

**PROTOCOLS AND CACHING
STRATEGIES IN SUPPORT OF
INTERNETWORK MOBILITY**

By

Mitchell Paul Tasman

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

Doctor of Philosophy
(Computer Sciences)

at the

UNIVERSITY OF WISCONSIN – MADISON

1994

© copyright by Mitchell Paul Tasman 1994

All Rights Reserved

Abstract

This thesis explores the provision of End System (ES) mobility on large, datagram-based, internetworks.

We describe a new type of network layer address, the hybrid address, which contains a Partial Destination Identifier, a Unique ID, and a Location Sequence number. The Partial Destination Identifier, which changes as a mobile ES relocates between areas of an internetwork, allows for efficient routing, while the Unique ID is used for ES identification at the lowest layers of a routing hierarchy. Finally, the Location Sequence Number, which a mobile ES increments as it relocates between areas, is used to compare the age of multiple addresses for a given mobile ES.

We present a mobility design that incorporates the hybrid address, and is based on the ISO OSI connectionless routing architecture and protocols. Each mobile ES has a home address and a current address. The home address is stored in a global database, and the ES's home area keeps track of the current address. After a mobile ES relocates to a new area, it sends Reconnect messages to the home and previous areas, each of which caches a forwarding pointer for

the mobile ES. ESs that communicate with a mobile ES send datagrams directly to the mobile ES. To accomplish this, each ES maintains cache entries for its mobile correspondents. When a datagram containing an out-of-date address is forwarded by an area, a Rewrite message is sent to the source ES, which updates its cache entry for the destination. An ES also updates its cache based on the source addresses of incoming datagrams.

The mobility algorithm has been implemented and tested in a simulation environment, and performs quite well. We also present the results of a study on strategies for caching mobile ES forwarding pointers at Intermediate Systems in the interior of an internetwork, based on the type and contents of transit control messages. Caching at interior Intermediate Systems based on Reconnect messages yields the greatest benefit, for both tree-shaped and general topology internetworks, while caching based on transit Rewrite messages is not recommended.

Acknowledgements

First and foremost, I thank my adviser, Lawrence H. Landweber, who provided support and encouragement even during extended periods when it seemed as though a thesis might never be forthcoming. Conversations over the years with Professor Landweber helped me to refine the ideas presented in this thesis, and his many comments on the content and organization of the thesis greatly improved the final result.

My parents provided constant, sometimes less-than-subtle, encouragement throughout my tenure in graduate school. “So, when *is* the thesis going to be finished?” was a frequent refrain. Although I did not always appreciate it at the time, I am very grateful for their support.

I thank Barton Miller, Miron Livny, James Goodman, and Murray Thompson for serving on my final defense committee. Barton Miller carefully read the thesis, and provided many helpful suggestions for improvement. Miron Livny made various comments on the thesis, and also deserves thanks for his DeNet simulation language and for teaching me much about writing simulations. The

entire committee contributed to a lively final defense, and made many constructive comments.

A former graduate school colleague, Edith Epstein, did an amazingly speedy yet quite detailed proofreading of the thesis, and provided much useful feedback. The staff, both past and present, of the department's Computer Systems Laboratory provided excellent support, and always graciously tolerated my pestering. Software and documentation helpful in conducting this research was donated by various organizations; I would particularly like to thank David Jacobs at Digital Equipment Corporation and Clarissa Jurgensen at McCaw Cellular for their assistance.

This research was funded in part by Corporation for National Research Initiatives grant AGR DTD 3-21-91, National Science Foundation grant NCR-9015556, and Department of Transportation/Federal Aviation Administration grant 93-G-003. Richard Olson at the FAA Technical Center deserves special thanks for his efforts in obtaining the FAA grant. I also thank Anthony Michel at Bolt Beranek and Newman for providing me with several opportunities to serve as a technical consultant.

Contents

Preface	x
1 Introduction	1
1.1 Motivation for Network Layer Mobility	3
1.2 Applications of Mobility	4
1.3 Problem Definition	6
1.4 Design Characteristics	7
1.5 Major Results	9
1.6 Roadmap to the Remainder of the Thesis	13
2 Previous and Related Work	15
2.1 TCP/IP Internet	16
2.2 ISO OSI Connectionless Routing Architecture	27
2.3 Flat Address	32
2.3.1 Xerox Network System	32
2.3.2 Sincoskie and Cotton	33

2.4	Distributed Systems	34
2.5	Packet Radio	35
2.6	Commercial Telecommunications Offerings	37
2.6.1	Cellular Telephony	37
2.6.2	Personal Number Calling/800 Service	38
2.6.3	ARDIS and RAM Mobile Data	39
3	Network Layer Addressing	42
3.1	Address Types	42
3.2	The Hybrid Network Address	47
4	A Method to Support Mobile End Systems	51
4.1	ISO Routing Architecture	51
4.2	Design Principles	54
4.3	Network Layer Addressing	55
4.4	Mechanism	58
4.5	Further Design Details	67
4.5.1	Processing at an IS	67
4.5.2	Processing at an ES	69
4.6	Optimality of Routes	71
4.7	Comparison with Sincoskie and Cotton	73
5	Caching Strategies at Interior ISs	76
5.1	Design Space	78

5.2	Options Chosen for Study	81
6	Design and Implementation of the Simulation	84
6.1	DeNet Language	85
6.2	Simulation Components	86
6.2.1	ES	86
6.2.2	IS	89
6.2.3	Database	92
6.2.4	Address Manager	92
6.3	Preprocessor tools	93
6.3.1	Mobility Scenario Generator	94
6.3.2	Internetwork Topology and Routing Generator	95
6.4	Simulation Job Submission	96
6.5	Graphing	96
7	Caching Study	98
7.1	Experimental Topologies	99
7.2	Mobility Scenarios	106
7.2.1	Parameter Files	106
7.2.2	Profile Files	108
7.3	Run Details	109
7.4	Numerical Results	110
7.4.1	Correction Factor	111

7.4.2	Dropped Datagrams	112
7.4.3	Suboptimally Routed Datagrams	116
7.5	Refinements to the Mobility Algorithm	121
8	Conclusions and Future Work	132
8.1	Conclusions	132
8.2	Areas for Future Research	134
8.2.1	Intra-area Mobility and Replicated Home ISs	134
8.2.2	Security and Authentication	137
8.2.3	Mobile Subnetworks	140
 Appendices		
A	Sample Input and Output Files	141
A.1	Mobility Scenario Generator	141
A.1.1	Sample Parameters File	141
A.1.2	Sample Profile File	143
A.1.3	Sample Output File	144
A.2	Internetwork Topology and Routing Generator	147
A.2.1	Sample Input File	147
A.2.2	Sample Output File	148
B	Transport Protocol Details	150
 Bibliography		153

Preface

This preface defines four acronyms that are used extensively in this thesis, but might not be familiar to some readers: **ES**, **IS**, **NSAP**, and **NET**. Definitions are also provided for the terms internetwork and TCP/IP Internet. The preface concludes by listing the acronyms that appear in this thesis. Throughout the thesis, acronyms are set in **sans serif** type.

An End System (**ES**) is a computer attached to a network; it may, for example, directly support users, provide services, or perform both functions. The literature has traditionally referred to such a computer as a “host,” but the International Organization for Standardization (**ISO**) has adopted the term **ES**. An Intermediate System (**IS**) is a device that performs network layer routing functions; an **IS** receives network traffic, determines an appropriate next hop, and forwards the traffic toward its destination. The term “router” has often been used to describe such a device, but the **ISO** has adopted the term **IS**. It is possible for a single computer to act as both an **ES** and an **IS**.

A Network Service Access Point (**NSAP**) is the conceptual point at which a user of the network layer interfaces to the network layer; an **NSAP** address

identifies a user of the network layer at a particular ES. It is possible for more than one NSAP to be associated with an ES. While an NSAP address identifies a *user* of the network layer, a Network Entity Title (NET) directly addresses the network layer at an ES or IS.

Finally, the term internetwork describes any set of networks that are interconnected by Intermediate Systems; the component networks will be termed subnetworks, in keeping with current usage. The term TCP/IP Internet refers to a specific internetwork that includes the NSFnet backbone along with various regional and campus subnetworks.

List of Acronyms

Organizations:

ANS	Advanced Network and Services
DARPA	Defense Advanced Research Projects Agency
EIA	Electronic Industries Association
FAA	Federal Aviation Administration
IETF	Internet Engineering Task Force
ISO	International Organization for Standardization
NSF	National Science Foundation

TCP/IP Internet Terms:

ARP	Address Resolution Protocol
IP	Internet Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

Packet Radio Terms:

PR	Packet Radio
PRNET	Packet Radio Network
PROP	Packet Radio Organization Packet

SURAN Survivable Adaptive Networking

ISO Terms:

AFI	Authority and Format Identifier
BIS	Border Intermediate System
CLNP	Connectionless-mode Network Protocol (informally, ISO IP)
ES	End System (Host)
IDI	Initial Domain Identifier
IS	Intermediate System (Router)
LSP	Link State Protocol Data Unit
L1 IS	Level 1 Intermediate System
L2 IS	Level 2 Intermediate System
NET	Network Entity Title
NPDU	Network Protocol Data Unit
NSAP	Network Service Access Point
OSI	Open Systems Interconnection
PDU	Protocol Data Unit
SNPA	Subnetwork Point of Attachment
TP	Transport Protocol
TP4	Transport Protocol Class 4
TPDU	Transport Protocol Data Unit

Other terms:

ATN	Aeronautical Telecommunication Network
CDPD	Cellular Digital Packet Data
ISDN	Integrated Services Digital Network
LAN	Local Area Network
LRU	Least Recently Used
MAS	Mobile Access Station
MSS	Mobile Support Station
PFS	Packet Forwarding Server
PNC	Personal Number Calling
SPF	Shortest Path First
TUBA	TCP and UDP with Bigger Addresses
VIP	Virtual Internet Protocol

Terms that are defined in this thesis:

LSN	Location Sequence Number
MUID	Mobile Unique ID
PDI	Partial Destination Identifier

Chapter 1

Introduction

This thesis studies protocols and caching methods that would enable a national-scale internetwork to support mobile End Systems (ESs). A convergence of two factors made this an opportune time to conduct such research:

1. Recently, there has been intense interest in wireless communication on the part of the computer and communications industries. For data communication, the primary focus has been on LANs, with wide-area internetworking treated as an afterthought; solutions are needed that tie these LANs together.
2. Computers have become increasingly portable, both in terms of physical size and power requirements.

The term *mobile* ES will be used to describe an ES that is able to relocate to an arbitrary point on an internetwork, while maintaining the ability to both

send and receive traffic. The capability to support mobile ESs in a wide-area environment will be referred to as *wide-area mobility*. A mobile ES would be able to disconnect from a wide-area data communication internetwork, relocate geographically, and later reconnect to the internetwork. After reconnecting, the ES would be able to receive new inbound traffic. Although not essential, it would be desirable for preexisting transport connections to survive the relocation process. This model of mobility does not preclude the use of a wireless link for local access. With such a link, disconnections could potentially be very brief (a fraction of a second), just long enough to reterminate the link at a new Intermediate System (IS). However, to be useful, a solution that provides wide-area mobility must be able to support a range of disconnection intervals, anywhere from seconds to hours to days. The term *physical mobility* will be used to refer to cases in which an ES physically moves. An ES does not necessarily need to physically relocate; *logical mobility* is also possible. For example, depending on the time of day, one of several physical ESs might be designated to act as an information server.

Wired and wireless mobility are similar enough that it should be possible to develop a solution that handles both. As an example of the similarity, an active location mechanism where ESs and ISs periodically send “Hellos” to each other should work well for wired as well as for wireless connections. Logical mobility is quite distinct from both forms of physical mobility. In particular, it does not make sense to maintain transport connections across a logical relocation, since

a different physical machine with different state will be involved. Also, designs for physical mobility can take advantage of locality of movement, but logical mobility allows for instantaneous relocation across an arbitrary distance. In this thesis, we deal only with physical mobility.

1.1 Motivation for Network Layer Mobility

A wide-area internetwork is typically constructed by interconnecting a set of ISs with point-to-point links leased from a telecommunications provider. A key observation is that the two end-points of each communication link are fixed. Thus, at the link layer, a wide-area internetwork has a fixed topology; links may periodically become nonoperational, but they do not move around.

It is this fixed-topology property at the link layer that motivates study of network layer mobility. If a mobility solution were to be implemented at the link layer, as a mobile ES roamed around an internetwork, it would have to somehow remain connected to the same IS. The IS end-point of the communications link would be fixed, but the ES end-point would be mobile.

There are separate considerations for wireless and wired communications links. Current technology constrains wireless links to a relatively short distance. An ES can use a wireless link to connect to a nearby IS, but if the ES moves to another geographic area, the connection is moved to a different IS. In the case of a wired link, if an appropriate switched (dial-up) telecommunications service were available, the ES could maintain a communications link with the

same IS by opening a new circuit after each move. However, tariffs for using such circuits over long distances are likely to be prohibitive.

Long distance links for ES \leftrightarrow IS communication are inadvisable for a second reason. As transmission speeds enter the gigabit range, there is increasing interest in supporting services that require performance guarantees; an example would be real-time video. In order to provide the necessary guarantees, a network must have a precise understanding of the behavior of its communication links. The shorter the link between an ES and IS, the more tightly the link's characteristics can be controlled.

1.2 Applications of Mobility

An obvious question when exploring the provision of wide-area mobility is how it would be used. One application is mobility in the traditional sense: an engineer or businessperson could travel with a personal workstation, and gain access to the same network services and environment that are available at the home location. The workstation would not be limited simply to polling for remotely stored data, such as electronic mail. It would also be able to make use of services that actively generate data — an hourly broadcast of stock quotes, for example. Further, while the workstation was remotely connected, its resources would be accessible to the engineer's or businessperson's colleagues. The Integrated Services Digital Network (ISDN), if universally deployed, could solve part of the mobility problem. With ISDN, a remotely connected workstation would be

able to originate traffic. However, ISDN as currently proposed would not allow services and users to locate the workstation in a timely manner; this would make it infeasible for the workstation to receive spontaneously generated data. The approach to mobility proposed later in this thesis does not suffer from the originate-only limitation.

Wide-area mobility is certain to find application in aviation, particularly for air traffic control. At present, the Federal Aviation Administration (FAA) is studying how best to build a “flying network,” in which plane-based computers would be in constant communication with FAA ground stations [52]¹. Mobility will also be critical for tactical applications. A battlefield situation is not likely to provide a fixed or stable communications infrastructure; “on the fly” reconfiguration is a must. The availability of a mechanism for wide-area mobility would also make the provision of dial-in network access significantly easier. Rather than requiring a separate ad-hoc solution, as is currently the case, dial-in ESs could be treated more generally as mobile ESs. Similarly, wide-area mobility might provide a limited “plug and play” capability for workstations: a workstation could be pre-configured, without knowledge of the specific network connection point.

¹However, as currently envisioned, the network will be modeled as a set of flying ISs rather than ESs.

1.3 Problem Definition

The characteristics of an internetwork and its users will certainly have an effect on the choice of a mobility solution. In advance of the deployment of a mobile ES capability, it is difficult to develop a precise characterization. Nevertheless, in this section, we attempt to present a reasonable internetwork profile.

The first question is one of scale. The current TCP/IP Internet covers the United States, and has a growing number of international links. However, the international portions of the TCP/IP Internet are operated by various independent authorities, and regulatory and tariff issues are likely to limit international mobile internetworking for the medium term. Also, the distance between North America and either Europe or Asia provides a natural constraint on the geographic area within which the typical mobile ES will travel. Thus, it is reasonable to develop mobility solutions for internetworks that are national in scale.

Any estimate of the potential number of networked ESs in the United States is highly speculative; however, for the medium term, it seems likely that the number of such ESs will be no greater than one tenth the number of telephone lines. Thus, we will estimate that the number of networked ESs will be on the order of 10 million. Of those, 50% will be potentially mobile, with no more than 15% on the move at any given time; these estimates are even more speculative. ESs will be evenly distributed across the geographic regions of the United States, with the number in a particular area roughly proportional to the population density. Mobile ESs will generally exhibit locality in terms of movement:

when an ES relocates, it is likely to move to a nearby location. In any case, the distance that a mobile ES can travel per unit time is constrained by available transportation technologies. Locality of access, however, is not necessarily guaranteed. Communication patterns among ESs result from virtual communities of interest, and members of those communities may be geographically separated. Thus, ESs communicating with a mobile ES may be quite far away.

We assume that ESs relocate frequently enough that a global dynamic database such as the X.500 Directory Service [9] is unsuitable for tracking an ES's current location. However, given a reasonable locality of movement, such a database can be used to point to an ES's "neighborhood." Fine location tracking should be done at the network layer. The network layer already handles internetwork routing, and implementing a mobility solution at that layer will allow higher layer protocols to function with minimal (or no) modification.

1.4 Design Characteristics

The mobility design created was developed as a research vehicle for exploring mobility, rather than as an end in itself. The focus was on internetworks that are datagram-based, and offer a connectionless service interface at the network layer. Examples of network layer protocols that provide such an interface are the Internet Protocol (IP) [46] and the Connectionless-mode Network Protocol (CLNP) [33].

We assume that ISs are fixed in location and somehow interconnected. The

goal is for a mobile ES to communicate via the “nearest” IS. The design handles mobile ESs, not mobile subnetworks. ES connections might be wireless or wired.

The design locates a reasonable amount of the “smarts” at the ESs. In particular, an ES keeps track of the addresses of the mobile ESs with which it is currently communicating². Also, should one of those addresses turn out to be out-of-date, the ESs implement a fairly sophisticated error recovery policy. This makes good sense from an engineering perspective, given the relative processing power of ESs and ISs. Also, one important mission of an ES is to communicate with its correspondents; who better, then, to keep track of the correspondents’ addresses? To some extent, the distribution of tasks between ESs and ISs is a philosophical issue.

The design created is explicitly not fully compatible with today’s TCP/IP Internet. While compatibility is a laudable goal, and creative, if imperfect solutions, have been proposed, maintaining strict compatibility is too limiting in the end. The design is loosely based on the ISO OSI connectionless architecture and protocols; some liberties have been taken with the network layer address structure. At the time this research was begun, it appeared that there would be a gradual transition from TCP/IP to the ISO OSI-based protocol suite. In comparison to TCP/IP, the ISO protocols provide greater flexibility in network layer addressing and routing, and are thus a better base for research on mobility. Within the ISO OSI context, our focus is on inter-area and inter-domain (versus

²Such ESs will be termed *correspondent* ESs or just *correspondents*.

intra-area) mobility. During the course of this research, it has become apparent that, at least in the United States, TCP/IP will not be supplanted by the ISO OSI-based protocol suite in the foreseeable future. It appears more likely that the network layer protocol eventually will be replaced with a “next generation” IP. One of the candidates for a “next generation” IP is the TUBA (TCP and UDP with Bigger Addresses) [21] proposal, which would replace the current IP with CLNP, the network layer protocol in the ISO OSI connectionless stack. Regardless of the fate of ISO OSI, we believe that this research, and the results obtained, will be useful to designers of future network protocols.

The design has been implemented and tested in a simulation environment constructed using the DeNet [38] simulation language. The simulation environment was invaluable for assessing protocol correctness, as we were able to iterate and refine the design as problems were discovered. The second use of the simulation environment was to perform a study on methods for caching the current address of mobile ESs at IS nodes in the interior of an internetwork. A variety of caching strategies were investigated, under various mobility workloads and with various internetwork topologies, in order to determine the effect of caching strategies on system performance.

1.5 Major Results

Simulation of our mobility design has indicated that it is indeed an effective method for supporting ES mobility. The design uses a new type of network

layer address, the hybrid address. A hybrid address includes hierarchical components (that partially identify an ES's location) plus a unique ID. Given that it uniquely identifies an ES, the unique ID can be used as a key for cache lookups. A hybrid address also includes a location sequence number, so that it is possible to compare the age of two addresses that contain the same unique ID. The basic mobility design relies on caching at ESs and at leaf ISs, in order to achieve optimal routing. The term *leaf* is used to describe ISs at the lowest level of the routing hierarchy; these ISs are located at the periphery of an internetwork and support connections to mobile ESs. The locality of movement of mobile ESs is exploited by using a two part strategy for tracking the location of a mobile ES. The home address of a mobile ES, which changes slowly, is stored in a global database such as the X.500 Directory Service [9], while the current address is cached at the home location. A mobile ES is responsible for sending a Reconnect message to its home IS after the ES relocates. The ES also sends a Reconnect message to an IS at the previous location, so that datagrams received at that location can be forwarded. If a datagram is addressed to a location where the destination ES is no longer present, an IS at that location rewrites the destination address and forwards the datagram, if a newer address is available. The IS also sends a Rewrite message to the source of the datagram. If a datagram arrives at an incorrect location, and a newer destination address is not available, the receiving IS drops the datagram and sends an Unreachable message back to the source ES. An ES that receives a Rewrite message updates its cache entry

for the destination ES. Upon receiving an Unreachable message, an ES clears its cache entry for the destination ES, and resends to that ES's home address.

Performance of the basic mobility design is excellent. Even with an extremely small forwarding pointer cache size at the leaf ISs, well under one percent of all datagrams are dropped due to the unavailability of a forwarding pointer. The percentage of dropped datagrams is a function of leaf IS forwarding pointer cache size, and falls dramatically as the cache size increases. The percentage of suboptimally routed datagrams is also quite low, in the 1.5 to 2.5 percent range. A datagram is said to have been routed suboptimally if it traveled a longer path than would have a datagram addressed directly to the destination's current address.

Since the basic algorithm is so effective, caching supplemental forwarding pointers at IS nodes at the interior of the internetwork does not have a dramatic effect, but the percentage of suboptimally routed datagrams can be reduced somewhat. Since interior ISs route Reconnect and Rewrite messages, it is possible for those ISs to "spy" on transit control messages, and update their caches accordingly. If an interior IS maintains a forwarding pointer cache, it is able to update the destination address of a transit datagram, in cases where a newer address is available in the cache. If an interior IS updates the destination address of a datagram, it will send a Rewrite message to the source.

We experimented with a range of interior IS forwarding pointer cache sizes, and had interior ISs spy on Reconnect messages, Rewrite messages, or both

message types. Spying only on Rewrite messages is the least effective method. With smaller cache sizes, the most effective method is to spy only on Reconnect messages. A Reconnect message contains more timely information concerning an ES's location than does a Rewrite message, since the Reconnect message was generated when the ES reconnected to the internetwork. When an interior IS's forwarding pointer cache size is small relative to the population of mobile ES's, spying on both Reconnect and Rewrite messages is less effective than just spying on Reconnect messages, since information obtained from Rewrite messages occupies cache entries that would otherwise contain more timely information from Reconnect messages. Only with very large cache sizes will spying on both Rewrite and Reconnect messages be equally (or in some experiments, slightly more) effective than just spying on Reconnect messages. With a very small forwarding pointer cache, it is possible to reduce the percentage of suboptimally routed datagrams slightly, to the 1 to 2 percent range, by spying on Reconnect messages. As the forwarding pointer cache size grows to approximately 20 percent of the mobile ES population size, the percentage of suboptimally routed datagrams drops to approximately .5 to 1 percent.

In addition to it being a less effective method for improving routing, we discovered that having the interior ISs spy on Rewrite messages can cause an out-of-date forwarding pointer cache entry to spread to other caches, and thence to still other caches. This is another reason for not having interior ISs cache information learned from transit Rewrite messages. In rare cases, we found that

an out-of-date forwarding pointer (leading to a location at which datagrams will be dropped) might be present along the path to both a destination ES's current address *and* its home address. We refined the mobility algorithm to allow such a cache entry to be bypassed, so that the source ES is able to recover.

Several other proposals for supporting mobility on datagram-based internetworks had suggested the possibility of caching the current address of mobile ESs at either ESs, ISs, or both. However, a systematic study of such caching methods had not been conducted prior to this thesis. The few performance results that have been presented previously are primarily concerned with the execution time at various individual internetwork components, rather than the behavior of the overall system. This is due at least in part to the fact that system behavior is extremely hard to capture without the use of a simulation environment.

1.6 Roadmap to the Remainder of the Thesis

The remainder of the thesis is organized as follows. Chapter 2 summarizes previous and related work. Chapter 3 compares and contrasts various types of network layer addresses, and introduces the hybrid network address. Chapter 4 describes a scheme for supporting mobility on ISO OSI-based internetworks that incorporates the hybrid network address. Chapter 5 outlines possible strategies for caching ES location information at ISs that are in the interior of an internetwork. Chapter 6 describes the simulation created to conduct the research

presented. Chapter 7 details the parameters of the caching study, presents the numerical results, and describes various refinements to the mobility algorithm. Chapter 8 lists conclusions and suggests directions for future research. Finally, Appendix A provides sample input and output files for the preprocessor tools developed to generate control files for the simulation, and Appendix B provides details of the transport protocol implemented in the ES component of the simulation.

Chapter 2

Previous and Related Work

During the past several years, various proposals have been made for supporting mobility on datagram-based internetworks. The vast majority of these proposals have been designed to be compatible with today's TCP/IP-based Internet. The proposals all have limitations, and do not appear to provide a truly practical method for supporting mobility on a national-scale, datagram-based, internetwork.

The following sections summarize previous and related work in several areas. In some cases, a paper or design may provide ideas that can be incorporated into a mobility solution for large internetworks; in others, it may provide an example of how *not* to support mobility. Related work is summarized by area, with a short critique following each summary or group of summaries. Related work is cited at appropriate points later in the thesis, to compare and contrast it with our ideas.

2.1 TCP/IP Internet

Many schemes have been proposed to allow the TCP/IP Internet to support mobile ESs. An Internet address is 32 bits long, and consists of two parts. The network number, which may be 1, 2, or 3 bytes long, identifies a subnetwork, and the remainder of the address identifies an ES on that particular subnetwork. The transport layer relies on the Internet address for ES identification; hence, the addresses of a pair of communicating ESs must remain constant for the lifetime of a transport connection.

Sunshine and Postel [56] proposed a scheme known as *virtual networks*, where one Internet network number would be assigned to each mobile “community of interest.” A mobile ES would be assigned an Internet address on the virtual network corresponding to the ES’s community of interest, and would retain that address as it relocated. ISs would not keep track of the location of mobile ESs. Instead, for each physical subnetwork that supported mobile ESs, an entity known as a *forwarder* would keep track of ESs on that subnetwork. A correspondent that wished to send a datagram to a mobile ES would first need to identify a forwarder for the subnetwork to which that mobile ES was *currently* attached. The authors suggested that a global dynamic database could be used to provide mobile ES address to forwarder mappings. Once the correspondent identified an appropriate forwarder, it would include a two-part partial source route in each datagram, as an IP option. The first part of the source route would specify the address of the forwarder, and the second part

would specify the mobile ES address. The virtual network proposal allows an ES to retain the same network address as it relocates, but the proposal has a serious deficiency in that it requires the presence of a dynamic global database to identify an appropriate forwarder.

