access only systems, but results in very poor performance when updates are performed. In a mobile environment, both updates and accesses are equally likely.
- DNS clients cache DNS records to reduce latency for future accesses and to reduce load on the name servers. There is no call back mechanism from servers to clients, in case cache entries become invalid.

A design for a distributed location directory service for mobile hosts was proposed by Awerbuch and Peleg in [4]. They formally proved an important theoretical result which established that a system cannot optimize both *access* and *update* operations[2]. Using the concept of *Regional Directories* (a type of cache) they proposed a distributed directory layout which guarantees that the communication overhead of *access* and *update* operations is within a poly-logarithmic factor of the lower bound.

As far as the Internet is concerned, distributed directory service based solutions do not appear very attractive since they cannot be deployed without changing existing host software. The Internet host population has already grown over tens of millions, which makes any change to host software almost impossible to achieve. Hence, an alternate solution method is required.

[2]In their paper they use terms `Find` and `Move` to denote these operations.

2.2.2 Mobility Problem: Internet View

When the Internet suite of protocols were originally developed, it was implicitly assumed that the *name* to *address* binding remained static. Thus, instead of referring to hosts through names, protocols were developed that referred to hosts through their addresses. A classic example is a TCP connection which is identified by a 4-tuple:

$$< \textit{source IP address, source TCP port,} \quad \textit{destination IP address, destination TCP port} >$$

If neither host moves, all components of the connection identifier will remain fixed, and thus a continuous TCP session can be maintained between the two hosts. If either end of the connection moves, we run into the following problem:

- If the mobile host acquires a new IP address, then its associated TCP connection identifier also changes. This causes all TCP connections involving the mobile host to beak.
- If the mobile host retains its address, then the routing system cannot forward packets to its new locations.

The fundamental problem is that in the Internet architecture, an IP address serves dual purposes. From the transport and application layer perspective, it serves as an *end-point identifier*, and at the network layer, the same IP address is used as a *routing directive*. This problem is not specific to the Internet architecture; in fact all contemporary connection-less network architectures, such as OSI[56], IPX[50] and XNS, suffer from this problem. Since our objective is to ensure that connection do not break when hosts move, we can say that:

In order to retain transport layer sessions, a mobile host's address must be preserved regardless of its point of attachment to the network.

An immediate consequence of this choice is that we can not rely on the existing routing system for delivering packets to a mobile host's new location. A solution might be to keep per-mobile-host routing information at all routers, but this completely breaks the hierarchical model of routing, causing unbounded growth in the size of routing tables. Thus, the problem of supporting mobile hosts within the Internet is not just keeping track of hosts. In addition, it has to do with designing a mechanism for packet forwarding to mobile hosts without modifying and compromising the scalable nature of the Internet routing mechanism.

2.3 Network Layer Solution Architecture

In this section we describe a network layer architecture that allows smooth integration of mobile end-systems within the Internet. Our objective is to highlight and analyze the essential aspects of providing mobility extensions in any connection-less network; the specific details involved in designing a mobile-networking system will be deferred until the next chapter. For ease of exposition, we will first introduce a few definitions.

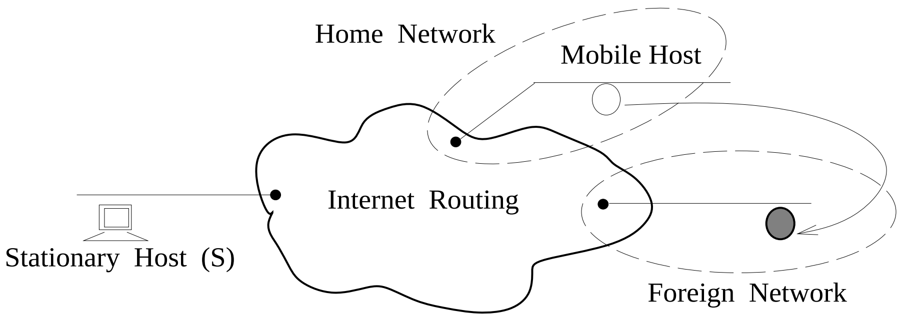


Figure 2.3: Illustration of Terms

Mobile Host: An internet host is called a *Mobile Host(MH)* if it frequently changes its point of attachment to the network. A change in the attachment point can also happen while one or more transport layer sessions involving the MH are in progress. It is assumed that the rate of change of location is slower than the time it takes to for the mobile routing protocols to learn about the mobile host's new location.

Home Address: Like any other internet host, a mobile host is also assigned an internet address which is referred to as its *Home Address (HA)*. A standard 32-bit internet address is allocated using the same guidelines that apply to stationary hosts. When the DNS is queried with a mobile host's name, it returns the home address of the mobile host.

Home Network: Within each administrative domain, network administrators find it easier to reserve one or more subnetwork(s) for mobile hosts. The home address of a mobile host is allocated from the address space of one of these subnetworks, referred to as the *Home Network* in the subsequent discussion. The terms *home address* and *home network* also apply to stationary hosts. The only difference is that stationary hosts always remain connected to their home network, while mobile hosts sometimes may not be found at their respective home networks.

Foreign Network: Any connected segment of an Internet, other than the home network of a mobile host, to which the mobile host is allowed to attach is referred to as a *Foreign Network*. If I denotes the set of all networks connected to the Internet, then any network in the set $I - \{Home\ Network\}$ is a foreign network to all hosts that derive their home addresses from the *Home Network*.

Notice that above definitions are relative to a mobile host. The same network could operate both as a home and as a foreign network, depending on which mobile host is connected to it. So long a mobile host remains connected to its home network, existing internet routing mechanism are sufficient to route packets up to its current location. It is only when it moves to a foreign network that additional mechanisms are required. If a mobile host moves within its home network (e.g., detach from one ethernet point and attach through another ethernet point), it does not constitute a move from the network layer point of view. Existing link layer bridging mechanism are capable of routing packets up to end-systems so long as they remains connected to the same layer 2 segment[3].

In the previous section, we made two crucial observations:

1. The home address of a mobile host cannot be used for routing packets to its current location (except when it is attached to its home network).
2. A mobile host's address must be preserved in order to retain all active transport connections involving the mobile host.

These are two conflicting requirements. From the first observation, when a host moves, a new address, reflecting its new point of attachment to the network, must be used for the purpose of routing. The second observation says just the opposite: the original address must be preserved to retain all active network sessions.

2.3.1 Two Tier Addressing

We introduce the concept of *two-tier addressing* to resolve the problem associated with the dual use of an internet address. Our solution involves associating two internet addresses with each mobile host(see Figure 2.4). The first component of the address reflects the mobile's point of attachment to the network while the second component denotes its home address. The first address component serves as a *routing directive*. It changes whenever a mobile host moves to a new location. The

[3]A collection of link layer networks, which are interconnected through bridges, is called a layer 2 segment. Within a layer 2 segment, a packet can be delivered solely on the basis of the destination node's link layer address; the network layer routing is not required

second component of the address serves as an *end-point identifier*. It remains static throughout the lifetime of a mobile host. The purpose of two-tier addressing is to decouple the dual role of an internet address into two disjoint, well defined functions.

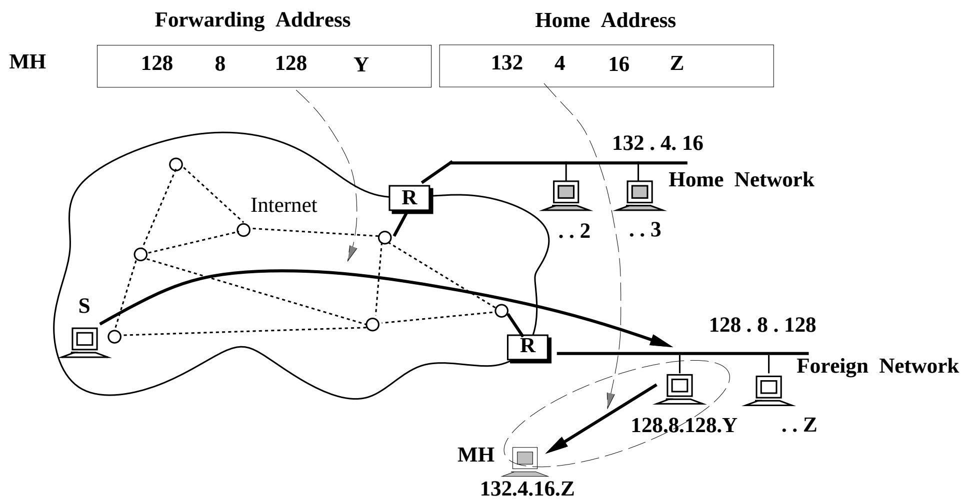


Figure 2.4: Two Tier addressing for Mobile Hosts

The concept of two-tier addressing is illustrated in Figure 2.4. Packets that are destined to mobile hosts contain the destination address in the two-tier format. The Internet routing system only looks at the first component of the address and routes those packets to the point where the mobile host is attached. At this point, the first address component is discarded. Only the second address component, the home address of the mobile host, is used in subsequent protocol processing. From an end-host's perspective this means that it notices no difference when it is attached to its home versus when it is located in a foreign network. In other words, the mobile host *virtually* remains connected to its home. Packets which originate from the mobile host and are destined to the stationary host (S) do not require any special handling, since the Internet routing system can deliver those packets based on their destination addresses. If S is also mobile, then the same two-tier addressing mechanism can be used to route packets to its current location.

It is important to note that the two-tier addressing is only a logical concept. Its realization doesn't necessarily require carrying two addresses in the destination address field of the network layer packets. In fact, doing so would require changes in the existing packet formats, necessitating changes to host and router software. It is desirable to support the two-tier addressing method using the existing mechanism available in the Internet Protocol suite. Below we describe how this can be achieved.

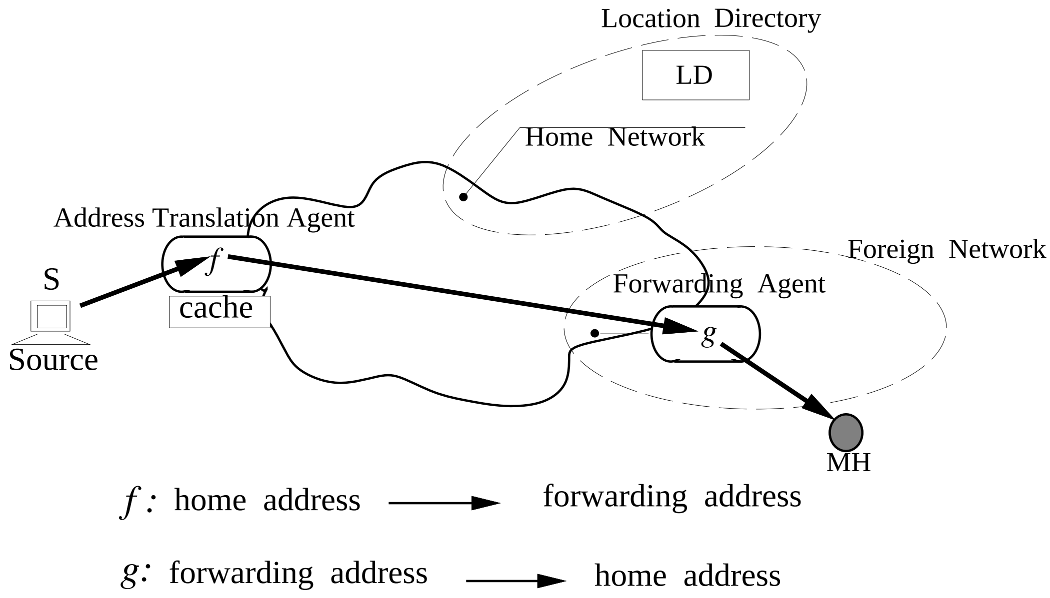


Figure 2.5: Packet Forwarding Model

2.3.2 Architecture Components

Forwarding Agent (FA)

When away from its home network, a mobile host can attach to the Internet through a foreign network. For the purpose of forwarding datagrams to its new location, an address derived from the address space of the foreign network must be used. Packets destined to the mobile host contain the address of a *Forwarding Agent (FA)* in the forwarding address sub-field of the two-tier address. An FA provides an access point through which mobile hosts can attach to the network. It receives packets on behalf of mobile hosts, and forwards them to appropriate mobile hosts after necessary protocol processing.

Conceptually, the processing at the FA involves stripping the forwarding address part of the two-tier address and exposing the home address of the mobile host. Once the packet arrives at the FA, the forwarding address is no longer required in the subsequent protocol processing. When a packet arrives at the FA, it contains the address of the FA in its destination address field. The FA, essentially, maps the contents of the destination address (the forwarding address) to the home address of the associated mobile host. We use the notation g to denote this mapping function:

$$g : (forwarding\ \ address) \rightarrow (home\ \ address)$$

An FA should be able to relay packets to the mobile host on the basis of its home address. This is easy if the FA and the MH are directly connected (normally over a wireless link). Otherwise, the routing protocol operating in the foreign network should advertise host specific routing information within the foreign network to facilitate routing of these packets to mobile hosts. Normally, we would expect a wireless base station to operate as an FA in which case both the MH and the FA would be directly connected over a wireless link.

A mechanism is required so that mobile hosts can discover the identity of an FA when they connect to a foreign network. Similarly, a mechanism is required so that the FA can determine the identities of all mobile hosts that require its service. The simplest way to achieve this is through a route advertisement and a registration protocol. Forwarding agents periodically advertise their presence in the foreign network. Beaconing, the periodic broadcast of messages over the wireless medium, is the most commonly used method. Mobile hosts can listen to broadcasts, determine the identity (address) of the nearest FA, and initiate a registration sequence.

Location Directory (LD)

The component in the architecture that records the association between the home and the forwarding address of a mobile host is called a Location Directory (LD). The LD contains the most up-to-date mapping between a mobile host and its associated FA. Mobile Hosts are required to send updates to the LD whenever they moves to a new location.

Since the number of mobile hosts is expected to be very large, a centralized realization of the LD is deemed infeasible. A policy for distributing LD components should take many factors into consideration, such as the cost of access, ease of locating LD components, and security and ownership of location information. Since the LD will be accessed very frequently, a good distribution method should exploit the locality of access patterns and provide uniform load balancing among all LD components. Given a model for the LD access pattern, the LD distribution can be formulated as an optimization problem[3]. Unfortunately, these mathematical results [3, 7, 6] cannot be directly applied in the Internet. The primary reason is that in the Internet factors such as ease of location, security, and ownership take precedence over any cost optimization considerations.

A feasible distribution scheme in the Internet is the *owner-maintains-rule*. According to this scheme, the LD entries for mobile hosts are maintained at their respective home networks. Within each home network, a good place for locating an LD component is at the home router. Advantages of this scheme are:

1. Each home network is responsible for maintaining, securing, authenticating, and distributing LD information for its mobile hosts. This policy fits well within the Internet philosophy of autonomous operation.

2. No special mechanisms are required to locate the LD components. It is important to point out that in a distributed scheme, in order for a source to send a query to the right LD component, the source is required to know the address of the LD component in advance. Under the *owner-maintains-rule*, a source simply sends a query that is addressed to the mobile host. The packet is delivered to the home network by normal internet routing where it is intercepted by the home router and subsequently relayed to the correct LD component.

This is certainly not the only possible distribution scheme. Later in this chapter we'll discuss other options while reviewing various MobileIP proposals.

Address Translation Agent (ATA)

Hosts that need to communicate with a mobile host insert the mobile's home address in the destination address field of all packets they issue. At some point during the routing process this address should be replaced by the address of the FA associated with the mobile host. The entity which performs this operation is called an *Address Translation Agent.* The process of address translation involves querying the LD, obtaining the FA address, and subsequently making use of this address in forwarding packets to the correct location of the mobile host. The address translation function is:

$$f : (home \;\; address) \rightarrow (forwarding \;\; address)$$

From a two-tier addressing perspective, an ATA initializes the forwarding address part of the destination address. In an actual implementation this could be achieved by replacing the original destination address of the packet with the FA's address. This operation can be performed at the source host; however, the only problem is that the function f cannot be computed without making changes to the existing host software of millions of hosts.

For performance reasons, an ATA may decide to cache LD entries which are frequently used in making forwarding decisions. Querying the LD before making each address translation operation could be prohibitively expensive, particularly so when the ATA and the LD are geographically separated. Caching, however, introduces a new requirement in the architecture; that of maintaining consistency between the LD and its cached entries throughout the Internet.

2.3.3 Location Update Protocol (LUP)

Keeping the LD up-to-date in the face of frequently changing host location is crucial. Keeping cached LD entries consistent with the master LD is an equally important consideration. Inconsistencies could make mobile hosts inaccessible and even cause the formation of routing loops. The purpose of *Location Update Protocol (LUP)* is to provide reliable mechanisms for keeping the LD and its cached copies consistent at all times.

To a large extent, the choice of the LUP depends on the caching policy used. Together, they determine the scalability and routing characteristics of a mobility solution. In systems which do not permit LD caching, ATAs must be co-located with the LD, since issuing an LD query for each packet that an ATA forwards is prohibitively expensive. In such systems, packets addressed to mobile hosts first travel all the way up to the home network before any address translation (operation f) is performed. Clearly, the paths that packets follow are non-optimal in this case. Caching improves the routing efficiency of a mobile networking system, as packets do not have to travel to home networks before being forwarded toward the FAs associated with the destinations. At the same time, caching makes the system more complex and vulnerable to security attacks. If cache entries are not properly authenticated, it is possible to redirect packets away from a mobile host and cause denial-of-service.

2.3.4 Packet Forwarding Operation

With the inclusion of address translation agents and forwarding agents, the operation of packet forwarding can be easily illustrated. Figure 2.5 illustrates how packets from a stationary host (S) are routed to a mobile host (MH). S sends out packets which are addressed to the home address of the MH. These are intercepted by an address translation agent which maps (using function f) the original destination of the packet to the address of the forwarding agent. Once these packets arrive at the forwarding agent, the FA remaps (using function g) the destination to the home address of the mobile host and delivers them to the mobile host. Along the path from the source to the destination, packets twice undergo an *address translation* operation. The end result of this translation process, the function gof, is an identity mapping, which means that the whole process of address translation is completely transparent to hosts located at both ends of the path. They communicate as if they were stationary. The transport layer protocols and the applications running on stationary as well as mobile hosts operate without any modifications whatsoever. This property of the solution architecture is termed as *transport layer transparency*.

The proposed architecture preserves *transport layer transparency* regardless of where and how in the network the LD, ATAs, and FAs are distributed. This flexibility enables us to capture the design choices made in other MobileIP proposals. Later in this chapter, we'll show that each one of these proposals can be viewed as a special case of the proposed architecture.

2.3.5 Address Translation Mechanisms

So far we described how various components of the architecture co-operate amongst each other to perform necessary address translation operations. The actual mechanisms for effecting those were not mentioned. Within the Internet architecture there are two possible ways of doing it: either using *encapsulation* or using *loose source routing.* A brief description of both follows:

Encapsulation

In the encapsulation method a new packet header is appended at the beginning of the original packet (see Figure 2.6). The outer header contains the address of the forwarding agent while the inner header contains the home address of the mobile host. Since the Internet routing system only looks at the outer packet header, it routes this packet to the forwarding agent. The forwarding agent strips the outer packet header and delivers the inner packet locally to the mobile host.

Figure 2.6: Illustration of Encapsulation and Decapsulation

Loose Source Routing (LSR)

Loose Source Routing is an option that is supported in IP which can also be used to perform address translation operation[4]. Using IP's source routing option, an address translation agent can cause packets addressed to a mobile host's home address to be routed via a forwarding agent. Figure 2.7 illustrates how this is done. An LSR option is used to specify a *nil* terminated list of addresses. The Internet routing system routes the packet containing the LSR option to each address, one by one, in the sequence it appears in the list. The current destination is kept in the destination address field of the packet header and a pointer points to the address which is to be visited next in the sequence. When the packet arrives at the current destination, the contents of the destination address field are swapped with the address pointed by the next hop pointer, and, the pointer is advanced to the next address in the list. This process is repeated until the packet is delivered to

[4]Originally it was included in IP not for this purpose, but to help in debugging network problems

the address which occurred last in the original list of addresses included in the LSR option. At this point the the next hop pointer in the LSR option points to *nil*.

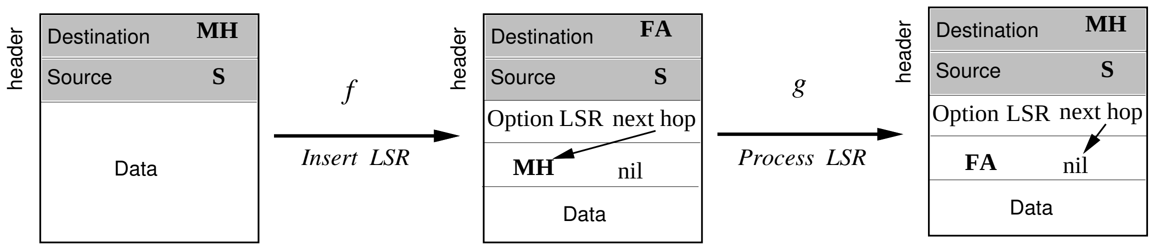


Figure 2.7: Using Loose Source Routing to perform address translation

An advantage of using the LSR oprion over encapsulation is that, as a natural consequence of the LSR option processing, the path that a packet follows (the list of addresses visited en-route) is automatically recorded in the packet. The destination can reverse this list and send a reply back to the source along the reverse path. In the next chapter, we'll show how we exploit this property to design a mobile networking scheme that co-locates the ATA with the source, and the FA with the destination. It is not possible to achieve this using any method which uses encapsulation, since when the packet arrives at the destination, it is already stripped of all useful routing information.

In this section we showed how components of the proposed architecture mutually co-operate to overlay a packet forwarding service on top of an existing routing infrastructure. It is important to point out that the ATA and the FA only represent functions that need to be supported, not machines that need to be deployed in the network. In fact, the proposed architecture allows sufficient flexibility in placement of these functions in the network. This flexibility allows us to experiment with various design alternatives and fine tune a solution for a specific target environment.

2.4 Mapping to candidate MobileIP proposals

Over the past several years, many proposals have been made for supporting host mobility on datagram-based internetworks. A vast majority of these proposals have been designed to be compatible with today's TCP/IP-based Internet. The candidate proposals differ widely in terms of the specific components they propose to add to the Internet, the mechanisms they use for address translation, and the policy they use for managing location updates. In this section, we'll show that all mobileIP proposals can be viewed as a special case of our proposed network architecture.

In our model, the ATA and FA represent the two basic functions that must be supported by any proposal that supports mobility. We'll demonstrate this fact by explaining the operation of each MobileIP proposal in terms these two functional entities. Basically, all proposals attempt to provide an address translation service through deployment of some additional entities in the network. They only differ in terms of their choice of where they locate these functions, the specific location update protocol they use, and whether they use encapsulation or source routing to effect address translation. Below, we present a short summary of related MobileIP proposals, with a short note following each proposal outlining how its operation can be captured by our proposed solution architecture.

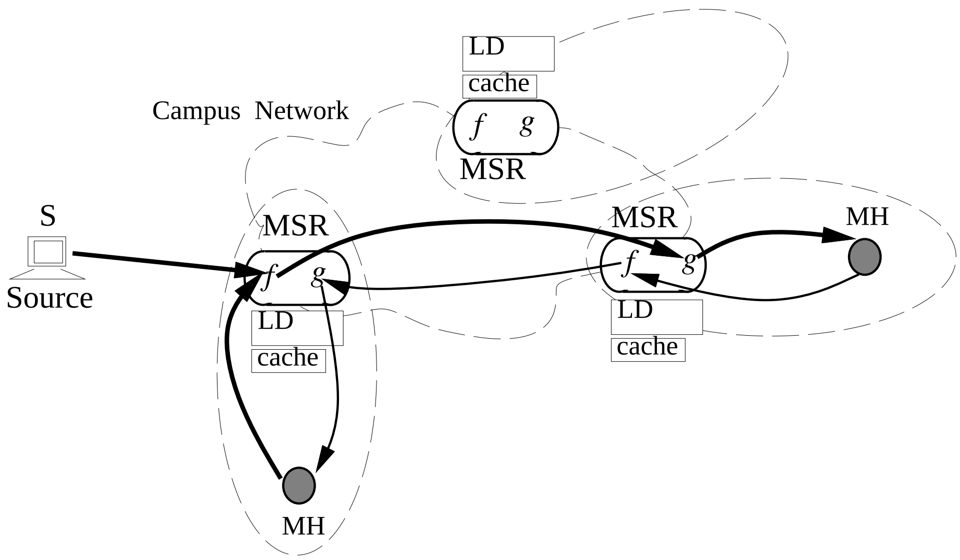


Figure 2.8: Mapping to Columbia Proposal

2.4.1 Columbia Scheme

The scheme proposed by Ioannidis [22, 23] is designed primarily to support mobility within a campus environment. Mobile hosts are allocated addresses from a subnetwork which is reserved for use by wireless hosts. A group of cooperating Mobile Support Routers (MSR), advertise reachability to the wireless subnet. MSRs provide an access point through which mobile hosts can connect to the campus back-bone, and are also responsible for forwarding traffic to and from mobile hosts. Each mobile host, regardless of its location within a campus, is always reachable via one of the MSRs. When a host sends a packet to a mobile host, it first gets delivered to the MSR closest to the source host. This MSR either delivers the packet (if the destination MH lies in its wireless cell), or forwards it to the MSR responsible for the destination MH. If an MSR does not know which MSR is currently responsible for a destination, it sends a `WHO_HAS` query to all MSRs in the campus and awaits a reply message from the responsible MSR. When sending a packet to the destination, an MSR encapsulates the packet and delivers it to the target MSR. Upon receiving this packet, the target MSR strips the encapsulation header and relays the original packet to the mobile host.