Deering [14] proposed a scheme known as *logical subnets*. A logical subnet is treated as a *single* unit for Internet routing purposes, but it can span multiple physical subnetworks. A mobile ES is assigned an address on one of the logical subnets. A procedure known as *tunneling* is employed to connect the separate physical subnetworks that comprise a logical subnet. Also, a mobile ES that “roams” away from its logical subnet can use tunneling to connect to one of the logical subnet’s physical subnetworks. Tunneling is typically accomplished by encapsulating an IP datagram within a second IP datagram. The logical subnet advertises the address of a single entry point (IS) to the Internet. Although this allows for straightforward Internet routing, routing within the logical subnet will often be suboptimal. This is especially the case with “roaming” ESs, since datagrams must flow via the logical subnet’s entry point, regardless of the ES’s current location.

The solution proposed by Ioannidis, Duchamp, and Maguire [27, 28] is essentially a hybrid of the Sunshine and Postel, and Deering schemes. A wireless mobile ES retains the same Internet address as it relocates; addresses are assigned such that it is possible to identify that a given address belongs to a

mobile ES. The first part of the solution concerns mobility within a small campus internetwork, and is similar to the Sunshine and Postel approach. Devices known as Mobile Support Stations (MSSs) are responsible for forwarding traffic to and from the mobile ESs; there is one MSS per wireless cell, and apparently at least one MSS per subnetwork. When sending a datagram to a mobile ES, an ES routes the datagram to an MSS on the local subnetwork. That MSS either delivers the datagram, or forwards it to the MSS responsible for the destination ES. If an MSS does not know which MSS is currently responsible for a destination, it sends a search request to the other MSSs. When sending a datagram, a mobile ES routes the datagram to the MSS that serves the wireless cell. If a mobile ES roams away from the campus internetwork, it informs one of the (campus) MSSs of its current location. As with the Deering proposal, the MSS uses tunneling to forward datagrams to and from the remote ES. The Ioannidis, Duchamp, and Maguire mobility approach is designed primarily to support mobility within one campus; roaming ESs are treated as a special case.

Teraoka [59, 60, 61, 62] proposed a mobility scheme known as Virtual Internet Protocol, or VIP. In addition to carrying the home address of a mobile ES, a datagram header can include the current address of the ES. All datagrams sent by a mobile ES that is away from its home subnetwork carry both source addresses. The datagram header also contains a timestamp that indicates the time at which the current address was assigned to the mobile ES. ISs and ESs that process a datagram can examine the source addresses and cache the mapping

between home address and current address. Then, when generating a datagram, if an ES or IS has a cache entry for the destination ES, the ES's current address can be included in the datagram's header. A transit IS that processes a datagram can look up the destination ES's home address in its local cache, and if necessary (based on the timestamp information) update the destination ES's current address in the datagram header. If a current address for the destination ES is not present in a datagram's header, and none of the transit ISs have a cache entry for the mobile ES, the datagram will be routed to the ES's home address. If a datagram reaches the home subnetwork, a home IS is responsible for adding the ES's current address and forwarding the datagram. Whenever a roaming mobile ES arrives at a new subnetwork, the ES sends a connect message to its home IS, which creates or updates a cache entry for the mobile ES. Likewise, when a mobile ES is about to disconnect, it sends a disconnect message to the home IS, which deletes the corresponding cache entry¹. Upon receiving a disconnect message, the home IS will also broadcast a disconnect message on all connected subnetworks. Any VIP-capable IS that receives such a message *and* has a cache entry for the referenced mobile ES deletes the cache entry and propagates the broadcast. This procedure results in a controlled flood of the internetwork, and is likely to scale poorly. Also, if some of an internetwork's ISs are not VIP-capable, it is possible that a disconnect message will not be able to reach all the ISs that have an out-of-date cache entry. Since cache entries have

¹Note that Teraoka has described more than one version of the disconnect procedure.

an idle timeout, this is not a fatal problem. If an out-of-date cache entry causes a datagram to be routed to the wrong IS, an error message is sent back to the source. Any IS or ES that processes the message deletes the appropriate cache entry, if one is present. Note that, in the process of conducting our research, we discovered that an error message sent back to a source is not necessarily routed via the IS that contains the errant cache entry. Therefore, the recovery procedure that Teraoka proposes is not entirely effective.

An enhanced version of VIP by Uehara, Teraoka, Sunahara and Murai [63] attempts to circumvent some of the limitations of the original proposal. The enhanced VIP header has various control flags. In particular, it is possible to inhibit the updating of a datagram's destination address at ISs other than the destination ES's home IS. Another control flag inhibits the creation of cache entries based on the source address of a datagram at ISs other than the source ES's home IS. The VIP header also includes a field to record the address of the ES or IS that most recently updated the destination ES's current address in the datagram's header. Additional control packets are defined that allow a cache entry at an arbitrary ES or IS to be updated or deleted; this capability presents significant security concerns unless authentication were to be employed. Instead of immediately deleting an out-of-date cache entry, an IS will mark that entry for deletion and start a timer. During the period before the timer expires, if a datagram is received where the destination home and current addresses match those contained in the cache entry, a cache invalidate message will be sent to the

ES or IS that last updated the destination ES's current address in the datagram's header. Although the enhanced version of VIP greatly improves on the original's ability to delete out-of-date cache entries, it has not solved all of the problems.

The Internet Engineering Task Force (IETF) has created a Mobile IP working group [44, 54] with the charter to develop a method for supporting mobility on the TCP/IP Internet. The design, which is still in progress, is simple. Each mobile ES is assigned a home address, which is an Internet address that remains the same regardless of the mobile ES's location. When the mobile ES has roamed away from home, it also has a care-of-address, which is an Internet address associated with the mobile ES's current point of attachment. The care-of-address either identifies the mobile ES directly or identifies a Foreign Agent that is responsible for providing access to visiting mobile ESs. When away from home, the mobile ES registers its care-of-address with a Home Agent; the Home Agent is responsible for intercepting datagrams addressed to the mobile ES's home address and tunneling them to the associated care-of-address. The design allows a limited form of mobile network (rather than just mobile ES) support, in that the registration message sent by a mobile ES to its Home Agent can specify a set of address prefixes that are reachable via the ES. If it chooses, the Home Agent can advertise reachability to the address prefixes listed in the registration message, and then tunnel received datagrams to the mobile ES. All datagrams addressed to a roaming mobile ES are routed via the Home Agent. When a mobile ES arrives at a new location, 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 ES must somehow acquire a care-of-address, then directly register with the Home Agent. Registration messages are to be authenticated, but the details are being hotly debated by the working group. At present, the plan is to calculate a hash value (using an algorithm such as MD5 [51]) over a secret value shared by a mobile ES and its Home Agent, followed by selected fields of the registration message, followed by another copy of the shared secret. The IETF design assumes that, if there is a Foreign Agent involved, a mobile ES will be able to send datagrams with the home address specified as source address. If there is not a Foreign Agent (that is, the care-of-address directly identifies the mobile ES), it seems likely that the foreign subnetwork would require the mobile ES to specify its care-of-address rather than home address as the source address. In this case, the mobile ES would need to tunnel datagrams back to its Home Agent.

The proposal by Wada, Yozawa, Ohnishi, and Tanaka [64] contains elements of the IETF's proposal and also bears some similarity to Teraoka's VIP. The proposal describes two modes: forwarding and autonomous. Forwarding mode operates nearly identically to the version of the IETF's mechanism where datagrams are tunneled directly to a mobile ES's care-of-address. Autonomous mode requires the presence of an "autonomous supporter" on the network that a mobile ES is visiting, and also software modifications to the correspondent

ESs. If both elements are in place, the correspondent ESs are able to cache a mobile ES's care-of-address and tunnel datagrams directly to the mobile ES. When a Home Agent receives a datagram destined to a mobile ES, it sends a location information message to the source ES, which (if appropriate modifications have been made) caches the mapping between home address and care-of-address. When a mobile ES roams to a new network, it attempts to locate an autonomous supporter. When the mobile ES registers with its Home Agent, it informs the Home Agent of the Internet address of the autonomous supporter, if one is available. The next time that the ES relocates, its Home Agent will send a location update message to the autonomous supporter at the previous network; that autonomous server is not required to record the mobile ES's new care-of-address. Since mobile ESs relocate, a cached care-of-address might become out-of-date, causing a datagram to be sent to a network where a mobile ES is no longer present. As described in the proposal, an autonomous supporter on that network will intercept the datagram; however, it appears that there is a problem in this regard, as the care-of-address may have been reassigned to another ES. The autonomous supporter must somehow selectively intercept datagrams that were tunneled to the mobile ES that has roamed away, while not interfering with those datagrams that are destined to the mobile ES that currently occupies the same care-of-address. The authors indicate that an autonomous supporter must have a promiscuous interface on a broadcast medium, but even this does not appear to provide a practical solution to the problem. If the autonomous

supporter is unable to reliably intercept improperly forwarded datagrams, such datagrams will be delivered to the ES to which the care-of-address has currently been assigned. The receiving ES must verify the destination home address, and discard the datagram if there is a mismatch. In any case, assuming that the autonomous supporter *is* somehow able to intercept a mis-addressed datagram, the supporter forwards the datagram onward if it is aware of a newer address for the datagram; otherwise, the supporter forwards the datagram to the mobile ES's home address. The autonomous supporter sends a location update message to the source of the datagram, so that the source can either update or clear (as appropriate) its cache entry. In addition to a possible problem with autonomous supporter interception of datagrams, a serious omission of the proposal is a timestamp (or similar mechanism) that would make it possible to determine which of two care-of-address mappings for a given home address is more recent.

Perkins [43] developed a clever but impractical mobility scheme using IP's Loose Source and Record Route (LSRR) option. Johnson [35] independently proposed a similar idea. As with the Sunshine and Postel scheme, each mobile ES is assigned an Internet address on a virtual network, and the address remains the same regardless of the ES's current location. Associated with each virtual network is a Mobile Router, which is responsible for advertising reachability to the virtual network, and for keeping track of the current location of each mobile ES that has been assigned an address on that network. Each wireless cell has

a transceiver known as a Mobile Access Station, or MAS. When a mobile ES roams into a new cell, the mobile ES informs its Mobile Router of the Internet address of the cell's MAS. The Mobile Router records this information, and also informs the previously recorded MAS that the mobile ES is no longer present. Datagrams from the mobile ES are routed optimally; however, unless special action is taken, datagrams sent to a mobile ES will end up at the Mobile Router. To forward a datagram to a mobile ES's current location, the Mobile Router adds an LSRR option to the datagram that specifies the appropriate MAS as a transit IS. When the mobile ES replies to a correspondent, the mobile ES also inserts a LSRR option in the outgoing datagram, again specifying the local MAS as a transit IS. In theory, when the correspondent receives the datagram, it will reverse the recorded route, and insert it as a LSRR option in future datagrams sent to the mobile ES; such datagrams will be routed via an optimal path to the mobile ES. Unfortunately, this requires a greater level of ES complexity than was originally intended for LSRR option, and even the basic LSRR specification is often mis-implemented. Another problem is that many IS implementations process the LSRR option quite inefficiently, leading to significant overhead.

A rather different approach to Internet mobility was taken by the designers of the Network Reconstitution Protocol [26]. Internet addresses used with the Network Reconstitution Protocol are redefined to contain a unique IS ID, and a unique ES ID; the protocol assumes IS-centric rather than network-centric routing. Datagrams are forwarded by the various ISs at which a mobile ES

has previously been located, but not indefinitely. It is the responsibility of a mobile ES to decide which of its previous addresses are still in use by its correspondents, and to request that the ISs identified in those addresses forward received datagrams directly to the ES's current location. Also, in order to communicate with mobile ESs, an ES's network layer must be modified. The goal is to take the (source, destination) Internet address pair that is specified by the user of the network layer at the source ES, and to provide that pair to the user of the network layer at the destination ES. However, either the source or destination ES might have relocated since a given transport connection was established. If it is necessary to rewrite the source or destination address of a datagram, the original value (as specified by the network layer user at the source) is stored as an IP option in the datagram header. When a datagram arrives at its destination, if the "old source" or "old destination" options appear in the header, the network layer will restore the original values before passing the datagram to the transport layer.

The various solutions to Internet mobility illustrate the limitations of the TCP/IP protocol suite, especially the Internet address format. As a result of TCP's reliance on the entire Internet address for ES identification, most of the solutions require that an ES maintain the same address as it relocates, and this leads to inefficient routing². The Network Reconstitution Protocol attempted to work around the problem by redefining the Internet address. However, 32

²The Teraoka, Wada and colleagues, and Perkins proposals allow for the possibility of optionally avoiding inefficient routing.

bits does not provide sufficient room to hold 2 reasonably-sized unique IDs. Also, since the IS ID portion of an address changes as an ES relocates, the designers found it necessary to provide a method by which a datagram's source and destination addresses can be restored to the values that a correspondent's transport layer expects to see. To some extent, the TCP/IP protocol suite is a victim of its own success: it has been so widely adopted that it is difficult to contemplate a major change at this point.

See Myles and Skellern [41, 42] and Perkins [44] for other surveys of mobility proposals for the TCP/IP Internet.

2.2 ISO OSI Connectionless Routing Architecture

Several mobility schemes have been proposed for the ISO OSI connectionless routing architecture. We provide a very brief summary of that architecture here; refer to Chapter 4 for a more detailed presentation. The ISO OSI connectionless architecture consists of a 3 level hierarchy. At the lowest level of the routing hierarchy, Level 1 (L1) ISs exchange information about the system IDs of ESs that are present within an area. A few of the L1 ISs are also designated as Level 2 (L2) ESs; these ISs exchange information about the identifiers of the areas that are interconnected to form a routing domain. Finally, certain of the L2 ISs are also Border Intermediate Systems (BISs); BISs exchange information about

the reachability of other domains.

In Carlberg's [8] proposal, each mobile ES is assigned a logical address; such an address is apparently distinct from any physical address, and is retained as the mobile ES relocates. Whenever a mobile ES enters an area, an L1 IS in that area associates a second address, known as a routing address, with the ES. The mapping between logical address and routing address is sent to the other L1 ISs in the area, via a routing update message. When an L1 IS encounters a datagram with a logical address as the destination, if the IS has a forwarding table entry for that logical address, it will forward the datagram based on the associated routing address. Otherwise, the IS will forward the datagram to an L2 IS. Each L2 IS periodically generates a routing update message that lists the logical addresses of *all* the mobile ESs that are present within the L2 IS's area of the domain. Note that this is a significant extension of L2 IS functionality; normally, an L2 IS would not track individual ESs, just the identifiers of areas in the domain. Carlberg's proposal also places new requirements on the BISs. When a BIS learns of a new mobile ES via an L2 routing update, it must update a global dynamic database to indicate that the mobile ES is present in the BIS's domain. When a BIS receives a datagram destined to a logical address that is not present in the local domain, the BIS must look up the address in the global database, to determine in which domain the mobile ES is currently present. Then, the BIS tunnels the datagram to the appropriate domain. Datagrams from a mobile ES are also tunneled, so that the ES's logical address will be hidden from transit

domains. When a BIS at a destination domain receives a tunneled datagram from a mobile ES, by examining the source address of the tunnel, the BIS can learn the domain at which the mobile ES is currently present. Also, when a mobile ES makes its presence known to a L1 IS, the ES identifies its previous domain, and this information is included in the next routing update message sent by the L1 IS. Upon receiving this information, an L2 IS will send a notification to the previous domain, so that the domain can update its information concerning the mobile ES's location. Several aspects of Carlberg's proposal require a high degree of overhead. Two routing update messages will be sent each time that a mobile ES relocates to a different subnetwork within an area [57]. At the second level of the routing hierarchy, L2 ISs must directly track the location of each mobile IS that is present within the routing domain. Finally, BISs must track the location of any mobile ESs that have correspondents in the local domain; this requires interactions with a global database.

Tanaka and Tsukamoto [57] studied mobility *only* within a single ISO area, and developed a method using forwarding to more effectively support mobile ESs. Within an area, ES addresses are unique, and ES locations are tracked directly. Each IS periodically sends a message (to all ISs in the area) that lists the set of ES addresses that are directly reachable via (i.e., neighbors of) that IS. This message is called a Link State PDU, or LSP. When an IS detects that an ES is no longer a neighbor, it generates an updated LSP. An LSP is also generated when an IS detects a new ES neighbor. The LSP mechanism

works adequately when the network topology is relatively static, but will cause problems if ESs relocate regularly. If the LSP generation rate is clamped, ISs will have inaccurate information concerning ES locations, and if the rate is not clamped, LSP generation and processing will become burdensome. Tanaka and Tsukamoto proposed a method to limit the need for LSP updates. The first IS in an area to see a given ES declares itself to be the default neighbor of that ES, and indicates this fact to the other ISs in the area by including a new option type in the LSP. When the ES relocates to another part of the area, and becomes a current neighbor of another IS, the current IS will send a migration notification message to the default neighbor IS, instead of generating an LSP. Datagrams for the ES will continue to be routed to the default neighbor IS, which will tunnel them to the IS identified in the most recent migration notification message. The current neighbor IS might for some reason choose to become the default IS, and indicate this fact by issuing an updated LSP. In this case, the previous default neighbor IS will cease to advertise the ES in its LSPs. ISs through which a migration notification message is routed can optionally observe and cache the current IS associated with the ES specified in the message; this allows datagrams to be directly tunneled to the ES's current location. An IS that is the current neighbor of an ES can also optionally attach this information (using a new option type) to datagrams that are being routed via the IS; this would allow other IS's that process the datagram to decode and cache the ES location information. Tanaka and Tsukamoto observe that cache entries can become

inaccurate, but if a datagram is misrouted, the receiving IS can always tunnel it back to the default neighbor IS, which will tunnel the datagram to the ES's current location. If a transit IS observes that a tunneled datagram originated from the default IS associated with the destination ES, the transit IS will *not* attempt to readdress the datagram. We believe that the method proposed by Tanaka and Tsukamoto is a promising technique for handling mobility within a single area.

One of the candidates for a next generation IP is the TUBA [21, 50] proposal, which would replace IP with CLNP. TUBA includes enhancements to CLNP that would allow datagrams to be routed to mobile ESs. The design is nearly identical to that proposed by the IETF Mobile IP working group; in fact, the draft specification [50] acknowledges that it was copied almost verbatim from the IETF's text.

The Cellular Digital Packet Data CDPD [10] network uses idle transmission capacity in the North American analog cellular telephony network to transmit IP and CLNP datagrams. CDPD follows the ISO OSI connectionless routing architecture, and support for mobility is similar to that proposed for TUBA. When a mobile ES is away from its home area, datagrams are intercepted at the home area, encapsulated, and forwarded to the area where the mobile ES is currently located. Given that CDPD is a commercial service, much attention has been given to management, authentication, and accounting issues. Since *all* datagrams destined to a roaming mobile ES must be routed via the home area

for both TUBA and CDPD, neither method appears to be ideal for supporting ES mobility.

In conjunction with MITRE Corporation, the Federal Aviation Administration is creating the Aeronautical Telecommunication Network (ATN) [3], in order to upgrade air-to-ground communications capabilities. The ATN project deals with flying ISs, rather than flying ESs. Each IS (one per plane) is tracked as a separate ISO OSI routing domain. While this approach may be suitable for the specific case of flying ISs, and makes efficient use of the limited bandwidth air-to-ground interface, the approach is not practical for the general case of ES mobility.

2.3 Flat Address

2.3.1 Xerox Network System

As described by Dalal and Printis [13], the Xerox Network System uses a 48-bit flat internet address to identify an ES; however, along with an internet address, the clearinghouse (a directory agent) returns a network number for use as a “very strong routing hint.” The 48-bit internet address is a globally unique ID, and is also used for subnetwork (that is, Ethernet) addressing. Dalal and Printis do not indicate what happens when the routing hint turns out to be wrong (for example, if an ES has relocated). To support mobile ESs, it would be best if the internetwork had some mechanism to forward a datagram to

the correct network, and to inform the source. Also, having just a two-level routing hierarchy (subnetwork and ES) may ultimately limit internetwork size. Nevertheless, the Xerox approach to internetwork *addressing* appears to provide a reasonable base for support of ES mobility.

2.3.2 Sincoskie and Cotton

Sincoskie and Cotton [55] outlined a network design that would route directly on link layer flat addresses. The network would be composed of a set of broadcast-capable subnetworks that are interconnected with link layer, self-learning bridges. To locate an ES that does not appear in any of the bridges' routing tables would normally require a broadcast across the entire network. For networks that exhibit a strong locality of reference, Sincoskie and Cotton propose a reduced broadcast algorithm. A broadcast would first propagate to nearby parts of the network; only if the destination did not reply would the broadcast be propagated further. Sincoskie and Cotton also observe that to update network state after an ES relocates, it is sufficient for the ES to transmit a message to its former location, and to have the bridges along the path update their routing tables. Although Sincoskie and Cotton suggest that their solution could scale nationally, it appears to make unrealistic demands on the bridges. A key problem with the solution is that a broadcast is required for initial ES location. Also, since the ESs do not cache information concerning the current location of ESs with which they are communicating, the ESs are unable

to provide this information to the bridges for use as a routing hint.

2.4 Distributed Systems

Powell and Miller [48] described process migration in the DEMOS/MP system. A process address contains a unique process ID and an identification of the last known machine that hosted the process. Each time that a process migrates, a forwarding pointer is created at the machine that most recently hosted the process. If a message arrives at a machine where a process has previously resided, the message is forwarded one or more times until it reaches the process at its current location. A message is sent back to the source process, so that the source can update its record of the destination's process address. In developing a generalized forwarding address, Fowler [22] extended the DEMOS/MP format to include a timestamp, which could be as simple as a count of the number of times that an object had relocated. Fowler's thesis studied the use of forwarding pointers as a method for object finding in a distributed system, and provided a mathematical performance analysis of several methods for managing a chain of forwarding pointers. In another study of forwarding address strategies, Man-sharamani and Livny [39], used a combination of mathematical analysis and simulation.

There are some important differences between support for migrating objects in a distributed system, and support for mobile ESs on a large internetwork. Distributed systems often assume a relatively simple network topology (perhaps

a LAN), where each processor can directly address any other processor and routing is straightforward. In the case of a large internetwork, the processors (mobile ESs) themselves are moving, and routing is anything but trivial. In a distributed system, a forwarding address strategy can be implemented by maintaining a chain of forwarding pointers at the processors that have previously hosted an object. For internetwork mobility, the only locations available for storing forwarding pointers are the internetwork's ISs. A network application running on an ES may have quality of service requirements for bandwidth, latency, jitter, or percentage of traffic dropped, and these will constrain the choice of a path through the internetwork. In general, the number of routing hops should be minimized. The constraints on routing, combined with the possibility of communication link and IS failures, make it inadvisable to have a long chain of forwarding pointers. Also, on an internetwork, an ES that wishes to begin communicating with another ES will generally not have a network address for that ES. Instead, the ES will have an Application Identifier or ES name, and a mechanism must be provided to translate this information into a network address.

2.5 Packet Radio

The DARPA Packet Radio Network (PRNET) [36] allows communication among a community of wireless mobile ESs. Although ESs on a PRNET often run the

Internet protocol suite, packet radio technology implements a general communication facility. Each ES is attached to a packet radio (PR), which periodically broadcasts a Packet Radio Organization Packet (PROP). The PROP includes the Internet (or other) address of the ES that is currently attached to the PR. The ES to PR mappings are recorded by the other packet radios, which make use of the mappings when introducing packets into the PRNET. This approach allows an ES to be addressed regardless of its current physical location or the particular packet radio to which it is attached. The DARPA Survivable Adaptive Networking (SURAN) program extended packet radio technology to larger, hierarchically routed, networks [37]. At the first level of the hierarchy, PRs are organized into clusters, and at the second level clusters are organized into a network. For very large networks, a third supercluster level is added. A PR will not know the address of a destination ES, unless the ES happens to be in the same cluster as the PR. To solve this problem, a SURAN address server allows a source PR to learn the hierarchical address of a destination ES's PR. Packet radio is *not* an internetwork technology; a set of PRs behaves as a single subnetwork. An ES and its attached PR can move around that (possibly large) subnetwork, but cannot relocate to other parts of an internetwork.

2.6 Commercial Telecommunications Offerings

2.6.1 Cellular Telephony

An important example of a technology that provides wireless physical mobility is cellular telephony. Until recently, however, mobility was limited primarily to a subscriber's home metropolitan area. A cellular subscriber was not able to receive calls while away from the home area unless the caller was aware of the subscriber's current location, and dialed a special access number. Also, a call was terminated if a subscriber happened to cross a boundary between two metropolitan areas. Now, some cellular providers are beginning to make a roaming capability "automatically" available to their customers [25]. A call to a cellular subscriber is always routed to the mobile exchange in the subscriber's home area. If the subscriber is roaming, and the mobile exchange has been informed of the subscriber's current location, a call is set up to the mobile exchange at the subscriber's current location. For the duration of the call, traffic is routed via the home area. To handle a subscriber who roams to one or more new areas *during* a conversation, it appears that the cellular providers plan to "cheat." Instead of making use of the long distance networks, preliminary information [12] indicates that providers are going to install direct links between cooperating mobile systems. In fact, the EIA Interim Standards on Intersystem Handoff [17] and Automatic Roaming [18] assume the presence of direct links. This essentially eliminates the wide-area network component, and will result

in traffic flowing via a potentially circuitous multi-hop path, one hop per area boundary that the subscriber has crossed.

In its current form, cellular telephony has a number of characteristics that limit its utility for wide-area data internetwork mobility. First, as described above, only limited capabilities exist for handling a subscriber who roams away from the home area. Also, a subscriber has no control over link characteristics; the way in which long distance and inter-system links are strung together will be very dependent upon the current location and previous path of the calling and called parties. Finally, current generation cellular technology in the United States provides a voice-grade analog channel, and this offers a limited bandwidth for data applications. Under ideal conditions, which may not often be reached in practice, a total of 14 K bits per second would be available to be shared among the two communicating parties.

2.6.2 Personal Number Calling/800 Service

Bellcore has proposed a concept known as Personal Number Calling (PNC) [15, 16] that would provide a form of logical mobility. With PNC, a telephone number would be assigned to an individual rather than to a line. Thus, a single “personal” telephone number could be used to reach an individual regardless of whether he or she happened to be at home, at work, or in an automobile. It is likely that PNC would be implemented using a technique similar to that developed for 800 Service. In North America, the “800” prefix is used to indicate

a toll-free number. In years past, 800-numbers were routed to special telephone lines, but this is no longer the case. Instead, a database is queried in order to translate the 800-number to a regular, non-800, number; the result of the translation can vary, for example, depending on the time of day [53]. One could imagine making a similar translation for a personal telephone number, yielding the telephone number of the individual's current physical location. In fact, Bellcore has developed a high-performance database architecture, called *Datacycle* [6], that would be capable of providing real-time name service for a network of 280 million users. Such a database might, for example, make it possible to assign a personal telephone number to each telephone subscriber in the United States.