Mapping In the Colombia proposal, an MSR performs both *encapsulation* and *decapsulation* operations, meaning that both functions, f and g, are co-located at the MSR. For packets addressed to MHs in its coverage area, an MSR acts like an FA. For packets addressed to other MHs it acts like an ATA. Each MSR maintains a table of MHs in its wireless cell. These tables together constitute the segment of the LD which is associated with mobile hosts on the campus network. This LD distribution scheme can also be thought of as a distributed realization of the *owner-maintains-rule*. Recall that in the *owner-maintains-rule*, the segment of the LD was co-located with the home router. An MSR in the Colombia scheme is a distributed realization of the home router. As a result, the table of mobile hosts maintained at an MSR constitutes a distributed segment of the LD that is required to be maintained at the home router.

MSRs acquire LD cache entries on a need-to-know-basis by sending a broadcast `WHO_HAS` query to all MSRs in the campus. The response to this query is generated by the MSR which possesses the primary copy (in other words, the MSR which is responsible for the destination MH). The Location Update Protocol uses a *lazy-update* approach. When a mobile host moves, only the primary copy of the LD entry is updated. Cached entries are assumed to be correct by default. In cases, when cached entries turn stale, the first packet which is forwarded using the stale entry generates an error message from the old MSR, causing the source MSR to flush its cache and then broadcast a `WHO_HAS` message.

Since functions f and g are required to be supported only in new entities (MSRs) that are added to the system, the Columbia proposal can operate without requiring any modifications to the existing host and router software. This proposal presents a good combinations of design choices for handling mobility within a campus environment. However, it has severe scalability problems. Since this proposal will require broadcast of `WHO_HAS` query to all MSRs located world-wide, it is not possible to scale this scheme to the Internet scale.

2.4.2 Sony Scheme

In Sony's proposal [49, 47, 48], a mobile host is assigned a new temporary address when it is attached to a new network. The router of the home network is notified of this new address through a special control message. Packets addressed to the MH, in addition to carrying its home address, can also carry its temporary address. Packets originating from an MH that is away from its home network always carry both home and temporary addresses in the source address field. Routers that forward these packets can examine the source addresses and cache the mapping (home to temporary) in their Address Mapping Tables (AMT). A source includes both addresses in all outgoing packets if it already has an AMT entry for the target host. Otherwise, packets are forwarded to the home address. If a transit router has an AMT cache entry for the destination, it can intercept the packet and forward it to its correct location. If none of the transit routers have a cache entry, the home router is eventually responsible for forwarding the datagram.

When a host moves to a new location, all AMT cache entries are invalidated through a special `disconnect` control message which is broadcast in the network. Since this message of invalidation is not reliable, there is also a timeout associated with all AMT cache entries, which, on expiration, causes AMT entries to be purged.

This method requires modifications to routers and host software and has problems inter-operating with the existing hosts since it also requires modifications to IP packet formats.

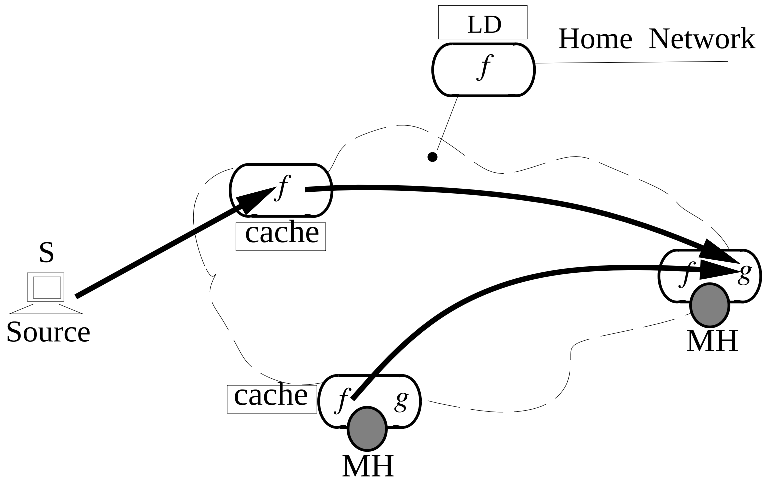


Figure 2.9: Mapping to Sony Proposal

Mapping: The Sony proposal co-locates the forwarding agent function, g, with mobile hosts. In other words, it requires each mobile host to act as its own forwarding agent. The advantage is that packets can be directly tunneled to the mobile host, without intervention from a forwarding agent. This is useful, particularly for wired mobile hosts, which may at times connect to foreign networks which have no forwarding agents attached. The approach of co-locating g with the mobile host has a disadvantage. It doubles the address space requirement for mobile hosts, since in addition to a home address, a temporary address is also required for operation. Given that IP address space is fast running out of available addresses, this is a serious problem.

In Sony's proposal, the home router acts as an address translation agent (f), and it also maintains the Location Directory for mobile hosts that have been assigned addresses on the home network. To avoid routing each packet via the home router, Sony proposal allows flexibility to co-locate f with internet routers. Since LD cache entries are carried in the source address field of the VIP protocol[5], routers can acquire these them just by inspecting the source address of packets they relay. Distributing LD caches all over the Internet improves routing performance; however, it makes updates very costly. Sony's proposal, therefore, has severe scalability problem. When a host moves to a new location, it is required to send a broadcast in the network to purge all cached LD entries.

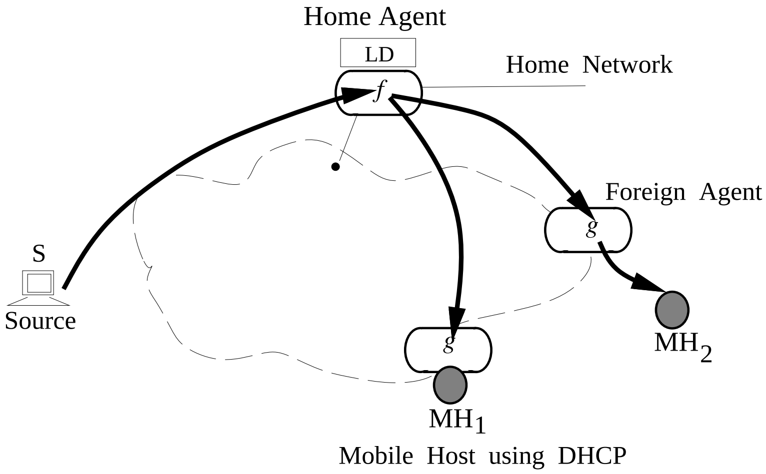


Figure 2.10: Triangle Routing: MobileIP Proposal

2.4.3 MobileIP working-group Proposal

IETF has created a MobileIP working group to come up with a proposal for near term deployment within the Internet. In this design [36], each mobile host retains its home address regardless of the mobile host's location. When the mobile host visits a foreign network, it is associated with a care-of-address, which is an Internet address associated with the mobile host's current point of attachment. The *care-of-address* either identifies the mobile host directly (if the address is acquired through Dynamic Host Configuration Protocol (DHCP)) or identifies a Foreign Agent that is responsible for providing access to visiting mobile hosts. When away from home, the mobile host registers its care-of-address with a Home Agent; the Home Agent is responsible for intercepting datagrams addressed to the mobile host's home address and tunneling (encapsulating) them to the associated care-of-address.

In this scheme all datagrams addressed to a mobile host are always routed via the Home Agent. However, the packets in the reverse direction, i.e., those originating from the mobile host and addressed to a stationary host, are relayed along the shortest path by the Internet routing system.

[5]The modified IP protocol

This gives rise to what is known as the triangle routing problem. Route optimization is possible if the location information is allowed to be cached; however, this proposal does not permit caching of LD entries because of security concerns. Currently, the Internet does not provided any secure mechanism for distributing cache entries. Any entity in the Internet can masquerade as a Home Agent and re-route traffic away from a mobile host just by re-distributing fake cache entries. This proposal, therefore, takes the stand that routing based on cached location information is insecure, and the best possible defense against security attacks is to not use it at all. The cost of this choice is that routing is always non-optimal.

When the mobile host arrives at a foreign network, it can listen for (or solicit) agent advertisements to determine whether a Foreign Agent is available. If so, the registration request to the Home Agent is sent via the Foreign Agent; otherwise, the mobile host must acquire a care-of-address (through DHCP), and then register with the Home Agent.

Mapping: The IETF-MobileIP proposal reflects a design choice that co-locates f with the Home Agent and g with the Foreign Agent. This proposal also allows g to be co-located with the mobile host. This happens when the mobile host acquires a temporary address via DHCP. The location update protocol is very simple; the mobile host notifies the Home Agent whenever it moves to a new location. Since the LD entries are never cached, the question of maintaining consistency doesn't even arise.

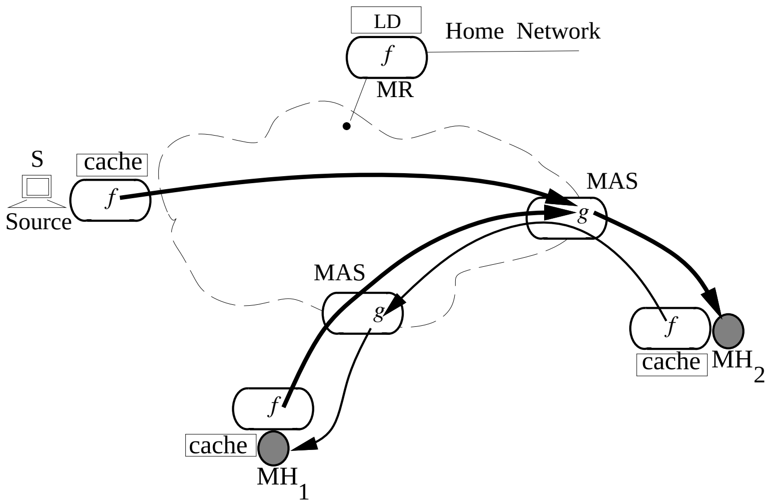


Figure 2.11: Mapping to LSR Scheme

2.4.4 LSR Scheme

In contrast with other proposals which are encapsulation based, the LSR proposal [9, 37, 38, 26] is based on the use of an existing IP option called Loose Source Route. A detailed description of this scheme will appear in the next chapter. Our scheme also allows each mobile host to retain its home address regardless of its current location. Associated with each home network is a Mobile Router, which is responsible for advertising reachability to the home network, and for keeping track of the current location of each mobile host that has been assigned an address on that network. In a foreign network, mobile hosts attach to the Internet via wireless base stations known as Mobile Access Stations (MAS). When a mobile host walks into the wireless cell of an MAS, it informs its Mobile Router the internet address of the current MAS. The Mobile Router records this information in its routing table, and also informs the previously recorded MAS that the mobile host has migrated from its wireless cell. The packets sent to the mobile host first arrive at the Mobile Router by the normal routing process. To forward a packet to the a mobile host's current location, the Mobile Router inserts an LSR option in the packet, specifying the current MAS as a transit router. The inserted LSR option causes this packet to be routed to the mobile host via the MAS. When the mobile host sends a reply to the source, it also inserts the LSR option in all outgoing packets, again specifying the current MAS as a transit router. When the stationary host receives this packet, it will reverse the recorded route, and insert it in all outgoing packets that are sent to the mobile host. Thus, subsequent packets originating from the stationary host will be automatically routed along an optimal path. Notice that route reversal is an integral part of LSR option processing. Our scheme exploits this feature to provide optimal routing between stationary and mobile hosts.

Mapping: In this proposal, the MR acts as an ATA, and is also responsible for maintaining the LD. The MAS acts an FA for mobile hosts that lie in its wireless cell. The key feature of this proposal is that it enables function f to be co-located with all internet hosts without requiring changes to host software. All internet hosts, when generating replies to packets that are received with the LSR option, are required to do the route reversal [12]. For TCP connections, the route reversal is performed by the protocol processing module, and in case of UDP connections, this responsibility lies with the applications. From our reference architecture view point, the process of route reversal amounts to the task that an ATA is required to carry out. Thus, this scheme effectively exploits mechanisms already available within IP protocol, and achieves co-location of ATA with end hosts without requiring any modifications to host software. It is worth mentioning that this feature cannot be achieved using any scheme that is based on encapsulation. Unlike LSR, encapsulation is not a part of the standard IP protocol specification. Therefore, no internet host can generate encapsulated IP packets without suitable software modification.

Another important feature of this scheme is that no special protocol is required for distributing and managing LD cache entries. LD entries are automatically acquired through the incoming LSR option. Recall that packets which arrive at a stationary host already contain the address of the MAS. This, together with the source address of the packet, constitutes an LD cache entry. When a host starts a new session with a mobile host, it has no LD cache entry for the destination. Naturally, the first packet is routed to the destination via the MR. When the ACK for this packet arrives, it contains the LD cache entry[6] in the incoming LSR option. This LD entry is maintained on a per-session basis, and it maintained only as long as the corresponding TCP session is alive. When the session terminates, the corresponding LD entry is purged. If the destination moves during an active session, the LD cache entry becomes inconsistent. However, it gets updated as soon as the next packet from the destination arrives at the source. This constitutes a pure on-demand-cache-update policy which has a good scaling property. Following a host's movement, only those LD cache entries are updated which are in use. Compared with Sony's proposal, which requires a message to be broadcast to the network, significantly fewer messages are exchanged. Naturally, an on-demand-cache-update policy lends a scalable design; both with respect to the size of the network, and the rate of host mobility.

2.5 Summary

In this chapter, we first identified network layer concepts that play a crucial role in the design of mobile networking systems. We showed that the process of *address translation* is fundamental to providing any solution to mobility at the network layer. Our proposed network architecture employs three basic set of entities: *Address Translation Agent*, *Forwarding Agent*, and *Location Directory*, which co-operate with each other to carry out the operation of address translation. The proposed architecture is *general* and *flexible*. The architecture's generality enables it to capture all possible scenarios of communication between mobile and stationary hosts. Its flexibility allows sufficient freedom in terms of placement of these entities in the network.

We showed that all candidate proposals for MobileIP can be visualized as special cases of our proposed architecture. We demonstrated this by showing a one-to-one mapping between the entities in our architecture, and those required by the candidate proposals. Mappings represent set of design choices (i.e., where in the network these entities are located) made in the candidate proposals (see Table 2.2).

In addition to these design choices, there are several other considerations such as inter-operability, backward-compatibility, security, and authentication, which also play a crucial role in the design of a mobile networking system. In the next chapter, we'll describe the complete design and implemen-

[6] All BSD 4.3 compliant TCP implementations copy this information in the TCP control block

tation process of the LSR based mobile networking system, pointing out factors which influenced our design decisions. Interested readers can refer to [48, 51, 23, 32] for a description of other mobileip systems.

Scheme	Address Translation Agent (f)	Forwarding Agent (g)	Location Directory	Location Update Protocol
Columbia	Co-located with MSR	Co-located with MSR	distributed among MSRs	Only primary copy is modified. *Lazy-update* policy is used for updating cache entries
Sony	Co-located with all hosts and routers	Co-located with mobile hosts	LD is maintained at home router. Cache entries are acquired by snooping a packet header	Only primary copy is modified by the explicit `connect` message. Cache entries are modified by broadcasting a `disconnect` message, or are auto-flushed by a timeout mechanism
MobileIP working group	Co-located with home routers	Co-located with Foreign Agent, or with mobile host if DHCP is used.	LD is maintained at home router only.	Due to security reasons, caching of LD entries is not allowed. This implies when a host moves only the primary copy is required to be modified. A simple location update message from the mobile host suffices for this purpose.
LSR scheme	Co-located with all hosts and home routers	Co-located with mobile hosts	LD is maintained at home router. Cache entries are acquired through incoming LSR option	Only primary copy is modified. Cache entries automatically get updated when packets with new LSR option arrive. On-demand update policy, no broadcasts.

Table 2.1: Functional comparison of MobileIP schemes

Property	LSR	Columbia		Sony	IETF Mobile-IP
		In-campus	Out-of-campus		
Optimal Routing	Always	Always	Never	Only if all routers are modified	Never
Address Translation Mechanism	Loose Source Routing	Encapsulation		Encapsulation	Encapsulation
Additional Address Space Required	None	None	Double	Double	None. But required when using DHCP.
Failure Modes	MR is a single point of failure, but it does not affect on-going sessions	Robust against local MSR failures		Non-local	Home Agent is a single point of failure, and it affects all on-going sessions
Scalability	Good	Good		Poor	Excellent
Route Updates	$O(mobiles)$	$O(mobiles)$		$O(mobiles$ x $routes)$	$O(mobiles)$
Route Aquisition	Automatic	On demand		Flooded	Sent to Home Agent
Compatibility with IP	Total, so long as hosts and routers conform to standards	Total		Requires major changes	Total
Security	Insecure	Partially Secure		Insecure	Fully Secure

Table 2.2: Comparison of MobileIP schemes

Chapter 3

MobileIP: Design and Implementation

It is desirable that the integration of mobile computers within the existing Internet be completely transparent to the transport and higher layers so that users of mobile computers can continue to run existing applications on their computers. Any acceptable solution for mobility should inter-operate with the existing infrastructure and not require any modifications to existing host or router software. In the previous chapter, we described a general network layer architecture which meets these objectives. In this chapter, we present a detailed design of such a system.

We consider a networking environment which consists of a set of base stations connected to the wired network. Base stations and mobile computers are equipped with wireless transmitters and receivers which enable them to communicate with each other. At any point of time a mobile host is reachable via some base station. Since a base station is stationary, its network address can be used to denote the location of all mobile hosts within its range. Our solution approach is to include location information within each IP packet destined to mobile hosts. We use an existing IP option (Loose Source Route) to carry the location information. The key feature of our design is that all functionality needed to manage and process location information is embedded in specialized entities, called Mobile Router and Mobile Access Station. This allows seamless integration of mobile hosts into current networking infrastructure without requiring any changes to existing routers and hosts. As mobile computers move around, they break and reset wireless connection to the closest base station. Despite transient disconnection, no disruption in service is noticed by applications. Thus, users continue to be in touch with networking resources regardless of their location and whether they are mobile or stationary.

The proposed scheme was implemented on a set of IBM PS/2 and a set of IBM RT-PCs. Each PC was equipped with a 1 Mb/s infrared wireless LAN card. In the following, we desribe an overview of our proposal and then present its implementation.

3.1 System Components

Our system involves participation of three types of entities, viz., *Mobile Host (MH)*, *Mobile Access Station (MAS)*, and *Mobile Router (MR)*. The networking architecture that we assume is that of a set of MASs connected through a wired backbone (see Figure 3.1). An MAS supports at least one wireless interface and functions as a gateway between the wired and wireless side of the network. Due to the limited range of wireless transceivers, a mobile host can setup a direct link layer connection with an MAS only within a limited geographical region around it. This region is referred to as an MAS's *cell*. The geographical area covered by a cell is a function of the medium used for wireless communication. The range of infrared cells is typically limited to about 20 feet, while that of radio frequency cells could be significantly larger.

Within one campus or administrative domain there could be multiple (sub)networks reserved for mobile hosts. Each (sub)network has a separate router which is referred to as Mobile Router (MR). Unlike other routers, an MR is not required to have an interface corresponding to the wireless (sub)net it serves. If an MR has a wireless interface then it can also function as an MAS. The association between an MH and its current MAS is kept in a Location Directory (LD), which is maintained at the MR.

A mobile host retains its address regardless of which MAS cell it is in. It can start sessions with other hosts (both mobile and stationary) and move into other MAS cells without disrupting any active sessions. The movement of a mobile host is completely transparent to the running applications, except possibly for a momentary pause which may occur while the cell switch takes place. An MH can reside in the cell of only one MAS at any given time. Even if cells of two MASs spatially overlap, an MH routes its outgoing packets through only one of them. An MAS can have multiple MHs in its cell.

We use the term *Correspondent Host (CH)* to refer to the host communicating with an MH. In the following discussion, a stationary correspondent host is also referred to as *Stationary Host (SH)*.

3.2 System Operation

Our scheme is based on the use of IP's LSR option. The LSR option provides a means for the source host to supply partial routing information to be used by routers in forwarding the datagram to the destination. A source can specify a list of routers which are to be visited in the specified sequence before the datagram is delivered to the final destination. According to the host requirements document [12], return traffic to the originator of the loose source routed packet is also sent with

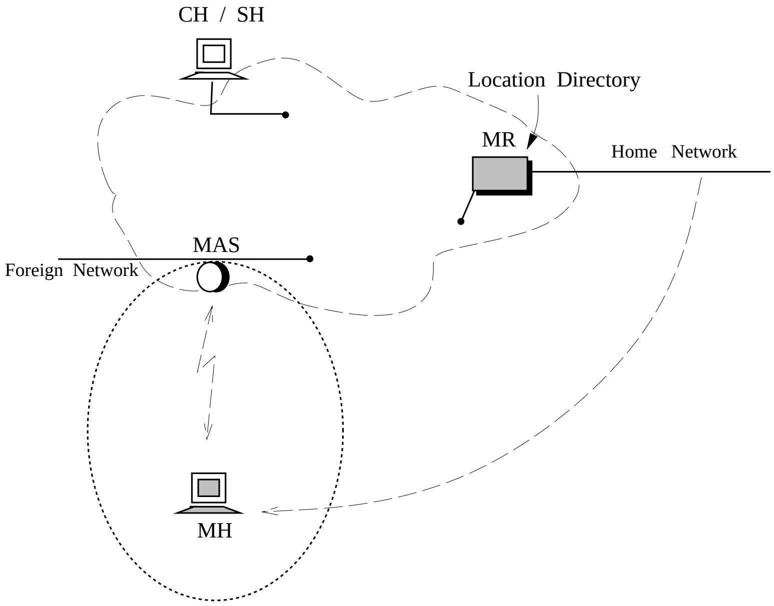


Figure 3.1: Mobile Networking System Components

the LSR option by reversing the route taken by the incoming packets. We use this technique to achieve optimal routing between an MH and a CH.

There are four possible communication scenarios depending on whether the CH is stationary or mobile and, if the CH is mobile, whether the MH and the CH are in the same cell or not. We consider each case separately and show how optimal routes are constructed in each scenario.

3.2.1 Mobile Host to Stationary Host

An MH, while communicating with an SH, issues packets with the LSR option which specifies that packets should be routed via the MAS serving the MH (see arcs 1 and 2 in Figure 3.2). The SH sends reply packets with the LSR option containing the reversed route. These packets are first delivered to the MAS which forwards them to the MH. Notice that if the LSR option is not used in the reply packets, then these packets would get delivered to the router (MR) for the MH's (sub)network (subsequently called the wireless subnet). The MR would eventually forward these packets to the MH; however, the complete path followed by the reply packets would not be optimal in this case.

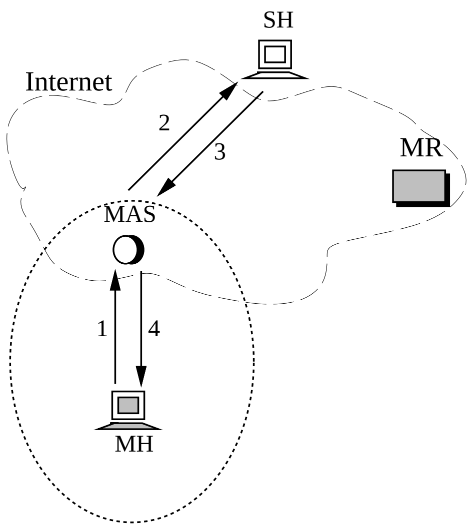


Figure 3.2: MH to SH

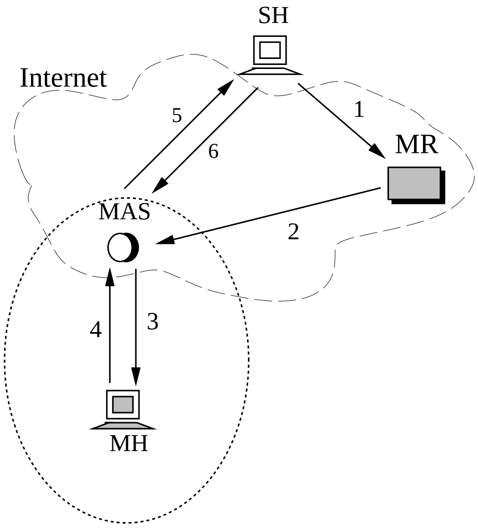


Figure 3.3: SH to MH

3.2.2 Stationary Host to Mobile Host

An SH need not be aware of the current location of the MH when it initiates a session. If it is not aware, the packet sent from the SH (denoted by arc 1 in Figure 3.3) arrives at the MR, which advertizes reachability to the wireless subnet. The MR, using the information in its Location Directory, inserts the LSR option in this packet which causes this packet to be delivered to the MH via the current MAS serving the MH (arcs 2 and 3 in Figure 3.3).