Although possibly useful for logical mobility, the PNC/800 Service approach is not suitable for physical mobility in datagram-based internetworks. In particular, the approach does not handle a relocation by one of the end-points after a connection has been established. Also, it is not certain whether it would be practical to update the database frequently enough to track the movement of a mobile ES.

2.6.3 ARDIS and RAM Mobile Data

ARDIS [1, 2, 65] is a wireless data network in the United States that is owned and operated by Motorola. ARDIS is designed for the transmission of short,

bursty, interactive traffic. Message delivery is guaranteed, and occurs approximately 4 to 10 seconds after transmission. A message can be sent between two subscriber units, or between a subscriber unit and a customer-owned computer. Full-duplex bandwidth over the wireless links is currently 4800 bits per second. Selected geographic regions are being upgraded to 19,200 bits per second, in order to increase channel capacity [65]. The ARDIS network is hierarchically organized and centrally routed. There are approximately 35,000 subscribers, 1350 base stations, 30 network control processors, and 2 message switches (the second message switch is for redundancy). A group of base stations is connected to a network control processor, and each network control processor is connected to both message switches. When a subscriber unit is powered on, it communicates with a base station, which in turn informs a network control processor of the subscriber's existence. The network control processor then informs the message switches that the subscriber has appeared. Thereafter, message routing is simply a matter of table lookup at the message switches. Although the ARDIS network is impressive, its centralized architecture and limited scalability do not make it a suitable model for ES mobility support on datagram-oriented internetworks.

RAM Mobile Data [49] is the United States licensee of Ericsson's MOBITECH technology. Like ARDIS, RAM Mobile Data operates a wireless data network in the United States; the company offers similar networks in various other countries. A message can be sent between two mobile terminals, or between a mobile

terminal and a customer-owned computer; bandwidth over the wireless links is 8000 bits per second. A feature provided by RAM Mobile Data is that, if a mobile terminal is temporarily inaccessible, messages destined for that terminal can be stored by the network and delivered at a later time. The network is hierarchically organized and hierarchically routed. Mobile radios communicate via a base station, and the base stations in a service area are connected to a local switch. Local switches are connected to a regional switch, which in turn is connected to the main exchange. Regional switches (except those that are used to interconnect multiple local switches within a single service area) are operated by a long distance provider rather than by RAM Mobile Data itself. Messages are “turned around” at the lowest hierarchical level shared by the source and destination of a message; for example, if the source and destination of a message are served by the same base station, that base station will handle message delivery. Each level of the switching hierarchy maintains a registry of all subscribers that are currently located at lower levels of the hierarchy. Subscriber addresses are divided into three classes, with varying degrees of geographic mobility: local, regional, and world-wide. A subscriber that is assigned a world-wide address is able to obtain service from any MOBITECH system. RAM Mobile Data’s design appears to scale well, as long as most subscribers roam at the lower levels of the switching hierarchy. Nevertheless, as with ARDIS, the strict hierarchical architecture of the network makes it impractical as a model for mobility support on datagram-oriented internetworks.

Chapter 3

Network Layer Addressing

3.1 Address Types

To implement mobility at the network layer, the first task is to choose a network layer addressing scheme. Historically, ES addresses on wide-area data communication internetworks have contained some amount of topological information. In the case of the TCP/IP Internet, for example, the Internet address contains a network number [46]. Such embedded information makes the task of routing significantly easier, in that transit ISs need not keep track of each and every ES. As the size of an internetwork grows, it makes sense to introduce increasing amounts of topological information into the ES address format, arranged in a hierarchical fashion. In a hierarchical system, routing can be done progressively (i.e., piece-by-piece), and routing information can be partitioned. Hierarchical addressing enables, but does not require, the use of hierarchical routing; when

appropriate, routing can be performed directly between two branches of a hierarchy. The international telephone numbering plan is a successful example of wide-area hierarchical addressing and routing. A similar scheme was adopted for the AT&T Databit [23], except that address components are text strings separated by “/” characters. Unfortunately, having a topology-dependent address does introduce one significant limitation: if an ES relocates, its address must change. An alternative is to assign a flat address to each ES; such an address uniquely identifies an ES, and would remain constant as an ES relocates. Although a flat address could also be called a name, it is customary (and less confusing) to speak of addresses rather than names when dealing with network layer addressing and routing.

There are two design questions relating to network layer addresses for ESs:

1. When sending traffic to an ES, what type of network address does a source present to the internetwork? Possibilities lie on a continuum from flat address to topology-dependent, hierarchical address. If the network address is flat, it stays the same as a mobile ES relocates; if hierarchical, there are two possibilities:
 - (a) The hierarchical address identifies the ES’s current location, and changes every time that the ES moves.
 - (b) The hierarchical address identifies the ES’s home location; if necessary, the home address is translated to the hierarchical address of the ES’s current location (cellular telephony, Deering [14], Ioannidis,

Duchamp, and Maguire [27, 28], IETF [54], Wada, Yozawa, Ohnishi, and Tanaka [64], TUBA [21], and CDPD [10]). Two variations are Teraoka [59, 60, 61, 62], where a datagram header can optionally carry the hierarchical address of an ES's current location in addition to the address of the home location, and Perkins [43], where a datagram header can optionally specify a loose source route to the destination ES.

2. On what network address does the internetwork route internally? Even if a source presents a flat address, it might be translated to a hierarchical address at the entry point to the internetwork (Personal Number Calling, 800 Service). Alternatively, the internetwork may route directly on the flat address (Sincoskie and Cotton [55]).

From the perspective of minimizing ES complexity, flat addresses are good in that the network address associated with a mobile ES never changes; that address can be recorded, and used whenever communication with the mobile ES is desired. Also, assuming that the flat address appears in the source address field of a datagram, the address can be used to identify the ES from which the datagram originated¹. Unfortunately, internetworks employing flat addresses (either externally or internally) tend to scale poorly in terms of size and degree of ES mobility. The problem is that there must be some means (such as a

¹Since the possibility of a deliberate lie on the part of the source ES or the internetwork exists, authentication at a higher protocol layer may be desirable.

database lookup or active search) for the internetwork to dynamically determine the current location of any ES. With increasing frequency of relocation, caching becomes less and less effective as a means to reduce the query (or search) load, and this is compounded by the fact that a flat address does not contain any routing hints.

A topology-dependent, hierarchical address is certainly easier for an internetwork to route efficiently; however, problems exist if ESs are allowed to be mobile. A hierarchical address identifies a Subnetwork Point of Attachment (SNPA), rather than an ES itself. Thus, datagram misdelivery cannot be detected by the internetwork: the datagram will arrive at the specified SNPA, but it is possible that the intended recipient is no longer there. Similarly, the internetwork is unable to reliably store forwarding pointers for mobile ESs, since a network address does not contain an identification of the intended ES. Consider the case where an SNPA is shared among a number of mobile or casually connected ESs. If a particular ES has recently been present at that SNPA, this mapping might have been cached by various ESs, or perhaps stored in a database. When the ES relocates, the cached mapping information can at least temporarily lead to mis-addressed datagrams. Teraoka solves the problem of ES identification by optionally including the hierarchical address of an ES's current location in a datagram's header, along with the hierarchical address of the ES's home location. ISs and ESs that process the datagram can cache the (potentially out-of-date) binding between current and home address. However,

even Teraoka's proposal is not immune to misdelivery problems. If a mobile ES roams to a non-VIP-capable portion of an internetwork, the home and current destination address fields of any datagram being forwarded to the ES must be swapped, so that the ISs can correctly route the datagram. A datagram will be delivered to whatever ES happens to be resident at the SNPA identified by the current destination address field of the datagram. If the receiving ES is VIP-capable, it can check for an incorrect value in the home destination address field, but if a mismatch is detected, the only reasonable option for the ES is to discard the datagram. If the receiving ES is not VIP-capable, misdelivery will not be detected.

A hierarchical network address can also cause problems for ESs that communicate with a mobile ES. The network address is used as part of the transport connection end-point identifier by both the Transport Control Protocol (TCP) [47] and the ISO Transport Protocol Class 4 (TP4) [31]. A correspondent's network address is assumed to remain constant for the lifetime of a transport connection; if datagrams start arriving from a different network address, they will be rejected. A solution employed in various forms by the mobility schemes for the TCP/IP Internet is to carry the original network address along with datagrams transmitted by a mobile ES, and to present that address to the receiving ES's transport layer. An alternative would be to develop a protocol that would allow a transport connection end-point to be rebound to a different network address. In general, having a network address identify an SNPA rather

than an ES can lead to authentication problems, since the transport layer can never be certain as to the source of a datagram.

3.2 The Hybrid Network Address

Although flat and hierarchical network addresses both have positive aspects, they have significant drawbacks as well. What is needed is a network layer address type that combines the best features of both; we will term this a *hybrid* network address. A hybrid network address is composed of hierarchical components (that partially specify an ES's physical location) plus a unique ID. When a mobile ES relocates, the hierarchical components of the address change, but the unique ID remains constant. The hierarchical components can be used by the upper levels of the routing hierarchy for efficient routing, while the unique ID is used for ES identification and flat routing at (or near) the bottom of the routing hierarchy. The internetwork can also use the unique ID as a lookup key for accessing mobile ES forwarding information.

Consider a North American telephone number:

608-262-1234

The “608” area code indicates that the telephone is in southwestern Wisconsin, and the “262” exchange further specifies that the telephone is located in Madison. “1234” simply selects one of the 10,000 telephone lines at the “262” exchange; it does not identify the telephone (i.e., ES) itself. The ES that is currently connected to that line may not be the ES that the calling party intended

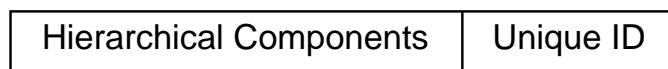


Figure 3.1: Basic Hybrid Address

to reach. In a hybrid addressing scheme, the “1234” line identifier would be replaced with a unique telephone ID, for example “987654321”, and the telephone number would have the following structure:

608–262–987654321

As before, an incoming telephone call can still be routed efficiently to Madison, Wisconsin. Now, however, it is possible for the Madison telephone exchange to identify the actual ES that the caller intended to reach, and potentially to forward the call if that ES has relocated. Of course, such a telephone number would be inconvenient for a human being to remember and dial, but this does not pose a problem for inter-computer communication. Figure 3.1 illustrates the basic hybrid address. A limited form of the hybrid address appeared in DEMOS/MP [48], in the context of process migration; a process address contained a unique ID and an identification of the last known machine that hosted the process. Fowler [22] extended the DEMOS/MP format to include a timestamp. In both DEMOS/MP and Fowler’s work, message delivery is guaranteed as long as the process or object still exists; if necessary, a message will be forwarded one or more times. The Xerox Network System [13] uses a 48-bit flat internet address, along with a network number that is used as a routing hint; the network number, however, is not considered to be part of the internet address. Finally,

Internet addresses used with the Network Reconstitution Protocol [26] contain a unique IS ID, and a unique ES ID; only the IS ID changes as an ES relocates.

Unlike the case of a strictly hierarchical address, if a datagram containing a hybrid address ends up at the wrong place, there are two possibilities:

1. If a forwarding pointer is available, the datagram can be forwarded, and a redirection can be sent to the traffic source.
2. If forwarding information is not available, an error can be returned to the traffic source, and the source can take appropriate action.

An internetwork implementation could attempt to guarantee that forwarding pointers are maintained at all locations that a mobile ES has visited, in which case Option 2 would not be needed. However, given that datagram-based internetworks offer only best-effort delivery, a traffic source must always be prepared to deal with indications that a correspondent ES is unreachable. Thus, as long as a means is provided for a traffic source to locate a currently valid network address for a mobile ES, it is reasonable to allow Option 2 as an exceptional case.

Over the lifetime of a mobile ES, and perhaps over the lifetime of a single transport connection, the ES's hybrid network address may change. The hierarchical components will change to reflect the current location of the mobile ES, though the unique ID will remain constant. Thus, it is useful to have a Location Sequence Number (like Fowler's timestamp), either as part of a hybrid address,

Hierarchical Components	Location Seq No	Unique ID
-------------------------	-----------------	-----------

Figure 3.2: Hybrid Address with Location Sequence Number

Partial Destination Identifier	Location Seq No	Unique ID
--------------------------------	-----------------	-----------

Figure 3.3: General form of a Hybrid Address

or carried along with the address, so that the newest address can be unambiguously identified. Figure 3.2 illustrates the hybrid address with the addition of a Location Sequence Number (LSN). A “may be mobile” flag that indicates whether an address is likely to change is also useful. If ISs or correspondent ESs can identify that a given ES is *fixed* (non-mobile), it might allow for simpler address management in the network layer.

The “hierarchical components” field of a hybrid network address does not necessarily need to be routed hierarchically. In fact, there is not even a need for the address components to have a hierarchical structure; for example, a flat address could be used to identify the destination’s “neighborhood.” Figure 3.3 shows the general form of a hybrid address, with the hierarchical components field replaced by a Partial Destination Identifier (PDI). The hybrid network address appears to be promising for wide-area mobility; however, by itself, an addressing scheme is *not* a mobility scheme. In the following chapter, we develop a mobility system that uses this addressing format.

Chapter 4

A Method to Support Mobile End Systems

In this chapter we describe a mobility scheme, incorporating hybrid network addresses, for the ISO connectionless routing architecture¹.

4.1 ISO Routing Architecture

The ISO connectionless routing architecture consists of a 3-level hierarchy. At the bottom level of the hierarchy are ESs. At the next level, an *area* is defined as a set of ESs and Level 1 (L1) ISs, and at the top level, a *routing domain* is defined as a set of areas. The hierarchy is illustrated in Figure 4.1. Routing within a routing domain is specified by the Intra-Domain Routing Information

¹Though some liberties have been taken with NSAP Address and PDU formats.

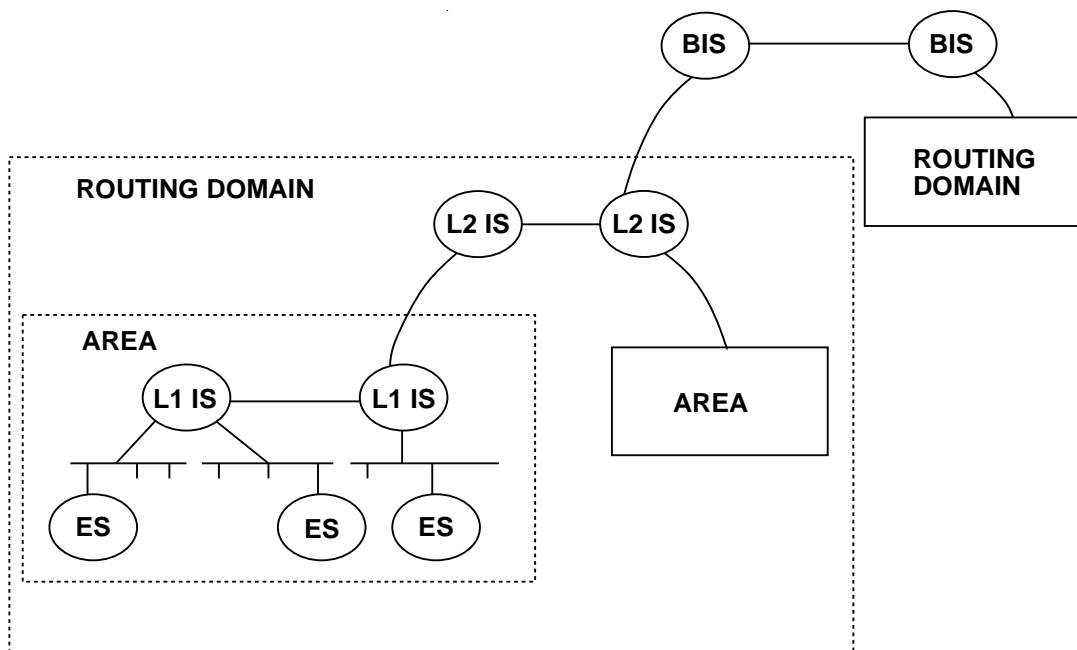


Figure 4.1: ISO Connectionless Routing Architecture

Exchange Protocol [29]. A Network Service Access Point (NSAP) address for use with the ISO routing protocol has the structure shown in Figure 4.2. The Area Address identifies the area, while the ID field identifies an ES within that area. The SEL (selector) field identifies a network service user at the ES. The identification of a routing domain is less explicit: a collection of areas that have the same Area Address prefix (i.e., a certain number of identical high-order bits) can be administratively defined as a routing domain. Every ISO NSAP address begins with an Authority and Format Indicator (AFI) and an Initial Domain Identifier (IDI); these are considered to be part of the Area Address. Together, the AFI and IDI specify an NSAP addressing domain, and indicate the syntax of the remainder of the address [32].

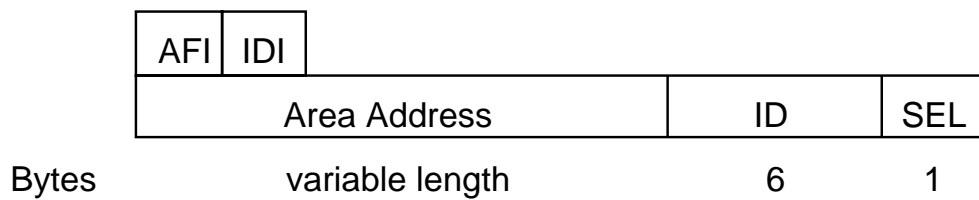


Figure 4.2: NSAP Address Structure

An area can itself be a reasonably large internetwork; the component subnetworks are tied together by L1 ISs, which are responsible for intra-area routing. Via the ES-IS Protocol [34], an ES establishes a relationship with an L1 IS that is on the same subnetwork as the ES; the ES and IS periodically send Hello PDUs to each other. A L1 IS directly keeps track of all ESs on its subnetwork, and is able to route datagrams to all ESs in its area. By default, an ES sends all datagrams to the local L1 IS; if the destination ES is on the same subnetwork as the source ES, the IS will send a Redirect PDU to the source ES. If the source and destination of a datagram are in the same area, but on different subnetworks, the datagram will be routed via L1 ISs. If an L1 IS receives a datagram destined outside its area, the IS will forward the datagram to a Level 2 (L2) IS.

Routing between areas (but within a routing domain) is handled by L2 ISs. When making routing decisions, L2 ISs examine only the Area Address portion of an NSAP Address, not the ID field. Note that the L2 ISs use the link-state method of exchanging routing information, as do the L1 ISs. An L2 IS that has a link to another routing domain is known as a Border Intermediate System (BIS). An inter-domain routing protocol [30] is employed to exchange information concerning the Area Address prefixes that can be reached via the

inter-domain links. A BIS advertises external routes to those prefixes to the L2 ISs with which it exchanges routing information.

4.2 Design Principles

A mobility scheme should support ES mobility across a community of ISO routing domains. An ES should be able to relocate to another area within the same routing domain and also be able to relocate to an area in another routing domain. Two properties should be preserved across a relocation:

- After relocating, a mobile ES should be able to send and receive new traffic. This means that datagrams from correspondent ESs must be able to reach the mobile ES at its new location.
- Preexisting transport connections should continue to operate, assuming a reasonably short disconnection interval.

A mobility solution could potentially be implemented at any level of a routing hierarchy. However, in the ISO connectionless routing architecture, the ID portion of an NSAP address is examined only when routing within an area. Inter-area and inter-domain routing are based strictly on the Area Address. Since a mobility solution will almost certainly involve the ID field, such a solution is best implemented at the lower-most levels of the routing hierarchy; namely, in the ESs and L1 ISs. An advantage of this approach is that only the

areas that wish to support mobility need to be modified; the rest of the routing hierarchy can remain unchanged.

4.3 Network Layer Addressing

As specified by the Intra-Domain Routing Protocol, the ID portion of an NSAP address only needs to be unique within an area. In fact, developers are specifically discouraged [11] from making engineering decisions that rely on the ID being globally unique; from the perspective of providing mobility, having a non-unique ID is certainly a bad idea! At least for NSAP addresses used in the United States (or a recognizable subset of those addresses), it should be possible to declare that IDs will be unique. Fortunately, the defacto practice is to use the ES's Ethernet address as the ID, and such IDs are globally unique. So, a mobile ES can be assigned a permanent unique ID, which will remain the same as the ES relocates. This ID will be termed a Mobile Unique ID (MUID). It is useful for ESs and ISs to be able to identify that a given NSAP address belongs to a mobile ES, so that special processing can be avoided for fixed ESs. We have chosen to add a “may be mobile” bit to the NSAP address format; a less satisfactory alternative would be to identify such NSAP addresses by their AFI/IDI prefix.

A mobile ES will retain the same MUID as it relocates, but what about the Area Address? The hierarchical structure of an NSAP address strongly associates an Area Address with a particular routing domain and area. For a

mobile ES to retain the same Area Address as it relocated across areas, that ES would have to be assigned a unique Area Address (distinct from that assigned to any physical area). To support mobility within a routing domain, the L2 ISs would have to maintain a routing table entry for the mobile ES's Area Address; for inter-domain mobility, the BISs would have to maintain a routing table entry. Unfortunately, this would defeat the information-hiding property afforded by hierarchical addressing and routing schemes, where detailed information about object location is kept only at lower levels of the routing hierarchy. Thus, it is not practical for a mobile ES to retain the same Area Address (and hence NSAP address) as it relocates.

Since the Area Address component of a mobile ES's NSAP address will change as the ES relocates, it is important to be able to distinguish between different instances of the NSAP address. Each time that a mobile ES relocates between areas, it increments a locally stored counter known as the Location Sequence Number (LSN), and the LSN is included as a field in the NSAP address. Most ESs have a nonvolatile configuration memory, and this can be used to store the LSN. Nevertheless, should an ES somehow "forget" its LSN, we describe a recovery procedure in the next section.

The sequence number space is managed in a circular fashion. At any given point in time, half of the LSN values are considered to be older than the current value, and half are newer. Circular treatment of the LSN avoids the need for an abrupt transition when the LSN counter overflows to zero. An LSN value

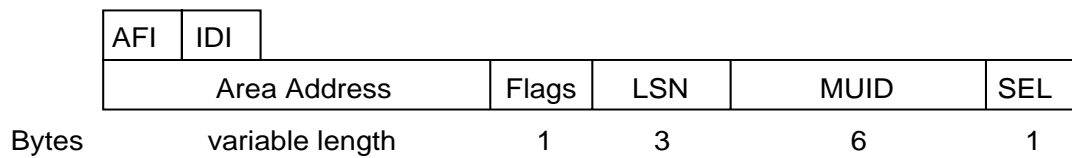


Figure 4.3: NSAP Address with LSN

is unambiguous until the sequence number space has wrapped halfway around from the point at which the value was assigned. A mobile ES's NSAP address does not change during all relocations, only those that cross an area boundary, and this moderates the rate at which the LSN increments. If three bytes are allocated for the LSN, and the LSN increments at an average rate of once per second, the sequence number space will wrap halfway around approximately every 90 days. If ESs relocate between areas at a more realistic (i.e., slower) rate, or if a faster wrap rate is permissible, fewer bits can be allocated for the LSN. Implications of the LSN wrap rate are mentioned at appropriate points in Sections 4.4 and 5.1.

Although modifying the standard NSAP format might present some obstacles with respect to the use of standard protocols, for purposes of experimentation a new address format that includes an LSN can be defined. Figure 4.3 shows the new NSAP format, which also includes a flags byte. The “may be mobile” indication is assigned to one of the bits of this byte.

As mentioned in the previous section, the SEL field of an NSAP address identifies a network layer user at the ES associated with the NSAP address. This field is ignored by ISs when making routing decisions and will not be

changed by our method.

4.4 Mechanism

Since the NSAP address of a mobile ES will change as the ES relocates, correspondent ESs need some method to learn the NSAP address currently associated with a mobile ES (unless the mobility scheme relies completely on forwarding by ISs). One possibility would be to store this information in a global dynamic database such as the X.500 Directory Service [9], and to update the ES \rightarrow NSAP address binding each time that a mobile ES relocates to another area. However, X.500 is explicitly not designed for storage of rapidly changing information, and in general, it may not be feasible to optimize a global database to support both a high query rate and a high update rate. Since an L1 IS already keeps track of all ESs on its subnetwork, it should be straightforward to have an L1 IS keep track of mobile ESs that are normally on its subnetwork, but have *roamed* to another area. At a given point in time, a mobile ES will be associated with a particular *home* area. This binding will be relatively static, and hence suitable for storage in a global database such as X.500. When a mobile ES moves between areas:

- It updates a forwarding pointer at the home area. This pointer remains in force until explicitly changed or deleted by the mobile ES.

- As a performance optimization, it leaves a forwarding pointer at the previous area. The forwarding pointer cache at an IS is bounded in size, and managed in an LRU fashion. A number of alternative methods *could* be used to bound the number and lifetime of pointers maintained at areas that a mobile ES has visited:
 - The mobile ES could explicitly delete the forwarding pointer the next time that it relocates.
 - The mobile ES could be required to periodically send a Hello PDU to the previous area. After an interval of non-communication, the previous area could delete the forwarding pointer.
 - As in Sincoskie and Cotton [55], the forwarding pointer could be deleted after an idle timeout (during which no traffic had been forwarded to the mobile ES).
 - The entire cache of forwarding pointers at previous areas could be flushed periodically; this observation is due to Elliott [19].

Even if a forwarding pointer for a mobile ES remains at a previous area after the mobile ES's LSN has wrapped halfway around, it will not interfere with correct operation of the mobility algorithm.

Correspondent ESs that are in active communication with the mobile ES at the time of a relocation learn the ES's new NSAP address, either as a result of receiving traffic from the mobile ES, or via a Mobile NSAP Rewrite message

received from an L1 IS at the previous area², and those correspondents update their address caches.

When a correspondent begins to send CLNP datagrams to a mobile ES, it will often be the case that the mobile ES will be resident in an area other than that specified in the Area Address portion of the destination NSAP address. When the datagram arrives at the (incorrect) area, one of two things will happen:

1. If the area has a forwarding pointer for the mobile ES, an L1 IS in the area will forward the datagram. Specifically, the IS will *rewrite* (change) the Area Address and Location Sequence Number fields of the destination NSAP address specified in the CLNP datagram, and forward the datagram via an L2 IS to the correct area. Also, to inform the traffic source of the mobile ES's current NSAP address (Area Address plus LSN plus MUID), the L1 IS will send a Mobile NSAP Rewrite message to the traffic source.
2. If the area does not have a forwarding pointer for the mobile ES, an L1 IS in the area will send a Mobile NSAP Unreachable message to the traffic source. If an unreachable diagnostic is received, the source will clear its cache entry for the mobile ES, and attempt to send to the mobile ES's current home NSAP address.