When a reply to this packet is sent, the MH reverses the LSR and sends the packet back to the SH via the MAS (as denoted by arcs 4 and 5 in Figure 3.3). Once the SH receives a source routed packet, it can send subsequent packets to the MH along the optimal path by reversing the incoming loose source route.

3.2.3 Mobile Host to Mobile Host within the same cell

An MH does not keep track of other MHs residing in the current MAS's cell. It always uses the current MAS as its default gateway for all outgoing traffic. When an MH initiates a session with another MH, it sends all packets to the MAS just as it would do if it were to send those packets to an SH. Since MAS keeps a list of all MHs residing in its cell, it can forward those packets to the destination MH. If the wireless link layer technology supports direct MH to MH communication then the MAS can also send an ICMP [40] redirect message to the source MH so that it can directly communicate to the destination MH rather than source routing its traffic through the MAS.

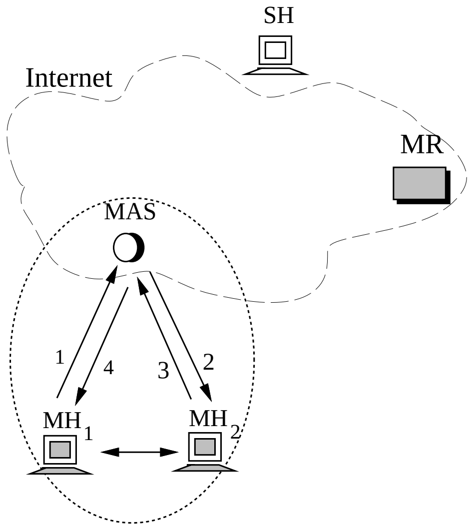


Figure 3.4: MH to MH (same cell)

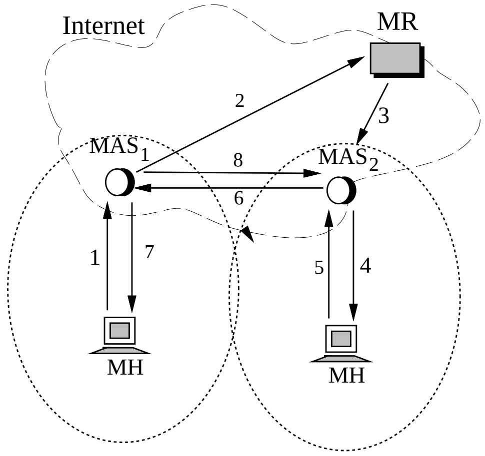


Figure 3.5: MH to MH (different cell)

3.2.4 Mobile Host to Mobile Host in different cells

An MH does not inspect the destination IP address to determine whether the destination host is a stationary host or a mobile host. Consequently, it always starts off by sending packets with the LSR option. By normal routing mechanisms, these packets are forwarded to the MR associated with the destination MH (see arcs 1 and 2 in Figure 3.5). The MR extends the existing LSR option by inserting the address of the the MAS presently serving the destination MH, and then forwards the packet. Normal routing procedure ensures that these packets get delivered to the MAS serving the destination MH, followed by the destination MH (as shown by arcs 3 and 4 in Figure 3.5). Notice that the LSR option list of the incoming packet contains addresses of two MASs – one serving the source MH, and one serving the destination MH. The reply packets are sent by reversing the incoming loose source route, which follow the optimal path as denote by arcs 5, 6 and 7 in Figure 3.5. Once the source MH receives a packet back from the destination MH, it can also send the subsequent packets along the optimal path.

3.3 Development Platform

Our development platform consists of a collection of IBM PS/2s and IBM RT personal computers. Each host is equipped with an infrared adapter and an infrared transceiver, or IRT (i.e. the part that contains electronics for converting a digital data stream into IR signals and vice-versa). These infrared devices operate at 1 Mbps speed and provide network coverage up to a distance of 20 to 25 feet. A local area network formed by a collection of IR devices operates much like an ethernet. The only difference is that optical transceivers are not capable of detecting collision. A brief hardware description of the IR card is included below to provide motivation for some of the design choices made in the MobileIP implementation.

3.3.1 Infrared LAN Card

The main function of the infrared network adapter card is to control the IRT. The IRT is designed to perform only transmit and receive functions. No protocol or network functions are present in the IRT. This design very closely resembles ethernet design, in which a network interface controller acts like a protocol engine while a transceiver operates like a slave unit.

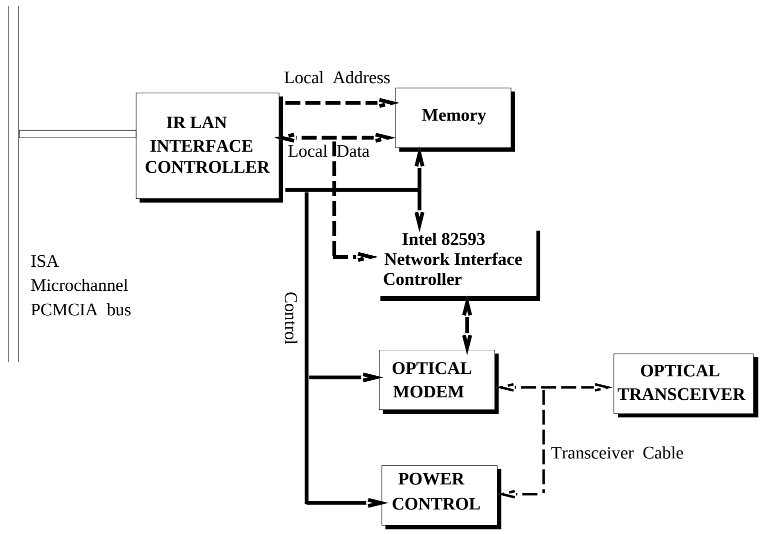


Figure 3.6: Block diagram of infrared LAN adapter

Figure 3.6 shows a block diagram of an IR adapter card. It consists of a Network Interface Controller (NIC), interface logic to the system bus, and interface logic to the IRT. The NIC consists of a CSMA based protocol engine and the necessary logic to support a speed matching FIFO between the system bus and the IR link. The IR network adapter card can plug into a variety of standard system bus interfaces, such as ISA, MicroChannel and PCMCIA. Its interface to a system bus is through an 8 bit wide data and address bus. Since the IRT only operates at 1 Mbps, an 8 bit wide interface is sufficient to operate the output link at full speed. The card also has a local SRAM for buffering transmit and receive data packets. The total buffering capacity is 32K, which is equally divided into 16K of transmit and receive buffers. On board buffers allow operating system software to manage incoming and outgoing data without fear of overrun or underrun conditions.

1 Mbps Infrared Transceiver: An infrared transceiver operates at 1 Mbps speeds. The IR signal (880 nm wavelength) emitted from transceiver is reflected off all solid objects and illuminates the whole room. Due to diffusion of light waves, transmitter/receiver pairs can communicate even if they are not in line-of-sight of each other. The IRT is connected to the NIC through an optical modem which is responsible for encoding and decoding digital signals transmitted or received from the NIC.

Network Interface Controller: The NIC used on the IR card is an Intel 83593 chip which is capable of supporting the IEEE 802.3 protocol. The basic operation of the NIC is very similar to an ethernet NIC. Since the 82593 chip was originally designed to interface with ethernet transceivers, a optical modem is required to interface the chip and the transciever. The main operation of the NIC is to manage packet transmission and reception. The programming interface to NIC is available through two I/O ports which are used by the infrared device driver (`eth_ir.c()`).

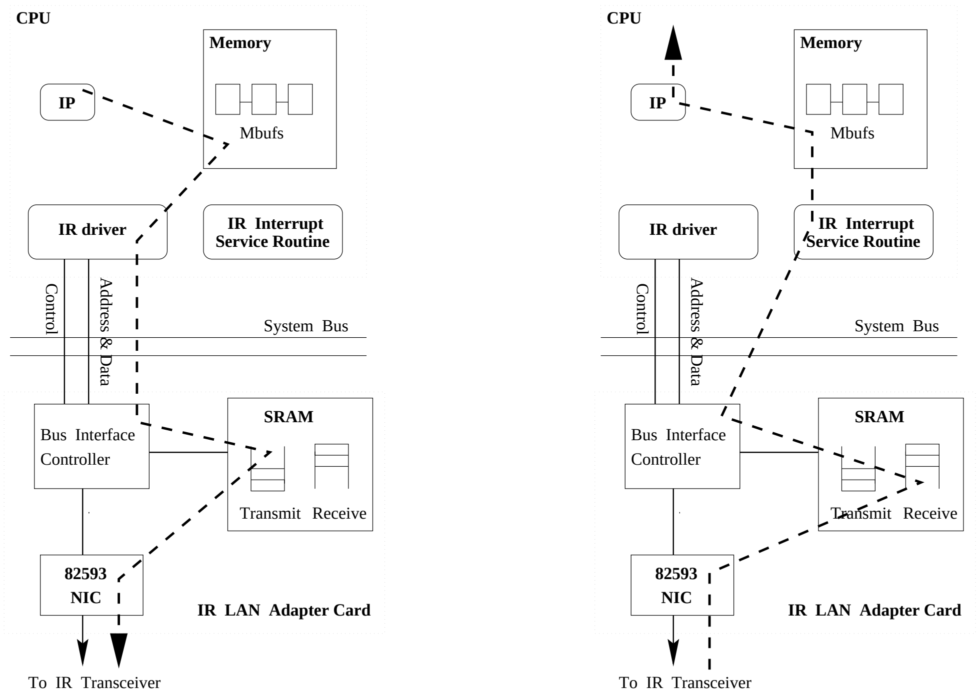
3.3.2 Infrared MAC Protocol

The infrared band constitutes a single channel which is shared among multiple stations by a simple *listen-before-talk* protocol. Each IR station has a globally unique 6 byte identifier, which is used as its MAC address. The access method uses a carrier sense mechanism to determine whether the wireless medium is available for transmission. Packet is transmitted if the carrier is found free. To reduce the chances of collision during transmission, the IR card implements a random backoff arrangement. Random backoff is used because collision detection is not possible on the wireless medium. Hardware does not provide any support for recovery from lost frames. It is up to higher layers to detect and support recovery from lost frames.

Although the MAC protocol allows direct point-to-point communication between any pair of nodes, this feature is rarely used in practice. In most cases, data is exchanged between mobile nodes and the access point. At the MAC layer, all nodes appear identical. There is no way to find out whether a specific MAC address corresponds to an access point. Access point discovery, therefore, must be supported at higher layers. In our implementation, access points periodically broadcast beacons which serve as access point advertisement messages. Ideally, this function should be provided by the link layer, but in this case, we had to implement it at the network layer.

3.3.3 Running IP over IR LAN

The first step toward building a working implementation of MobileIP was to layer the IP protocol stack atop the infrared interface. This was achieved by mapping the IP layer's output routine to the IR driver's entry point and coupling the driver interrupt service routine with IP's input queue. Similar changes were needed for the Address Resolution Protocol (ARP) as well. Figure 3.7 illustrates the sequence of steps involved in packet transmission. The IP layer passes packet in the form of an *mbuf*[1] chain to the IR device driver. The IR driver strips *mbuf* headers, adds a MAC header to the packet, and copies it onto the transmit buffer. When the wireless medium is free, the NIC empties the contents of the transmit buffer and issues a transmit complete interrupt to the CPU. The CPU at this point can reschedule the driver output routine and process the next packet for transmission.



a. Transmit Path b. Receive Path

Figure 3.7: Layering IP over infrared LAN adapter

On the receive side (see Figure 3.7), the NIC channels the incoming bit-stream directly to the receive frame buffer, and issues an interrupt to the CPU when frame reception is complete. At this point, the driver interrupt service routine takes over the charge. It first strips the MAC header from the received frame, converts it into *mbuf* format, and then copies it to the main memory. The IP layer picks up this packet for further processing.

Once it became possible to encapsulate IP frames within IR MAC frames, two hosts could begin to communicate over wireless. Some problems, however, quickly surfaced:

[1] *mbuf* is a data dynamic data structure used for storing packets in the memory.

- Hosts from two different networks could not communicate over wireless even if they were within direct communication range. This is because the IP subnet model does not invoke ARP to discover the destination's MAC address if it is located on a foreign network.

- Neither IR hardware nor IP layer provided any mechanism to detect when a host moved.

In the next section, we show these problems were resolved by MobileIP.

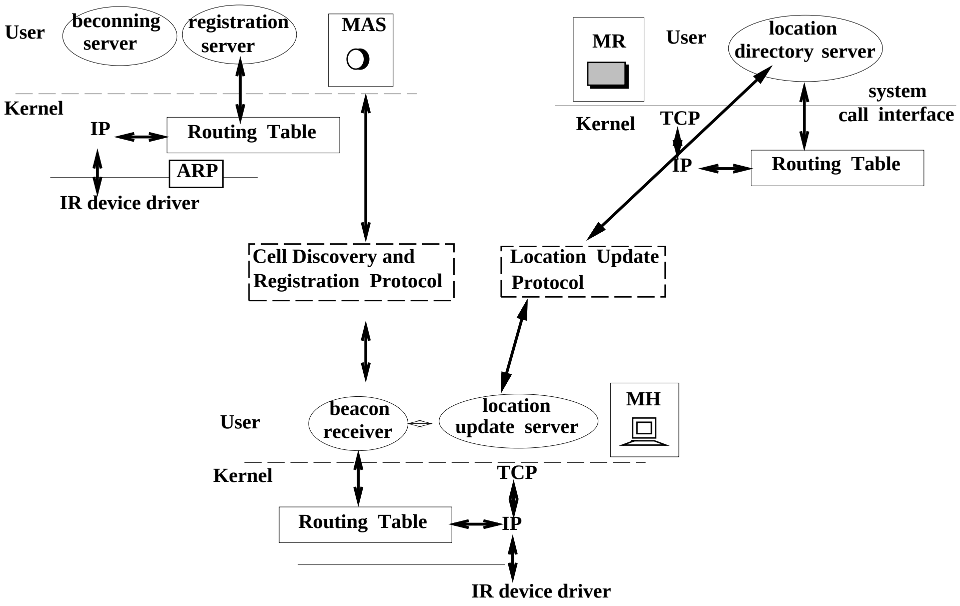


Figure 3.8: System Overview

3.4 System Description

The MobileIP system can viewed of as consisting of two parts, viz., the *packet routing part* and the *location information management part* (see Figure 3.8). Actions related to packet routing are performed in the kernel. To avoid creating new data structures, location information is stored implicitly in the kernel routing table. This approach has some obvious advantages. First, minimal kernel modifications are needed to route packets to and from MHs. Secondly, with a little modification, the existing *route* command can be used to manipulate the location information. In the following, we first describe how packets are routed among various components and how location information is managed. Later, a description of the processing required at each component is included.

3.4.1 Packet Routing

For each MH, which has an address on the wireless (sub)net served by an MR, a host route is maintained by the MR. The current location information of the MH, i.e., the address of the MAS serving the MH, is kept in the gateway field of the routing table entry. This routing table entry is distinguished from other entries by the presence of a new flag called `RTF_MOBILE`. Since the MR advertizes reachability to the range of addresses on the mobile (sub)net, an IP packet destined to an MH is first routed to the MR for further delivery. At the MR, one of the host routes with the `RTF_MOBILE` flag is chosen to route this packet. The MR knows how to interpret the `RTF_MOBILE` flag; that flag specifies that the MR should insert the LSR option in this packet (by using the MAS[2] as the intermediate hop) before forwarding it. Due to the inserted LSR option, this packet is delivered to the MAS currently serving the destination MH. The LSR option is processed here and, finally, the packet is delivered to the MH.

An MAS performs the forwarding function between the wired and the wireless interface. It keeps a host route corresponding to each MH residing in its cell. Thus, an IP packet which is destined to one of the MHs in its cell is delivered to the MH by the MAS.

Processing of packets originating from MHs is relatively simpler. An MH always keeps a default route to the MAS currently serving it. This routing table entry also has the `RTF_MOBILE` flag set, meaning that the LSR option should be used on all outgoing packets. Packets originating from an MH are source routed via the current MAS and are delivered to the final destination by the normal routing mechanism. The reply packets use the reverse route and are delivered back to the MH by the optimal route.

3.4.2 Location Information Propagation

We now describe how the location information, which is implicitly maintained by routing table entries at the MR, MAS, and MH, is updated when an MH switches cells.

A server program (`beacon_snd`) running on the MAS periodically broadcasts beacons on the wireless interface. These beacon packets contain the IP address of the MAS. An MH, upon entering the MAS's cell, receives these beacons and sets up the MAS as its default router. At the same time the MH also sends an `mh2mr` message to its MR. This message contains the new location information of the MH and a timestamp. A server (`mr_serv`) running at the MR receives this message, uses the location information contained in it to update the MR's routing table, and notifies the previous MAS serving the MH about its migration by sending it a `delete_host_route`

[2]The IP address of the MAS currently serving the MH is available from the gateway field of the routing table entry.

message. The previous MAS, upon receiving this message, deletes the host route corresponding to the migrated MH.

All messages are exchanged using UDP packets. The message exchange protocol is very simple and completely implemented by user level processes running at the MH, MR, and MAS.

3.4.3 Processing at a Mobile Host

The software running on a mobile host performs primarily two functions: cell discovery and insertion of the LSR option in all outgoing IP packets.

The purpose of the cell discovery operation is to send a notification and to supply the address of the MAS to the network layer whenever an MH performs a cell switch. It is desirable that the cell discovery feature be supported by the link layer. However, due to lack of hardware support, a user level process (`beac_rcv`) performs this operation at each MH. This process continuously monitors beacon packets that are broadcast by MASs and changes the default route at the MH whenever a cell switch occurs. If the cells of two or more MASs spatially overlap, the MH may potentially receive beacons from multiple MASs. The cell discovery mechanism should avoid rapid cell switching by the MH in this scenario. In our implementation, if beacons from the current MAS continue to arrive at regular intervals, then beacons from other MASs are ignored. The MH switches to another MAS cell, i.e., changes its default route, if it does not receive any beacon from the current MAS for 3 seconds. After an MH has determined the cell to which it belongs, it is required to notify its MR about its current location. This is accomplished by sending an *mh2mr* message to the MR.

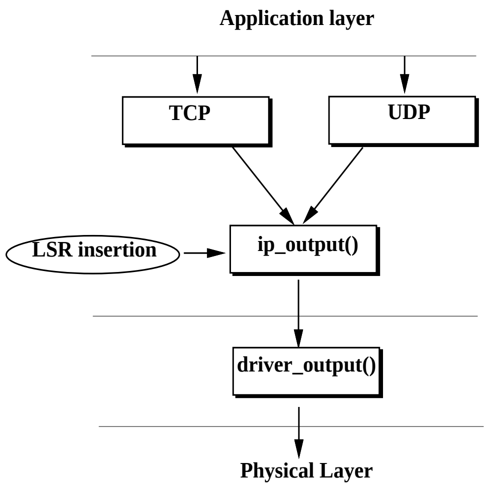


Figure 3.9: Kernel processing at Mobile Host

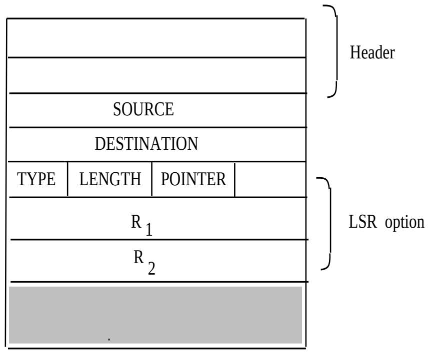


Figure 3.10: An IP header with LSR option

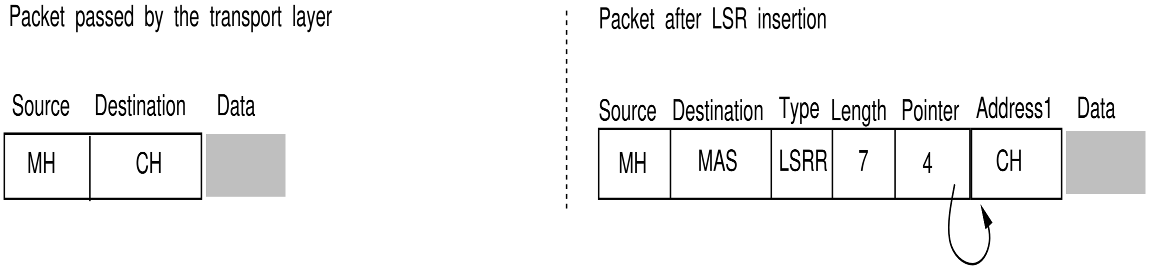


Figure 3.11: LSR insertion by an MH

Unlike cell discovery which is performed by a user level process, the LSR option insertion is performed in the MH kernel (see Figure 3.9). The transport layer (TCP, UDP) running at an MH passes packets to the IP layer for further delivery. The IP layer at the MH is required to modify this packet before passing it on to the network interface. If the outgoing packet does not contain the LSR option, then conceptually the MH must insert the LSR option and specify the MAS as the only intermediate router. In some cases, the packet passed by the transport layer already contains the LSR option. This option is formed by reversing the route contained in the LSR option of an incoming packet. An existing LSR option in an outgoing packet has already been extended by

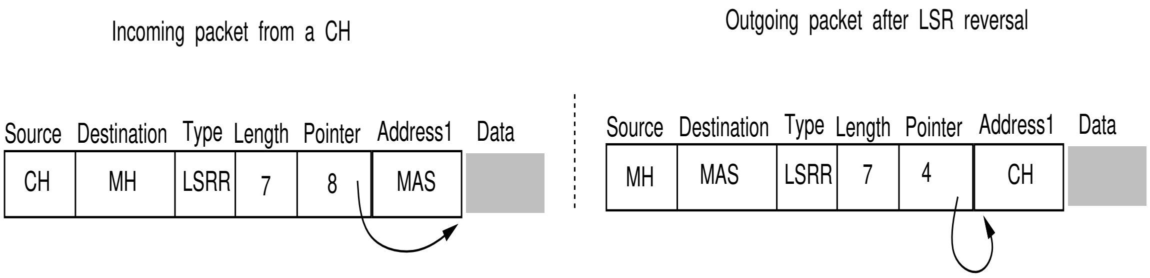


Figure 3.12: LSR reversal by an MH

the MR to include the local MAS as a required intermediate router for the source route. This model of operation conforms to the description given in Comer[16]. However, since in most current implementations all routing is done on the basis of the destination address, the actual operation is somewhat different (see Figures 3.11 and 3.12). The only routine which needs modification is `ip_output()`. The changes are described below:

```
if ( pkt does not have LSR option) {
    insert LSR option in pkt;
    first address in option list = pkt.destination;
    pkt.destination = MAS address;
} else      /* option already exists by TCP route reversal */
    if ( pkt.destination != MAS address )
        pkt.destination = MAS address;
```

Suppose a packet from a CH arrives at the MH and thus has an LSR option. Suppose also that before a reply is sent, the MH switches to another MAS cell. The destination address contained in the reply packet will be that of the previous MAS. Since the MH has already left the previous MAS cell, it is necessary to modify the destination address of this packet to the address of the current MAS. After this packet reaches the CH, the subsequent packets originating from the CH will follow the reverse route and arrive at the MH via the current MAS. Thus, no interruption in the active transport layer session will be observed.

3.4.4 Processing at a Mobile Access Station

The primary function of an MAS is to provide an access point through which an MH, regardless of its address, can attach itself to the network. To forward packets to MHs, an MAS should keep a host route corresponding to each MH that resides in its cell. The host route is installed by the MAS kernel as soon as the first packet (e.g, ARP packet) is received from the MH. Beyond this point, the existing `ip_forward()` code is able to forward packets to the MH via the wireless interface. When an MH moves out of the MAS cell, the corresponding host route at the MAS should be deleted, otherwise the MAS would continue to transmit packets to the MH which no longer exists in its cell. As mentioned earlier, a server process (`mas_serv`) running on the MAS performs the deletion of the host route when it receives a `delete_host_route` message from the MR.

After an MH moves out of the MAS cell, the packets addressed to it may still arrive at the MAS. These packets could be directly forwarded to the MAS currently serving the migrated MH. This, however, would require the MAS to maintain forwarding pointers for all MHs even after they migrate out of its cell, thereby, unnecessarily increasing the protocol complexity and memory overhead. We wanted to design a system where there was no need for any information exchange among MASs. Therefore, packets which cannot be delivered by the MAS are returned to the MR associated with the destination MH. Note that the MAS need not know the address of the MR for this purpose. The MAS only forwards this packet on to the wired side of the network. The normal routing mechanism automatically delivers this packet to the MR, since it is the router to the MH's home subnet. It is necessary to mark these returned packets from MAS so that MR can distinguish them from other packets. Since there is no extra space available in the packet where this marking can be incorporated, we do it implicitly by setting the last address in the LSR option equal to the packet destination address.

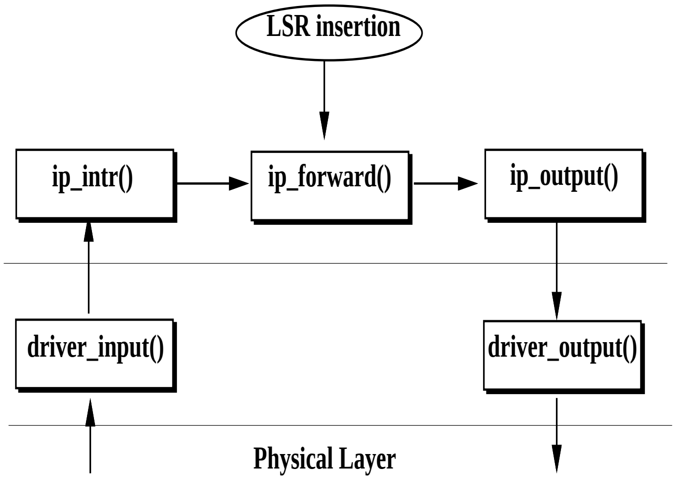


Figure 3.13: Kernel processing at Mobile Router

3.4.5 Processing at a Mobile Router

An MR performs the following two functions:

1. Maintains location information of all MHs on its (sub)net.
2. Forwards packets to MHs (via an MAS) by inserting/updating the LSR option.

Location information is nothing but the necessary routing information which is maintained at an MR. Therefore, the most convenient place for storing it is the kernel routing table. Corresponding to each MH, a host route is maintained at the MR. The gateway field of the route table entry stores the address of the MAS which servers the MH. Each time an MH switches to another cell, it sends a location update message (`mh2mr`) to the MR. A user level process (`mr_serv`) receives this message and updates the routing table by executing a `route` command.

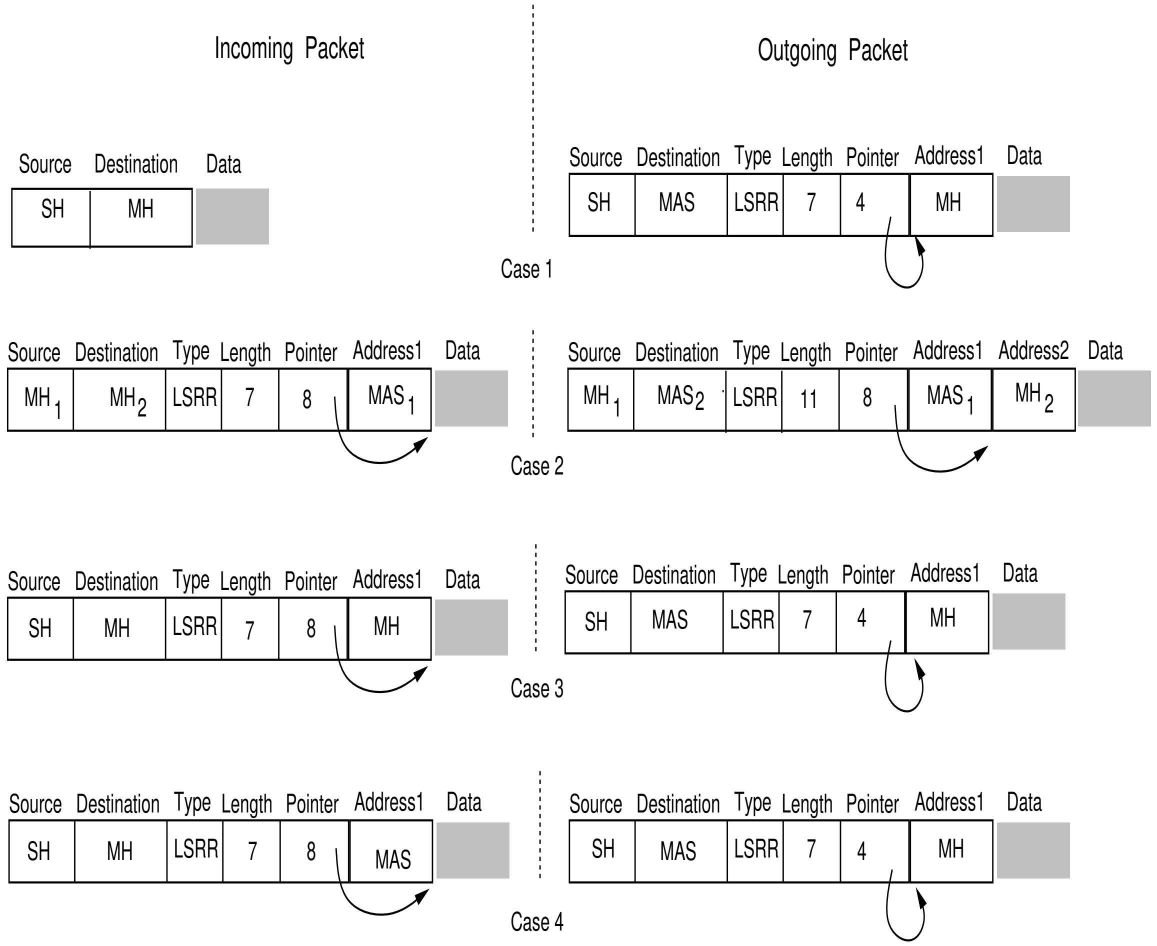


Figure 3.14: Packet processing at MR

Actions related to the LSR option insertion are performed in the kernel. The only routine that needs modification is `ip_forward()`. Packets arriving at the MR, which are destined to some MH, can be classified into four categories depending on the contents of their LSR option. For each case, Figure 3.14 shows the format of incoming and outgoing packets. The steps taken by the MR are outlined below.

1. Packets originating from a Stationary Host : The packet originating from a stationary host and destined to some MH usually does not contain any LSR option until it reaches the MR associated with the destination MH. The MR modifies this packet (as shown in case 1 of Figure 3.14) by inserting an LSR option in it. Notice that the original packet destination address is moved in the LSR option list and the new packet destination is set equal to the address of the MAS presently serving the MH. This modification causes this packet to be forwarded to the MAS, where the LSR option is processed, and finally the packet is delivered to the MH.

2. Packets originating from a Mobile Host : An MR can also receive packets which already contain the LSR option. This can happen, for example, when an MH (say MH1) starts up a session with another MH (say MH2). Suppose, in addition, that MH1 and MH2 reside in the cells of MAS1 and MAS2, respectively. When the packet originated from MH1 arrives at the MR, it appears as shown in case 2 of Figure 3.14. The destination address in the packet header contains the IP address of MH2. The LSR option list contains the address of MAS1 with the pointer pointing beyond the end of the address list.

The MR takes the destination address and appends it to the end of the option list and places the IP address of MAS2 into the destination field of the packet header. The normal forwarding mechanism first delivers the packet to MAS2, and the packet is finally delivered to MH2.

3. Packets returned from a Mobile Access Station : A packet arriving at the MR could also be from an MAS which failed to deliver this packet to the target MH. As in the previous case, this packet also contains the LSR option with one address and pointer pointing beyond the end of the address list (see case 3 in Figure 3.14). However, unlike the previous case, the MR should forward this packet to the current MAS serving the MH without extending the LSR option list. To do that, the MR should be able to distinguish this packet from the packets mentioned in case 2. The MR compares the destination address of this packet against the address contained in the LSR option list. If it is a packet returned from an MAS, then these two addresses are same. In this case, the MR sets the destination address field in the packet to the address of the current MAS serving the MH, resets the pointer in the LSR option to the beginning of the address list, and forwards the packet.

4. Incorrectly processed packets from a Stationary Host : When an MH starts a conversation with an SH, the packets arriving at the SH contain the LSR option. If the SH correctly processes the LSR option, the reply packets will automatically follow the reverse path without any involvement of the MR. Unfortunately, several IP implementations, such as 4.3 BSD and SUN OS 4.1, do not correctly process the LSR option. Even though the incoming source route is reversed and included in the reply packet, the destination address is set equal to the original source and the pointer is made to point to the end of the option list. This has the effect of sending the packet straight back to the original source, rather than sending it along the reverse path. Since the destination of this packet is an MH, normal routing mechanism delivers this packet to the MR associated with the MH. A sample of such a packet is shown in case 4 of Figure 3.14.

The MR checks the last address in the LSR option list. If this address is of the MAS associated with the destination MH, then the packet is coming from an SH which does not correctly process LSR options. The MR swaps the addresses stored in the destination address field and LSR option field, resets the pointer to the beginning of the address list, and forwards the packet to the network interface output routine.

3.5 Cell Switching

In the MobileIP implementation, the process of packet routing is controlled by the contents of the routing tables maintained at the MH, MAS, and MR. The MAS maintains a host route for each MH in its cell. Similarly, each MH maintains a default route to its current MAS. The Location Directory (LD) is maintained in the form of a routing table at the Mobile Router. When a mobile host moves to a new location, the cell discovery and location update protocols ensure that routing tables at the MH, MAS and MR are updated in a co-ordinated fashion.

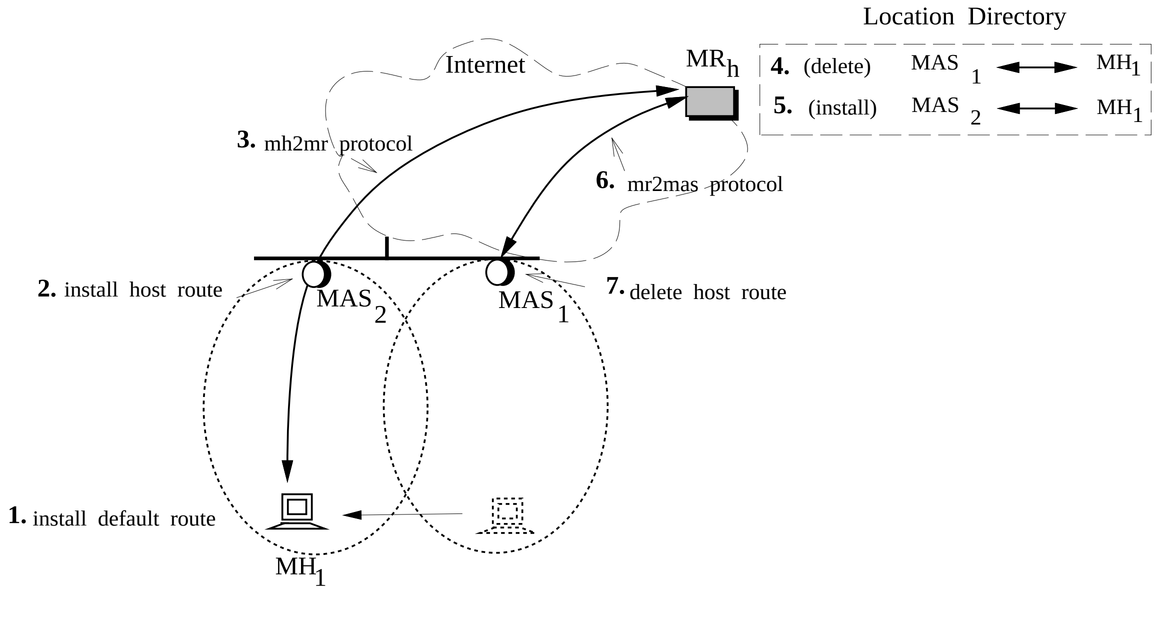


Figure 3.15: Cell Switching

Figure 3.15 illustrates the set of messages that are exchanged and the sequence of events that take place when the mobile host moves from the cell of MAS_1 to MAS_2. Figure 3.16 shows the temporal relationship among the events that occur during the process cell switch. t_s denotes the time when MH_1 enters the wireless cell of MAS_2 and t_f marks the completion of the location update protocol. MH_1 starts receiving beacons from MAS_2 the moment it enters the wireless cell of MAS_2, but it continues to route its outgoing packets via MAS_1 until it moves out of the range of MAS_1. The `beacon receiver` at MH_1 is responsible for detecting this event. In Figure 3.16, the interval $(t_1 - t_s)$ denotes the latency of detecting the cell switch.

The process of cell switching starts by changing the default routing table entry at the mobile host. This change is sufficient to cause all packets to be relayed via the new MAS. The MH_1 is responsible for notifying its home router, MR_h, about its new location. This is accomplished by the use of `mh2mr` protocol handshake (the details of this protocol can be found in appendix A). When the first packet from MH_1 arrives at MAS_2, it updates its routing table to add a host route corresponding to MH_1. This modification was embedded in the ARP protocol processing at MAS_2. When the first packet from a new host arrives at the wireless interface (usually the ARP request packet), before the reply is generated, a route is automatically installed. Notice that this processing is not required for each packet that arrives at MAS_2. In fact, doing so would be computationally very expensive.

The `mh2mr` message carries the new location information of the mobile host. When this message arrives at MR_h, it is required to update the LD to reflect this change. Since location directory is implicitly maintained in the form of the routing table, updates to the LD are performed in two steps: 1) deletion of the previous routing table entry, followed by 2) installing a new table entry. This is because, the unix system call interface does not support any update operation on routing table entries. The period $(t_5 - t_4)$ should be as small as possible to prevent loss of any packets that arrive during the interval for which MR_h does not have the appropriate routing table entry.

Location updates issued from MH_1 must be communicated to MR_h in a reliable fashion. Therefore, MR_h is required to send an `mh2mr_ack` to MH_1, when update to its routing table is committed. Instead of choosing TCP to relay `mh2mr` packets, we decided to design our own handshake protocol using UPD messages. There were several reasons behind it.

- Latency of the UDP based `mh2mr` protocol is 3 times less than that of TCP; TCP requires 3 round-trips worth of time (connection setup, data-ack, FIN-Close) to transmit a single packet.
- Processing overhead of TCP is more than UDP based protocols. For a single data-ack type of transaction, TCP is a very heavy duty protocol to use. In addition, if the mobility rate is very high, setting up a TCP connection for each arriving update will be infeasible due to memory and CPU limitations.
- Since `mh2mr` packets are likely to be lost, relying on TCP's retransmission policy will cause large delays in the execution of the LD update protocol. A UDP based protocol provides flexibility to implement more aggressive retransmission policies, and fine tune the retransmission timer for a specific mobile environment.

When the MR sends an ACK to the MH, it also sends a notification to the previous MAS asking it to delete its host route corresponding to the migrated mobile host. The details of this protocol

Time Interval	Description	Average Value
$(t_1 - t_s)$	Cell switch detection latency	3.5 sec.
$(t_2 - t_1)$	Message propagation time over the wireless link	4 ms
$(t_3 - t_2)$	Message delay over the wired network	1.2 ms - 60 ms
$(t_4 - t_3)$	Time to process *mh2mr* message. It includes time for context switching, and protocol processing.	450 μs
$(t_5 - t_4)$	Delay between route updates	see table 3.2
$(t_6 - t_5)$	Message transmission latency at MR	500 μs
$(t_f - t_6)$	Transmission delay over path from MR to MH.	same as $(t_3 - t_1)$.
$(t_f - t_1)$	Location Update Delay	

Table 3.1: Table of Latencies

can be found in appendix A. It is important to delete this route for otherwise the previous MAS will continue to relay packets destined to the MH in its wireless cell. Similar to the `mh2mr` protocol, `mr2mas` packets are also required to be ACKed by the MAS. MR retransmits lost packet up to a limit set by max retry counter.

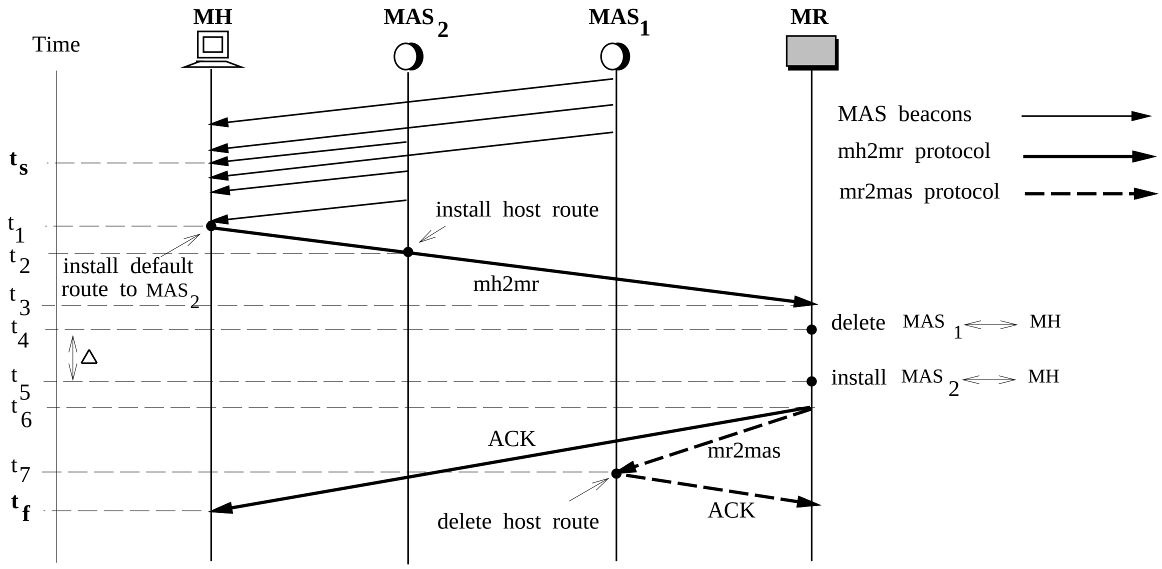


Figure 3.16: Timing Diagram

3.5.1 Optimizing LD Updates

We mentioned earlier that during the period $(t_5 - t_4)$, no routing information for the MH is available. If a packet destined for the MH arrives at MR during this interval it can potentially create a routing loop. Since in this period, the MR does not have a host specific routing table entry for this destination, it will use its default route to relay this packet to the next hop router. But the next hop router will bounce this packet back to MR since MR is the router which advertised reachability to the destination. In the following we describe three approaches for solving this problem, where each one was an improvement over the previous one.

Application Controlled Updates

Due to its simplicity, this approach was used in the first version of the implementation. AIX 1.2 provides a user level command, `route`, which can be used to install and delete routing table entries. When an `mh2mr` packet arrives at the MR, the location update server `forks` and then spawns a child process which executes the `route` command. Table 3.2 shows the average time to execute the `route` command at low and high priorities.

System Call Assisted Updates

As an improvement over the previous scheme, the next step was to directly call `rt_ioctl()` system call from within the location update server and avoid the cost of spawning a new process.

Supporting Atomic Updates

To improve things further, we designed a new system call interface to support update operations on routing table entries. We added a new SIOC_UPDTRT parameter to the `ioctl()` system call. This allowed us to directly manipulate routing table entries. Since updates to shared data structures are done within critical sections, an update operation to the routing table is atomic.

Performance Comparison: Since the PS/2 hardware does not provide access to the hardware clock, to access the relative performance improvements of these proposed modifications, we re-implemented the new system call interface on a RS/6000 machine. The table 3.2 shows the results of our measurements.

LD Updates	Application directed		System Call assisted		Atomic Update
	p_{low}	p_{high}	p_{low}	p_{high}	
mean	50.97ms	40.34ms	0.162ms	0.152ms	0.067 ms

Table 3.2: Performance Comparison of LD update strategies

3.5.2 Overhead

The processing and memory requirements of our scheme are very moderate. The overhead of storing the LSR option is 8 bytes per packet (and 12 bytes in case of MH to MH communication, when they reside in different cells). Assuming that the average packet size is 500 bytes and most of the communication is between MHs and SHs, the overhead of carrying LSR option is less than *2%*. Each host route requires 128 bytes of memory as it is stored in an *mbuf*[3]. At an MAS, the number of host routes is equal to the cell population. If we assume 20 as an upper bound on the cell population, only 2K bytes would be required to keep the routing information. At an MR, the number of host routes will equal the number of mobiles which are assigned addresses on the wireless subnet. Suppose an MR serves 256 mobile hosts, then it will require about 32K bytes to store the location information. Given that we only require one MR per wireless subnet, this is a very reasonable figure. It is worth mentioning that it is not necessary to keep the location information in the kernel routing table. We did it only to simplify our implementation effort. A further reduction in data memory requirement can be achieved if a separate table is used to keep the location information, although more processing would be required.

3.6 Critique

Alternative solutions for supporting mobility are based on the encapsulation approach. This approach relies on the presence of a packet forwarding agent which can perform a packet encapsulation/decapsulation function. A packet destined to a mobile host is intercepted by a packet forwarding agent. Intercepted packets are encapsulated and forwarded using the address of another agent that is located close to the mobile host. The destination agent strips the encapsulation header and relays the original packet to the mobile host.

The major problem with this approach is that routing is always sub-optimal, unless the packet forwarding agent is located close to the source. Another problem is the fragmentation and reassembly required at the packet forwarding agent. It has been observed that the majority of packets in the network are either very short or very large in size. Packet fragmentation may occur at the packet forwarding agent if the extra bytes added for encapsulation cause a packet to exceed its maximum transmission size. This may also happen at an MR when it inserts the LSR option. However, the risk of fragmentation is higher in an encapsulation based approach because the added encapsulation header size is greater than the LSR option size.

The LSR approach also has a few shortcomings. The first problem is that UDP[39] agents

[3]mbuf is a data structure that is used for dynamic memory allocation in Unix

do not perform route reversal. As a result, UDP packets originating from a stationary host are always routed sub-optimally via a mobile router. Notice that if there is no encapsulating agent close to the stationary host, any scheme based on encapsulation approach will also suffer from this problem. One approach to get around this problem is to distribute the MR functionality among multiple cooperating MR agents and place them close to the UDP sources. This also improves the robustness of the system.

Another problem with the LSR option is that there is a potential security risk associated with its use. The authentication mechanisms which are based on the source address lookup can easily be broken with the use of the LSR option. For this reason, many routers disable this. Note that encapsulation based methods also have the same problem unless each address translation agent (such as an Mobile Support Router (MSR) in the Columbia scheme [22]) knows every other forwarding agent. We should not overlook the benefits of source routing only on the basis of security limitation. IP itself has a lot of security holes. Besides, mobile networks, due to wireless links, are quite prone to security attacks. Better methods for data and address encryption are required to make wireless networks robust against impersonation attempts. In the future, some of those techniques can be used to make the LSR usage more secure.

Another concern that has been raised is related to the performance aspect of LSR. Compared to normal IP packets, more processing time is required to process packets which contain options. Existing routers manufactured by router vendors have specialized code to rapidly forward IP packets that do not contain any IP options. It is, however, not difficult to optimize this, since a similar performance enhancement could be made for packets containing precisely the IP LSR option with option data of the expected length(s). In the normal case, routers would not have to do any further processing, but would only have to forward the IP packet unchanged.

The following is a brief summary of the comparison of the two approaches.

Criteria	LSR	Encapsulation
Existing Implementation	Usually Poor	Nonexistent
Impersonation	Easy spoof	Easy spoof
Router Efficiency	Not optimized	Optimized
Additional Fragmentation	Less	More
ICMP error propagation	Correct	Incorrect
Fragmentation/Reassembly	End-to-End	Done by encapsulating agent

3.7 Summary

In summary, we have shown that LSR provides a fast and elegant solution to the problem of providing seamless networking for mobile hosts using the Internet Protocol. The use of LSR also enables optimal routing in most cases, without incurring the expense of agents interceding for the destination hosts, as long as those hosts implement IP as it has been specified now for a decade.

Besides using LSR, our approach demonstrates a modular construction with good separation between layer 2 (MAC/link layer) and layer 3 (IP) functions (see Figure 3.8). The beaconing, cell discovery, and cell switch mechanisms are all quite distinct from location update and routing mechanisms. Either feature could be replaced without substantially affecting the other. Indeed, new cell discovery mechanisms are likely to become commercially available at protocol layer 2, and our system will still work in almost exactly the same way.

Lastly, our system has shown itself to be quite practical. We have had test systems running in our laboratory for over two years with the current implementation. The mobile hosts are quite usable, and all new kernels and other programs have been installed by wireless file transfers of various sorts. The application transparency has been perfect; X11, NFS, and all the normal networking software has required absolutely no change. The only change above the IP network layer has been to eliminate the route-caching feature built into TCP for performance reasons. We have not noticed any significant slowdown in operation of the wireless network except for bulk data transfers, and that is due to the relatively slow speed of our wireless adapter hardware.

Chapter 4

Seamless Mobility

MobileIP enables distributed networking applications to run in a non-disruptive fashion while allowing hosts running these applications to change their attachment points to the network. Implicit in the MobileIP scheme is an assumption that a mobile host has single *physical* interface which is used to connect to the network. This interface is assigned an IP address, which remains fixed during the lifetime of the host. Though a mobile host frequently changes its location, it continues to use the same physical interface to connect to the network. In other words, the layer 2 medium through which the mobile host connects remains unchanged. An example is a laptop equipped with an infrared adapter card. When the user carries it to another room, it uses the same physical adapter card for communication. Only the access point changes.