After an IS rewrites a datagram's destination NSAP address, it must recalculate the datagram's header checksum. As a result, even if the IS were to have corrupted the datagram header due to faulty processing, the header would still

²The Mobile NSAP Rewrite message is a performance optimization.

carry a valid checksum upon exit from the IS. With standard CLNP and IP, an IS must adjust the value stored in the Lifetime field of every datagram header, and this requires recalculation of the checksum. Thus, the mobility algorithm does not add any *new* vulnerability to undetectable header corruption. If the concern is specifically that the destination MUID field might be altered due to incorrect IS processing, one could imagine augmenting the datagram header to carry a checksum for that field. However, the benefit would be marginal.

When a mobile ES reconnects after a relocation, it learns its new Area Address via the ES-IS Protocol by receiving a Hello PDU from an L1 IS on the local subnetwork. The ES will inform the home area and previous area of the ES's new NSAP address. While in an area, a mobile ES behaves normally, maintaining an ES-IS relationship with one of the L1 ISs in the area. However, when the mobile ES roams to another area, it will maintain a *virtual* ES-IS relationship with the previous area, as well as with the home area. Several new logical functions are needed for the ES-IS Protocol:

- When an ES arrives at an area, it does one of the following:
 - Registers as a home mobile ES. This must be done in concert with an update of the global database:
 1. Register at new home area
 2. Update database
 3. Unregister at old home area

- Registers as a roaming (transient) mobile ES. This relationship can time out or otherwise terminate without an explicit “unregister” being needed.

Note that requiring the ES to be physically present in an area in order to register as a home or transient mobile ES provides a limited form of security.

- When a mobile ES relocates to another area, it continues to maintain an ES-IS relationship with the home area and the previous area. However, the ES-IS Protocol is defined as a *subnetwork*, rather than internetwork, protocol. To send an ES-IS PDU to an L1 IS in the home or previous area, the mobile IS encapsulates the PDU inside a CLNP datagram, and addresses that datagram to the IS’s Network Entity Title (NET).
- (Enter Quiescent State) Before leaving an area, the mobile ES signals the local L1 IS that it is about to disconnect, and this signal is acknowledged by the IS. If the disconnection interval is brief, a possibility would be for the L1 IS to buffer traffic during the period that the mobile ES is disconnected. Limitations at the link layer might make it impossible for a mobile ES to signal the local IS before leaving an area. In this case, traffic for that mobile ES will be dropped until the ES-IS protocol detects that the mobile ES is no longer present, or the IS receives a Reconnect message from the mobile ES at its new location.

- (Mobile ES Reconnect) When a mobile ES reconnects at a new area, it informs the home and previous areas that it has reappeared. If an L1 IS at the previous area has buffered datagrams during the disconnection interval, it will deliver the datagrams at this time, after updating each datagram's destination NSAP address. A Reconnect message is acknowledged.
- (End ES-IS relationship (Unregister)) After a mobile ES registers as a home mobile ES in another area (and updates the global database), it unregisters at the old home area. An Unregister message is acknowledged.

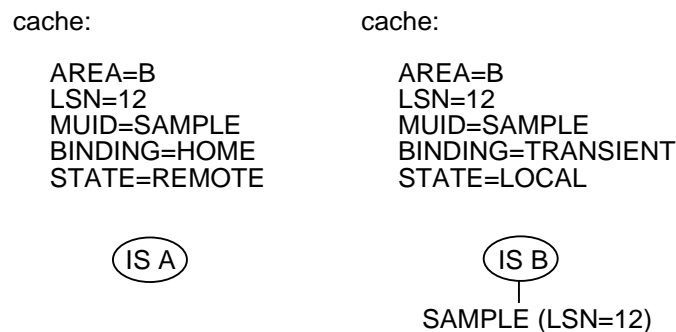


Figure 4.4: Initial Cache Contents; ES SAMPLE's Home is Area A

Figures 4.4–4.6 provide an example of a mobile ES SAMPLE moving from area B to area C; the ES's home is area A. At each step, the ISs' forwarding pointer cache entries for ES SAMPLE are shown. For simplicity, shorthand area names are listed in the AREA fields; in reality, these fields would contain full, hierarchically structured, Area Addresses. Together, an Area Address, LSN, and MUID form an NSAP address.

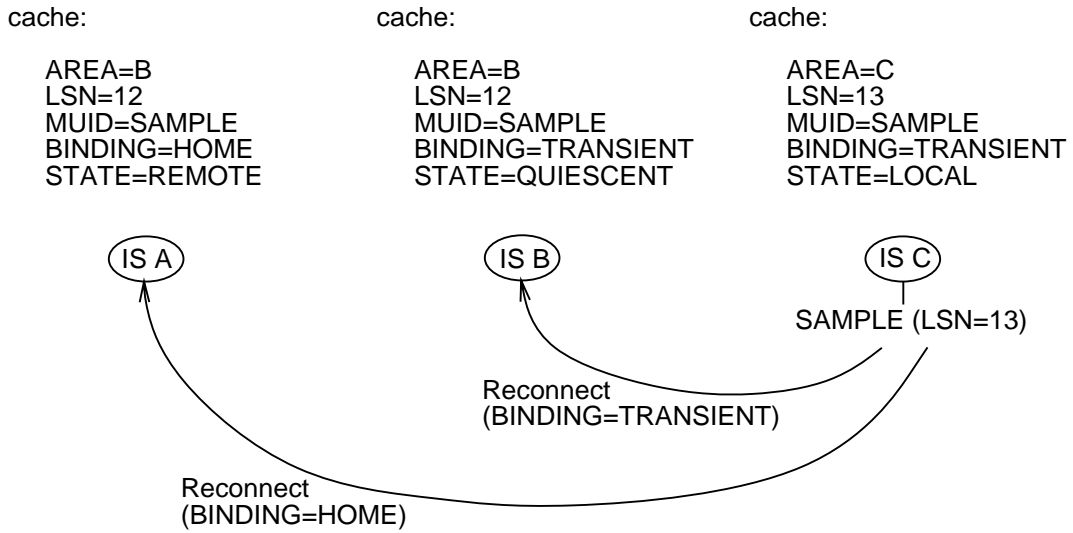


Figure 4.5: ES Moves to Area C, Sends Reconnect Messages to Areas A and B

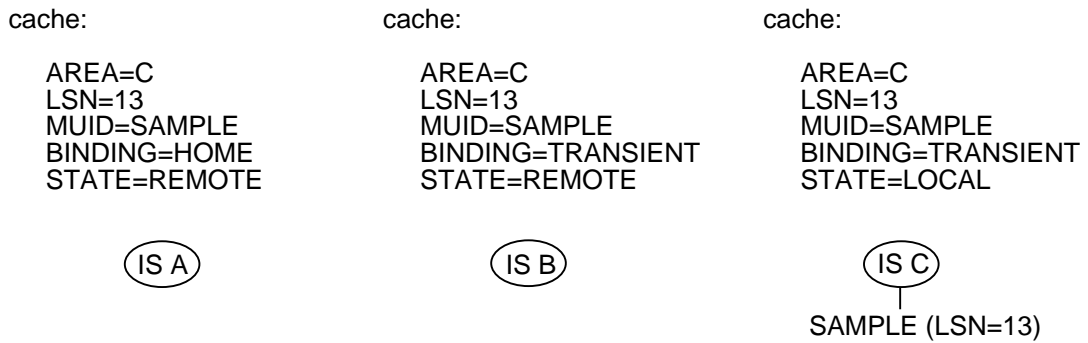


Figure 4.6: Cache Contents After Reconnect Messages Have Been Processed

A mobile ES updates its home NSAP address in the global database whenever the ES designates a new home area. A mobile ES also should update its home NSAP address when the ES's LSN is about to wrap halfway around from the point at which the last database update occurred. Otherwise, the home NSAP address stored in the database will appear to be newer than the ES's current NSAP address. The update procedure is as follows. The next time that the mobile ES relocates between areas, the ES increments its LSN twice. The value after the first increment is used as the LSN for the updated home NSAP address; the remaining fields of the home NSAP address are unchanged. The value after the second increment is used to construct the ES's current NSAP address.

An ES must “remember” its LSN, even across reboots. Although the LSN can be stored in nonvolatile configuration memory, it is conceivable that the LSN could somehow be forgotten. In such a case, a mobile ES could consult the global database to learn its home NSAP address. Then, to learn its current (before the reboot) NSAP address, including the LSN, the mobile ES could contact an L1 IS in the home area. Our current mobility specification does not define a message type for this transaction.

Each ES that wants to fully support communication with mobile ESs will maintain a mobile address cache at the network layer. The address cache maps MUIDs→full NSAPs:

- The cache is populated by queries on the global database and source NSAP addresses of incoming datagrams from mobile ESs.

- The cache is updated by Mobile NSAP Rewrite messages and source NSAP addresses of incoming datagrams from mobile ESs.
- Cache entries are cleared by LRU management, Mobile NSAP Unreachable messages, and possibly by idle timeouts.

When an ES receives a Mobile NSAP Unreachable message, it will clear the address cache entry for the corresponding MUID, and requery the global database. Note that if the home area is allowed to change, it *must* be possible to requery the global database by presenting an MUID, to learn the new home NSAP address. To support such queries, either the MUID can have a hierarchical structure, or one can be imposed for lookup purposes. Since the home NSAP address is likely to change infrequently (if at all), an option for ESs would be for each cache entry to record the home NSAP address associated with the MUID, as well as the current NSAP address. This would significantly reduce the frequency at which the global database needs to be queried.

A consequence of allowing mobile ES NSAP address information for the mobile address cache to come from multiple sources is that contradictory (or out-of-sequence) information may be received. The LSN field allows these conflicts to be resolved.

4.5 Further Design Details

In this section we present further details of the mobility algorithm. The need for certain of these details became apparent during the caching study. In particular, the “rewrite by destination only” flag was introduced to fix a problem with out-of-date cache entries at interior ISs. By necessity, presentation of the full rationale is deferred until Section 7.5.

4.5.1 Processing at an IS

When routing a datagram, an IS looks up the destination mobile NSAP address in its forwarding pointer cache. If the cached NSAP address is newer (based on a comparison of LSNs) than that contained in the datagram, the IS rewrites the datagram’s destination NSAP address and increments a “rewrite count” field in the header. An exception to this rule occurs if the datagram has a “rewrite by destination only” flag set. In this case, only an L1 IS in the area specified by the Area Address field of the destination NSAP address is allowed to rewrite the NSAP address and clear the “rewrite by destination only” flag. This feature allows a traffic source to force a route to a specific destination, for example to a home NSAP address: the destination NSAP address is not allowed to be rewritten until after the datagram arrives at the Area Address specified by the sender. When an IS rewrites a destination NSAP address, the IS also sends a Mobile NSAP Rewrite message to the source of the datagram. The Rewrite message contains the new NSAP address, and also a copy of the “rewrite by

destination only” flag from the original datagram.

It is possible, due to a combination of out-of-date cache entries and finite sized caches, that an IS will encounter the following conditions while attempting to route a datagram:

- a) The datagram is addressed to the IS’s area.
- b) The destination mobile ES is not locally connected or marked as quiescent.
- c) A newer cache entry is not available for the destination mobile ES.

In such a case, the IS sends an Unreachable message to the source of the datagram. The Unreachable message contains the destination NSAP address of the undeliverable datagram, and copies of the “rewrite count” field and the “rewrite by destination only” flag from that datagram. Any IS that caches forwarding pointers must examine each Unreachable message that passes by:

- If the IS does not have a cache entry for the NSAP address contained in the Unreachable message, or if the cached NSAP address is newer, the Unreachable message is ignored.
- If the NSAP address contained in the Unreachable is identical to that in the cache entry, the cache entry is deleted.
- If the NSAP address contained in the Unreachable is newer than the corresponding cache entry, the cache entry is deleted *only* if the “rewrite count” field in the original datagram contained a value greater than zero.

The rationale is as follows. A traffic source can forge a destination NSAP address with an arbitrarily large LSN. Such a datagram will be routed to the (forged) destination, and an Unreachable message will be generated. In this case, we do not want legitimate cache entries for the destination mobile ES to be deleted as a result of the mis-addressed datagram. If the original datagram contained a rewrite count of zero, an IS does not trust the NSAP address contained in the Unreachable, and only deletes a cache entry if it exactly matches the NSAP address contained in the Unreachable. With an exact match, the IS knows that the cached NSAP address is actually unroutable. With a non-zero rewrite count, the IS knows that the NSAP address contained in the Rewrite message was rewritten by another IS, and hence can be trusted.

4.5.2 Processing at an ES

When an Unreachable message arrives at an ES, the LSN contained in the referenced NSAP address is compared with that in the corresponding cache entry. If the cache entry has a newer LSN, or there is no cache entry, the Unreachable message is ignored. Otherwise, the cache entry is replaced by the home NSAP address for the mobile ES that is referenced in the Unreachable message. The home NSAP address is obtained from the global database, or from a copy stored along with the cache entry, if such a copy is available. A “do not rewrite” control bit is set for the cache entry; this control bit is present as part of each cache

entry.

If in the process of preparing a datagram to be sent, a **ES** discovers that the cache entry for the destination **NSAP** address is marked “do not rewrite,” the **ES** sets a “rewrite by destination only” flag in the datagram. This ensures that the datagram’s destination **NSAP** address will not be rewritten until the datagram reaches the Area Address specified in the destination **NSAP** address.

Setting the “do not rewrite” bit in a cache entry is straightforward; deciding when to clear that bit is more challenging. If a datagram arrives at an **ES**, the local cache is checked for an entry for the source mobile **ES**. As long as the cache entry is not newer than the datagram’s source **NSAP** address, the cache entry is updated, and its “do not rewrite” bit is cleared.

It is also possible to clear the “do not rewrite” bit upon receiving certain Rewrite messages. After sending a datagram with the “rewrite by destination only” flag set, a mobile **ES** is likely to receive a Rewrite message from the destination’s home location before receiving traffic from the destination itself. However, it is critical to distinguish that Rewrite from other Rewrites with out-of-date information that might still be traveling through the internetwork. Therefore, a Rewrite message includes a copy of the “rewrite by destination only” flag from the datagram that triggered the Rewrite. Thus, if a mobile **ES** receives a Rewrite marked “original was rewrite by destination only,” it can be reasonably confident that the Rewrite is valid. As long as the **NSAP** address contained in the Rewrite is newer than the corresponding cache entry, the entry

is updated, and its “do not rewrite” bit is cleared. If the Rewrite is *not* marked “original was rewrite by destination only,” but the corresponding cache entry has the “do not rewrite” bit set, the Rewrite message is ignored.

4.6 Optimality of Routes

Datagrams sent between a pair of communicating ESs will nearly always be routed via an optimal path³, even if one or both of the ESs are mobile. The only exceptions are a brief interval as communication begins, and a brief interval after one of the ESs relocates.

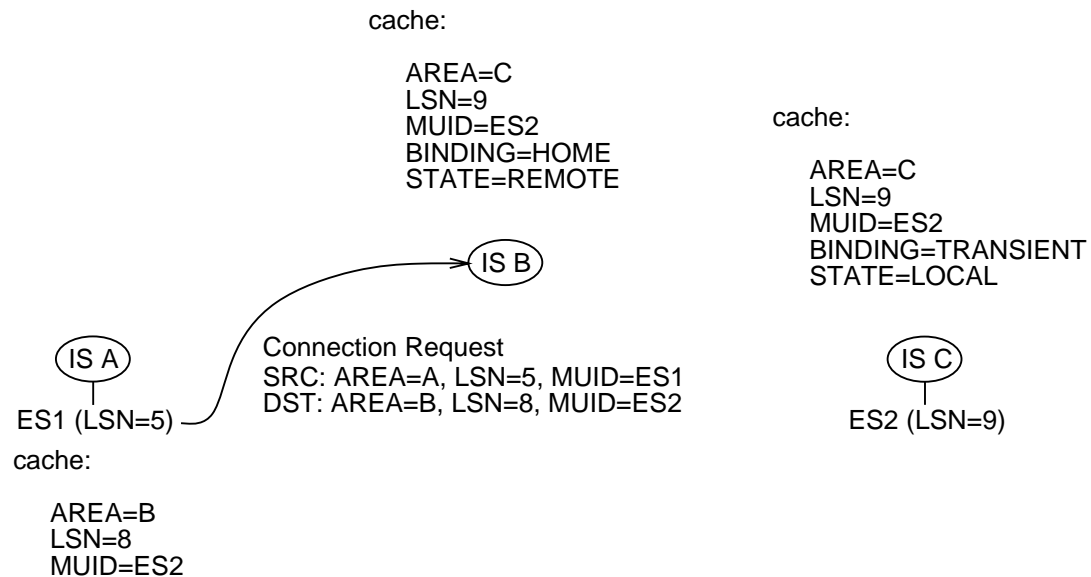


Figure 4.7: ES1 Sends a Connection Request to ES2’s Home NSAP Address

³An optimal path for a datagram is one that is equal in length to the path that the datagram would travel if both the source and destination ES are non-mobile.

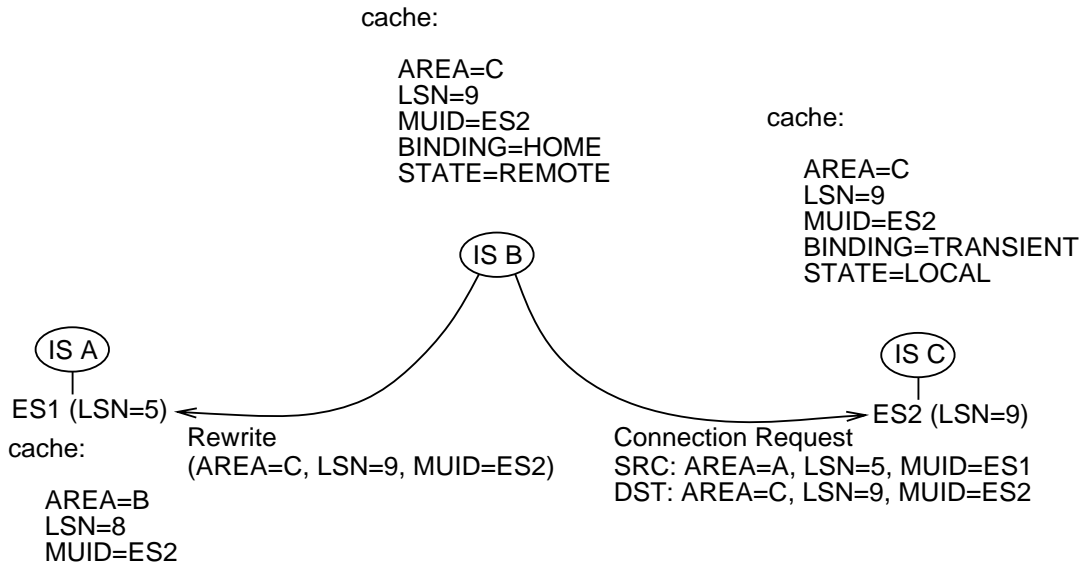


Figure 4.8: IS B Forwards the Connection Request and Sends a Rewrite to ES1

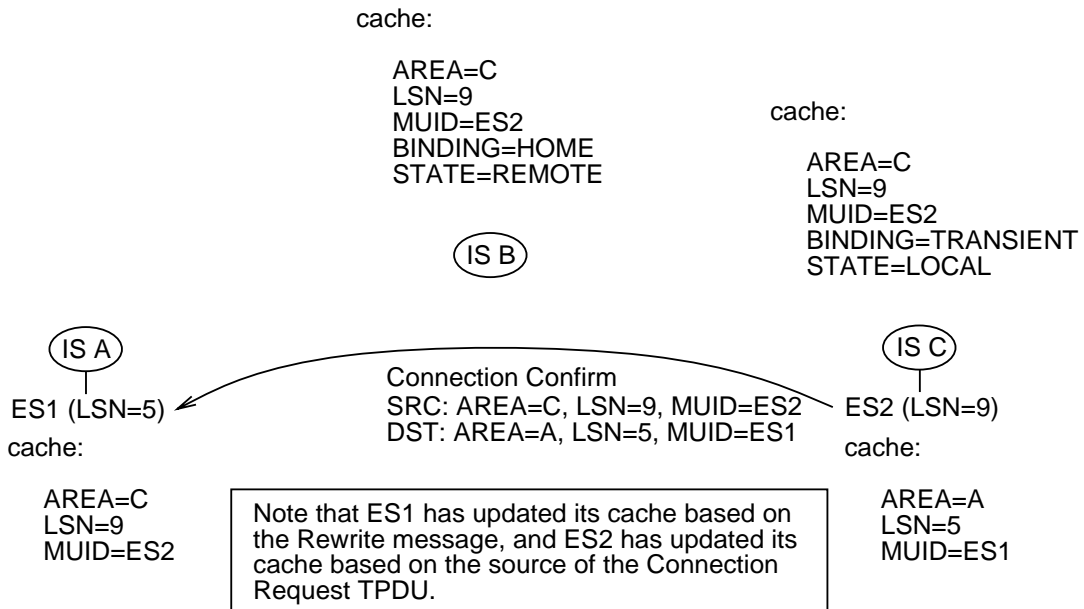


Figure 4.9: ES2 Sends a Connection Confirm to ES1's Current NSAP Address

As an example, consider an ES ES1 that opens a transport connection to ES ES2. Assume that ES1 has cached ES2's home NSAP address; this information likely was obtained from a global database. Figures 4.7–4.9 illustrate the message exchanges that occur. In Figure 4.7, ES1 sends a Connection Request TPDU to ES2's home NSAP address; the TPDU is routed to an IS in ES2's home area. As shown in Figure 4.8, the IS rewrites the destination address of the Connection Request and forwards it onward to ES2. The IS also sends a Rewrite message to ES1, in order to inform it of ES2's current NSAP address. Upon receiving the Rewrite message, ES1 updates its cache entry for ES2, and upon receiving the Connection Request, ES2 creates a cache entry for ES1 (or updates an existing entry), based on the datagram's source NSAP address; Figure 4.9 shows the updated cache entries. Also in Figure 4.9, ES2 sends a Connection Confirm TPDU to ES1; since ES2 is able to specify ES1's current NSAP address, the datagram is routed directly to ES1. From this point on, until one of the ESs relocates, datagrams sent between ES1 and ES2 will be routed via an optimal path. Note that even if the Rewrite message had not been received by ES1, ES1 would have learned ES2's current NSAP address upon receiving the Connection Confirm.

4.7 Comparison with Sincoskie and Cotton

To some extent, one could view our mobility proposal as an extension of Sincoskie and Cotton's [55] design. Relevant features of Sincoskie and Cotton's

design are summarized here, followed by a comparison with our proposal.

In Sincoskie and Cotton, an address contains a unique ID only; there is no information that would partially identify the destination. In order to locate an ES, a controlled flood of the network must be performed, and the ES must send a datagram in response. When an ES sends a datagram, any bridge that forwards the datagram will cache information concerning the link on which that datagram arrived. Then, if a datagram later needs to be sent to that ES, it can be forwarded on the link from which the earlier datagram arrived. A bridge must maintain cache entries for all ESs that currently are receiving datagrams via that bridge; if a cache entry for an ES is deleted prematurely, a flood will be required the next time that a datagram destined to the ES arrives. To keep the cache from overflowing, entries are timed out after an idle period. For Sincoskie and Cotton's design to work, the network must use exactly the same route (path) in both directions, when exchanging datagrams between a pair of ESs. Further, the network topology must be a spanning tree, or have a spanning tree overlaid on the physical topology for routing purposes. Finally, an ES is unable to provide any information to the bridges concerning the current locations of its correspondents, since addresses contain a unique ID only.

Our mobility proposal has a number of practical advantages over the Sincoskie and Cotton design, when applied to the ISO connectionless routing architecture. First, it is possible to support an arbitrary internetwork topology

of ISs. Since an NSAP address contains a partial destination identifier in addition to a MUID, the internetwork is able to perform normal routing on NSAP addresses belonging to mobile ESs. No flooding is required for any purpose: neither search, update, nor clear. No aggressive cache timeouts are required; caches can be managed in an LRU or similar fashion. An ES caches the NSAP addresses of the mobile ESs with which it is communicating, which allows for optimal routing without an undue burden on the ISs.

Since mobile ESs exhibit a locality of movement per unit time, the “home and roam” approach is a good paradigm. The home NSAP address of a mobile ES changes slowly or not at all, which makes it suitable for storage in a global database. The current NSAP address of a mobile ES can change often, so it is cached at the network layer, at correspondent ESs and certain ISs.

Chapter 5

Caching Strategies at Interior ISs

In the standard ISO OSI connectionless architecture, the ES ID is examined only at the lowest level of the routing hierarchy; hence, it is most practical to store forwarding pointers only at that hierarchical level. In the general case of a multi-level routing hierarchy, however, information concerning mobile ES locations can be cached at ISs throughout the hierarchy. When the internetwork forwards a datagram, there are two parameters that should be simultaneously optimized.

First, the datagram's path should be as short as possible relative to a properly addressed datagram. The optimal location for a forwarding pointer is dependent upon the relative positions of the source, the addressed location, and

the correct destination. However, in the case of an internetwork that is tree-shaped and routed hierarchically, to update all relevant IS nodes it is sufficient to send a message from an ES's current location to the ES's previous location, and also from the current location to the ES's home location. As long as *all* nodes along the way cache forwarding pointers to the new location, routing will be optimal. This observation is due to Sincoskie and Cotton [55].

The second parameter is that the source ES should be informed of the correct destination NSAP address in a timely manner. Here the principle is simple: the closer a forwarding pointer is to the source, in terms of routing hops, the faster the source will be informed. Since locality of reference is not guaranteed, a datagram addressed to a mobile ES may have been sent by a source that is quite distant. To inform such sources in a timely manner, it is not sufficient to locate forwarding pointers just in the general neighborhood of a mobile ES; forwarding pointers are also needed in portions of the routing hierarchy that are nearer to distant traffic sources. Note that this is a performance, rather than correctness, issue. A source will eventually learn the correct NSAP address of a correspondent mobile ES, as soon as it receives a datagram from that ES.

We first review the range of design possibilities for caching forwarding pointers, and then select a reasonable set of options for study. The results of the caching study are presented in Chapter 7.