In this chapter, we consider a mobile computing environment where a variety of layer 2 connectivity options are made available to roaming hosts. We assume that a mobile host is equipped with multiple physical interfaces, such as infrared, RF wireless, and/or Ethernet. More than one network interface could be in use at the same time. Since each host interface is required to be associated with an IP address, a host of this kind will have multiple IP addresses. In the Internet terminology, a host which is equipped with multiple communication interfaces is termed a *multihomed* host [13] . A multihomed host has multiple identities (IP addresses), and each one of those can be used to set up a network connection to the host. Depending on the address that is used during the connection set up phase, the associated physical interface is used to send or receive packets.

Consider an office environment where a mobile user connects a laptop to the network via two interfaces: ethernet and infrared. The ethernet interface is used to connect the laptop to a file server and a compute server. The infrared interface is used to form a personal area network of computing devices located in the office premises, such as printers, phones, active badges, and PDAs. Suppose this user now carries the laptop to a conference room where ethernet connectivity is not available; the network access is possible only through an infrared interface. Despite a change in the layer 2 medium, it is natural that the user will desire access to the same networking environment that is available from the office location. To provide this functionality, it will be necessary to switch some connections from the ethernet to the infrared interface.

The following two requirements naturally follow from the description of this scenario:

- A technique for switching from one communication medium to another without disrupting active network connections.
- A method for moving specific connections; instead of moving all active connections from one medium to another at the same time.

We show that support for this flexibility can be provided within the MobileIP framework by co-locating the forwarding agent and the mobile host. The support for media switching can be embedded at any layer in the protocol stack. In section 4.1, we discuss various design consideration that motivate a network layer solution to this problem. In section 4.3 we describe our solution. Finally, in section 4.5 we describe how our solution can be extended to support connection level switching.

4.1 Design Objectives

The following list reviews the high level design considerations, which motivate a network layer solution to this problem.

Media Independence: A variety of wireless services today are available. Some are designed to provide only in-building coverage while others are suitable for use only in a wide area setting. No wireless technology is expected to provide universal coverage, since each caters to a niche application area. It is expected that mobile hosts will connect to the Internet through a variety of link types, depending on their location. Ensuring that the same solution works across an array of different technologies is an import design consideration. As long as the underlying link layer supports the encapsulation of IP packets, it should be possible to connect mobile hosts through a new communication medium and resume suspended sessions if any exist. In addition, support for media switching should be independent of the underlying communication medium.

Media Adaptation: The ability of a system to respond to variations in link availability and bandwidth is important. Since mobile hosts sometime move out of the coverage area of the wireless link, a mechanism should be available so that alternate media can be utilized to retain active sessions.

Software Intensive Philosophy: Certainly, it is possible to design network interfaces that provide support for media switching at the hardware level. A network interface of this type will support connections to multiple network types and hide media switching from the protocols running on mobile hosts. However, such a solution is bound to be inflexible and expensive. It will limit switching capability to only a few network types. Moreover, the added complexity will make the hardware costly. Software-based solutions reduce the complexity of network hardware by moving functionality into the software of the end-node. This makes software solutions more economical, and at the same time, independent of any specific hardware technology.

Backward Compatibility: A switch from one communication medium to another should be completely transparent to existing applications. It should not compromise the ability of existing internet hosts to communicate with multihomed mobiles and vice-versa. This is possible only if media switching is completely hidden from protocols at and above the transport layer.

Media switching essentially means a change in a host's attachment point to the network. Whether or not this change is visible at the network layer depends on the configuration of host interfaces. For example, if a host equipped with two network interfaces is assigned only one IP address, then a change in the network interface does not constitute a move at the network layer. Interface switching becomes visible at the network layer when each interface has a different IP address. In the following, we consider various addressing options for multihomed hosts and discuss their implications for supporting media switching at the network layer.

4.2 Addressing Considerations

In the subsequent discussion, a host equipped with multiple network interfaces will be referred to as a *multihomed* host. For a multihomed host, each network interface represents a separate *physical* point of attachment to the network. The notion of attachment that is visible at the network layer, however, is quite different. Each IP address that is assigned to the host represents a separate *logical* interface or logical point of attachment to the Internet. Usually there is a one to one correspondence between the physical and the logical interfaces of a host. However, other types of mappings are also possible and are not ruled out by the Internet architecture.

In the Internet architecture, transport layer entities do not have a separate identification. Network layer addresses in conjunction with port numbers are used for the identification of transport layer entities. In the case of a multihomed host, for each connection originating or terminating at the mobile host, one of the IP addresses (among the pool of many) must be chosen as an endpoint identifier. If some fault within the network renders that IP address unreachable (e.g., the fault in the network interface, router, or the physical wire constituting the last hop of the connection) all active connections break which are destined to or originating from that logical entity. Note that transport sessions do not break when routers in the internet fail or come back up, since the routing protocols continue to route packets through alternate routes. Since multihomed hosts do not engage in routing table exchange with first hop routers, the failure mode described here cannot be fixed by routing protocols. The problem here is that even though an alternate routing path exists, the routing system is not able to locate that path and utilize it.

From an Internet routing viewpoint each IP address represents a separate logical host. The fact that all addresses assigned to a multihomed host belong to the same host is not known to the routing system. Therefore, when one interface fails, additional mechanisms are required at the network layer for rerouting packets via another interface. Below, we'll show how this can be achieved within the MobileIP routing framework.

4.3 Switching Across Interfaces

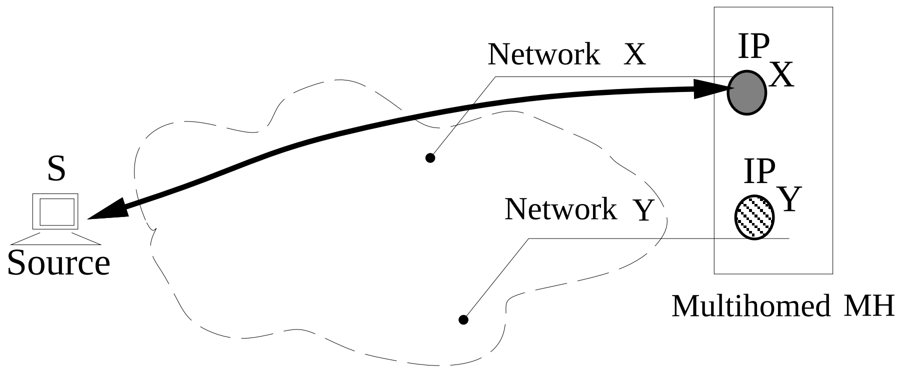


Figure 4.1: Setting TCP connections with a multihomed host

Consider a mobile host (MH) which is equipped with two network interfaces (see Figure 4.1). The interface which connects it to *network x* is assigned an address IP_X. Similarly, the other network interface, IP_Y, connects it to *network y*. To set up a connection with the multihomed host, a source host first sends a query to the Domain Name Server (DNS). The DNS returns a list of all addresses that are allocated to the MH. The source can pick any address from this list to set up transport layer connections with the MH. The interface associated with the chosen address is the used to send and receive packets for that connection. Suppose the source chooses IP_X to set up a connection with the mobile host. If the MH is connected to its home network (as shown in Figure 4.1), then packets will be delivered to the MH by the normal routing process. Otherwise, if the MH is connected via a foreign network, MobileIP will ensure correct delivery of packets.

Our objective is to design a solution that will ensure correct delivery of packets even if the associated interface fails or is disconnected. When the interface X is disconnected, there are two possibilities depending on whether the second network interface is connected to the home network or to a foreign network. We consider each case separately and show how our proposed solution works in both cases.

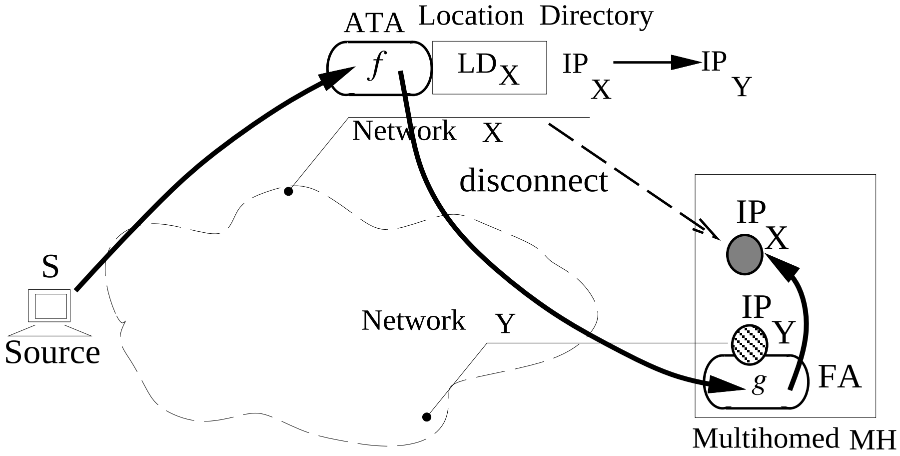


Figure 4.2: Switching within Home Network

4.3.1 Switching within Home Network

Suppose an active TCP session between the source (S) and IP_X is in progress, and, in the middle of this connection, the interface IP_X is disconnected from *network x* (see Figure 4.2). When the MH detects this event[1], it can make the necessary changes in its routing table so that packets originating from the MH are routed through the second interface (IP_Y). These packets must contain IP_X in the source address field, otherwise the TCP connection between S and the MH will break. The normal IP routing process will cause these packets to be delivered to S. The problem, however, is that packets which are addressed to IP_X will never arrive at the MH since they will continue to be routed to *network x*.

In the previous two chapters, we demonstrated how MobileIP enables hosts to migrate from one network to another without disrupting any transport layer connections. The basic idea here is to view interface switching as a special case of host migration. Within the Internet architecture, each IP address denotes a logical host. Failure of a host interface can be interpreted as the migration of the associated logical host from its home network to a foreign network which is denoted by the address of the second interface. Borrowing from MobileIP proposals, where the address of a forwarding agent is used to route packets to a host's current point of attachment, we propose to

[1]Either by an explicit notification from the hardware or by loss of beacons.

use the address of the second host interface in place of the forwarding agent address. Since the second host interface is connected to its home network[2], its address can be used for the purpose of forwarding packets to the MH. However, in order for this scheme to work, the MH must be capable of acting as its own *forwarding agent*.

When the network interface X is disconnected, the MH sends a location update message to the location directory (LD_X) associated with its home, *network x*. This message contains the address of the second interface (IP_Y) through which it can send/receive packets. The LD interprets IP_Y to be the address of the forwarding agent associated with the MH and records the association $IP_X \rightarrow IP_Y$ in its table. When the source (S) sends packets which are addressed to IP_X, they are delivered to *network x* by the normal internet routing process. At this point the home router performs the necessary address translation operation on the packet and relays it to the forwarding agent associated with the MH (in this case, the interface IP_Y). To perform address translation, the home router appends a new header if using encapsulation or inserts a new address in the source route option. Once the packet arrives at the MH, it can strip the extra addressing information and again relay the packet to the network layer protocol processing module for further processing. This amounts to twice the normal network layer protocol processing: once for the processing of the outer IP encapsulation header and again for the inner IP header. If LSR is used instead of IP within IP encapsulation, the overhead is considerably less, since in addition to the normal IP header processing a single LSR option processing is required.

4.3.2 Switching within Foreign Network

Figure 4.3 shows the scenario in which the alternative interface is connected to a foreign network. This happens when the MH sets up a TCP connection to a host (S), moves to a foreign network (Z), and then connects to a local forwarding agent (address = IP_Z) via its second interface. In this case, the MH takes the following two actions:

1. The MH sends a location update message to its home, *network x*, indicating that its new forwarding agent is IP_Y. The LD_X records the association $IP_X \rightarrow IP_Y$ in its table.

2. The MH also sends a location update message to its second home, *network y*, notifying it of the address of the forwarding agent IP_Z through which its second interface is connected to the foreign network. The LD_Y records the association $IP_Y \rightarrow IP_Z$ in its table.

The purpose of the first step is to register the address of the second interface as the forwarding address for all packets destined to IP_X. Once the registration sequence is complete, the MobileIP

[2]The next section covers the case when this is not true

protocol will set up an encapsulation or source routing tunnel to the forwarding agent. What is interesting about this case is that the normal Internet routing is not capable of delivering packets to the end of the tunnel since the tunnel end point (IP_Y) is not attached to its home. Thus, a second level of tunneling is required for carrying packets.

The purpose of the second step is to set up the tunnel to the address of the forwarding agent associated with IP_Y. It ensures that all packets addressed to IP_Y are forwarded to FA_Z. Notice that the two tunneling methods (essentially the MobileIP protocols) used here need not be identical. This modularity is very useful, since it allows two different MobileIP methods to coexist.

Packets that are addressed to IP_X, first arrive at *network x* by normal IP routing. ATA_X (which is normally co-located with the router for *network x*) forwards these packets to IP_Y. From the ATA_X viewpoint, the MH is connected to a forwarding agent whose address is IP_Y. The specific mechanisms used to deliver packets up to IP_Y are immaterial. It could be MobileIP (as shown in the figure), or it could be CDPD[19], or it could even be MDL[34].

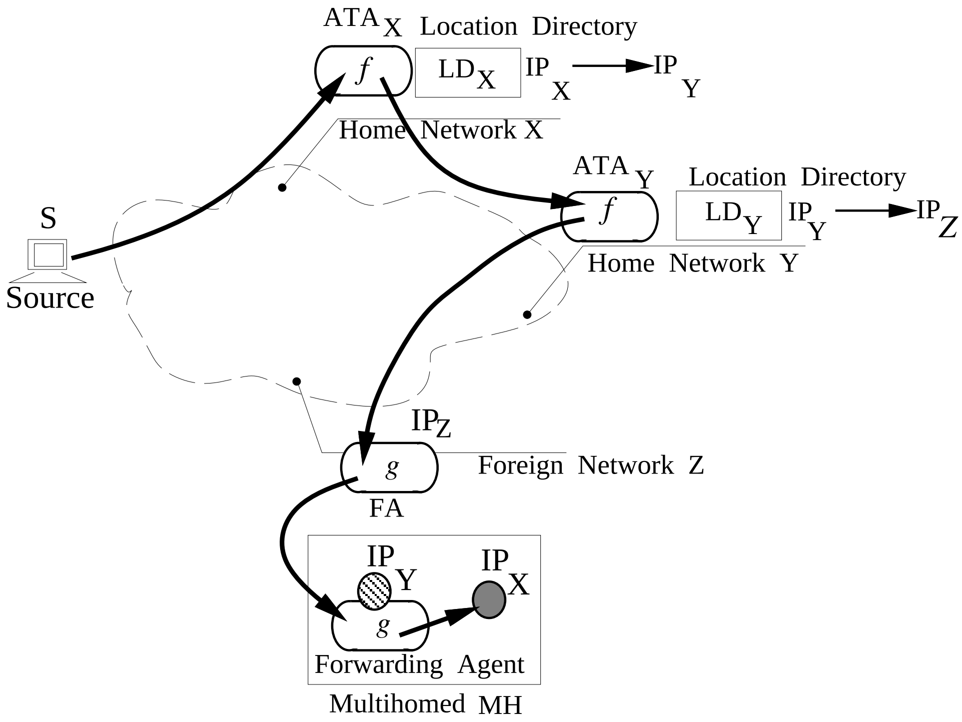


Figure 4.3: Switching within Foreign Network

4.4 Operational Description

To support media switching within a MobileIP system, the only modification required is at the mobile host. From a Mobile Router point of view, it makes no difference whether the mobile host and the forwarding agent are two separate entities or they are co-located. In both cases, the MobileIP protocol tunnels packets to the address of the forwarding agent.

The mobile host performs some additional protocol processing when the two entities are co-located. If Loose Source Routing is used as an address translation mechanism, then co-location is achieved without any software modifications. Although incoming packets undergo two iterations of IP processing, the normal LSR processing is sufficient to carry out both steps.

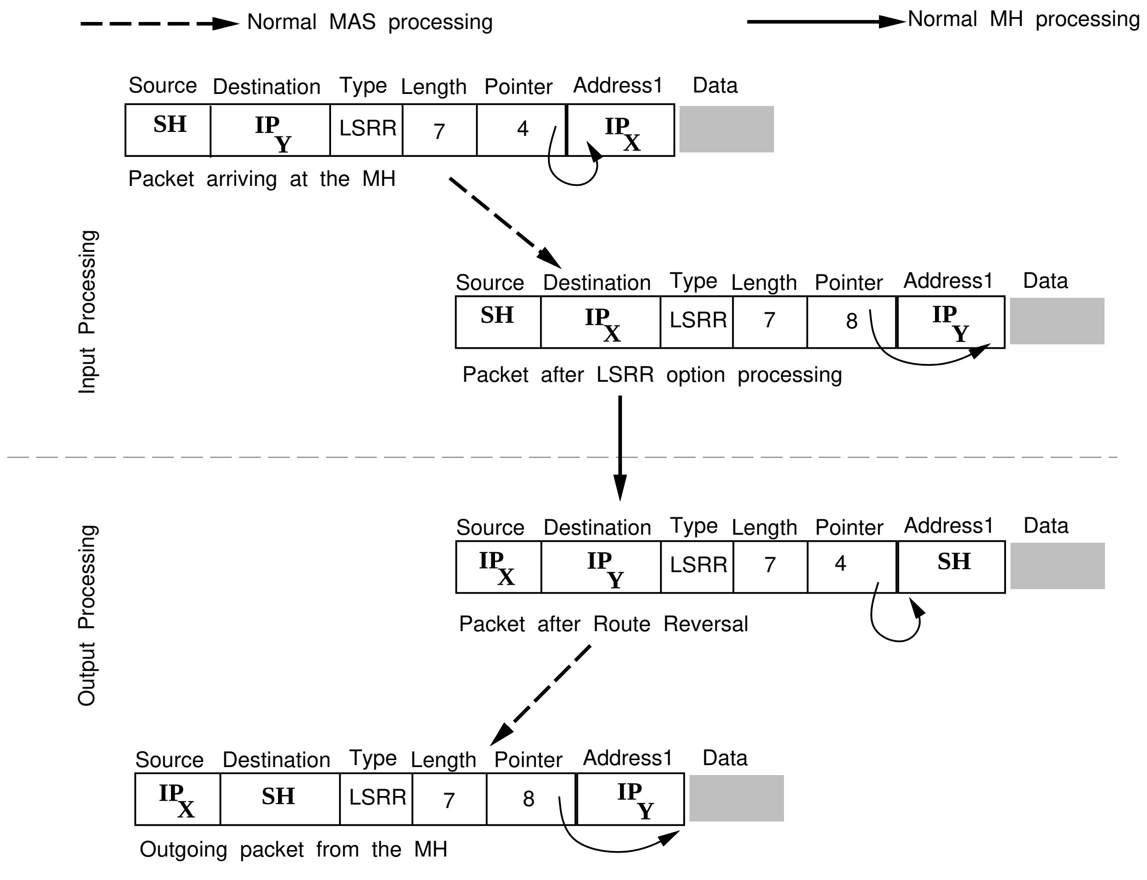


Figure 4.4: Processing at the MH

Figure 4.4 shows various stages of IP header processing at the mobile host. Both input and output are now two-phase processes. In the first phase, processing on behalf of an MAS is carried out. Packets arriving at the mobile host contain the address of the secondary interface (IP_Y) in the destination address field. By the normal LSR option processing, the destination is set to IP_X.

At this point the packet looks as if it just arrived from an MAS (compare this with Figure 3.12). In the second phase, the normal MH processing is carried out. This involves saving the incoming LSR option in the TCP control block and then passing this packet to the TCP layer for further processing.

During output, the same two steps are carried out in the reverse oder. When the packet is received from the TCP layer, the MH inserts the LSR option in the packet. By normal route reversal operation, the destination is set equal to IP_Y (which happens to be address of the second interface). This packet loops back into the IP input processing queue. When the LSR option processing is carried out, the new destination is set equal to the address of the stationary host (SH). At this point, the packet is released in the network.

Naturally, some extra packet processing overhead is incurred during input and output processing. However, the overhead of additional processing is nominal. It does not cause any slowdown in the operation of the system. We verified this by conducting large file transfer experiments between a stationary and a mobile host. We measured ftp transfer times before and after interface switching for a range of file sizes. A summary of measurements is shown in Table 4.1. Our mobile host (gomati.cs.umd.edu) is an IBM RT-PC workstation equipped with an infrared and an ethernet interface. The stationary host (tapti.cs.umd.edu) is an IBM RS6000 workstation. Another IBM RT-PC (notrump.cs.umd.edu) operates as the mobile router. In these experiments, the ftp session was set up between the stationary host and the infrared interface of the mobile host. Before interface switching, packets are relayed to the mobile host via a base station. After the ftp session is switched to the ethernet interface, the mobile host starts performing two-phase protocol processing. Yet, we do not observe any degradation in throughput. On the contrary, throughput improves. This is because the ethernet is 10 times faster than the infrared LAN. To make a fair comparison, we performed another file transfer experiment in which ftp session was set up directly to the ethernet interface of the mobile host. In this case the mobile host carries out normal protocol processing.

File transfer times in column 3 and 4 are roughly the same. The difference is in the range of 100ms - 300 ms. This is because when the interface switching is performed, the first few packets of the ftp session are routed through the mobile router. This introduces a few round trip worth of delays in the ftp session. Once both ends of the session have exchanged one round of packets, the traffic starts to follow the optimal path; i.e., it is not routed via the mobile router. Since the time difference between column 3 and 4 is not proportional to the file size, we can conclude that the protocol processing is not the bottleneck; the performance of the system is limited by the link speed.

File Size (Mbytes)	File Transfer Time (sec.)			Throughput (Kbytes/sec.)		
	Before Switching (infrared)	After Switching (ethernet)	Direct ethernet	Before Switching (infrared)	After Switching (ethernet)	Direct ethernet
1.63	75.74	14.01	13.72	21.52	114.1	116.5
0.8	34.30	6.63	6.476	23.32	119.6	122.5
0.1	4.31	0.92	0.81	23.20	108.6	123.45

Table 4.1: Performance Comparison of Switching Overhead

4.5 Connection level Rerouting

The mechanisms proposed here, in effect, cause all packets addressed to one specific network interface to be re-routed via another interface. Since switching is performed at the network layer, all sessions are moved at the same time. The system can be made more flexible by adding control mechanisms that allow switching to be performed on a session by session basis. This is easily achieved by modifying the `mh2mr` protocol. The basic idea is to include the TCP/UDP port number in the `mh2mr` message. When a mobile host decides to switch a specific session to another network interface, it sends an `mh2mr` message to the mobile router. The `mh2mr` message includes the session port number and the address of the new network interface. The mobile router records this information it its routing table. The mobile router is now required to maintain two types of routing table entries: normal and special. Special entries are those which record a port number in addition to the destination address. When a `mh2mr` message containing a port number is sent to the mobile router, a special type routing entry is created. If the message does not contain a port number, the information carried in the `mh2mr` packet is used to install a normal routing table entry.

When a packet destined to the mobile host arrives at the mobile router, the mobile router also inspects the port number carried in the packet. If a match occurs on both the destination address and the port number, the special type entry is used to forward the packet. Otherwise, normal routing procedures are followed. Thus, only packets belonging to a specific session are routed via the second interface. Other packets continue to arrive via the first interface. The process of LSR route reversal ensures that outgoing packets are routed via the correct network interface.

4.6 Application Areas

4.6.1 Mobility Across Communication Medium

The proposed solution allows mobile clients to seamlessly move from one communication medium to another without disrupting any active network connections. This flexibility is very valuable, given the variety of wireless solutions that are being developed by different vendors. These products operate in different frequency bands, use widely different sets of MAC/link layer protocols, have differences with respect to link speed and geographic coverage area (indoor vs outdoor). Given these constrains, none of the emerging wireless solutions is likely to provide ubiquitous coverage. Universal coverage would only be possible by interconnecting different types of wireless segments and enabling roaming across multiple medium through flexible internetworking protocols.

4.6.2 Tolerance against Network Failures

Enabling mobility across different communication medium also lends an easy and elegant solution to the network fault-tolerance problem. The traditional approach to network-fault tolerance relies on sending multiple copies of messages on different networks to protect applications against network failures[33]. In contrast, our approach relies on re-routing network traffic through an alternate network interface when network failures occur. Since re-routing is done at the network layer, no application modifications are required. Moreover, this approach to network-fault tolerance is independent of any specific hardware technology.

4.7 Summary

It is expected that the form of network connection will vary depending on the environment where mobile computing devices are used. For example, when disconnected they will use RF or infrared links; when docked they will use physical wires; and when used in an outdoor setting they will use digital wireless cellular links (such as CDPD [19]). A solution is required that will enable smooth operation and transition across different communication medium.