5.1 Design Space

The presence of a forwarding pointer at each mobile ES's home location is sufficient to ensure correct internetwork operation; caching additional forwarding pointers at higher levels of the routing hierarchy potentially will enhance performance. The term leaf IS describes an IS at the lowest level of the routing hierarchy; leaf ISs are located at the periphery of an internetwork, and are able to support mobile ES connections. ISs at higher levels of the routing hierarchy are termed interior ISs.

There are several design questions:

- Under what conditions should additional forwarding pointers be cached at interior ISs¹? There are two options:

1. When the internetwork observes that an ES is away from its home location (The “just in case” strategy). ISs can “spy” on Mobile ES Reconnect messages sent by a roaming ES to its home and previous locations. For a tree-shaped internetwork, the Sincoskie and Cotton technique that was described above can be applied, thus minimizing forwarding path length. The Sincoskie and Cotton technique should be also be of at least some benefit for general (non-tree) internetwork topologies.
2. When the internetwork observes that a datagram has been forwarded

¹A conversation with Dr. Baruch Awerbuch of M.I.T. was helpful in developing a general approach to this question.

(The “after a mistake” strategy). ISs can spy on Mobile NSAP Rewrite messages. This option could be appropriate for deciding when to place a forwarding pointer close to a distant source. Where along the path of the message to cache the pointer, or pointers, is an independent decision.

Another possibility is to combine both options. However, for small IS cache sizes, the danger exists that information gleaned from Rewrite messages will compete for cache space with information from Reconnect messages. If one of the message types turns out to provide better information than does the other, ISs should spy only on the more useful message type.

- Where in the routing hierarchy should additional forwarding pointers be cached? When a message with certain characteristics triggers an IS decision to cache a forwarding pointer, there are two alternatives:
 1. A single pointer can be cached at an optimal location ² in the routing hierarchy. It may, however, be very difficult to determine an optimal location based on the source, destination, and type of a given message.
 2. Forwarding pointers can be stored along the entire path between the source and destination of the message. While this alternative is

²An optimal location is one that will achieve the maximal reduction in excess hops traversed by datagrams that carry out-of-date destination NSAP addresses.

straightforward to implement, it may result in unnecessary forwarding pointers being cached.

- Under what conditions should a cache entry be updated or deleted? One or more of the following techniques could be employed:
 1. An IS could use LRU management to limit cache size. This is fine, since the forwarding pointers stored at interior ISs are not required for correct operation.
 2. In an IS without adequate hardware support for NSAP address look-ups, an incremental cost may be incurred for each cache entry, when comparing the destination NSAP address of a datagram against the cache. In such cases, there may be an incentive to aggressively prune the cache, perhaps by deleting entries after a period of non-use.
 3. An IS could watch Mobile NSAP Rewrite messages and Mobile ES Reconnect messages, and update or delete cache entries when new NSAP address information is received.
 4. An IS could examine the source NSAP addresses of datagrams that are passing by (ignoring any NSAP addresses without a “may be mobile” indication). If a source NSAP address contained a newer LSN than that found in a corresponding cache entry, the entry would either be deleted or updated. However, the cost of examining and processing the source NSAP address of *every* datagram, rather than

just those containing control traffic, makes this option impractical.

Regardless of the techniques chosen, a cache entry must be deleted before the associated mobile ES's LSN has wrapped halfway around from the value contained in the cache entry. Otherwise, the cache entry will appear to be newer than the mobile ES's current NSAP address, and this may cause NSAP addresses to be rewritten incorrectly.

5.2 Options Chosen for Study

Certain of the options listed in the previous section cannot reasonably be implemented at interior ISs. A caching policy that is based on the routing distance of an IS from either the source or destination of an individual datagram is not practical, as such distance information is not likely to be readily available in a time-efficient manner (or at all). Thus, it is not possible to cache a single forwarding pointer at an optimal location in the routing hierarchy. Also, as mentioned in the previous section, examining and processing the source NSAP address of *every* datagram by a transit IS is too expensive, and must be eliminated from consideration. Note that the Teraoka [59, 60, 61, 62] proposal does require that transit ISs examine the source and destination NSAP address of every datagram.

We have studied the remaining caching options; these are as follows. An interior IS can be configured to spy on transit Reconnect messages, Rewrite

messages, both message types, or neither. Further, it is possible to configure an interior IS so that it will spy on transit control messages only if the IS is at or below a specified level of the routing hierarchy. This is of use mainly when studying tree-shaped networks, which have multiple levels. Interior IS forwarding pointer caches are of a fixed, configurable size, and are managed on an LRU basis. Any interior IS that maintains a forwarding pointer cache processes all transit Unreachable messages.

Although this chapter deals primarily with supplemental caching at interior ISs, note that control messages also pass through the leaf ISs. Spying on transit Reconnect messages by a leaf IS will not provide useful information, since such messages could only have come from a directly connected ES, and the IS will already be aware of the presence of such ESs. In theory, spying on Rewrite messages by a leaf IS could have two benefits. First, if a Rewrite message is directed at one of the ESs that is currently connected to an IS, the IS will be able to apply the information contained in the Rewrite to datagrams originating from other connected ESs. However, there is no guarantee that the other ESs will ever generate a datagram destined to the ES referenced in the Rewrite message. A second benefit is that if the ES referenced in a Rewrite was previously attached to a leaf IS, that IS can update (or recreate) the cache entry for the ES. The probability that a Rewrite message will happen to reference a previously connected ES, however, is fairly small.

There is a significant danger in having a leaf IS spy on transit Rewrite

messages that outweighs the possible benefits. A leaf IS tries to maintain forwarding pointer cache entries for mobile ESs that were connected locally, but have roamed elsewhere. If such an IS also caches NSAP addresses obtained from transit Rewrite messages, these NSAP addresses (which might or might not be of use) will displace cache entries for ESs that were previously local. Thus, it is best for leaf ISs not to spy on transit control messages, a conclusion supported by the results presented in Chapter 7.

Chapter 6

Design and Implementation of the Simulation

To be able to experiment with mobility designs, a substantial internetwork simulation was designed and constructed. The simulation, which is a program written in the DeNet [38] language, enables exploration of mobile internetworking. The simulation has proven to be a valuable and effective emulation of an internetwork environment.

The simulation supports an arbitrary internetwork topology of ISs. Datagrams are routed on a Shortest Path First (SPF) basis, with path length based on the number of hops. There are two categories of ISs: leaf and interior. ESs can connect to the leaf ISs, while interior ISs support store and forward routing only. An arbitrary number of mobile ESs can travel among the leaf ISs, while conversing with other ESs. The internetwork topology, as well as the patterns of

movement and communication among the mobile ESs, can be specified at run-time, without the need to recompile the simulation software. The simulation gathers extensive statistics during a run, and presents a summary upon completion. Varying degrees of status information can be displayed as the simulation executes; the amount of information desired can be selected at run-time.

The simulation can be used to compare the performance of various designs, including IS caching policies, and also to validate correctness. The simulation does not provide a fine-grained model of ES or IS internals, as we are primarily interested in the behavior of internetwork routing, including the readdressing and forwarding of datagrams by ISs.

6.1 DeNet Language

DeNet is a simulation language created by Miron Livny at the University of Wisconsin-Madison. DeNet provides an event-driven model. It is a superset of the Modula-2 language, and is implemented as a pre-compiler along with various Modula-2 modules. Development of the simulation software was done on a DECstation 5000, and the Modula-2 code generated by the DeNet pre-compiler was then compiled using the Gardens Point Modula [24] compiler from the Queensland University of Technology.

6.2 Simulation Components

6.2.1 ES

An ES can periodically relocate; the relocation pattern and frequency are configurable. Details of the DeNet environment make it necessary to pre-attach (from the perspective of the *simulator*, not the simulated internetwork) an ES to all the ISs that it might visit during a particular run. These attachments then determine the set of ISs to which that ES might roam.

One of two types of relocation pattern can be specified: random or circulate. If the random pattern is specified, an ES randomly chooses (with a uniform distribution) among the ISs to which it has been pre-attached. A random pattern is best specified for an ES when the pre-attached ISs are clustered in a region. If the circulate pattern is specified, an ES moves back and forth through the ISs to which it is attached; when the ES reaches one end of the IS list, it turns around and moves in the other direction. The initial direction in which an ES moves (up or down) is chosen randomly, with the alternatives evenly weighted. A circulate pattern is most appropriate for an ES when the pre-attached ISs are scattered over a wide distance. This is because a human (and the accompanying mobile ES) that travels an extended distance is likely to move in a linear, rather than random, pattern.

The IS that an ES initially visits is chosen randomly, with a uniform distribution, among the ISs to which it has been pre-attached. As simulated, the IS that

an ES initially visits becomes the home location for that ES; the home location does not change. Therefore, the Unregister message described in Chapter 4 is not implemented.

An ES will periodically open transport connections to other ESs. If a cache entry for the destination ES is not available, communication is preceded by an access of the simulation's database component, in order to obtain the destination's home NSAP address. The communication frequency and pattern of an ES are configurable. For each ES, a set of communications groups is specified; a group is defined as a contiguous range of ES node numbers. A communications probability is specified for each group, which is used to determine the likelihood that a given group will be chosen as the target for the next transport connection. Within a communications group, the target ES is chosen randomly, with a uniform distribution among the members of the group. The ES that will be opening the transport connection is excluded from consideration as a target.

The ES module of the simulation includes a reliable, sequenced, simplex transport protocol that we designed for this purpose; the protocol is loosely modeled on TP4. Refer to Appendix B for details concerning the design of the transport protocol. Upon opening a transport connection, the user of the transport layer specifies the length of time that the connection should remain open. After a transport connection has been established, fairly long delays are inserted between the generation of successive Data TPDU's, in order to prevent the simulator (as distinct from the simulated internetwork) from becoming overloaded

with traffic. This has the effect of increasing the apparent percentage of suboptimally routed datagrams, because the ratio of Connection Request TPDU's to Data TPDU's is much higher than it would be in real life. While preventing the simulation from overloading is certainly a good thing, it means that a correction factor must be calculated and applied to the simulation's results. The correction factor adjusts the results to reflect the estimated number of Data TPDU's that would normally have been transmitted, but were not.

As a datagram is routed through the simulated internetwork, a hop count field is incremented each time that the datagram is transmitted via a communications link. When a datagram arrives at the destination ES, that ES is able to record statistics concerning the datagram's journey. By consulting the simulation's distance matrix (generated by the tool to be described in Subsection 6.3.2), the ES is able to determine the minimum number of routing hops that the datagram could have traversed, if it had been correctly addressed. By comparing that value with the actual number of routing hops, the ES is able to identify datagrams that were routed via a suboptimal path, due to an out-of-date destination NSAP address having been specified.

Configurable parameters for each ES include:

- Link latency when sending to an IS.
- Transmission time (first bit to last bit) when sending to an IS.
- Average interval between ES relocations [exponentially distributed].

- Relocation pattern: random or circulate.
- Number of ES communications groups with which this ES will communicate.
- The range of ES node numbers associated with each communications group.
- The probability of communication with each communications group.
- Average interval between transport connection starts [exponentially distributed].
- Average transport connection length, in seconds [exponentially distributed].
- Average interval between the generation of successive Data TPDU's [exponentially distributed].
- Retransmission interval for all TPDU types except Connection Request.
- Retransmission interval for Connection Request TPDU's.

6.2.2 IS

There are two types of IS, leaf and interior. Mobile ESs visit leaf ISs, and these ISs are connected to interior ISs. Interior ISs store and forward datagrams, but

do not directly support visits by ESs. Interior ISs can optionally cache forwarding pointers for mobile ESs. A decision as to whether to cache can be based on various parameters, including the IS's hierarchical level. An IS's hierarchical level is specified by a user of the simulation as part of the internetwork topology, and is not examined by the IS other than when making caching decisions. We have experimented with various schemes for caching forwarding pointers at interior ISs, trying combinations of the design possibilities discussed in Chapter 5. At run-time, an IS dynamically decides that it is a leaf IS if any of its communications links are capable of supporting an ES visit.

A leaf IS handles home and transient registrations by mobile ESs, and will maintain forwarding pointers for those systems. As simulated, when a mobile ES prepares to relocate, it sends an explicit disconnect to the local IS, and the IS enters "quiescent" state. During the relocation interval, datagrams for the mobile ES might arrive at the mobile ES's last known location (i.e., the IS from which it has just disconnected). Rather than simply discarding such datagrams, the IS queues the datagrams, and forwards them upon the IS's receiving a Reconnect message from the mobile ES at its new location.

After a mobile ES reconnects at a new IS, it sends a Reconnect message to its home IS and another Reconnect message to its previous IS. The simulation can be configured so that transit ISs that route the Reconnect messages will "spy" on those messages, and update their caches accordingly. The simulation can also be configured so that transit ISs that route Rewrite messages will update

their caches based on the contents of those messages. The simulation can be configured so that only those interior ISs below a specified hierarchical level will spy on transit Reconnect and Rewrite messages. Finally, whether or not leaf ISs will spy on transit messages is a configurable parameter.

Non-permanent cache entries at ISs are managed on an LRU basis; the maximum number of such entries is a configurable parameter. Non-permanent cache entries include information concerning ESs that previously visited an IS (on a transient basis), but have roamed elsewhere, as well as information gathered from transit Reconnect and Rewrite messages. As simulated, cache entries do *not* time out.

As simulated, a Rewrite message is generated each time that an IS rewrites the destination NSAP address in a datagram; the Rewrite is sent to the source NSAP address contained in the datagram. In a real implementation, the Rewrite generation rate would most likely be clamped, perhaps by associating a “last used” timestamp with each cache entry. Also as simulated, links are reliable; no datagrams are dropped. This is true for both the IS to IS and IS to ES links.

Configurable parameters for each IS include:

- Link latency when sending to another IS.
- Transmission time (first bit to last bit) when sending to another IS.
- Link latency when sending to an ES.
- Transmission time (first bit to last bit) when sending to an ES.

- Maximum number of non-permanent cache entries if this is a leaf IS.
- Maximum number of non-permanent cache entries if this is an interior IS.
- Should the IS “spy” on transit Reconnect messages?
- Should the IS “spy” on transit Rewrite messages?
- Hierarchical level at or below which “spying” by interior ISs is permitted.
- If this is a leaf IS, is it permitted to “spy?”

6.2.3 Database

The database component maintains the mapping between mobile ES MUID and home NSAP address; it is the responsibility of an IS at the home location to keep track of the NSAP address associated with the mobile ES’s current location. The interface to the database is simple. Procedures are provided to update (or create, if necessary) a database entry, and to lookup an ES based on its MUID. In an actual implementation, the database would also provide mappings from high-level application identifiers to the associated NSAP addresses. As simulated, database queries do not incur a time penalty.

6.2.4 Address Manager

The address manager maintains a single, read-only copy of each NSAP address in the system; all accesses are by reference only. Although such handling of

addresses would be infeasible in an actual implementation, it proved to be quite valuable in a simulation environment. The address manager is used by the IS, ES, and database components of the simulation. The manager provides primitives to create (construct), install, acquire, and release an **NSAP** address. When acquiring or releasing an **NSAP** address, the type of access (create, transit, cache, or database) must be specified; thus, the address manager can keep reference count information for each address. Upon termination, the reference counts for all active **NSAP** addresses can be displayed; this was an invaluable tool for evaluating protocol correctness, since the cause of a suspicious reference count could then be investigated.

6.3 Preprocessor tools

Fairly early on in the development of the simulation environment, it became clear that some semi-automated method to generate simulation control files was needed. The alternative was to build (and repeatedly modify!) large control files by hand, something definitely to be avoided.

To aid in the construction of control files, two tools were designed and implemented: a mobility scenario generator, and a topology and routing generator. The output from these two tools can be concatenated to form a control file for input to the simulation. Generating the two components of a control file separately means that either component can be saved and reused as desired.

6.3.1 Mobility Scenario Generator

The mobility scenario generator was implemented in C. Input to the tool consists of two files. The first, a *parameters* file, specifies values for the various parameters that control the behavior of the simulation. Some parameters control global behavior of the simulation, such as how much detail should be displayed upon termination. Other parameters specify ES and IS behavior, and many of these were detailed in Section 6.2. An example would be the average ES relocation interval.

The second input file, the *profile*, specifies the general movement and communications patterns of the ESs. Rather than specifying the behavior of individual ESs, the profile deals with groups of ESs; the members of a group, which may be of arbitrary size, will all exhibit similar behavior. Grouping makes the task of creating a profile significantly more manageable. The definition of an ES group includes the set of IS nodes to which members of the group will be pre-attached, and the relocation pattern (circulate or random) that will be used when visiting the ISs. The definition of an ES group also lists the groups to which transport connections will be opened, and provides a probability value for each group. A probability value indicates the likelihood that the corresponding group will be chosen as a target for the next transport connection. Refer back to Subsection 6.2.1 for further details concerning ES movement and communications.

Appendix A.1 provides a sample parameters file and a sample profile file,

```

node-definition:  node nodenum/hierarch-level: neighborlist end
neighborlist:    neighbor | neighborlist neighbor
nodenum:        INTEGER
hierarch-level: INTEGER
neighbor:       INTEGER

```

Figure 6.1: Syntax for a Node Definition

along with the corresponding output from the mobility scenario generator.

6.3.2 Internetwork Topology and Routing Generator

The internetwork topology and routing generator was created using the LEX and YACC (parser generator) utilities. The tool takes as input a compact description of an internetwork topology, in the form of a set of IS node definitions. The syntax for a node definition is shown in Figure 6.1. Associated with each IS is a node number, hierarchical level, and one or more neighbor IS nodes.

The hierarchical level is passed along unchanged; the real work is done on an IS's node number and neighbor list. The tool parses the input file, generates a connectivity matrix and verifies that all inter-node links are bidirectional. Then, the tool performs a SPF routing calculation (Dijkstra [58]), and verifies that the topology is not partitioned. Path length is calculated based solely on the number of routing hops. Finally, the topology data, along with next-hop and distance routing matrices, are output in a form that can be parsed by DeNet's input processor.

Refer to Appendix A.2 for a sample input file and the corresponding output file. Note that, due to readability and space constraints, the internetwork topology described in the appendix is smaller than those used when running the simulation.

6.4 Simulation Job Submission

For a given pair of parameters and profile files, along with a particular internetwork topology, we would typically want to try out a variety of caching parameter values. To accomplish this, we would create a base parameters file with all parameters except those relating to caching at ISs, and then “on the fly” tack on the various combinations of caching parameters. C-shell scripts were written to automate the production of a set of simulation control files, by feeding appropriate inputs to the mobility scenario generator and to the topology and routing generator. The scripts are also able to produce a Condor [7] control file, so that the set of simulation jobs can be submitted to the Condor batch job control system.

6.5 Graphing

The volume of statistical data generated can be overwhelming, especially when comparing a number of simulation runs. As such, tools were developed to process the data generated by the simulation, and display it in an easy to

comprehend graphical form. We employed the AWK and C-shell utilities to postprocess the simulation's output, and Jgraph [45] to generate graphs in the PostScript language.

Various example graphs will be provided in Chapter 7.

Chapter 7

Caching Study

This chapter presents the results of a study of the effect of caching at interior IS nodes on system performance. Specifically, we show that appropriate combinations of the caching parameters described in Chapter 5 can reduce the percentage of datagrams that are routed suboptimally. A datagram is said to have been routed suboptimally if it traveled a longer path than would have a datagram addressed directly to the destination ES's current NSAP address. The results are stable across a variety of internetwork topologies and mobility scenarios.

We first describe the internetwork topologies and mobility scenarios used to conduct the study, along with details of the simulation runs that were performed. Next, the numerical results are summarized. Finally, the rationale is presented for certain features of our mobility design; these features were added as a result of lessons learned during the caching study.

7.1 Experimental Topologies

As with any simulation environment, DeNet has limits as far as execution speed: as the size of a simulation grows, the number of simulated events per second that can be handled eventually becomes a bottleneck. Tests showed that, for our simulation, the practical size limit is approximately 120 ESs and 31 ISs. In the experimental topologies, ISs 1 through 15 are designated as interior nodes; these nodes route traffic, but do not directly support ESs. ISs 16 through 31 are leaf nodes, to which an arbitrary number of ESs are able to attach. Each of the leaf ISs could be considered to model an ISO OSI area. The same number of ISs is used for each topology, in order to allow the results to be compared, and to enable use of the same mobility scenarios.

Given the Sincoskie and Cotton technique, caching at interior ISs should be most effective when an internetwork is tree-shaped; hence one of the topologies is a five level tree, as illustrated in Figure 7.1. Note that three communications links are shown exiting each leaf IS; these are intended to suggest points at which an *arbitrary* number of ESs might attach. The number three is simply for the purpose of illustration. The abbreviated name shown in parentheses (5leveltree) in the figure's caption will be used when later referring to the topology.

In practice, internetwork topologies are not tree-shaped, nor is a spanning tree typically imposed for routing purposes. So, to be generally applicable, caching strategies need to be effective for a variety of topologies. We constructed six additional experimental topologies, which though certainly not exhaustive,

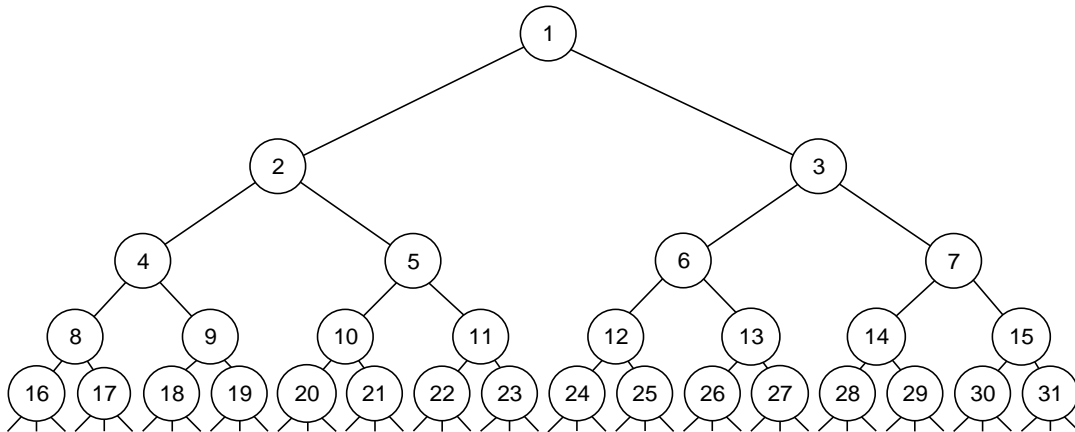


Figure 7.1: Five Level Tree (5leveltree)

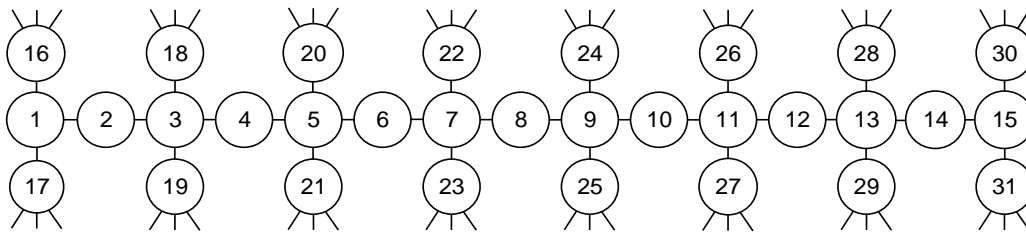


Figure 7.2: Linear with Leaf Nodes Evenly Distributed (lineardistrib)

do cover a range of possible internetwork topologies. In Figure 7.2, the interior ISs are arranged linearly, with the leaf ISs evenly distributed across the internetwork. In Figure 7.3, the interior ISs have the same arrangement, but the leaf ISs are grouped at either end of the internetwork.

In Figure 7.4, the interior ISs are arranged in a single ring, with the leaf ISs distributed evenly around the ring. Figure 7.5 shows the same arrangement of ISs, but with additional communications links cutting across the ring.

Figures 7.6 and 7.7 show internetwork topologies in which the interior ISs are arranged in two rings, with a single link between the rings. In the first

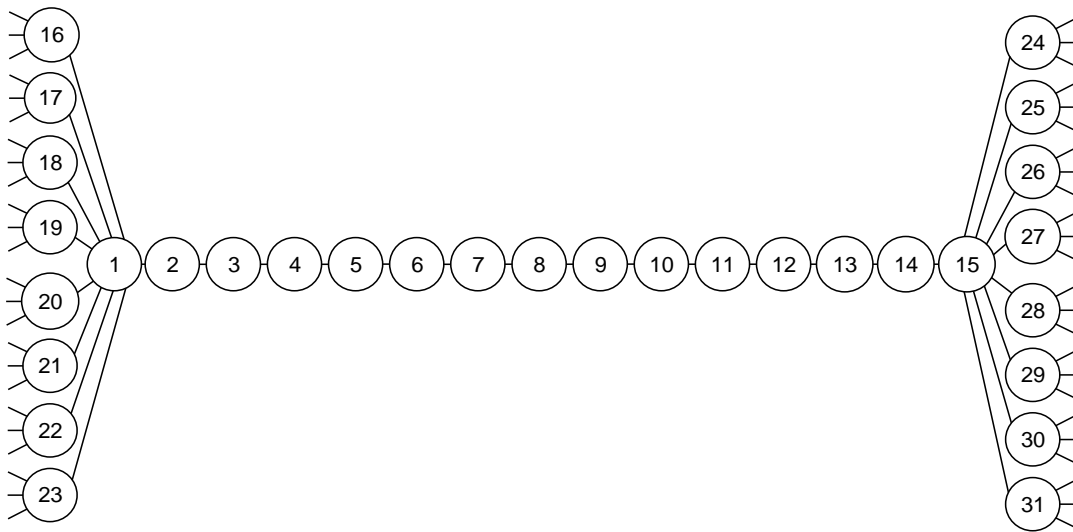


Figure 7.3: Linear with Leaf Nodes at Both Ends (linearatend)

figure, the leaf ISs are evenly distributed around both rings, while in the second figure, all but two of the leaf ISs are on one of the two rings.

Finally, Figure 7.8 models the ANSnet¹ T3 backbone topology as of February, 1994. Table 7.1 indicates the geographic location of each of the 15 interior ISs. With just 16 leaf ISs available, it was not possible to fully model the regional subnetworks connected to the backbone; we ranked the interior ISs by the number of regional interconnections, and placed a pair of leaf ISs at the 8 sites with the most interconnections.

¹ANSnet is used to route the NSFnet's traffic.