In this chapter, we showed that switching across communication medium can be viewed as a special case of host migration. The MobileIP architecture provides an elegant solution to this problem by co-locating the forwarding agent with the mobile host. We described how this is accomplished in our MobileIP implementation. We also demonstrated that the overhead of performing extra protocol processing was negligible.

While the original motivation for developing this solution was to support seamless mobility across different communication medium, the proposed scheme has wide applications in the area of fault-tolerance. Stationary hosts can be protected against interface and network failures by using the same mechanisms that make switching network interfaces totally transparent to the running applications on mobile hosts.

Chapter 5

Wireless Access and Session Performance

In chapters 2, 3 and 4, we described the *functional* aspect the mobile internetworking problem. In this chapter, we shift our focus to the *performance* issue. MobileIP only provides a mechanism for packet exchange between mobile and stationary end-systems. Performance of these data exchange sessions depends crucially on the error characteristics of the wireless medium and the Medium Access Control (MAC) protocol employed at the wireless link layer. The environment we consider is that of a wireless LAN being used to gain access to a wired backbone network (see Figure 5.1). A base station relays packets between wired and wireless links, and all mobile users gain access to the backbone network via a base station.

Hosts on the existing wireless LANs (WLANs) use applications and end-to-end transport protocols that were originally developed for wired networks, where the underlying physical links are fairly reliable and packet losses are random in nature. However, due to radio propagation characteristics, wireless links are significantly error prone. Within the ISM[1] band, due to multi-path fading, the error characteristics of wireless channels are bursty and time varying. In addition, user mobility and frequent handoffs cause burst packet losses. Current generation wireless LANs operate within the ISM radio spectrum, with typical link rates within a few Mbps at a typical range of 50-100 meters. Under these conditions, the performance of transport protocols such as the Transmission Control Protocol (TCP) degrades significantly.

A method for fast recovery from losses during handoff has been proposed in [15]. It exploits TCP's fast retransmit option to reduce TCP's loss detection latency. This method, however, cannot be effectively used to recover from burst errors due to fading. In [5, 55] another scheme is proposed that splits a TCP connection into two separate connections; one between the mobile host and the base station and the other between the base station and the stationary host. It attempts to isolate the dynamics of TCP's congestion control from the interference of losses on the wireless

[1] Industrial, Scientific, Medical band (902-928MHz, 2400-2483.5 MHz, and 5725-5850MHz) approved by FCC for unlicensed use.

channel. The solution proposed in [2] achieves the same objective with less bookkeeping and state maintenance at the base station. Unfortunately, these techniques do not perform well when losses on the wireless channel are bursty.

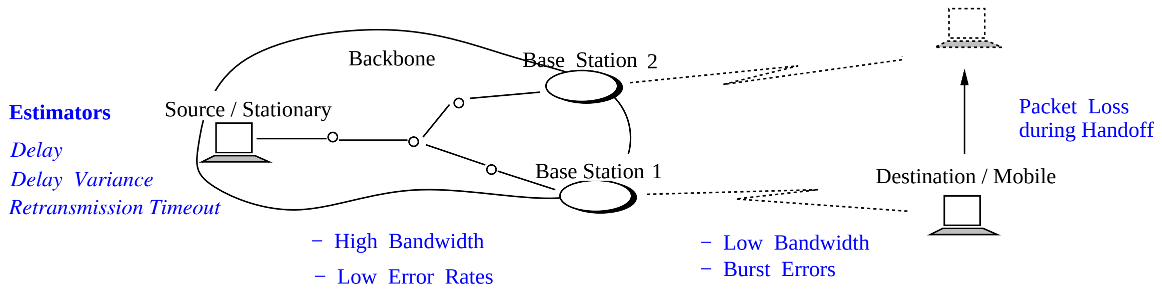


Figure 5.1: Intergration of Wired and Wireless Medium

In this chapter we investigate the design space of link layer mechanisms for recovering from burst errors. We begin by carefully studying the effect of burst packet losses on the end-to-end throughput of a TCP connection. We also study the effect of adding link level error recovery on TCP's performance. We have observed that due to the coarse-grained timer (of granularity 500 msec) used in TCP implementations loss detection latencies are very high. This latency allows sufficient time at the link layer to recover from burst errors. We investigate the performance of multiple TCP connections over a wireless LAN which employs a link level error recovery protocol. This is a likely scenario, in which mobile users gain access to the backbone network via one base station. The channel is subject to burst errors independently for each connection. We observe that under FIFO packet scheduling the head of the line packet, if encountering burst losses, can block the transmission of other packets. In addition to reducing the overall throughput, this phenomenon causes unfairness, as a particular receiver in a burst error state can reduce the throughput of all other receivers.

We propose link level channel state dependent packet (CSDP) scheduling methods to overcome the problems associated with multiple sessions sharing the wireless link. We show that appropriate packet transmission scheduling can increase overall throughput and reduce the unfairness problem, without interfering with TCP's retransmission schemes. We use TCP as an example protocol to motivate the problem and illustrate the effectiveness of our solution approach. CSDP scheduling methods can be implemented at the base station's wireless device driver. Since it is a link layer solution, not only TCP, but all UDP based applications such as `rpc` [29] and `NFS` [52] also benefit from it. Another advantage is that a CSDP scheduler can be used in conjuction with the higher layer techniques for improving performance over the wireless channels [15, 5, 55]. Thus, it is possible to combine the benefits of both approaches.

This chapter is organized as follows. First, we briefly describe packet loss characteristics commonly observed in indoor wireless RF LANs. In section 5.2, we present some TCP traces collected over a wireless testbed to demonstrate TCP's reaction to burst packet losses. In section 5.3, we illustrate the head of the line blocking effect that is observed when error-recovery is employed at the wireless link layer. In section 5.4, we describe the CSDP approach and show, through simulations, the interaction of First in First out (FIFO), Round Robin (RR), Earliest Timestamp First (ETF), and Largest Queue First (LQF) schedulers with the dynamics of multiple TCP sessions. We conclude by presenting a summary of our results and observations.

5.1 Wireless LAN Channel Characteristics

In the following, we briefly discuss various factors that give rise to burst losses on the wireless medium and show how these errors degrade throughput and channel utilization.

Error characteristics of the multi-access wireless channel differ significantly from that of the wired medium. Packet losses on the wired medium are very rare and random in nature. In contrast, the errors on the wireless medium are bursty and the wireless channel is distinct and time-varying for each mobile user. The distinction between the wired and wireless channels arises for many reasons. As users move, the received signal strength varies significantly. In addition, there are effects due to fading, interference from other users, and shadowing from objects, all of which degrade the channel performance [25]. Frequency-hopping WLANs are further subject to a unique error phenomenon. The WLAN may hop onto a frequency channel which is particularly susceptible to interference (e.g., a subset of the mobile users may be close to some other entity transmitting at that frequency). This phenomenon leads to bursty errors, since the effect is likely to disappear with the next change of frequency channel. Digital Cellular channels are also subject to the same problems for the same reasons. However, WLAN channels are more susceptible to losses because of the higher bit rates and lack of sophisticated signal processing techniques such as adaptive power control, forward error correction, and bit interleaving. Measurements of a particular WLAN system [20] show that packet-error rates critically depend on the distance between the transmitter and receiver, and surprisingly, are not monotonically increasing with this distance. Thus, the channel varies with each user, depending on their location with respect to the base station.

5.1.1 Channel Loss Model

Many earlier studies in the literature [45, 53] have established that finite state Markovian models can be effectively used to characterize bit error patterns observed on RF channels. Figure 5.2 shows an example of the bit sequence for a burst error channel. A guard section is defined as an error free section and a burst section is defined as the section sandwiched between guard sections. The

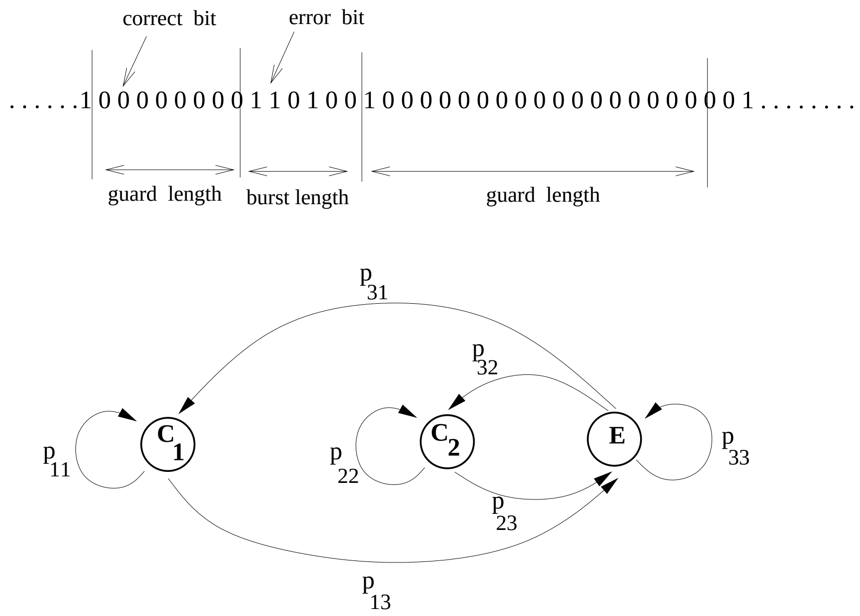


Figure 5.2: Three-state Markov Model: Transition Diagram

guard is specified as an error-free section longer than a fixed burst limit. In [43] authors have shown that a 3-state Markovian model provides a good characterization of the burst error channel. Their model consists of 3 states C_1, C_2, and E. C_1 shows a guard section while C_2 and E show a burst section. C_1 and C_2 show correct states, and E shows the error state. p_{qr} denotes the probability of transition from state q to state r. The wireless channel probabilistically switches between the three states.

Note that for most systems, a single bit error on the channel translates into the loss of a complete packet (see Figure 5.3 in which 0 indicates good reception and 1 indicates an error at the bit level). In practice, we have observed that mean residency time in states $C2$ and E is longer than a single packet transmission time. This results in correlated packet losses, i.e., a single packet loss would be followed by many back-to-back packet losses. We use this observation to capture *packet loss characteristics* by a 2-state Markov model. This model conforms with the observation reported in some of the earlier empirical studies of packet loss behavior on wireless LANs [20]. At any point in time, we model the channel as being in one of the two possible states, G (good) or B (bad). We use the term, p^S to denote the packet loss probability in state S, where $S \in \{G, B\}$. We assume that $p^B \gg p^G$. Roughly speaking, a packet transmission is usually successful provided the channel stays in the good state during the packet transmission duration. We assume that the time spent in the Good and Bad periods are exponentially distributed, but with different average values. Note that our objective here is not to provide an accurate characterization of the channel. Rather, it is to illustrate the behavior of transport sessions when packets are subject to burst losses. An approximate characterization of the wireless channel is sufficient to illustrate these effects.

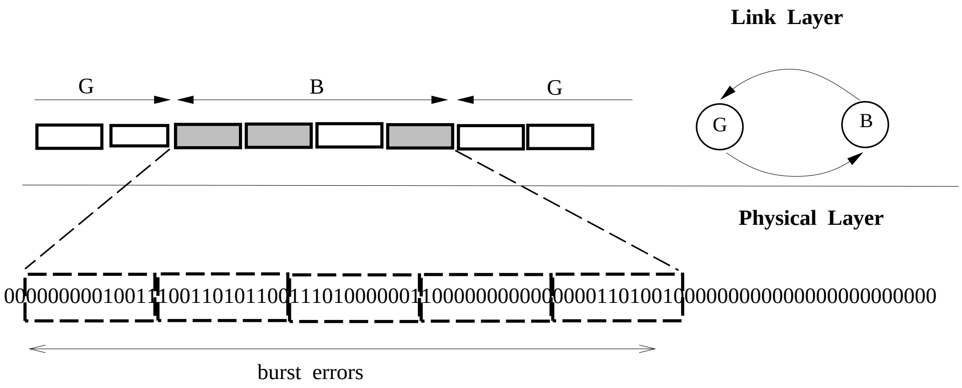


Figure 5.3: Error Characteristics

5.2 TCP's reaction to Packet Losses

Figure 5.4 shows the trace of a single TCP connection whose packets were subject to burst losses. It shows a plot of sequence number versus time for an ftp connection between an IBM RS/6000 (tapti.cs.umd.edu) and an IBM RT-PC (narmada130.cs.umd.edu). The RS/6000 is connected to an ethernet while the RT-PC is connected to a 1Mb/s infrared wireless LAN. The ethernet and the wireless LAN are connected via another RT-PC machine notrump.cs.umd.edu. While ftp was in progress packets on the wireless segments were dropped using a 2-state loss model described in the previous section. The controlled dropping of packets was accomplished by modifying the wireless device driver at the base station. The trace was collected using tcpdump [31] running on a DEC alpha which is connected to the ethernet.

In the trace collected above, round trip time values are in the 20ms range, the receiver window size is 4K, and the maximum segment size is 1466 bytes. The trace corresponds to a 600K file transfer which, under a no-loss situation, completes in approximately 10.5 seconds. During the course of the experiment the wireless channel switched into burst mode only 3 times (at $t = 4.3sec, t = 5.8sec$ and $t = 12.8sec$). Yet observed loss in throughput is about 40% (see Table 5.1). Below, we enumerate some of the problems that we observed during our experiments.

Loss Detection Latency: Following the first burst error period, which occurs at $t = 4.3sec$, TCP waits approximately 1.48 seconds before it starts retransmission. Similarly, after the 3^{rd} burst error period, which occurs at $t = 12.8$ seconds, TCP takes 1.01 seconds to resume the *slow start* [24] phase. In [15], similar phenomenon was pointed out in the context of losses during handoff. A microscopic view of the ftp trace reveals that the time spent in the slow start phase is

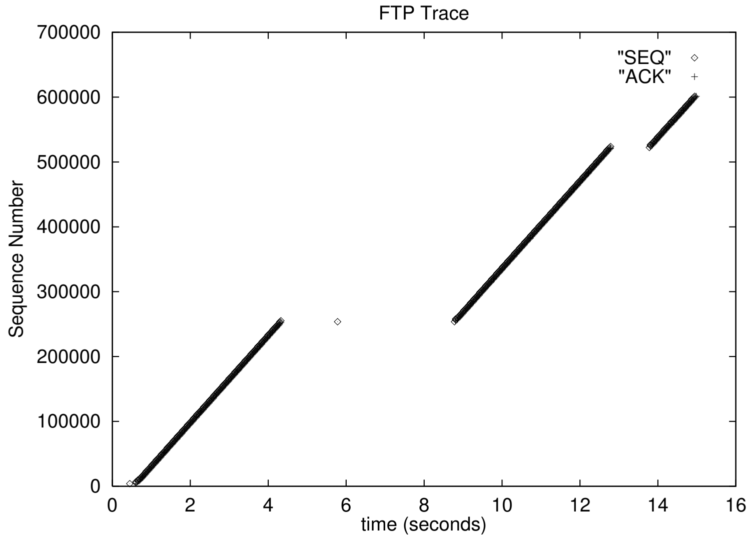


Figure 5.4: An ftp connection subject to burst losses

	No Losses	Burst Losses
Throughput	0.508 Mb/s	0.321 Mb/s

Table 5.1: Effect of burst errors on TCP throughput

negligible compared to the latency of loss detection. This is because it only takes approximately RTT*$O(log_2(W/mss^2))$ time to restore the window size to its original value [28]. Over local area wireless connections, where round trip times are short and the window is small, this period is negligible compared to the latency of loss detection.

Failure of the Fast Retransmit Feature: The fast retransmit feature was added in TCP to reduce the latency of loss detection. This feature is triggered whenever a TCP source receives a threshold number (usually 3) of back to back duplicate acknowledgements. Arrival of triplicate ACKs is a good indication that a packet loss has occured, and there is no need to wait for a timeout before retransmitting. The fast retransmit feature fails on the wireless medium because of two reasons. First, due to the low bit rate available on wireless channels, TCP window size is often chosen to be small (some times even less than 3 packets). Second, back to back packets and ACKs are both likely to be lost during fade periods implying that fast recovery will fail to trigger. Moreover in the case of interactive applications, a TCP source may not even have additional packets ready for transmission which would force the receiver to generate ACKs.

[2] W is the window size and mss is the segment size.

Exponential Retransmit Back-off: If retransmission occurs during a period when the channel is in a burst error state, the retransmit timeout value is doubled by Karn's [27] exponential retransmit back-off algorithm. For example, in Figure 5.4, the packet retransmitted at $t = 5.8sec$ falls in the burst error zone. Since the ACK for this packet is not received, TCP doubles its retransmit timeout period to 3 seconds and retries only after it expires. During this period the channel is not utilized.

Delayed Connection Setup & Denial of Service: At the connection setup time, TCP starts with an initial RTT estimate of 6 seconds (for lack of any better estimate). If the SYN packet is lost, the connection setup is delayed by 6 seconds. Loss of the 2^{nd} SYN packet further delays connection setup by an additional 24 seconds. The connection setup fails if the 3^{rd} SYN packet is also lost.

The main reason behind these problems is that TCP was designed around the assumption that links are reliable. It interprets packet losses as signs of network congestion and consequently reacts by invoking congestion control mechanisms [24]. This is justified if underlying links are reliable, in which case the reason behind the packet loss is likely to be congestion and the only remedy is a reduction of input rate. However, triggering a congestion avoidance mechanism in response to each packet loss in wireless media may lead to a unacceptable level of performance. In fact, if the wireless link is error prone then TCP's slow start mechanism may repeatedly get triggered, resulting in a a scenario where TCP's window size stays at the minimum possible value.

The problem is due to the inability of current mechanisms to distinguish between losses due to congestion and those due to burst errors in the wireless link. Better control mechanisms can be designed if some form of feedback is available from the wireless hardware about the quality of wireless link. Unfortunately, current generation wireless adapters do not have any provision to provide this information. Even if provisions are made, it would only solve one end of the problem since the other end of the TCP connection which resides on the wired segment would never be able to receive this feedback.

Another plausible solution to this problem is to use a fully reliable link layer protocol over the wireless segment. A reliable link layer would substantially reduce the effective packet loss rate seen by the transport layer. This approach, too, as will be shown in the next chapter, does not warrant good performance. The primary reason being the complex interaction of the link layer retransmission mechanisms with TCP's round trip time estimator.

5.3 Loss Recovery at the Link Layer

A natural solution for the problems discussed in the last section lies in adding error recovery at the wireless link layer[3]. Although transport layer protocols are capable of recovering from packet losses on the wireless medium, as pointed out in the previous section, the latency of timeout based loss detection mechanisms is very high in the transport layer. Timeouts used in the link layer protocols, on the other hand, use a much smaller timeout value and, therefore, can recover faster from lost segments.

5.3.1 Adding Reliability at the Link Layer

Due to the high packet loss rate anticipated on the wireless medium, a need for explicit MAC layer acknowledgement for each data packet has been widely recognized within the IEEE 802.11 subcommittee. One of the recommendations [18] is to use CSMA/CA + priority ACK. The CSMA/CA part of the protocol is designed to reduce the collision probability between multiple stations accessing a medium. The priority ACK part of the protocol supports recovery from lost frames. To allow detection of a lost frame (due to collision, fading, or interference) an ACK is returned by the destination station immediately following a successful reception. ACK packets use a smaller carrier sense interval which gives them priority over access to the medium by all other stations which are waiting for the medium to become available. The ACK is transmitted by the receiving station only when the CRC of the received frame is found correct. If an ACK is not received immediately following a packet transmission, the source quickly times out and retransmits the lost frame after a random Retransmission-Backoff. Retransmission is either attempted by the wireless LAN card (if a retry function is supported in the card hardware) or by the link layer protocol (which is implemented as part of the device driver). The packet is eventually dropped if R_{max} successive frame transmission attempts fail.

5.3.2 FIFO Dispatching

The current generation device drivers of WLAN cards maintain a FIFO queue of ready-to-transmit packets. In the transmit phase, the driver picks up the packet at the head of queue, copies it into the WLAN adapter's on-board memory or sets up a DMA channel, and issues a transmit command

[3] In [17], the authors point out that error recovery employed at the link layer could potentially interfere with TCP's timeout computation mechanism. This would be true if TCP used a very accurate clock to measure round trip times. In practice, however, TCP implementations use a very coarse timer (granularity of 500ms) to sample round trip time (RTT) values. The coarse grain timer makes TCP's adaptive RTT estimator insensitive to small fluctuations in RTT samples.

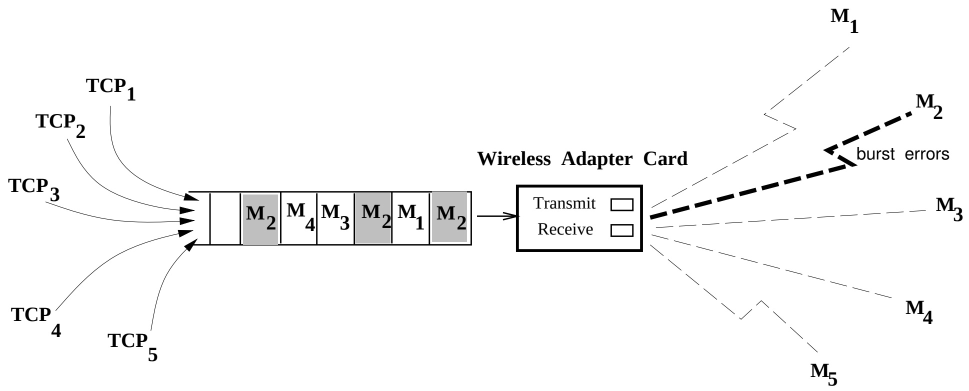


Figure 5.5: HOL blocking in FIFO scheduler

to the card controller. The WLAN card issues a transmit complete interrupt to the CPU when transmission of this packet is complete. At this point the device driver can pick the next packet at the head of the queue and prepare it for next transmission. The FIFO packet dispatching sequence works fine so long as the packet losses on the wireless medium are random. When losses on the wireless medium are bursty, this approach leads to many performance problems. We have observed that in RF wireless LANs typical burst periods are 50 to 100 ms long. During this period, all packet transmission attempts (to a specific destination) fail with a very high probability. The result is that receivers which are located in fade zones claim more wireless resources in comparison to receivers located in good zones. Moreover, packets which are lost during burst periods are interpreted by end-to-end transport layers as signs of congestion [15]. Transport layer mechanisms react to these losses by throttling their transmission rates causing poor throughput. We conducted a simulation study to perform a careful investigation of this interaction and to develop solutions to alleviate this problem. We chose TCP as an example of an end-to-end transport protocol due to its widespread deployment within the Internet. The observations made below are not specific to TCP, rather they apply to any end-to-end transport protocol, such as those employed in DEC-BIT, RPC etc.

We study a simple network topology consisting of a single base station providing coverage to N mobile hosts. There are N TCP connections, each starting from a stationary host and terminating at a mobile host. The wireless link between the base station and mobile hosts has a bandwidth of 1Mb/s. This bandwidth is shared among all transmitters by the CSMA/CA + priority ACK protocol that has been implemented as a separate component in the simulator. The transmission line between the source and the base station has a bandwidth of 10Mb/s and a propagation delay of τ.

Each TCP session involves a 0.8Mbyte file transfer. The propagation delay τ is set equal to 16ms which is representative of the delay in typical LAN environments. We expect the wireless base station to be the bottleneck for each TCP session. This is due to the fact that the wireless

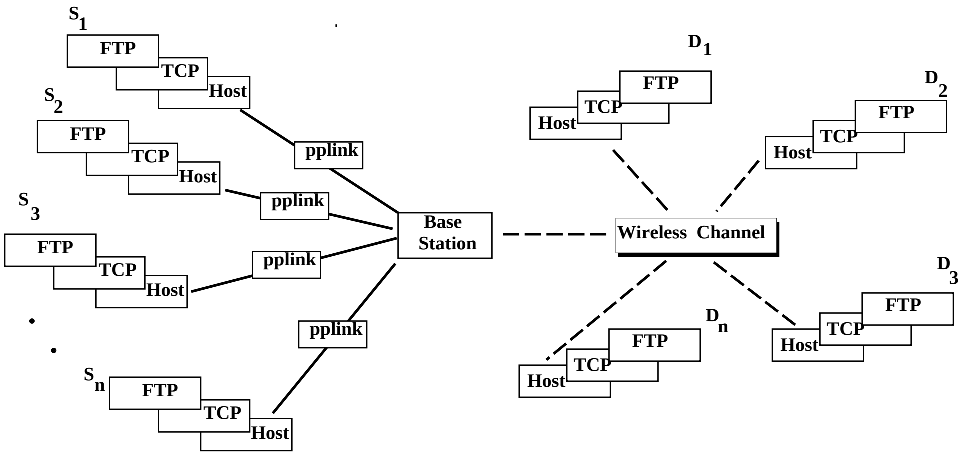


Figure 5.6: Simulation setup

link is at least an order of magnitude slower than the speed of the wired segment. We assume sufficient amount of buffering at the base station, implying that buffer overflow is not a problem. This assumption is justified by two reasons. First, due to low bandwidth of the wireless medium we expect TCP sessions to operate with small window sizes. Secondly, the user population within each wireless segment is expected to be small.

Figure 5.7 shows a comparison of file transfer times for 10 TCP connections when FIFO and CSDP-RR schedulers (see section 5.4) are used at the base station. For these experiments, mean burst length was set to 100ms, mean good period was set to 1 sec, and R_{max} was set to 8. The following observations became clear from these experiments:

1. **Blocking Effect:** Head-of-the-Line (HOL) blocking is observed because the wireless LANs constitute a single wireless channel which is shared on-demand among multiple receivers. When the head of the line packet is destined to a receiver which is located in a fade zone, it prevents other packets in the FIFO queue from being transmitted. Thus, if the channel to a specific receiver is in the burst error state, all receivers in that cell suffer throughput degradation.

2. **Poor Utilization of the wireless medium:** When the channel to a specific destination is in a burst error state, the CSMA/CA + priority MAC protocol repeatedly attempts to transmit a packet to its destination. During the burst error period, all transmission attempts fail with a very high probability causing significant degradation in channel utilization.

3. **RTT growth:** A direct consequence of HOL blocking effect is seen on the end-to-end level, since the round trip time estimates of all the connections passing through the base station increase. This further increases the timeout values computed by the source TCP. HOL blocking

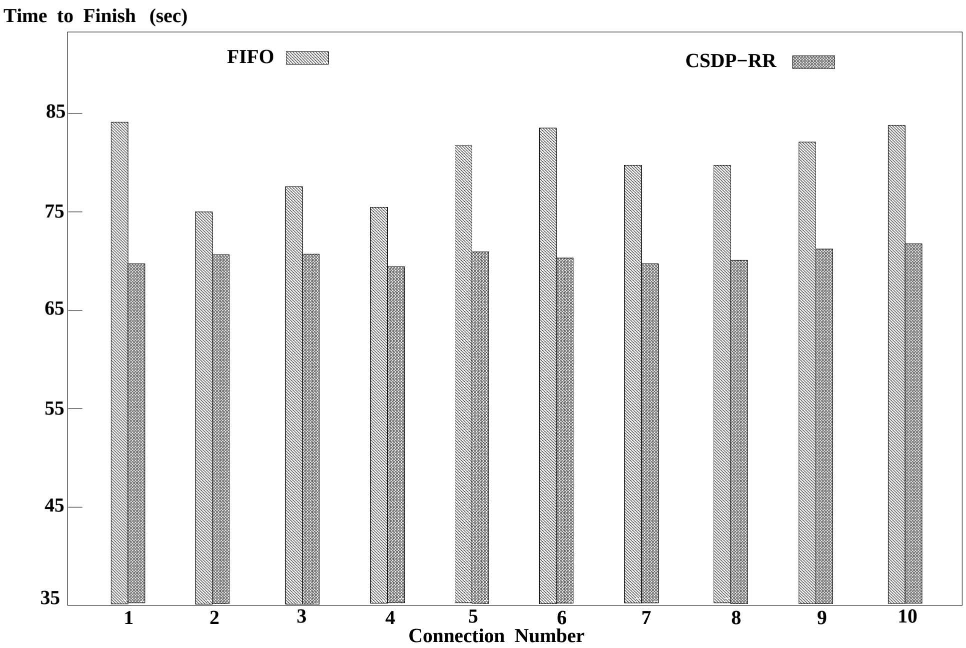


Figure 5.7: Comparison of FIFO and CSDP-RR, burst=100ms

also increases the latency observed by interactive applications.

4. **Fairness Problem:** Over a large time interval, the FIFO scheduler allocates access to the output link in proportion to the amount of traffic input by various sources. TCP's window flow control is a mechanism for bounding the amount of traffic queued up at the bottleneck node at any point of time. When all connections use identical window sizes, a FIFO scheduler provides fair access to the output link. This property no longer holds when errors are introduced on the output link. Since lost packets are transmitted multiple times, the share of the wireless channel received by different packet streams is a function of the channel loss characteristics. Under these conditions, FIFO scheduling fails to enforce any reasonable interpretation of fairness. See the completion time of various connections in Figure 5.7.

In summary, FIFO scheduling at the link layer yields poor end-to-end throughput. Interaction of various scheduling policies with the end-to-end flow/congestion control has been analyzed before in the literature [44]. However, these studies do not consider packet losses on the output link. One of the important design considerations is to ensure that all sessions receive a fair share of the wireless bandwidth. When the medium is error free, a fair MAC protocol is sufficient to ensure this property. However, in the presence of burst errors, additional mechanisms are required. In the next section, we propose a class of packet scheduling policies which can be deployed at the base station wireless interface to increase the utilization of the wireless channel. These schedulers

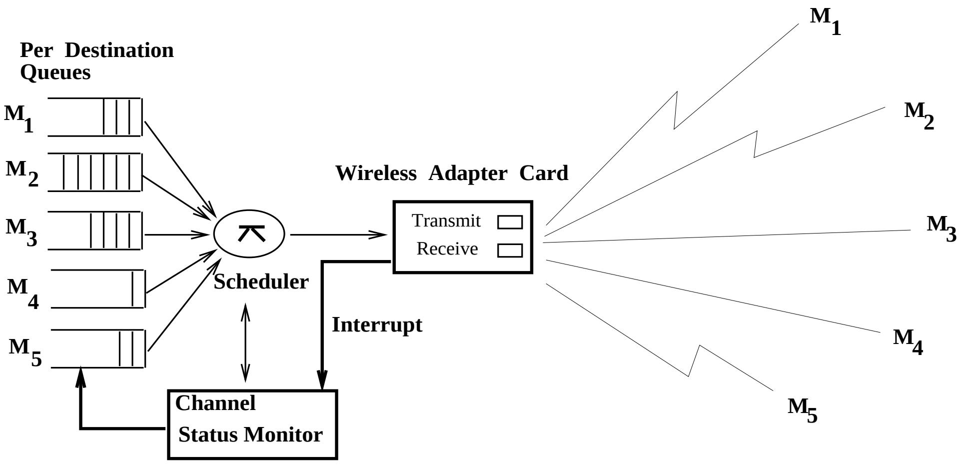


Figure 5.8: Channel State Dependent Packet Dispatching

provide fairness while effectively shielding the end-to-end transport sessions from the effects of burst errors on the wireless medium.

5.4 Channel State Dependent Packet Scheduling

The primary cause of inefficiency is that the CSMA/CA + priority ACK protocol makes repeated attempts to transmit a packet even when the channel is in a burst error state. Moreover, the FIFO packet dispatcher at the WLAN device driver level continues to dispatch packets without taking channel characteristic into consideration. [4] By the time TCP detects packet loss and starts to throttle its transmission, the fade period is usually over. To alleviate this inefficiency, a mechanism is required to defer scheduled transmissions until the start of the next good period. Since the burst periods of different channels are mutually independent, packets for other destinations can be successfully transmitted during this interval.

It may seem that there is a potential risk associated with this approach. If the deferred period length is more than the timeout period of the transport protocol, the source will timeout and retransmit a copy of the delayed packet, thereby unnecessarily increasing the load on the system. In practice, however, we have observed that the timeout period of TCP[5] is on the order of seconds, while the duration of burst periods is on the order of milliseconds. This time difference is sufficient for link layer mechanisms to attempt loss recovery by retransmission over the wireless link.

[4]Prior studies [44] of TCP dynamics have shown that packets from a TCP session have a tendency to arrive in groups. This behavior, in conjunction with FIFO packet dispatcher, makes TCP sessions more susceptible to back-to-back packet losses.

[5]Minimum timeout value is often 1 second, and it grows exponentially for each failed retransmission attempt

5.4.1 CSDP Scheduling Mechanism

Our solution involves modifying the packet dispatching function at the base station WLAN device driver layer. A scheduler which can take the characteristics of each wireless channel into consideration in making packet dispatching decisions would improve wireless link utilization. We refer to the class of such schedulers as Channel State Dependent Packet (CSDP) schedulers. A CSDP scheduler has three components:

- a set of per destination queues
- a link status monitor
- a packet dispatcher.

For each mobile host in the coverage area, the CSDP scheduler at the base station maintains a separate queue of packets. Within each queue, packets are served in FIFO order. However, service across per destination queues depends on the specific policy employed within the packet dispatcher. The link status monitor (LSM) is responsible for monitoring the state of channels between the base station and each mobile station. Whenever the channel between the base b and the mobile station i is in burst error mode, LSM marks the queue for destination i, Q_i. The channel is assumed to be in burst error state whenever the MAC layer ACK is not received despite multiple transmission attempts on that channel. Q_i is unmarked after an estimated burst period length[6]. It is possible to design more robust mechanisms for detecting burst error periods if the physical layer can inform the MAC layer about packet reception status. For example, since the physical layer manages CRC, it should be able to inform the MAC layer about CRC failures. In addition, the physical layer should be able to inform the MAC layer if a packet is received out of range (for example if power reception is below a certain threshold). This notification will help the MAC layer to make a distinction between errors due to collision and errors due to signal attenuation. It is important to note that this information is only available at the receiver end. However, this information can be piggy-backed with ACKs to notify the transmitter as well.

5.4.2 Scheduler Operation

A CSDP scheduler operates by choosing a packet (at the head of the line) from one of the unmarked queues. If all unmarked queues are empty, then it picks up a packet from one of the marked queues. Immediately following the packet transmission, if an ACK is not received within a short timeout period, CSDP returns the packet to the head of the queue Q_i, increments a counter, R_i, which

[6] If burst periods are exponentially distributed, one possible estimator could be $mean + 2 * variance$

records the number of packet transmission attempts, and selects another packet according to the scheduling policy. If R_i exceeds R_{max}, the packet is dropped (i.e., it is not put back at the head of Q_i) and the counter R_i is reset to 0. Notice that CSDP mechanism avoids retransmitting the lost packet immediately following an errored transmission. Thus, in the presence of burst errors, HOL blocking is significantly reduced. Overall, this mechanism yields better wireless link utilization (as results in the next section show) at a marginal cost of software complexity. The pseudocode of the CSDP scheduler is shown in 5.13. The notations are described in Table 5.4.

5.4.3 CSDP Implementation Complexity

A CSDP device driver is required to manage per destination queues, which are accessed and updated whenever either packets are received from the network layer or an interrupt is issued by a WLAN card (see Figure 5.9). At any point in time, the number of queues is equal to the number of active [7] stations within a single base station coverage area. In typical WLAN environments, this number is expected to be on the order of 10 to 15, implying that the overhead of managing per destination queues is negligible. Compared to traditional device drivers, a CSDP device driver requires more data copying operations. This is because following each packet loss event a CSDP device driver dispatches a new packet. This requires a single `bcopy` operation since the contents of the transmit buffer can be overwritten. Table 5.2 shows the memory to memory copy latency on a 33Mhz, 486DX processor based machine. Since memory to memory copy latency is less than the carrier sense interval (which is on the order of $50\mu s$) it is possible to operate the CSDP device driver without compromising throughput.

packet Size	128	256	512	1024
Memory Copy Latency (μs)	5.6	8.4	14.2	26.9

Table 5.2: Memory to memory copy latency

5.4.4 CSDP Scheduling Policies

The CSDP scheduling framework lends an easy implementation of a variety of *work conserving* scheduling policies. Since a separate queue is maintained for each destination, it is possible to enforce QoS and fairness constraints on a per-host basis by suitable choice of a scheduling policy (π). It is important that the scheduling algorithm be simple so that the required processing can be carried out in real-time. Our results indicate that complex scheduling functions do not always

[7] By active we mean those stations which have active, on-going data transfer sessions

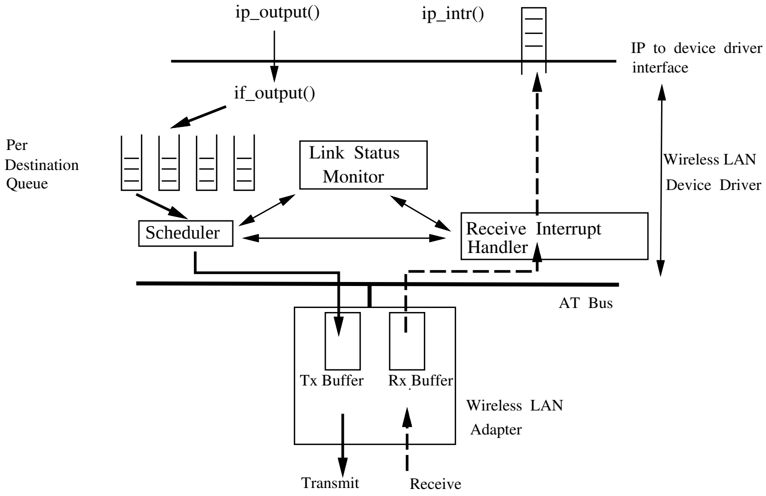


Figure 5.9: Implementation of per destination Queuing

yield good results and, interestingly, as will be shown in the results section, some of the seemingly good techniques exhibit anomalous behavior.

We experimented with CSDP versions of three well known scheduling methods: Round Robin (RR), Earliest Timestamp First (ETF), and Longest Queue First (LQF).

CSDP-RR: Set of unmarked and marked queues are maintained as two separate circular lists. Each set is served in a round robin fashion. Packets are first chosen from the unmarked set. If this set is empty, the scheduler serves packets from the marked set. If packet transmission fails, the retry counter is incremented, and the corresponding queue is marked. Retransmission is attempted when this queue is chosen next time by the scheduler. Notice that a Pure-RR scheduler will continue to retransmit the HOL packet until it succeeds or the max retry counter expires.

CSDP-ETF Packets arriving from the wired side are first timestamped, and then added to the respective queues. From the unmarked set, if it is non-empty, the packet with the earliest timestamp is chosen for service. Otherwise, the packet with the earliest timestamp from the marked set of queues is selected. This policy is very similar to FIFO, except that it tries to avoid transmitting packets to destinations that are known to be in errored state. FIFO can be viewed as a special case of CSDP-ETF when no feedback from the LSM is available.

CSDP-LQF CSDP-LQF scheduler always tries to minimize maximum backlog by always chosing packets from the longest queue. In [46] it is proved that if errors on the wireless channels are statistically identical and packet arrivals follow a poisson distribution then LQF policy will provide fairness and maximize channel utilization [46].

5.5 Numerical Results

The mutual interaction between TCP's flow control and the link layer dispatching mechanism in quite complex. Since the behavior of TCP sources cannot be captured by any closed form analytical expression, it is not possible to analyze this system mathematically. To perform a careful investigation of this interaction we decided to conduct some experiments on the NETSIM [21] simulator. Each simulation run involves data transfer over N simultaneous TCP sessions, each starting from a stationary host and terminating at a mobile host. During the course of data transfer, channels between the base station and mobile hosts are independently subject to burst errors. Using the same seed for the random number generator, we repeat this experiment with different CSDP policies. A fixed seed ensures that the same error pattern is repeated in each experiment. It allows us to directly compare results from different experiments. Under each policy π, we measure the time it takes to complete each TCP session (T_i^π). From the collected data set we compute:

$$T_{min}^\pi = min\{T_1^\pi, \ldots, T_N^\pi\}$$
$$T_{max}^\pi = max\{T_1^\pi, \ldots, T_N^\pi\}$$

T_{max}^π denotes the time when the last TCP session completes. We say that a policy π_1 provides better wireless channel *utilization* compared to policy π_2 if $T_{max}^{\pi_1} < T_{max}^{\pi_2}$. Time difference between the finish time of the first and the last session $(T_{max}^\pi - T_{min}^\pi)$ provides us a measure of *fairness*. In a perfectly fair system, identical sessions should finish at the same time. Below, we compare the performance of FIFO, CSDP-RR, CSDP-ETF and CSDP-LQF schedulers using these two evaluation criteria. Results in the next section demonstrate how CSDP schedulers interact with TCP sources. They also provide a basis for making comparison across different CSDP policies.

5.5.1 CSDP with Perfect Channel Estimation

Figure 5.10 shows file transfer times for 10 TCP connections. Each session involves a 0.8 Mbyte file transfer. In this experiment, mean burst length was 100 ms, mean good period was 5 sec and the maximum retries limit was set to 8. Packet loss probability in burst period is 0.8. Our results indicate that:

- CSDP schedulers provide better link untilization since $T_{max}^{CSDP} < T_{max}^{FIFO}$.

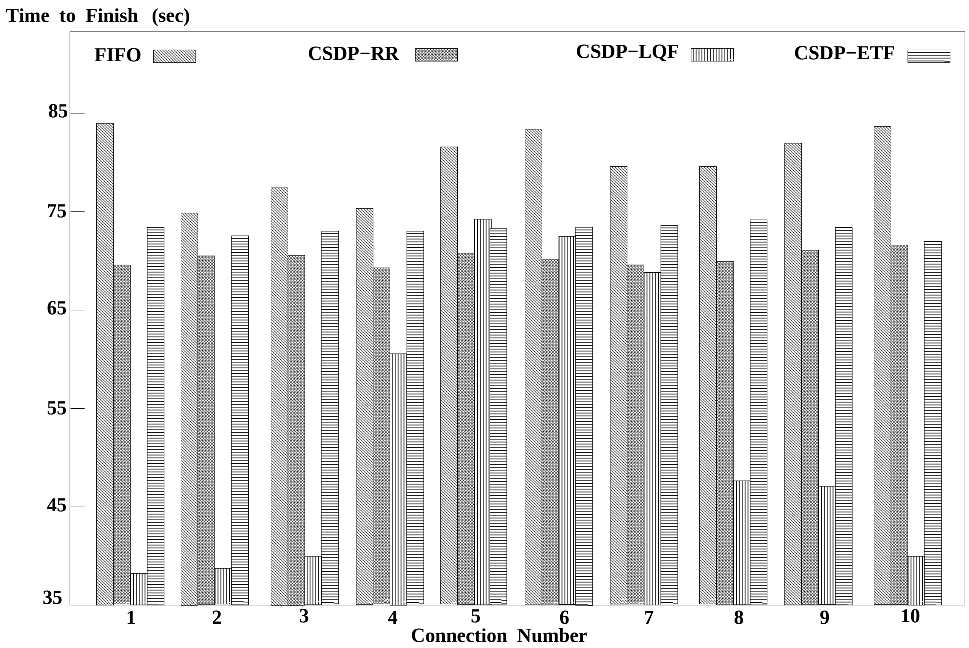


Figure 5.10: CSDP scheduling policies, burst=100ms, R_{max}=8

- Compared to FIFO, sessions under CSDP-RR and CSDP-ETF receive fair service since $|T^{CSDP}_{max} - T^{CSDP}_{min}| < |T^{FIFO}_{max} - T^{FIFO}_{min}|$

CSDP scheduling policies differ widely in terms of their interaction behavior with TCP sources. Since all TCP sources are identical, relative difference among their transfer times is a good measure of fairness. Under a RR policy all connections finish about the same time. Results for LQF policy are rather surprising. Some connections finish very fast while other remain blocked. For Poisson sources and a random error model, LQF is known to be the best policy [46] for this system. This result, however, does not hold when input traffic and loss characteristics on output channels are correlated. As LQF attempts to serve the queue with the maximum number of packets first, the corresponding TCP source receives a bigger share of the bottleneck link. It keeps expanding its window size while packets in other queues wait for their turn for transmission. The result is that some connections 'win' in the beginning while other remain 'blocked' until other connections finish or suffer burst errors. Overall, CSDP-RR scheduler outperforms all other policies, both in terms of performance and implementation complexity. A summary of our results and a qualitative comparison of various CSDP scheduling methods is shown in Table 5.3.

Data reported in figures 5.10 and 5.11 is representative of typical simulation runs. We have observed similar performance improvements when the mean length of the burst period was varied from 50ms to 500ms while keeping the ratio between mean good and bad periods equal to 10.

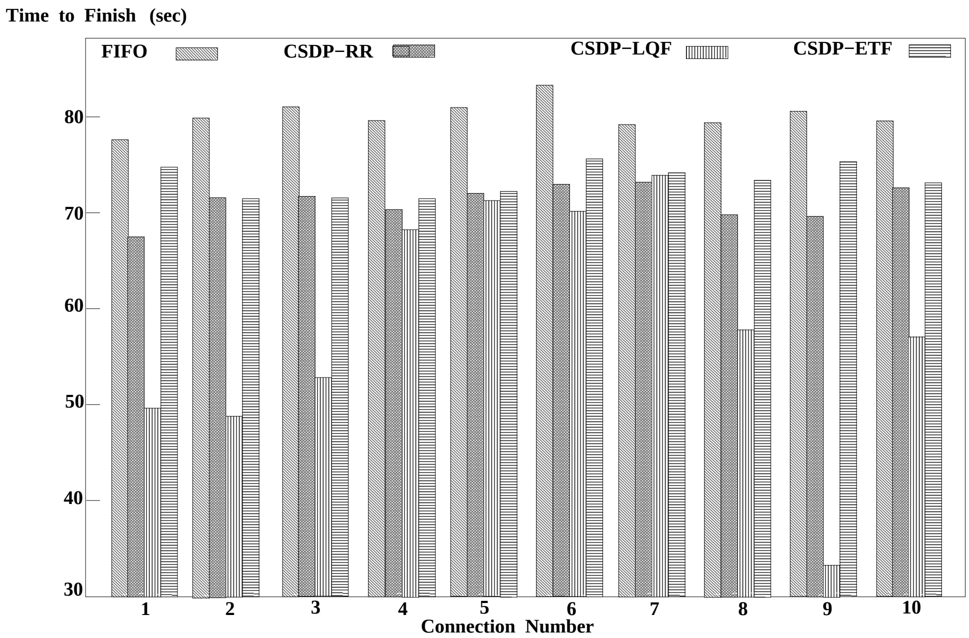


Figure 5.11: CSDP scheduling policies, burst=500ms, R_{max}=8

Policy	FIFO	RR	LQF	ETF
Channel Utilization	worst	best		
Fairness		best	worst	
Throughput	worst	best		
Implementation Complexity	$O(1)$	$O(1)$	$O(n)$	$O(n)$

Table 5.3: Comparison of CSDP schedulers

5.5.2 CSDP with Imperfect Channel Estimation

The level of actual improvement depends on the degree of accuracy in characterizing channel loss behavior. Results reported in the previous section were performed assuming full channel knowledge. Under these conditions we have observed as much as 10-15 % improvement in channel utilization. In practice, however, this may not always be achievable. To demonstrate the effectiveness of the CSDP approach in realistic settings, we operated the CSDP-RR scheduler without any feedback from the Link Status Monitor. We refer to this scheduler as Zero Channel Knowledge Round Robin (ZCK-RR) scheduler. ZCK-RR serves all queues in round robin sequence without differentiating between marked and unmarked destinations. Following a packet loss, it simply defers retransmission of the lost packet until the next round. The deferred period can be thought of as a simple channel

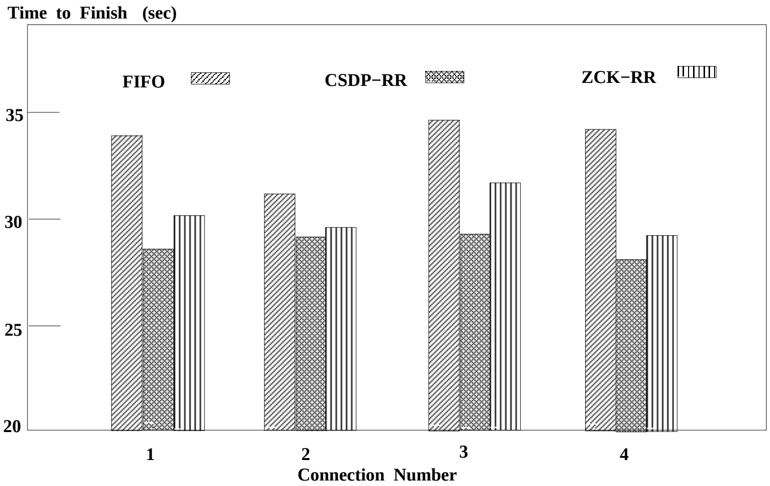


Figure 5.12: Comparison of FIFO, CSDP-RR and ZCK-RR. Mean burst length = 100ms, $p_B = 0.8$

estimator, and, as our results show (in Figure 5.12), this simple estimator also works quite well in practice. If errors on the wireless channel are random, the link status monitor will mark all destinations as in "burst" state. In this case, the operation of a CSDP-RR scheduler will be identical to a ZCK-RR scheduler. Thus, CSDP-RR will always outperform FIFO, regardless of what the error characteristics are and whether a good channel estimator is available or not.

5.6 Critique

Link layer error recovery mechanisms in current wireless LANs, such as those in the forthcoming IEEE 802.11 standards, are effective only in the presence of random errors. Recent performance studies [20] and our experience with RF wireless LANs have shown that error behavior on wireless channels is often bursty. We have also observed that using a FIFO packet dispatcher in conjunction with typical wireless link layer protocols results in poor channel utilization, unfairness, and throughput degradation. We have described a simple technique for link layer error recovery that provides an efficient way to recover from the inefficiencies caused by bursty packet errors. Further, the Channel State Dependent Packet (CSDP) scheduling utilizes the distinct states of the channel to each mobile user to increase the efficiency of the link. Since the burst periods are relatively long, there are many performance benefits in monitoring channel state information.