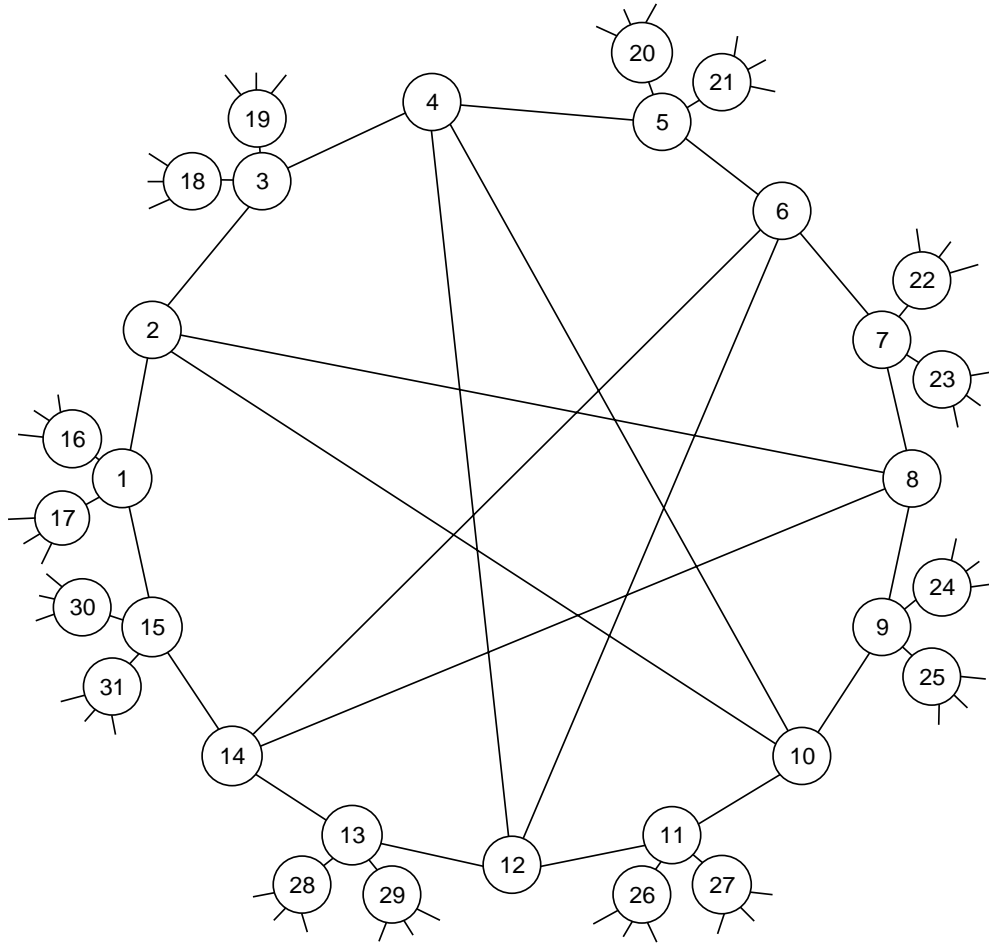


Figure 7.5: One Ring with Additional Links (1ringstardistr)

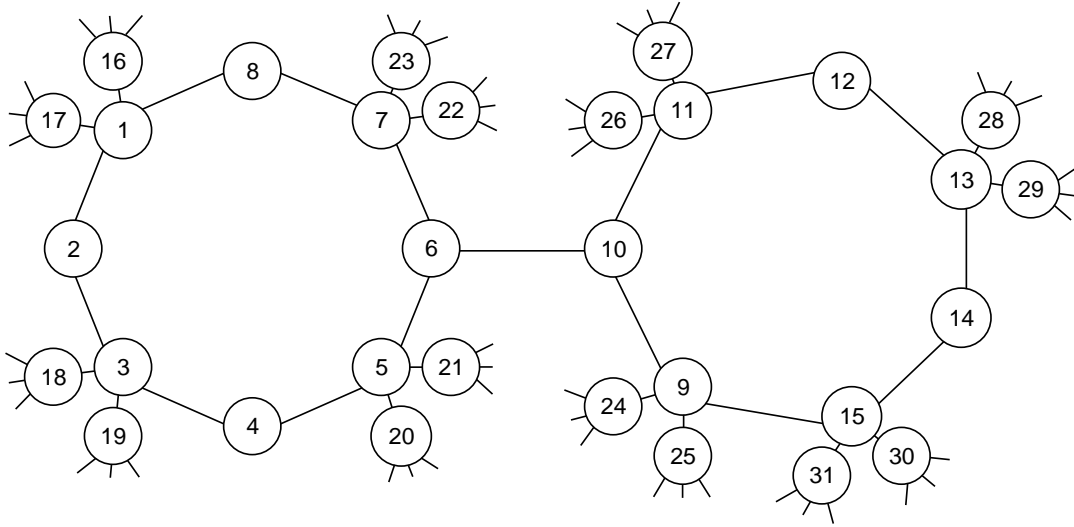


Figure 7.6: Two Rings with Even Leaf Distribution (2ringdistr)

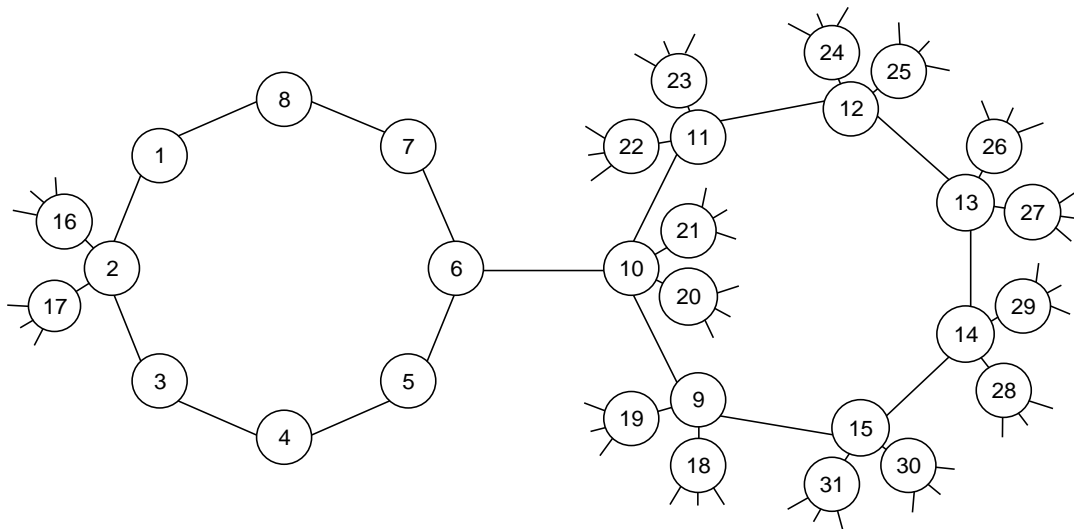


Figure 7.7: Two Rings with Uneven Leaf Distribution (2ringuneven)

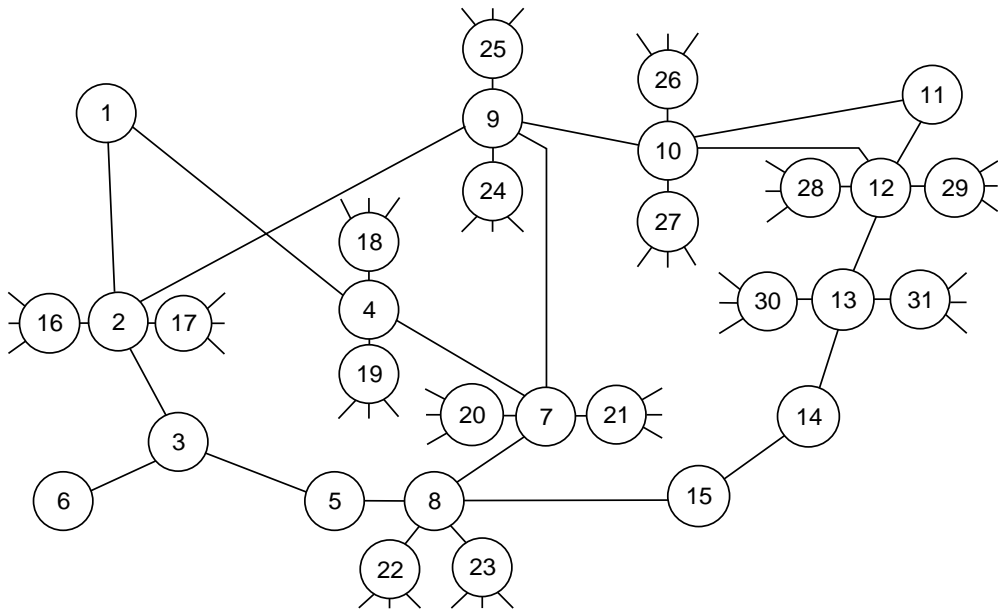


Figure 7.8: A Model of the ANSnet T3 Backbone (nsftop)

IS	Location
1	Seattle, WA
2	San Francisco, CA
3	Los Angeles, CA
4	Denver, CO
5	Albuquerque, NM
6	Honolulu, HI
7	St. Louis, MO
8	Houston, TX
9	Chicago, IL
10	Cleveland, OH
11	Hartford, CT
12	New York, NY
13	Washington, DC
14	Greensboro, NC
15	Atlanta, GA

Table 7.1: Location of ANSnet T3 Backbone Nodes

7.2 Mobility Scenarios

It would be nearly impossible to develop a *single* model of mobile ES movement and communication and to prove that the model mirrored reality. Fortunately, this is not necessary, as a reasonable alternative is to experiment with a range of workloads.

The mobility scenario generator tool described in Section 6.3.1 was employed to generate patterns of ES movement and communication for the simulation. The parameters and profile files used as input to the tool are described in the two following subsections.

7.2.1 Parameter Files

Four parameters files were created, params1 through params4. These files share certain common parameter values, as shown in Table 7.2. The latency on an IS↔IS link is 10 times that of a an ES↔IS link; this is realistic given the typical

Parameter	Value (in milliseconds)
ES↔IS link latency	0.5
ES↔IS datagram transmission time	4
IS↔IS link latency	5
IS↔IS datagram transmission time	1
Average Data TPDU generation interval	200
Transport connection retransmission interval	300

Table 7.2: Common Parameter Values

distances involved. Both latency values are fairly high, but this helps compensate for the limited size of the simulated topologies. As was described in Subsection 6.2.1, it is necessary to limit the rate at which Data TPDU's are generated by a transport connection, so as not to overwhelm the simulator. The average Data TPDU interval (exponentially distributed) shown in Table 7.2 translates to an average rate of 5 Data TPDU's per second. The datagram transmission times translate to a rate of 1000 datagrams per second on an $IS \leftrightarrow IS$ link and 250 datagrams per second on an $ES \leftrightarrow IS$ link. The latter value is rather high, but the limit on Data TPDU generation means that $ES \leftrightarrow IS$ links will not be utilized at anywhere near capacity.

The parameters values that are specific to each parameters file are listed in Table 7.3. All of these parameters are average values (with an exponential distribution), and the selected combinations of parameters will put a variety of stresses on the IS and ES caches. For `params1` through `params3`, an ES relocates every 2 seconds. An ES in `params4` relocates once per second, or at twice the rate of the other three files. In `params1`, an ES opens 2 transport connections between successive relocations, and each connection lasts for 1 second. An ES in `params2` opens 1 transport connection during every *other* relocation interval. For `params3`, an ES opens 1 transport connection between successive relocations, and that connection remains open during 2 relocation intervals. Finally, for `params4`, an ES opens 1 transport connection per relocation interval, and that connection lasts for one interval.

	params1	params2	params3	params4
Parameter	Average Value (in seconds)			
Interval between ES relocations	2	2	2	1
Transport connection start intvl	1	4	2	1
Transport connection length	1	1	4	1

Table 7.3: Parameter Values Specific to Each File

7.2.2 Profile Files

Three profile files were generated, `prof120region`, `prof120long`, and `prof120-hybrid`. Each profile contains 120 ESs, which are divided into 4 groups of 30 ESs each. When an ES chooses a target for a transport connection, there is a 40% chance that it will select one of the other ESs in its group, and a 20% chance for each of the remaining groups that it will select an ES from that group. Once an ES group has been selected, the choice of a target *within* that group is made randomly, with a uniform distribution.

The three profiles differ in the distance across which ESs relocate geographically, and the patterns of those relocations. In `prof120region`, members of an ES group roam within a defined cluster of IS nodes. Each group is assigned to one quarter of the internet network's leaf IS nodes, and the clusters do not overlap. When a ES relocates, it chooses randomly, with a uniform distribution, among the IS nodes in its cluster. This profile models a mobile ES owner who roams within a limited geographic area.

In `prof120long`, members of an ES group roam among a set of widely dispersed IS nodes. An ES circulates among the set of leaf IS nodes to which its group has been assigned. The ES visits each IS on its list in succession; when the end (or beginning) of the list is reached, the ES turns around and moves in the opposite direction. This profile models a mobile ES owner who regularly travels back and forth across an extended distance.

`Prof120hybrid` is a hybrid of the other two profiles. Two of the ES groups relocate randomly among a cluster of IS nodes, and the other two groups circulate among a dispersed set of IS nodes. The profile models a more typical mix of mobile ES owners, where some remain within a geographic region and others travel to distant areas.

7.3 Run Details

Simulation runs were done with a cross product of the 8 topologies, 4 parameters files, and 3 profile files. For each combination of topology, parameters, and profile, a variety of interior IS caching parameters were tried. Runs were performed with IS forwarding pointer cache sizes of 2, 10, 20, 30, 40 and 100 entries; for any given run, the same cache size was specified for the interior ISs and the leaf ISs. Interior ISs were configured to “spy” on transit Reconnect messages, transit Rewrite messages, both message types, or neither. In the last case, which will be labeled as NONE on the graphs in the following section, no interior IS caching is performed, and the curves show the performance of the

basic mobility algorithm.

For the 5leveltree topology, an additional variable was the hierarchical level at or below which interior ISs were permitted to cache forwarding pointers. This parameter was varied from 1 (all interior nodes may cache) to 4 (only the lowest level of interior nodes may cache). A final parameter for all topologies was whether *leaf* ISs were permitted to spy on transit Rewrite messages.

7.4 Numerical Results

When an ES opens a transport connection to a mobile ES, it is sometimes the case that the source ES has an out-of-date cache entry for the destination. This will occur if the destination mobile ES has relocated since the source last cached the destination's NSAP address. If the Connection Request TPDU is sent to an out-of-date NSAP address, it typically is forwarded one or more times by leaf ISs, and eventually arrives at the mobile ES's current location. If interior ISs are configured to cache forwarding pointers, these ISs might participate in the forwarding process as well. As soon as the source ES receives a Rewrite message from an IS or a Connection Confirm TPDU from the destination ES, the source is able to update its cache entry. A similar process can occur if the mobile ES relocates after the transport connection was opened, with a small number of datagrams being forwarded via the mobile ES's previous location.

Forwarding pointer caches at ISs are of finite size, and a leaf IS is not required to retain cache entries for ESs that were previously local (on a transient basis),

but have roamed elsewhere. Also, all cache entries at interior ISs are optional. Thus, on rare occasion, the forwarding process described above will lead to an IS that does not have a cache entry for the destination ES. The IS drops the datagram and sends an Unreachable message to the source ES. That ES, in turn, resends the datagram to the destination's home NSAP address, and it is forwarded to the destination's current location. Again, the source updates its cache entry upon receiving a Rewrite message from the home location or a datagram from the destination mobile ES.

Except for brief transients at the very start of a transport connection and after a correspondent ES relocates, routing is optimal, since each ES caches the current NSAP address of its correspondents.

7.4.1 Correction Factor

The preceding discussion was necessary to motivate the need for a *correction factor* to be applied to the results produced by the simulation. As mentioned in Subsections 6.2.1 and 7.2.1, it is necessary to limit the rate at which Data TPDU's are generated, in order to prevent the simulator (as opposed to the simulated internetwork) from overloading. In the experiments conducted for this caching study, generation of Data TPDU's for a transport connection was limited to 5 per second. What this means is that certain statistics generated by the simulation, such as the percent of datagrams routed suboptimally, are larger than they would be, had the Data TPDU generation rate not been artificially

limited. The additional Data TPDU_s that would otherwise have been generated would have increased the total population of datagrams, without increasing the number of datagrams that could have been suboptimally routed.

We need to calculate a correction factor, in order to adjust the statistics produced by the simulation for the disparity in Data TPDU generation rates. In the experiments conducted for this caching study, ES↔IS links were configured with a capacity of 250 datagrams per second. However, it is likely, even without a limit on datagram generation rates, that an ES would have used only a fraction of that capacity. Based on observed ES transmission rates on today's internetworks, we will make a conservative estimate that a single transport connection would have used 10% of the link capacity, for a rate of 25 datagrams per second, versus 5 datagrams per second with the limit on Data TPDU generation. To adjust the percentage values generated by the simulation for the 5 to 1 ratio, we therefore need to multiply each value by a correction factor of 0.20. This correction factor is an estimate, rather than an exact value, but it errs on the conservative side. So, at worst, the results will appear to be slightly less good than they ought. All graphs presented in this chapter show the adjusted values.

7.4.2 Dropped Datagrams

Figure 7.9 plots percent dropped datagrams versus IS cache size. In this experiment, leaf IS nodes did not spy on transit Rewrite messages. Several elements

of notation are worth describing, as they will be used during the remainder of this chapter. The string

```
nsftop:params3+prof120hybrid
```

indicates that the experiment used internet network topology `nsftop`, parameters file `params3`, and profile file `prof120hybrid`. These elements were described in Sections 7.1 and 7.2. The dotted curve, labeled NONE, shows the performance of the basic mobility algorithm, with no caching at interior IS nodes. The remaining curves show the effect of having interior IS nodes spy on Rewrite messages, Reconnect messages, and both message types.

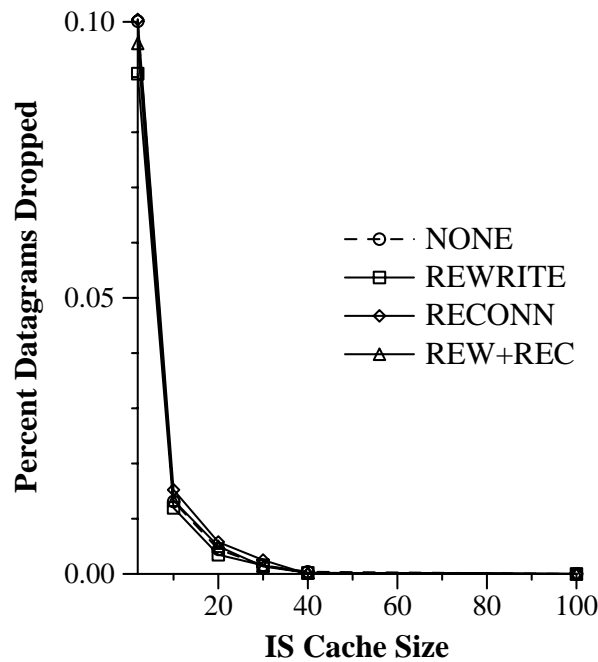


Figure 7.9: Experiment `nsftop:params3+prof120hybrid`

Figure 7.10 plots results from the same experiment as Figure 7.9, *except* that in cases where interior ISs were configured to spy on transit Rewrite messages,

leaf ISs were also permitted to spy on those messages. For small IS cache sizes, spying on Rewrite messages (either alone or in combination with spying on Reconnect messages) caused a higher percentage of datagrams to be dropped than if no spying occurred. This is because information obtained by the leaf ISs from transit Rewrite message competes for cache space with forwarding pointers for mobile ESs that were previously local. The Rewrite information, which *might* possibly be of use, has displaced forwarding pointers that are definitely useful. Thus, we conclude that spying on Rewrite messages at leaf ISs is not a good policy, and such caching has not been performed for any of the remaining experiments described in this chapter.

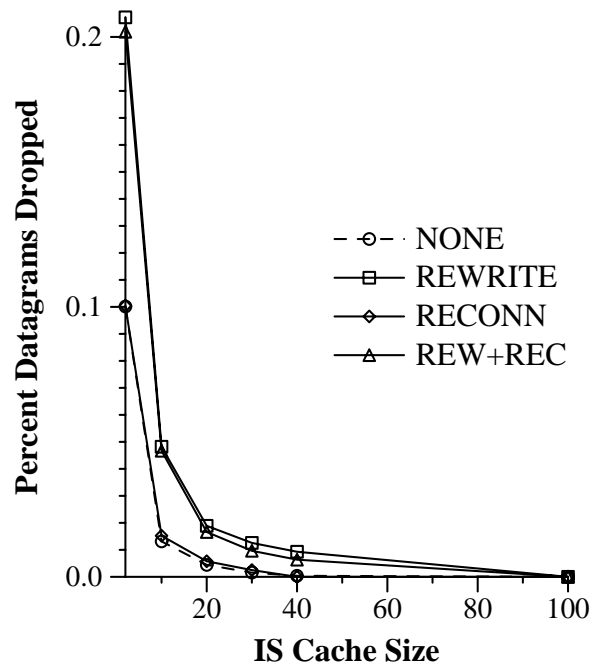


Figure 7.10: Experiment nsftop:params3+prof120hybrid (with leaf IS spying)

Figure 7.9 is representative of all the experiments except those involving params4. Those experiments produced results like those plotted in Figure 7.11. The curves are similar in shape to those involving the other parameters files, but approximately 3 times the number of datagrams were dropped. However, the percentage of dropped datagrams is still extremely low. The increase in drop rate is due to a doubling of the ES relocation rate with params4. ES cache entries become out-of-date more quickly, and the chains of forwarding pointers maintained by leaf ISs grow more rapidly in length. Both factors increase the likelihood that a datagram addressed to an out-of-date NSAP address will arrive at a leaf IS where no forwarding pointer is available.

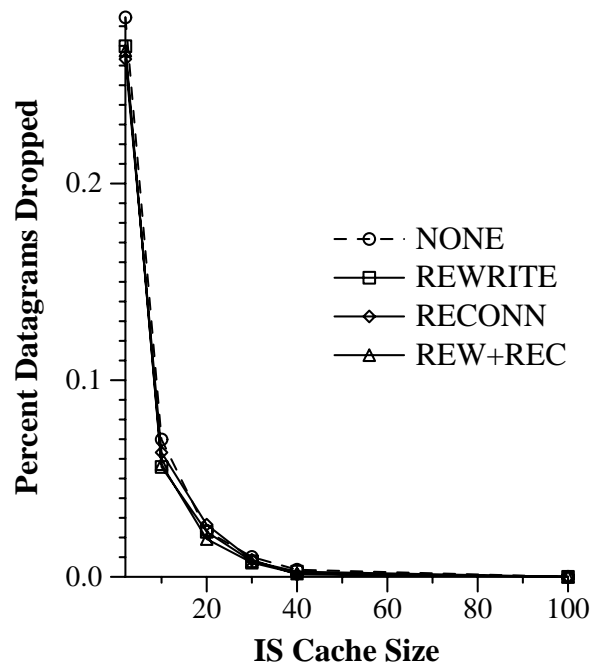


Figure 7.11: Experiment nsftop:params4+prof120hybrid

Several conclusions can be drawn from the results presented in this subsection. First, spying on transit Rewrite messages by leaf ISs is not a good idea, since it is more important for those ISs to maintain forwarding pointers for ESs that were local, but have roamed elsewhere. Caching policies at the interior ISs have a negligible effect on dropped datagrams. The percentage of dropped datagrams is primarily a function of leaf IS cache size: as the cache size increases, the percentage drops rapidly. The percentage of dropped datagrams is also a function of average ES relocation rate: for a given leaf IS cache size, as the ES relocation rate increases, the percentage of dropped datagrams will also increase. Finally, with our mobility algorithm, the percentage of dropped datagrams is quite small, and is stable across a variety of topologies and mobility scenarios.

7.4.3 Suboptimally Routed Datagrams

The left-hand graph in Figure 7.12 plots percent suboptimally routed datagrams versus IS cache size. The number of datagrams used in the denominator when computing this percentage is the number successfully received by destination ESs, and hence does not include datagrams that were dropped at ISs due to the lack of a forwarding pointer. This means that the percentage values are slightly higher than they would be otherwise; again, this errs toward the conservative.

Suboptimal routing increases the total number of routing hops experienced by datagrams in the internetwork, relative to a situation in which all datagrams

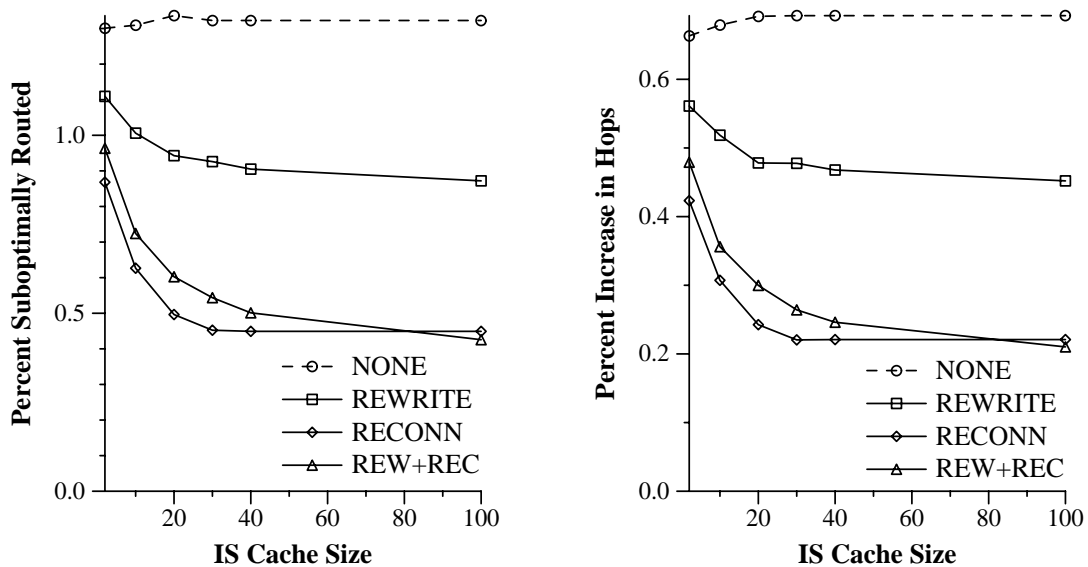


Figure 7.12: Experiment 1ringdistr:params3+prof120region

were routed via the most direct path. The right-hand graph in Figure 7.12 plots percent increase in routing hops versus IS cache size. Figure 7.12 is representative of experiments using prof120region and params1 through params3. Figure 7.13 shows results for the same topology and profile file, but using params4. As with the dropped data graphs, the percentage of suboptimally routed datagrams is higher for params4 than for the other parameters files, due to the faster ES relocation rate; however, in this case, the difference in percentage values is less than a factor of 2.

As illustrated in Figures 7.12 and 7.13, spying on Rewrite messages alone is the least effective interior IS caching policy. This observation is true for all experiments conducted. In these figures, spying on Reconnect messages is more effective for smaller interior IS cache sizes than is spying on both Rewrite and

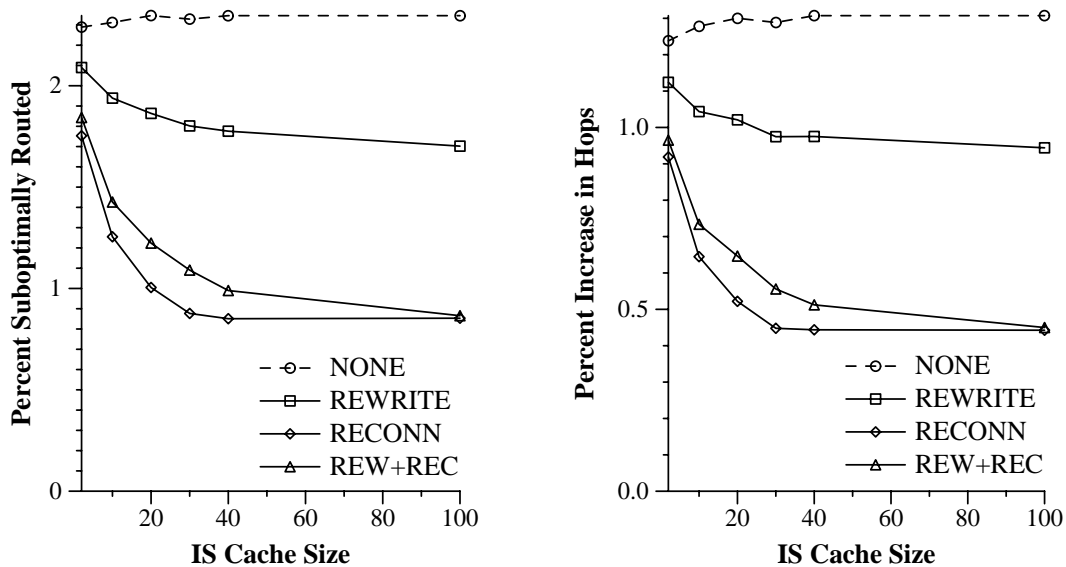


Figure 7.13: Experiment 1ringdistr:params4+prof120region

Reconnect messages. Only with the largest cache size does spying on both message types yield a greater benefit. Spying on both Rewrite and Reconnect messages yields inferior results for small interior IS cache sizes because cache entries created from Rewrite messages displace the more useful cache entries created from Reconnect messages. Reconnect messages provide more timely information, since they are generated when a mobile ES reconnects to the internetwork. With the largest cache size, there is enough room for all cache entries, so the added information from Rewrite messages can improve performance marginally.

The deleterious effect of spying on both Rewrite and Reconnect messages is most evident in the experiments using prof120region, where each mobile ES roams among a cluster of ISs. Spying on Reconnect messages is of particular benefit with this profile because Reconnect messages are likely to pass

through the interior ISs that route traffic from outside the region to both the ES's old and new location. Experiments using the 5leveltree topology along with prof120region showed that caching Reconnects can improve performance, even if only those interior ISs at the lowest two levels of the routing hierarchy participate in caching. This again makes sense, because Reconnect messages travel via a limited region of the internetwork. Figure 7.14 shows an experiment with the topology in which caching is performed at all interior IS nodes, and Figure 7.15 shows the same experiment, but with caching only at the two bottom-most hierarchical levels of interior ISs. The plots are nearly identical.

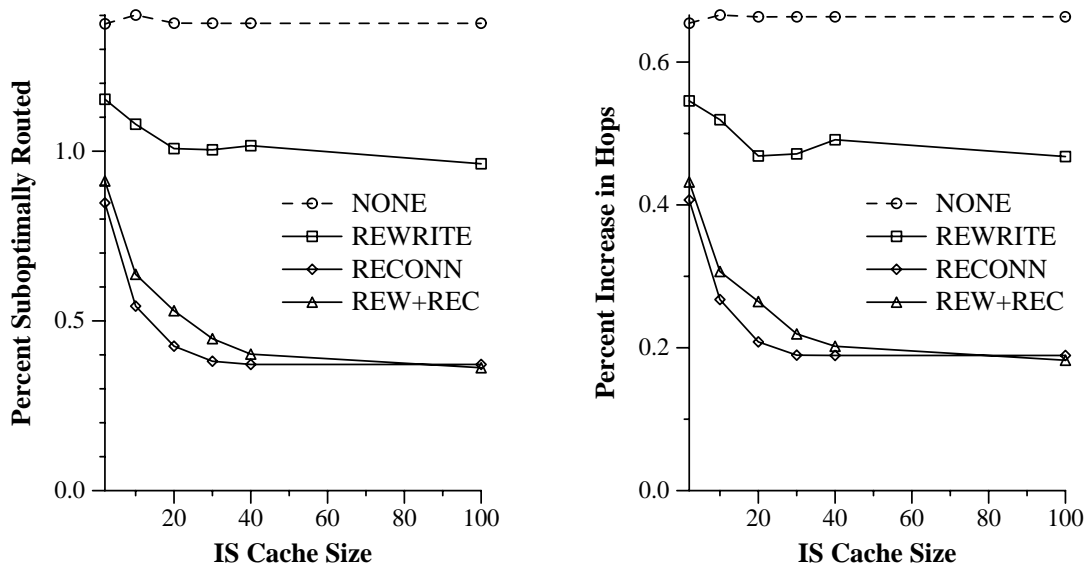


Figure 7.14: Experiment 5leveltree:params2+prof120region (hier-level = 1)

With prof120region and small interior IS cache sizes, spying only on Reconnects has a clearcut advantage over spying on both Rewrites and Reconnects. This is not true for the other two profiles, prof120hybrid and prof120long; see

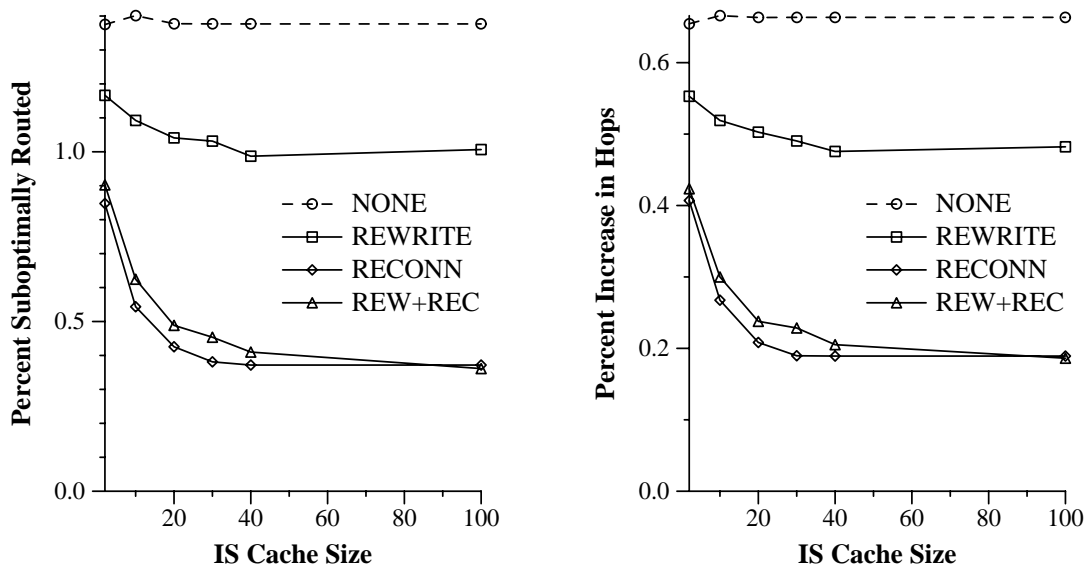


Figure 7.15: Experiment 5leveltree:params2+prof120region (hier-level = 3)

for example, Figure 7.16. Regardless, it was never the case, for *any* experiment performed for this caching study, that spying on both messages types had more than a marginal benefit over just spying on Reconnects. It certainly is not worth the resources that would be required to process transit Rewrite messages at interior ISs.

Another difference between Figure 7.16 and the earlier figures is in the ratio between percent datagrams suboptimally routed and percent increase in routing hops. In the earlier figures, a 1 percent value for suboptimally routed datagrams yields a .5 percent increase in routing hops, while in Figure 7.16, the ratio is approximately 1 to 1. The reason is that for prof120long, the penalty for sending to an out-of-date NSAP address is higher, since the destination ES's current location is likely to be a significant distance away.

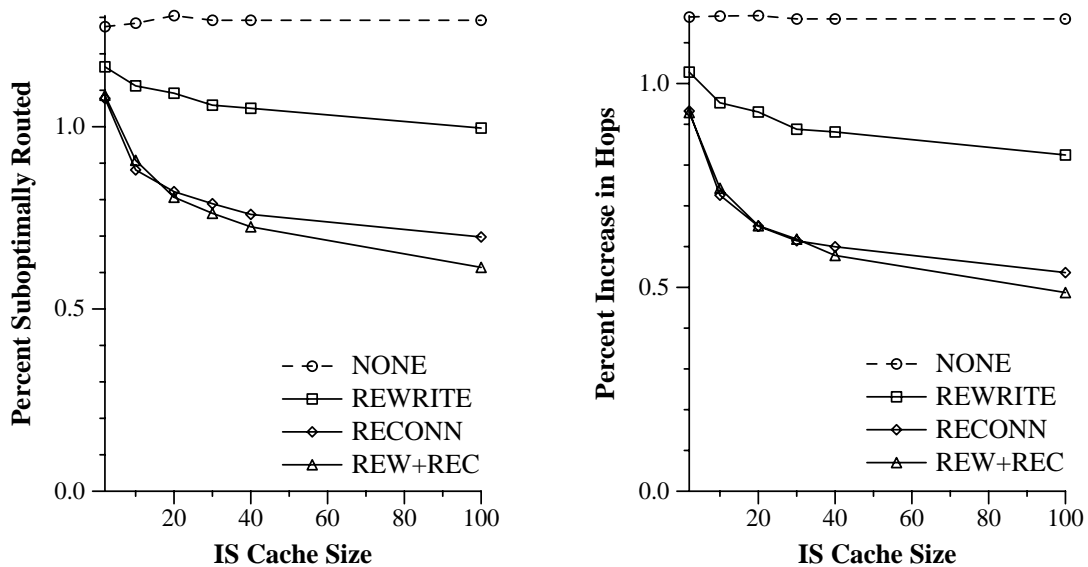


Figure 7.16: Experiment 2ringdistr:params3+prof120long

A conclusion can be stated. If caching is to be performed at interior ISs, spying on Reconnects is the correct policy. This policy will reduce the number of suboptimally routed datagrams, but since the basic mobility algorithm is quite effective and robust at optimally routing datagrams, the added benefit of caching at interior ISs is not dramatic.

7.5 Refinements to the Mobility Algorithm

Certain details of the mobility design presented in Section 4.5 resulted from intuition gained during the caching study. In this section, we explain the rationale for those details.

A surprising phenomenon exists where out-of-date forwarding pointer cache entries at ISs can spread via Rewrite messages to other ISs, “contaminating” the

caches at those ISs. It is possible that a valid cache entry will be deleted (due to finite cache size and LRU replacement) and later replaced by an out-of-date entry for the same mobile ES.

A small percentage of out-of-date cache entries will, when used to rewrite the destination NSAP address of a datagram, cause that datagram to eventually arrive at an IS that no longer maintains information concerning the location of the destination mobile ES; this is a known aspect of the mobility design. The IS drops the datagram and sends an Unreachable message to the source, and the source resends the datagram to the destination's home NSAP address. In rare cases, however, the same out-of-date cache entry that was on the path from the source to the destination NSAP address cached by the source is *also* on the path to that destination's home NSAP address. We now describe an example that actually occurred during one of the caching experiments. In this experiment, interior ISs were configured to spy on both Rewrite messages and Reconnect messages. Having a simulation environment was extremely valuable; had we been working with an experimental prototype, it is much less likely that the phenomenon would have been detected.

Initial conditions are illustrated in Figure 7.17. Mobile ES DEST relocated from area B to area C, incrementing its LSN from 4 to 5. Interior IS A spied on the Reconnect message sent by DEST to IS B, and updated its forwarding pointer cache accordingly, as shown in Figure 7.18. However, ISs have a finite

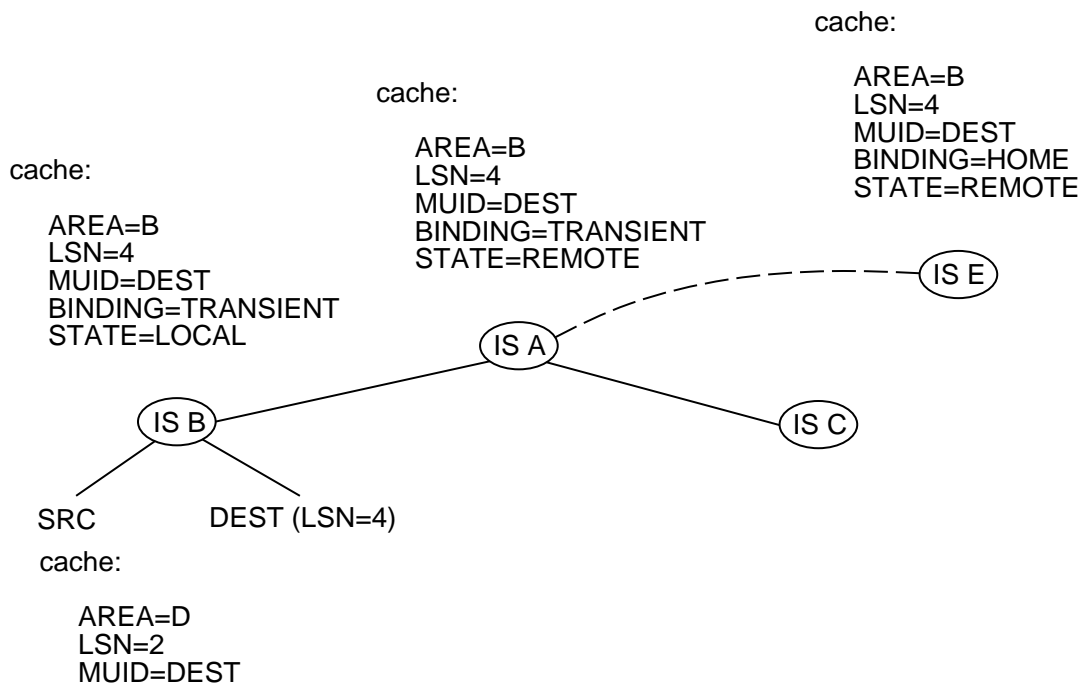


Figure 7.17: Initial Cache Contents

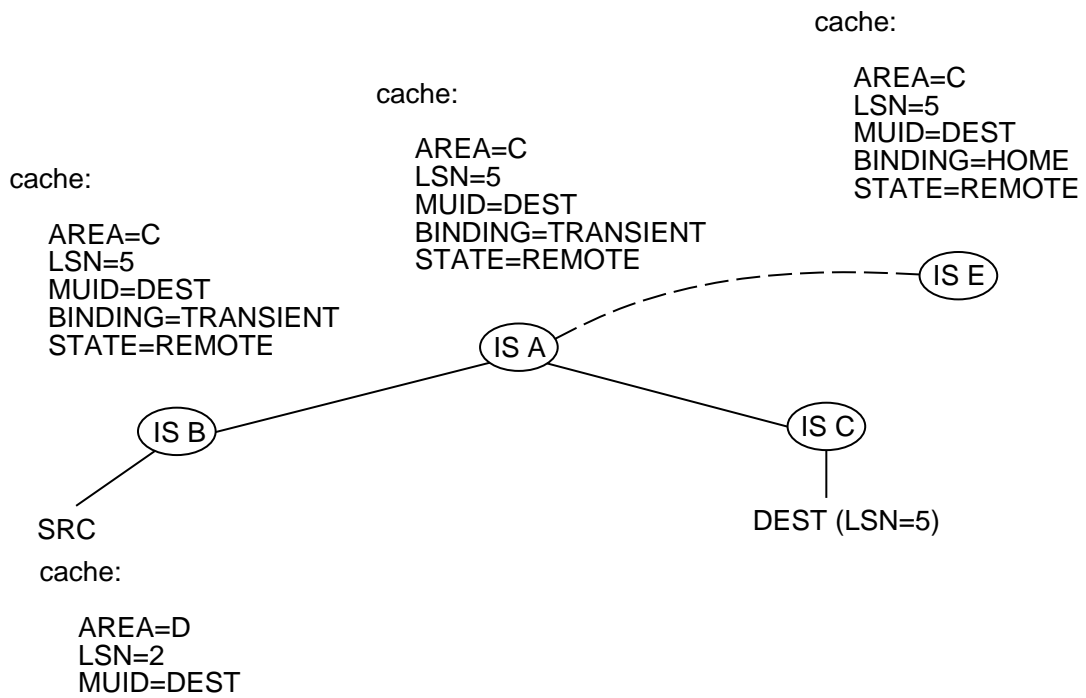


Figure 7.18: ES DEST Has Relocated from Area B to Area C

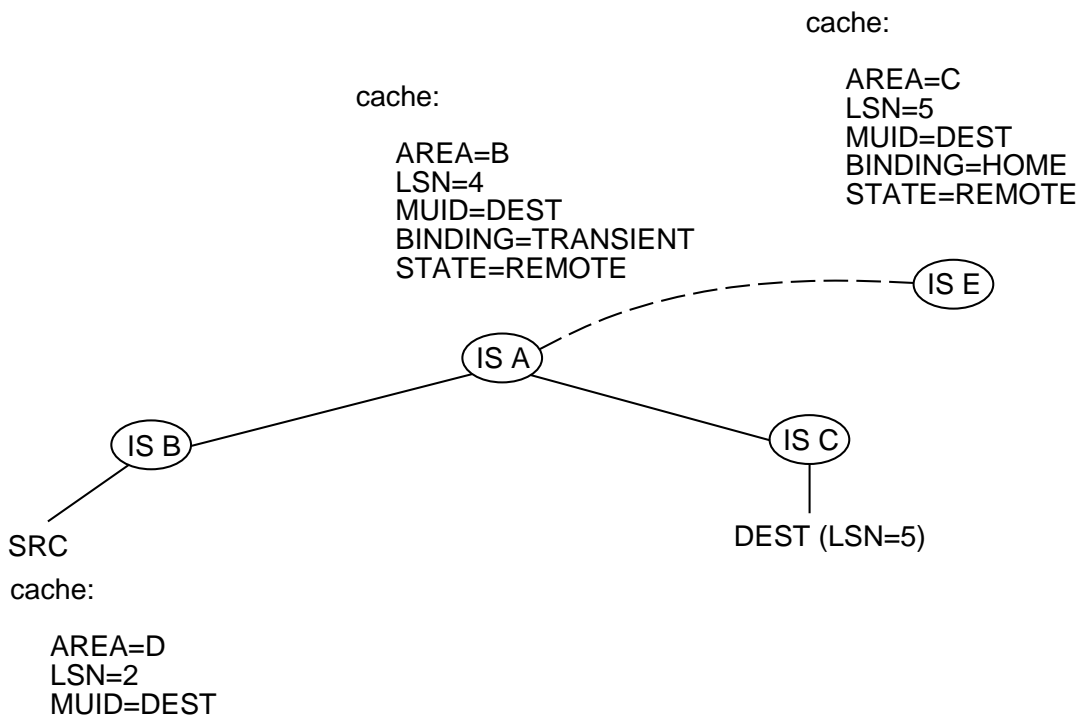


Figure 7.19: New Cache Contents at IS A and IS B

cache size for supplemental entries, and IS A's cache slot for DEST was eventually reused. IS B's cache slot for DEST was also reused. Later on, IS A spied on a Rewrite message that contained the NSAP address "DEST (LSN=4)," and recorded that NSAP address in its cache. The source of the Rewrite was an IS that had an out-of-date cache entry for DEST. Cache contents were now as shown in Figure 7.19. At this point, there is a serious problem. Assume that "DEST (LSN=1)" is DEST's home NSAP address, and that datagrams from ES SRC to areas D and E are routed via IS A.

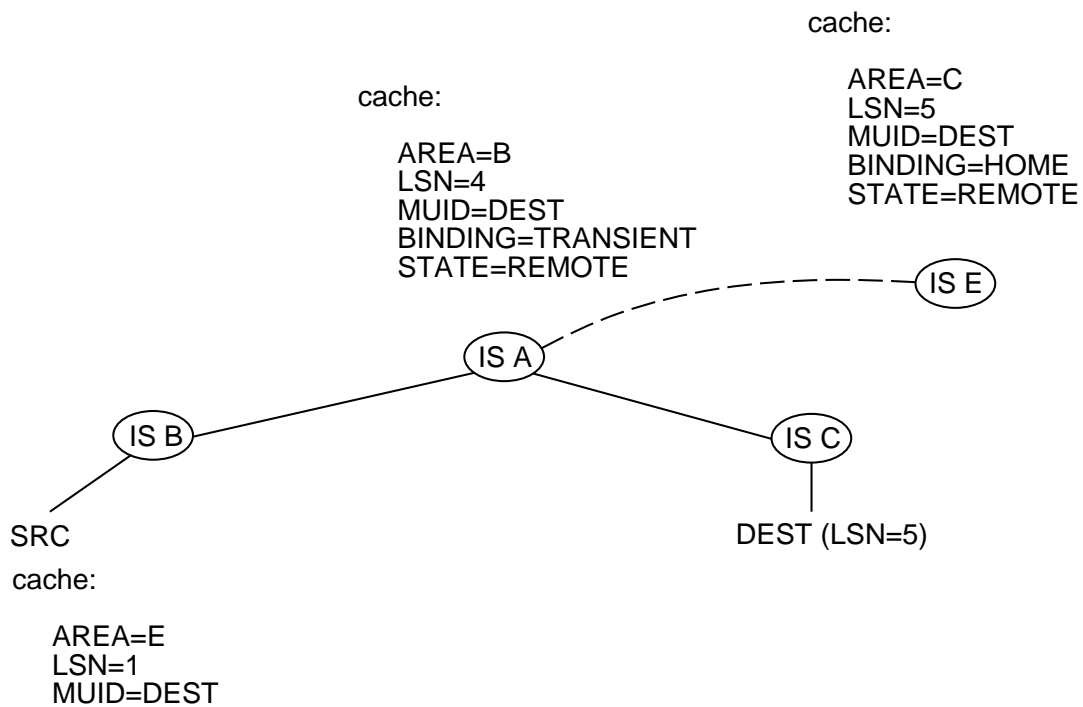


Figure 7.20: ES SRC Has Cached DEST's Home NSAP Address

ES SRC was not actively communicating with DEST at the time DEST last relocated, and thus has an out-of-date cache entry for DEST. SRC now

attempts to open a transport connection to DEST. The destination address of the Connection Request TPDU is rewritten by IS A to be “DEST (LSN=4),” and IS A forwards the datagram back to IS B. IS B detects the fact that DEST is not local, and since no forwarding pointer cache entry is available, it drops the datagram and sends an Unreachable message to SRC. Note that IS A does *not* see the Unreachable message, since IS A is not on the path by which the Unreachable is routed. Thus, IS A’s out-of-date cache entry for DEST will remain. Upon receiving the Unreachable message, SRC deletes its cache entry for DEST and replaces it with DEST’s home NSAP address, “DEST (LSN=1).” See Figure 7.20. When the retransmit timer fires, SRC sends a Connection Request TPDU addressed to DEST’s home NSAP address. Unfortunately, this new datagram *also* is rewritten by IS A to “DEST (LSN=4)” and forwarded to IS B. IS B drops the datagram and sends an Unreachable message to SRC. The cycle continues indefinitely, effectively rendering SRC unable to transmit to DEST, unless special measures are taken.

The example described above motivated the addition of a “rewrite by destination only” flag to the datagram header, plus enhancements to the ES recovery policy, as described in Section 4.5. Upon receiving an Unreachable message, an ES replaces its cache entry for the destination ES with the destination’s home NSAP address and marks the cache entry “do not rewrite.” When a datagram is next sent to the destination, the “do not rewrite” bit causes the source ES to set a “rewrite by destination only” flag in the datagram. This flag forces the

datagram to be routed to the Area Address specified in the destination NSAP address. Once the datagram reaches the specified area, an L1 IS in that area is allowed to rewrite the NSAP address and clear the “rewrite by destination only” flag. When SRC resends the Connection Request TPDU to DEST’s home NSAP address in the example described above, the ‘rewrite by destination only’ flag causes the out-of-date cache entry at IS A to be bypassed. SRC updates its cache entry for DEST and clears the “do not rewrite” bit upon receiving a datagram from DEST, or upon receiving a Rewrite message marked “original was rewrite by destination only.” In their enhancements to VIP, Uehara and colleagues [63] independently proposed a control bit with similar semantics to our “rewrite by destination only” flag.