The strength of the CSDP scheduling approach lies in its ability to hide effects of burst errors from the transport and application layer protocols. This is achieved by recovering from packet losses at the link layer rather than relying on higher layer error recovery mechanisms. It allows existing protocols and applications to run unmodified over wireless channels and, at the same time, it provides performance comparable to wired networks. Since it is a layer 2 solution, not only TCP, but all UDP based applications such as `rpc` and `NFS` also benefit from it. A layer 2 solution is attractive for many reasons. It only requires a software modification to the device driver which makes it easy to deploy and test. It can also be used in LAN-to-wireless bridges. Solutions that rely on a transport layer agent [5, 55], or require a soft-state maintenance at the wireless access point cannot be used in wireless bridges.

Recently, several techniques for improving performance over wireless channels have been proposed [14, 2, 5, 30]. These techniques provide fast recovery from losses due to handoff and errors on the wireless channels. A majority of them operate at the layer 4 of the protocol stack. For example, the proposal by Carceres et. al. [15] requires modification to TCP; the proposal by Bakre et. al. [5] requires a layer 4 proxy-agent; and the proposal by Katz et. al. [2] maintains a log of TCP packets at the base station. Though the CSDP approach attempts to resolve the same set of problems[8], it is not an alternative to these proposals. Since ours is a layer 2 solution, it can be used in conjunction with any of these proposals. Deploying a CSDP scheduler underneath a layer 4 solution will yield the benefits of both approaches. Thus, the CSDP proposal complements rather than compete with other proposals.

The level of achievable performance of the CDPS approach depends on the accuracy of the channel state predictor. Under the assumption of full channel knowledge, we have observed up to 15% improvement in channel utilization [8]. Even in absence of the channel state information, we have shown that the round robin version of the CSDP scheduler provides significant performance improvement over a pure FIFO dispatcher. The ability to differentiate between losses due to collision and losses due to channel errors is crucial for the efficient operation of a CSDP scheduler. If a loss due to collision is misinterpreted as occurring due to channel errors, then the CSDP scheduler would unnecessarily defer transmissions for that destination. Based on received signal strength. a wireless receiver might easily be able to distinguish between the two cases. For example, a collision would cause the received signal strength to increase by a few db, while typical errors due to signal fade would cause the received signal strength to decrease by a few db. Many current generation cellular products already use adaptive power control mechanisms to mitigate the effects of fading and other channel fluctuations. It also might be possible to differentiate between collisions and other channel errors by utilizing these sophisticated control mechanisms.

[8] except for losses due to handoff

5.7 Summary

In this chapter, we investigated the effect of burst packet errors and error recovery mechanisms employed in wireless MAC protocols on the performance of transport protocols such as TCP. Most wireless LAN link layer protocols recover from packet losses by retransmitting lost segments. When the wireless channel is in a burst error state, most retransmission attempts fail, thereby causing poor utilization of the wireless channel. Furthermore, in the event of multiple sessions sharing a wireless link, FIFO packet scheduling can cause HOL blocking resulting in unfair sharing of the bandwidth. This observation leads to a new class of packet dispatching methods which explicitly take wireless channel characteristics into consideration in making packet dispatching decisions. We compared a variety of channel state dependent packet (CSDP) scheduling methods with a view towards enhancing the performance of transport layer sessions. Our results indicate that by employing a CSDP scheduler at the wireless LAN device driver level, a significant improvement in channel utilization can be achieved in typical wireless LAN configurations.

$S_i(t)$	the status of the channel between host i and the base station at time t. $S_i(t) \in \{G, B\}$
$A(t)$	Set of destinations such that $\forall i \in A(t), S_i(t) = G$
$P(t)$	Set of destinations such that $\forall i \in P(t), S_i(t) = B$
π	Scheduling policy. $\pi \in \{FIFO, RR, LQF, ETF\}$
Q_i	FIFO queue of packets ready-to-be transmitted to destination i
$\xi_{bi}(t)$	remaining duration in state B, of the channel between host i and the base station, at time t

Table 5.4: CSDP Scheduler Notations

```
Receive(packet𝒫)
        j ← destination of 𝒫
        Enqueue(𝒫, Q_j)
        if (TxFree)
        Schedule Transmit event.

Transmit(t)
        if (Q_{A(t)} ∪ Q_{P(t)} = ∅) then {
                Txfree ← TRUE ;
                return;
                }
        if (Q_{A(t)} ≠ ∅) then
                j ← π(A(t)) ;
        else
                j ← π(P(t));
        𝒫 ← Dequeue( Q_j );
        transmit 𝒫;
        if (ACK not received within T) {
                enqueue(𝒫, Q_j)
                S_j(t) ← B
                A(t) ← A(t) − j
                P(t) ← P(t) ∪ j
                Schedule event LinkToggle(j, t) event at
                t ← t + ξ_{bi}(t);
        }
        Schedule next Transmit Event;

LinkToggle(j, t)
        S_j(t) ← G
        A(t) ← A(t) ∪ j
        P(t) ← P(t) − j
```

Figure 5.13: CSDP Scheduler Operation

Chapter 6

Conclusion and Future Work

6.1 Contribution Summary

This dissertation makes several important contributions to the area of mobile internetworking.

- We have established that supporting an *address translation* service at the network layer is fundamental to supporting host mobility within connection-less network architectures. We have proposed a network layer solution architecture which enables smooth integration of mobile end systems within the existing Internet. This architecture not only generalizes the previous schemes proposed for MobileIP [47, 51, 22, 36], but it also provides a basis for evaluating the trade-off between various design alternatives for mobile networking systems.

- Exploiting IP's LSR option, we have proposed a MobileIP protocol which provides location independent network access to TCP/IP compliant hosts. Unlike other Mobile-IP proposals that are encapsulation based, our approach provides optimal routes for all TCP sessions, requires less overhead, and is more suitable for deployment in the next generation IP protocol. The scheme is fully distributed and scalable.

- We have demonstrated that the LSR based MobileIP scheme also provides an elegant solution to the problem of switching networking interfaces in the middle of active network connections.

- Finally, we have shown that end-to-end reliable transport layer mechanisms do not provide efficient error recovery when packet losses on the wireless medium are bursty. To support efficient operation, additional mechanisms at the link layer are required. We have proposed Channel State Dependent Packet (CSDP) scheduling, a generalized packet dispatching method, which utilizes the wireless channel state knowledge to reduce packet losses on the wireless medium. CSDP can be employed in conjunction with any wireless link layer protocol [35] to improve throughput over multiple access wireless channels. Recently, some higher layer solutions have also been proposed to support efficient operation over wireless channels [15, 5, 55, 2]. Layering them on top of CSDP will yield the benefits of both approaches.

We acknowledge that the issues addressed in this dissertation represent only a few points in the vast territory of open problems that remain to be explored in the area of mobile computing. Research in this area is only beginning, and it will take many more years of research effort before we develop a full understanding of all design tradeoffs.

6.2 Future Research Direction

Over the last 2 to 3 years, much of the work on MobileIP has been motivated by the need to provide *location transparency* to users of mobile computers. Mobile-IP ensures that a mobile host "virtually" remains connected to its home regardless of its current point of attachment. This allows existing applications to operate over mobile nodes without any modifications. Portable computers, like desktop computers, need to access resources such as NFS file servers, name servers, etc. Existing distributed systems are configured to make use of resources available on their home networks. As a result, portable systems continue to access services from their home network even when they are not physically connected to it. This restriction gives rise to many performance problems since every access to a home server is routed across multiple, possibly slow, links. Slow links introduce delays and cause performance degradation visible to the mobile user. To alleviate this problem, new mechanisms are required that enable mobile hosts to discover and access resources on the foreign network [10].

ATM is rapidly evolving as a high speed packet switching technology which promises to provide support for integrated services traffic. Integrating mobile end-systems within a connection-oriented ATM environment is a challenging open problem. Since standards and protocols for ATM are still evolving, ATM provides an unconstrained platform for developing next generation mobile internetworking protocols and applications. The two tier addressing approach can be integrated with ATM signaling protocols to set up virtual circuits to mobile ATM nodes. Extending quality of service guarantees of ATM virtual circuits over wireless segments is another challenging problem that is worth investigating.

Over the long term it is important to design protocols and provide operating system support to enable portable computers achieve the same degree of functionality as stationary hosts. There are two major physical constraints which make this task difficult. First, wireless channels are error-prone, bandwidth limited, and provide intermittent network connectivity due to signal fading. Second, portable computers have to operate under limited battery power. Pure hardware solutions to these problems cannot be satisfactory since applications have widely different resource requirements. To obtain the best results, operating systems should provide an interface which exposes changing operating conditions to the running applications, leaving policy decisions to the software. Layered on top of this interface, adaptive applications and communication protocols can be developed, tested, and deployed. These developments will allow users to work efficiently and fully enjoy the flexibility of mobile computing.

Appendix A

Packet Formats

A.1 mh2mr Packet Type

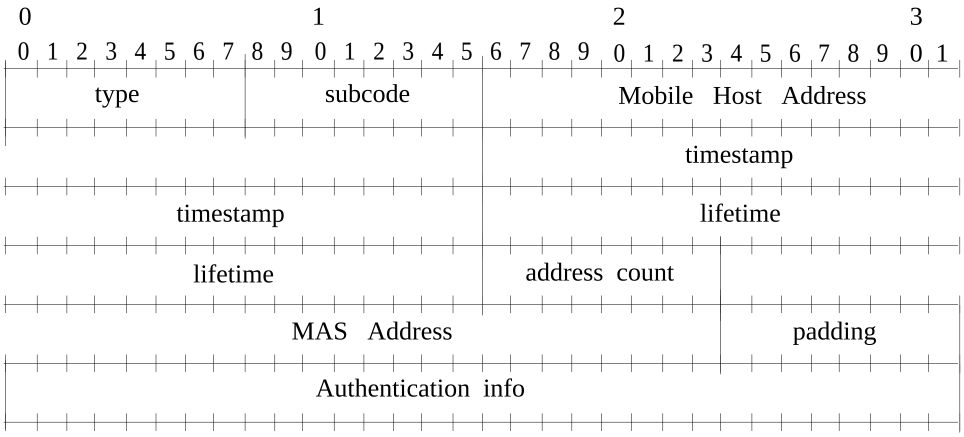


Figure A.1: mh2mr packet format

Packet Type:

MH2MR_DATA or MH2MR_ACK

Subcode:

Indicates whether authentication information is present or not

Mobile Host Address:

The home address of the Mobile Host.

Timestamp:

A 32-bit sequence number.

Lifetime:

A 32-bit field specifying the maximum time the MR receiving the `mh2mr` message can rely on the information in the message.

MAS address:

This is the IP address of the wired-interface of the MAS in whose cell the MH is located at the time of sending this message.

Authentication Info:

A password or token that the MR uses to decide whether the MH has the credentials to be given service. Multiple authentication fields may be present to accommodate a variety of authentication mechanisms.

A.2 mr2mas Packet Type

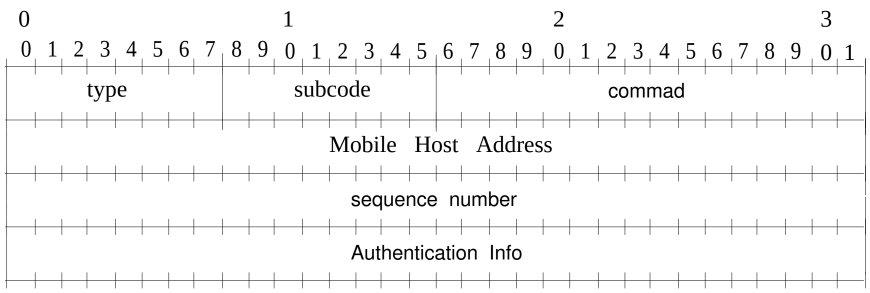


Figure A.2: mr2mas packet format

Packet Type:

MR2MAS_DATA or MR2MAS_ACK

Subcode:

Indicates whether authentication information is present or not.

Command:

either `DELETE_HOST_ROUTE` or `ADD_HOST_ROUTE`

MH address:

IP address of the mobile host.

Sequence Number:

A 32-bit sequence number, used to match ACK against DATA packets.

Authentication Info:

A password or token that the MAS uses to decide whether the MR has the credentials to issue add/delete commands. Multiple authentication fields may be present to accommodate a variety of authentication mechanisms.

A.3 Beacon Format

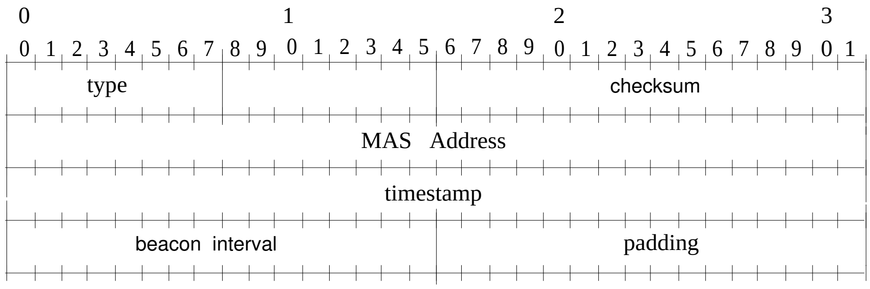


Figure A.3: beacon format

Packet Type:

BEACON

checksum:

16-bit checksum of the beacon packet.

MAS address:

IP address of the wired-interface of the MAS.

Timestamp:

A 32-bit sequence number.

Beacon Interval:

An MAS broadcasts beacons at a constant rate. This field denotes the beaconing period (in milliseconds). MHs may decide that they have moved away from a cell if they do not receive any beacon in a time interval larger that this field.

Appendix B

IP Loose Source Route Option

The following description from [41] describes the IP Loose Source Route Option.

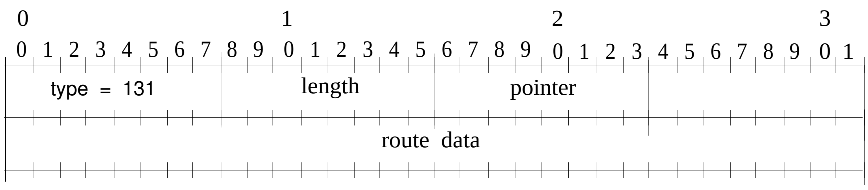


Figure B.1: Loose Source Route Option

The loose source and record route (LSRR) option provides a means for the source of an internet datagram to supply routing information to be used by the gateways in forwarding the datagram to the destination, and to record the route information.

The option begins with the option type code. The second octet is the option length which includes the option type code and the length octet, the pointer octet, and length-3 octets of route data. The third octet is the pointer into the route data indicating the octet which begins the next source address to be processed. The pointer is relative to this option, and the smallest legal value for the pointer is 4.

A route data is composed of a series of internet addresses. Each internet address is 32 bits or 4 octets. If the pointer is greater than the length, the source route is empty (and the recorded route full) and the routing is to be based on the destination address field.

If the address in destination address field has been reached and the pointer is not greater than the length, the next address in the source route replaces the address in the destination address field, and the recorded route address replaces the source address just used, and pointer is increased by four.

The recorded route address is the internet module's own internet address as known in the environment into which this datagram is being forwarded.

This procedure of replacing the source route with the recorded route (though it is in the reverse of the order it must be in to be used as a source route) means the option (and the IP header as a whole) remains a constant length as the datagram progresses through the internet.

This option is a loose source route because the gateway or host IP is allowed to use any route of any number of other intermediate gateways to reach the next address in the route.

Must be copied on fragmentation. Appears at most once in a datagram.

Bibliography

[1] ISO 8473. Protocol for Providing the Connectionless-mode network service. ISO standards document.

[2] E. Amir, H. BalaKrishnan, S. Seshan, and R.H. Katz. Efficient tcp over networks with wireless links. In *The First International Conference on Mobile Computing and Networking 1995*, Berkeley, CA, November 1995.

[3] V. Anantharam, M. L. Honing, U Madhow, and V. K. Wei. Optimization of a database hierarchy for mobility tracking in a personal communications network. In *Proceedings of Performance 93*, Septembero 1993.

[4] Baruch Awerbuch and David Peled. Online Tracking on mobile users. In *Proceedings of ACM SIGCOMM*, 1991.

[5] Ajay Bakre and B. R. Badrinath. I-TCP: indirect TCP for mobile hosts. In *Proceedings of the International Conference on Distributed Computing Systems*, Vancouver, Canada, 1995. IEEE.

[6] Amotz Bar-Noy and Ilan Kessler. Tracking Mobile Users in wireless communication Networks. *IEEE Transactions on Information Theory*, pages 45–65, Jan 1994.

[7] Amotz Bar-Noy, Ilan Kessler, and Moshe Sidi. Mobile Users: To Update or not to Update? In *Proceedings of IEEE INFOCOM*, pages 570–576, Toronto, Canada, June 1994.

[8] P. Bhagwat, P. Bhattacharya, A. Krishna, and S. K. Triphati. Using channel state dependent packet (csdp) scheduling to improve throughput over wirless lans. Technical Report RC 20093, IBM T. J. Watson Research Center, 1995.

[9] Pravin Bhagwat and Charles Perkins. A Mobile Networking System based on Internet Protocol(IP). In *Proceedings of USENIX Symposium on Mobile and Location Independent Computing*, pages 69–82, Cambridge, MA, Aug 1993.

[10] Pravin Bhagwat, Charles Perkins, and Satish K. Tripathi. Transparent resource discovery for mobile computers. In *Workshop on Mobile Computing Systems and Applications*, Dec 1994.

[11] Pravin Bhagwat and Satish K. Tripathi. Mobile computing: A research perspective. In *Networks, 94*, Dec 1994.

[12] R. Braden. Requirements for Internet Hosts – Communication Layers. RFC 1122, Oct. 1989.

[13] R. Braden. Requirements for Internet Hosts – Communication Layers. RFC 1122, Oct. 1989.

[14] R. Caceres and L. Iftode. Improving the Performance of Reliable Transport Protocols in Mobile Computing Environment. *IEEE Journal on Selected Areas in Communications*, 1994.

[15] Ramon Caceres and Liviu Lftode. The effects of mobility on reliable transport protocols. In *Proceedings of the International Conference on Distribnuted Computing Systems*, pages 12–20, Poznan, Poland, June 1994. IEEE.

[16] Douglas E. Comer. *Internetworking with TCP/IP*. Prentice Hall, 1991.

[17] Antonio DeSimone, Mooi Choo Chuah, and On-Ching Yue. Throughput performance of transport-layer protocols over wireless lans. In *Proceedings of the Conference on Global Communications (GLOBECOM)*, pages 542–549. IEEE, 1993.

[18] Wim Diepstraten. IEEE 802.11: Wireless Access Method and Physical Specification, May 1993. Doc:IEEE P802.11-93/70.

[19] Documnet Standards. Cellular Digital Packet Data System Specification. Release 1.0, July 1993.

[20] D. Duchamp and N.F. Reynolds. Measured performance of a wireless lan. In *17th Conference on Local Computer Networks*, pages 494–499, Minneapolis, MN, September 1992. IEEE.

[21] Andrew Heybey. NETSIM: The Network Simulator, Sept 1990.

[22] John Ioannidis, Dan Duchamp, and Gerald Q. Maguire Jr. IP-based Protocols for Mobile Internetworking. In *Proceedings of ACM SIGCOMM*, pages 235–245, 1991.

[23] John Ioannidis and Gerald Q. Maguire Jr. The Design and Implementation of a Mobile Internetworking Architecture. In *Proceedings of Winter USENIX*, pages 491–502, San Diego, CA, Jan 1993.

[24] Van Jacobson. Congestion avoidance and control. *ACM Computer Communication Review*, 18(4):314–329, August 1988. Proceedings of the Sigcomm '88 Symposium in Stanford, CA, August, 1988.

[25] W.C. Jakes. *Microwave Mobile Communications*. IEEE Press, Piscataway, NJ, 1994. Reissue of 1974 edition.

[26] David B. Johnson. Mobile host internetworking using ip loose source routing. Technical Report CMU-CS-93-128, Carnegie Mellon University, Pittsburgh, PA 15213, February 1993.

[27] Phil Karn and Craig Partridge. Improving round trip time estimates in reliable transport protocols. *ACM Transactions on Computer Systems*, 9:364–373, November 1991.

[28] T. V. Laksman and U. Madhow. Performance Analysis of window-based flow control using TCP/IP: the effect of high bandwidth-delay products and random loss. In *High Performance Networking*, 1994.

[29] B. Lyon. Sun Remote Procedure Call Specification. Technical Report, Sun Micro Systems, 1984.

[30] Pietro Manzoni and Deepak Ghoshal. Design and Analysis of a Protocol to Support Host Mobility in a Wide Area Network. *IEEE Journal on Selected Areas in Communications*, 1994.

[31] Steven McCanne and Van Jacobson. The BSD packet filter: A new architecture for user-level packet capture. In *1993 WInter Usenix Conference*. Usenix, January 1993.

[32] Andrew Myles and David Skellern. Comparing Four IP Based Mobile Host Protocols. In *Joint European Networking Conference*, May 1993.

[33] Pankaj Jalote. *Fault Tolerance in Distributed Systems.* Prentice-Hall, Inc., 1994.

[34] Baiju V. Patel, P. Bhattacharya, Yakov Rekhter, and Arvind Krishna. An Architecture and Implementation Toward Multiprotocol Mobility. *IEEE Personal Communication Magazine*, 2(3):32–42, Jun 1995.

[35] S. Paul, E. Ayanoglu, T.F. La Porta, K-W.H. Chen, K. K. Sabnani, and R.D. Gitlin. An asymmetric protocol for digital cellular communications. In *IEEE INFOCOM 95*, Boston, MA, April 1995. IEEE.

[36] Charles Perkins. draft-ietf-mobileip-protocol-09.txt. Draft RFC - work in progress, June 1995.

[37] Charles Perkins and Pravin Bhagwat. A Mobile Networking System based on Internet Protocol. *IEEE Personal Communication Magazine*, 1(1):32–41, Feb 1994.

[38] Charles Perkins and Yakov Rekhter. Short-cut Routing for Mobile Hosts. Internet draft, July 1992.

[39] J. Postel. User Datagram Protocol. RFC 768, Aug 1980.

[40] J. Postel. Internet Control Message Protocol. RFC 792, Sep 1981.

[41] J. Postel. Internet Protocol. RFC 791, Sep 1981.

[42] J. Postel. Transmission Control Protocol. RFC 793, Sep 1981.

[43] Takuro Sato, Manabu Kawabe, Toshio Kato, and Atsushi Fukasawa. Throughput analysis method for hybrid arq schemes over burst error channels. *IEEE Trans. on Vehic. Tech.*, 42(1), February 1993.

[44] Scott Shenker, Lixia Zhang, and Dave Clark. Some observations on the dynamics of a congestion control algorithm. *ACM Computer Communication Review*, pages 30–39, October 1990.

[45] F. Swarts and H.C. Ferreira. Markov characterization of digital fading mobile VHF channels. *IEEE Trans. Vehic. Tech.*, pages 977–985, November 1994.

[46] L Tassiulas and A. Epremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, 39(2):466, March 1993.

[47] Fumio Teraoka and Mario Tokoro. Host Migration Transparency in IP Networks. *Computer Communication Review*, pages 45–65, Jan 1993.

[48] Fumio Teraoka, Keisuke Uehara, Hideki Sunahara, and Jun Murai. VIP: A Protocol Providing Host Mobility. *Comminications of the ACM*, 37(8), August 1994.

[49] Fumio Teraoka, Yasuhiko Yokote, and Mario Tokoro. A Network Architecture Providing Host Migration Transparency. In *Proceeding of ACM SIGCOMM*, Sept 1991.

[50] Paul Turner. NetWare Communications Processes. Novell Research, September 1990.

[51] Hiromi Wada, Takashi Yozawa, Tatsuya Ohnishi, and Yasunori Tanaka. Mobile Computing Environment Based on Internet Packet Forwarding. In *proceeding of Winter USENIX*, pages 503–517, San Diego, CA, Jan 1993.

[52] Dan Walsh, Bob Lyon, and Gary Sager et. al. The Overview of the Sun Network File System. In *Proceedings of Winter USENIX*, pages 117–124, Dallas, TX, Jan 1985.

[53] H.S. Wang and N. Moayeri. Finite state markov channel – a useful model for radio communication channels. *IEEE Trans. Vehic. Tech.*, pages 163–171, February 1995.

[54] R. W.Scheifler and J. Gettys. *X Window System.* Digital Press, 1992.

[55] Raj Yavatkar and N. Bhagwat. Improving end-to-end performance of tcp over mobile internetworks. In *Mobile 94 workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, December 1994.

[56] H. Zimmermann. OSI Reference Model - The ISO Model of Architecture for Open Systems Interconnections. *IEEE Transactions on Communications*, 28:425–432, April 1980.