The “rewrite by destination only” bit allows SRC to recover from the out-of-date cache entry stored at IS, but the mechanism does not necessarily cause that entry to be expunged permanently. In the particular example that we provided, a Rewrite message will be sent from DEST’s home area to SRC, via IS A and IS B, as illustrated in Figure 7.21. Since IS A “spied” on an earlier Rewrite message (which contained out-of-date information), it also spies on this Rewrite. For the moment, IS A’s cache will contain the current NSAP address for DEST, as illustrated in Figure 7.22. However, it is possible that the cache entry will be deleted later on, and then replaced by out-of-date information from a future Rewrite message. The cache entry at the IS that generated the Rewrite message will eventually age, due to LRU management; an active timeout

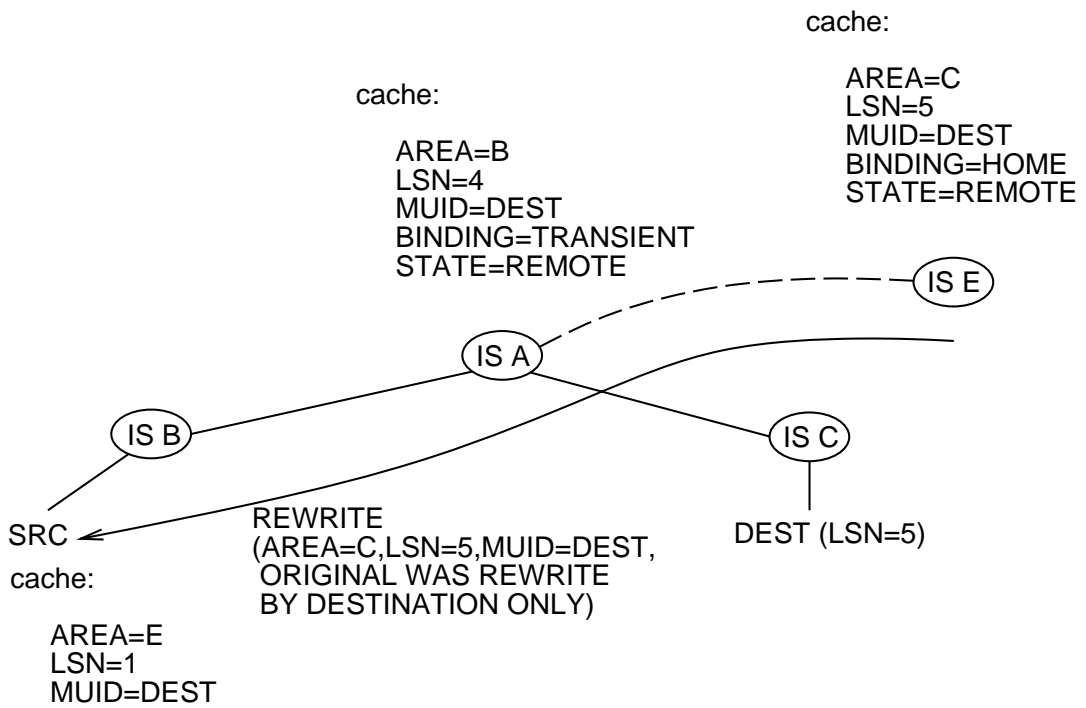


Figure 7.21: IS E Sends Rewrite Message to SRC

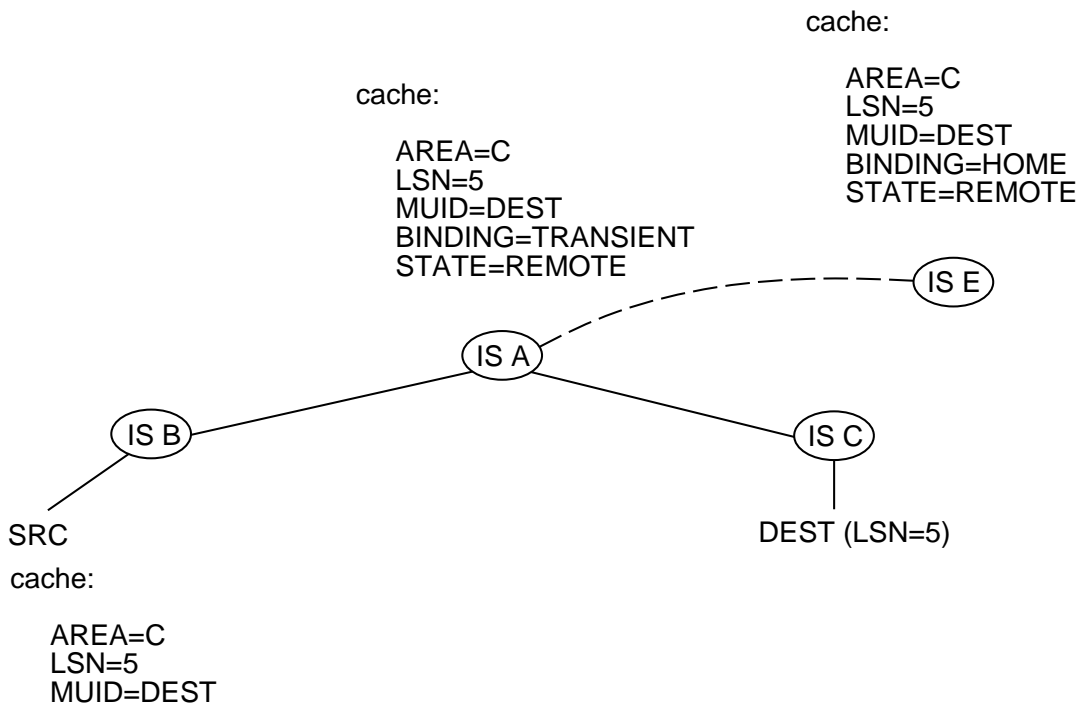


Figure 7.22: Cache Contents After Rewrite Has Been Processed

mechanism would also be a possibility. Uehara and colleagues described an elaborate mechanism where the identity of the last ES or IS to update the destination address of a datagram is recorded in the datagram's header. That ES or IS can be notified, should the datagram turn out to be undeliverable. Although this technique suggests an alternative means by which an out-of-date cache entry could be deleted at IS A, the technique would not solve the problem of IS A's later reacquiring an out-of-date NSAP address via a Rewrite message, since it would not affect the source of IS A's information. A reasonable conclusion is that interior ISs should not be configured to "spy" on transit Rewrite messages, especially since results of the caching study showed that there are more effective means to improve system performance.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

We have developed a new type of network layer address, the hybrid address. The hybrid address combines the best features of flat addresses and topology-dependent, hierarchical addresses, and provides an excellent base for supporting ES mobility. We presented a mobility design for the ISO OSI connectionless routing architecture that incorporates the hybrid address. As demonstrated by experiments in a simulation environment, the design behaves well across a variety of internetwork topologies and mobility scenarios. The basic design relies on caching at ESs and leaf ISs. A small number of datagrams (well under 1%) are dropped at a leaf IS due to the unavailability of a forwarding pointer; in such cases, the source ES is notified and takes appropriate action. The percentage of dropped datagrams is a function of leaf IS forwarding pointer cache size and

mobile ES relocation rate, and drops to zero as the IS cache size increases. An ES caches the current NSAP address of each correspondent ES with which it currently is communicating, and thus routing is normally optimal. At the very start of a transport connection, or immediately after a correspondent ES relocates during a transport connection, it is possible that datagrams will be suboptimally routed, until the source ES learns the correspondent's current NSAP address.

We conducted a study of caching methods at interior ISs, to investigate whether such caching could reduce the number of suboptimally routed datagrams. The study showed that the percentage of suboptimally routed datagrams can be reduced by having interior ISs create forwarding pointer cache entries based on the NSAP addresses contained in transit Reconnect messages. However, since our basic mobility algorithm is so robust, the impact of caching at interior ISs based on transit Reconnects is not dramatic. Caching based on transit Rewrite messages offers a much smaller benefit, and can lead to surprising problems with out-of-date cache entries spreading from one IS to other ISs.

Simulation was a good vehicle for trying out the mobility design and gathering results in a controlled environment. The next logical step is to experiment with the design in a real (versus simulated) environment, by modifying an existing ES implementation and an existing IS implementation of the ISO OSI

protocol suite. This will identify any compatibility problems with the existing protocols and software implementations, and serve to further validate the design. Other possible areas for future research are listed in the next section.

8.2 Areas for Future Research

In the context of a thesis, it was impossible to study *all* of the issues that relate to wide-area mobility. The intent in developing the research plan was to identify a coherent, interesting, and manageable subset of those issues, and to leave the remainder for future research. We conclude by listing several areas that remain to be addressed:

8.2.1 Intra-area Mobility and Replicated Home ISs

This thesis dealt with mobility between ISO OSI areas, but an area itself might be a moderately complex internetwork. Moves within an area are straightforward. Since the Area Address portion of a mobile ES's NSAP address does not change, the LSN does not need to be incremented, and it is not necessary to send a Reconnect message to the home area¹. In the standard ISO OSI architecture, the L1 ISs in an area directly track the system IDs of all ESs in that area. Therefore, datagrams destined to a mobile ES automatically will be routed to

¹Since the current area is the same as the previous area, it is also unnecessary to send a Reconnect to the previous area.

the correct subnetwork as the mobile ES moves within an area. However, in order for the tracking process to be successful, a L1 routing update must be issued each time that a mobile ES moves to another subnetwork. The frequency of such routing updates will eventually become burdensome as the number of mobile ESs in an area increases. A solution proposed by Tanaka and Tsukamoto [57] provides an approach to the problem, though one that *only* handles intra-area moves. The first L1 IS in an area to see a mobile ES declares itself to be the default neighbor IS of the ES, and advertises this information in an L1 routing update. When the mobile ES roams to another subnetwork in the same area, the local L1 IS informs the default neighbor IS that the ES has relocated, rather than issuing a routing update. If the default neighbor IS receives datagrams for the mobile ES, it tunnels them to the ES's current neighbor IS.

Another problem concerns the reliability and availability of L1 ISs at a mobile ES's home area. As presented, a mobile ES maintains an ES-IS relationship with *one* IS in the home area. This will either be a physical or virtual relationship, depending on whether the ES is currently home or roaming. It is possible that the particular L1 IS in the home area with which the ES communicates will fail or become unreachable. In such a case, responsibility for a roaming ES needs to be transferred to another L1 IS in the home area.

An L1 IS failure presents two sub-problems. First, datagrams destined for a mobile ES that arrive at the home area must be forwarded, even if the designated L1 IS fails. Second, the mobile ES needs to be able to contact an alternate L1

IS, in order to inform the home area of future ES relocations. We propose an extension to the Tanaka and Tsukamoto [57] solution that could handle the first sub-problem. In addition to advertising default neighbor designations, an L1 IS could advertise the current NSAP address associated with each ES for which it is acting as the default neighbor. Since a mobile ES's current NSAP address only changes when the ES crosses an area boundary, the frequency of L1 IS routing updates should be tolerable. If an L1 IS failed, another L1 IS in the area could act as a default neighbor for those mobile ESs that were associated with the failed IS, and continue forwarding datagrams to the ESs.

The mobile ES still needs to solve the second sub-problem, which is to be able to contact an alternate L1 IS, without having an address available. A possible solution would be to reserve certain values in the MUID field of an NSAP address for multicast purposes. One such reserved MUID value would mean "all L1 ISs" (or perhaps "any L1 IS") in an area. The Intra-Domain Routing and ES-IS protocols already reserve certain 6-byte *subnetwork* addresses for multicasting purposes; the same values might be used for internetwork multicasts. This would allow a mobile ES to contact an alternate L1 IS in the home area, should the ES's home IS stop responding.

The possibilities described above deserve further study.

8.2.2 Security and Authentication

If a pair of ESs require a high degree of confidence in each other's identity, authentication should be performed end-to-end, at the transport layer or higher in the protocol stack. Network layer authentication, which we will discuss here, can provide a limited benefit, but it is not a substitute for authentication at a higher protocol layer.

All datagram-based internetworks have security and authentication problems. ES mobility introduces surprisingly few new problems. We distinguish between the presumed and actual threats, and discuss possible approaches to the latter category. Bellare [5] provides an overview of the many security problems on the TCP/IP Internet. One of the more obvious-seeming problems with mobility over wireless links is vulnerability to eavesdropping. In fact, some wired Local Area Network (LAN) technologies are equally or more vulnerable to eavesdropping. Ethernet is a broadcast medium, where all ESs on the LAN tap into the same cable. Any ES can monitor all the traffic flowing across the Ethernet. A human "snooper" does not necessarily need to be local; it is possible to remotely access an ES via an internetwork, compromise its security, and then monitor all the traffic on the ES's LAN. In a major security breach of the TCP/IP Internet [20], this technique was employed to capture tens of thousands of user account names and passwords. Returning to the issue of wireless links, appropriate modulation techniques (such as spread spectrum) possibly combined with link layer encryption should provide sufficient security.

Another concern of potential users of mobile technologies is that of location privacy. By using the technology, will the user reveal his or her current geographic location? In the mobility design presented in this thesis, we specifically are *not* trying to hide the location of a mobile ES; a hybrid network address contains a partial destination identifier, and hence location, hint. While location privacy may be a valid concern for personal communications services, we do not consider it be a major issue for ES mobility.

There are two separate authentication problems. The source NSAP address of datagrams entering the internetwork from a mobile ES should carry an authenticated MUID. This is important, since an ES will cache the current NSAP address of correspondent mobile ESs, based on the source NSAP address of incoming datagrams. The second authentication problem is to ensure that forwarding pointer cache entries at ISs contain valid NSAP addresses for mobile ESs. Otherwise, datagrams could be misdirected to a rogue or nonexistent destination. Again, these problems are not unique to mobile environments; Bellovin describes several methods by which routing to ESs can be disrupted on the TCP/IP Internet.

Upon reconnecting to an internetwork, a mobile ES can authenticate itself to an L1 IS in the current area. A public key-based approach seems to be the most promising. It is probably not practical for the L1 IS to authenticate itself to the mobile ES, since the ES will not have the means to verify the credentials of an arbitrary IS. Leaf ISs at the periphery of an internetwork should not allow

datagrams from a mobile ES to enter the internetwork unless the source NSAP address has been authenticated.

It is feasible for a mobile ES and its home IS to mutually authenticate each other; this could also be accomplished with public key methods, or with a shared key. Reconnect messages could carry a public key-based authenticator, which would allow an IS receiving the message to determine that it indeed originated from the claimed MUID. The authenticator could be stored in the IS's forwarding pointer cache and propagated in future Rewrite messages; this is important, since the source of a Rewrite (an arbitrary IS) cannot easily be authenticated. Validating the authenticators contained in Rewrite and Reconnect messages would incur a performance penalty; the relative costs and benefits of such authentication techniques will need to be evaluated.

Playback attacks, where old traffic is recorded and later retransmitted, are a possibility. To combat such attacks, transactions between an ES and its home IS may need to incorporate a timestamp; the IETF Mobile IP working group is considering various possible approaches. The body of relevant literature is also growing. A recent paper by Molva, Samfat, and Tsudik [40] looks at authentication in environments with mobile users, and another paper by Aziz and Diffie [4] describes a protocol to provide privacy and authentication for wireless LANs.

8.2.3 Mobile Subnetworks

The mobility design that we have presented tracks individual mobile ESs as they move around an internetwork. Mobile subnetworks are also a possibility. The simplest method to handle such a subnetwork is to treat it as a collection of mobile ESs; however, this will become inefficient as the number of ESs on the subnetwork grows. If the ESs on a mobile subnetwork always travel together, a straightforward extension to our mobility design is possible. The MUID field could be partitioned into a group ID and an ES ID, with all the ESs on a subnetwork sharing the same group ID. The task of sending Reconnect messages could be delegated to the IS responsible for providing routing services to the subnetwork, and such messages would apply to the entire group of ESs. In an environment where partitioning of a mobile subnetwork is possible, or in which ESs move from one mobile subnetwork to another, the group ID proposal is not effective. Extending the mobility design to deal with general mobile subnetwork topologies, including nested mobile subnetworks, is an issue for further study.

Appendix A

Sample Input and Output Files

A.1 Mobility Scenario Generator

A.1.1 Sample Parameters File

```
managerdevm      101      # node number of the address manager
mesbase          120      # ES nodes number start at this value
nonleafismin     1        # IS nodes that don't host ESs are
nonleafismax     15      # consecutively numbered between
                  # nonleafismin and nonleafismax
leafismin        16      # IS nodes that host ESs are numbered
leafismax        31      # between leafismin and leafismax
mescachesize     1000    # max size of an ES cache
twinsize         2       # transport connection window size

mesdontcacheupdates
                  FALSE   # TRUE if ES should _not_ cache based on
                  # source adrs and Rewrite messages
printdenettop   FALSE   # TRUE if "Print;" should appear in the
                  # topology file produced by the mobility
```

```

# scenario generator
simtrace      FALSE # TRUE if DeNet should log EVERY event
mesprintcache FALSE # TRUE if an ES should print cache table
                # upon termination
mesprintprobes FALSE # TRUE if an ES should print probe data
                # upon termination
manprintaddr  FALSE # TRUE if manager should print address
                # table on termination
manrawcache   FALSE # TRUE if calls to PrintCacheTable
                # should print the raw cache
isprintcache  FALSE # TRUE if an IS should print cache table
                # upon termination
isprintprobes FALSE # TRUE if an IS should print probe data
                # upon termination
isspyatleaves FALSE # TRUE if a leaf IS should populate its
                # cache based on transit control messages

# The following parameters are all in units of seconds.

intermovetime 2.0 # average interval between ES moves
interconvtime 4.0 # on average, an ES waits this long
                # before starting another transport
                # connection
convlength    4.0 # average conversation length
trexmitintvl  0.3 # retransmit interval for transport
                # connections
trexmitforcr  0.3 # retransmit interval for
                # Connection Request TPDU only
intertmsgtime 0.2 # average interval between generation of
                # transport data messages
mesislatency  0.0005 # latency on an ES->IS link
mesisxmittime 0.004 # transmission time on an ES->IS link
ismeslatency  0.0005 # latency on an IS->ES link
ismesxmittime 0.004 # transmission time on an IS->ES link
isislatency   0.005 # latency on an IS->IS link
isisxmittime  0.001 # transmission time on an IS->IS link
simtime       40.0 # run simulation for this many time units

```

```

# The following parameters control the behavior of supplementary
# caching at IS nodes.  Although these parameters are
# technically part of a parameters file, we typically add them
# "on the fly," during processing.

leafiscache      20      # maximum size of a leaf IS's cache for
                    # non-permanent entries
nonleafiscache  20      # maximum size of a non-leaf IS's cache
                    # for non-permanent entries
isspyonreconn   FALSE   # populate IS cache based on transit
                    # Reconnect msgs?
isspyonrewrite  TRUE    # populate IS cache based on transit
                    # Rewrite msgs?
issupcachelevel 1      # supplementary IS caching at or below
                    # this hierarchical level

```

A.1.2 Sample Profile File

```

# declare groups of ES nodes
# each line lists a "logical" group number and that group's size.
# the mobility scenario generator will assign a range of ES node
# numbers to each group.
listgroups
  1 10
  2 10
endlist

# now, we need to define each group of ESs that has been declared
# above.

definegroup 1
16 20 24 28      # a list of IS node numbers to which all
                    # members of this ES group are connected
circulate        # movetype, "random" or "circulate"
# the communications list -- each line specifies an ES group
# number and the probability that the next transport connection

```

```

# will be opened to an ES belonging to that group. Note that
# once a target group has been selected, the particular ES
# in that group is selected randomly [uniform distribution].
# The probabilities must, of course, sum to 100.
1 60
2 40
enddefine

definegroup 2
17 21 25 29          # a list of IS node numbers to which all
                    # members of this ES group are connected
circulate            # movetype, "random" or "circulate"
2 60
1 40
enddefine

```

A.1.3 Sample Output File

```

(* Define the principal DEVM *)
0 := {
    40.0    (* number of simulated time units *)
    FALSE  (* log every event? *)
};

(* Declare and define the manager DEVM *)
101 := manager;
101 := {
    FALSE  (* print address table upon termination? *)
    FALSE  (* print raw cache tables? *)
};

(* Declare the interior ISs *)
1..15 := is;

(* Declare the leaf ISs *)
16..31 := is;

```

```

(* Define the ISs *)
1..15,16..31 := {
    0.005    (* link latency in seconds on an IS link *)
    0.001    (* transmission time in seconds on an IS link *)
    0.0005   (* link latency in seconds on an ES link *)
    0.004    (* transmission time in seconds on an ES link *)
    20       (* max num non-permanent cache entries for an
              * interior IS *)
    20       (* max num non-permanent cache entries for a
              * leaf IS *)
    FALSE    (* populate a leaf IS cache based on transit msgs? *)
    FALSE    (* populate cache based on transit Reconnect msgs? *)
    TRUE     (* populate cache based on transit Rewrite msgs? *)
    1        (* supplementary caching at or below this
              * hierarchical level *)
    FALSE    (* print cache table upon termination? *)
    FALSE    (* print probe data upon termination? *)
};

(* Declare ES group 1 *)
120..129 := mes;

(* Declare ES group 2 *)
130..139 := mes;

(* Define ES group 1 *)
(* Group is connected to ISs: 16 20 24 28 *)
120..129 := {
    4.0      (* average quiet time between conversations *)
    4.0      (* average conversation length, in seconds *)
    0.2      (* average interval between transport data
              * messages *)
    2        (* number of target communications groups *)
    (* percent probability for each communications group *)
    { 60 40 }
    (* lowest ES node number for each communications group *)
    { 120 130 }
    (* highest ES node number for each communications group *)
};

```

```

{ 129 139 }
2.0      (* average interval between ES moves *)
2        (* movetype [1=RANDOM, 2=CIRCULATE] *)
0.0005   (* link latency in seconds *)
0.004    (* transmission time in seconds*)
1000     (* maximum number of cache entries *)
FALSE    (* don't update cache based on src addr and Rewrite
          * msgs? *)
FALSE    (* print cache table upon termination? *)
FALSE    (* print probe data upon termination? *)
2        (* transport connection window size *)
0.3      (* retransmit interval for transport connections *)
0.3      (* retransmit interval for CR TPDU only *)
};

(* Define ES group 2 *)
(* Group is connected to ISs: 17 21 25 29 *)
130..139 := {
  4.0     (* average quiet time between conversations *)
  4.0     (* average conversation length, in seconds *)
  0.2     (* average interval between transport data
          * messages *)
  2       (* number of target communications groups *)
  (* percent probability for each communications group *)
  { 60 40 }
  (* lowest ES node number for each communications group *)
  { 130 120 }
  (* highest ES node number for each communications group *)
  { 139 129 }
  2.0     (* average interval between ES moves *)
  2       (* movetype [1=RANDOM, 2=CIRCULATE] *)
  0.0005  (* link latency in seconds *)
  0.004   (* transmission time in seconds*)
  1000    (* maximum number of cache entries *)
  FALSE   (* don't update cache based on src addr and Rewrite
          * msgs? *)
  FALSE   (* print cache table upon termination? *)
  FALSE   (* print probe data upon termination? *)
  2       (* transport connection window size *)

```

```

    0.3      (* retransmit interval for transport connections *)
    0.3      (* retransmit interval for CR TPDU only *)
};

```

```

(* Connect up ES group 1 *)
FOR node := 120 to 129 DO
    [ node | 16,20,24,28 ] := MsgEtoI;
    [ 16,20,24,28 | node ] := MsgItoE;
END;

```

```

(* Connect up ES group 2 *)
FOR node := 130 to 139 DO
    [ node | 17,21,25,29 ] := MsgEtoI;
    [ 17,21,25,29 | node ] := MsgItoE;
END;

```

A.2 Internetwork Topology and Routing Generator

A.2.1 Sample Input File

```

topname="4 level tree"
numnodes=15
begindefs
node 1/1: 2 3 end
node 2/2: 1 4 5 end
node 3/2: 1 6 7 end
node 4/3: 2 8 9 end
node 5/3: 2 10 11 end
node 6/3: 3 12 13 end
node 7/3: 3 14 15 end
node 8/4: 4 end
node 9/4: 4 end

```

```

node 10/4: 5 end
node 11/4: 5 end
node 12/4: 6 end
node 13/4: 6 end
node 14/4: 7 end
node 15/4: 7 end
enddefs

```

A.2.2 Sample Output File

```

(* IS topology name = "4 level tree" *)

(* Declare and define the router DEVM *)
102 := router;
102 := {
    15      (* number of nodes *)
    { (* hierarchical level array *)
        1 2 2 3 3 3 3 4 4 4 4 4 4 4 4
    }
    { (* nexthop matrix *)
        { -1 2 3 2 2 3 3 2 2 2 2 3 3 3 3 }
        { 1 -1 1 4 5 1 1 4 4 5 5 1 1 1 1 }
        { 1 1 -1 1 1 6 7 1 1 1 1 6 6 7 7 }
        { 2 2 2 -1 2 2 2 8 9 2 2 2 2 2 2 }
        { 2 2 2 2 -1 2 2 2 2 10 11 2 2 2 2 }
        { 3 3 3 3 3 -1 3 3 3 3 3 12 13 3 3 }
        { 3 3 3 3 3 3 -1 3 3 3 3 3 3 14 15 }
        { 4 4 4 4 4 4 4 -1 4 4 4 4 4 4 4 }
        { 4 4 4 4 4 4 4 4 -1 4 4 4 4 4 4 }
        { 5 5 5 5 5 5 5 5 5 -1 5 5 5 5 5 }
        { 5 5 5 5 5 5 5 5 5 5 -1 5 5 5 5 }
        { 6 6 6 6 6 6 6 6 6 6 6 -1 6 6 6 }
        { 6 6 6 6 6 6 6 6 6 6 6 6 -1 6 6 }
        { 7 7 7 7 7 7 7 7 7 7 7 7 7 -1 7 }
        { 7 7 7 7 7 7 7 7 7 7 7 7 7 7 -1 }
    }
}

```

```

{ (* distance matrix *)
  { 0 1 1 2 2 2 2 3 3 3 3 3 3 3 }
  { 1 0 2 1 1 3 3 2 2 2 2 4 4 4 }
  { 1 2 0 3 3 1 1 4 4 4 4 2 2 2 }
  { 2 1 3 0 2 4 4 1 1 3 3 5 5 5 }
  { 2 1 3 2 0 4 4 3 3 1 1 5 5 5 }
  { 2 3 1 4 4 0 2 5 5 5 5 1 1 3 }
  { 2 3 1 4 4 2 0 5 5 5 5 3 3 1 }
  { 3 2 4 1 3 5 5 0 2 4 4 6 6 6 }
  { 3 2 4 1 3 5 5 2 0 4 4 6 6 6 }
  { 3 2 4 3 1 5 5 4 4 0 2 6 6 6 }
  { 3 2 4 3 1 5 5 4 4 2 0 6 6 6 }
  { 3 4 2 5 5 1 3 6 6 6 6 0 2 4 }
  { 3 4 2 5 5 1 3 6 6 6 6 2 0 4 }
  { 3 4 2 5 5 3 1 6 6 6 6 4 4 0 }
  { 3 4 2 5 5 3 1 6 6 6 6 4 4 2 }
}
};

```

```

(* connect up the ISs *)
[ 1 | 2, 3 ] := MsgItoI;
[ 2 | 1, 4, 5 ] := MsgItoI;
[ 3 | 1, 6, 7 ] := MsgItoI;
[ 4 | 2, 8, 9 ] := MsgItoI;
[ 5 | 2, 10, 11 ] := MsgItoI;
[ 6 | 3, 12, 13 ] := MsgItoI;
[ 7 | 3, 14, 15 ] := MsgItoI;
[ 8 | 4 ] := MsgItoI;
[ 9 | 4 ] := MsgItoI;
[ 10 | 5 ] := MsgItoI;
[ 11 | 5 ] := MsgItoI;
[ 12 | 6 ] := MsgItoI;
[ 13 | 6 ] := MsgItoI;
[ 14 | 7 ] := MsgItoI;
[ 15 | 7 ] := MsgItoI;

```

Appendix B

Transport Protocol Details

This appendix provides various details concerning the reliable, simplex transport protocol that was designed and implemented as part of the internetwork simulation.

TPDU types are as follows:

Type and Acronym	Fields
Connection Request CR	srcref, destref=0, requested window size
Connection Confirm CC	srcref, destref, initial window allocation
Data DT	srcref, destref, seqno
Acknowledgement AK	srcref, destref, next seqno expected, window
Disconnect Request DR	srcref, destref
Disconnect Confirm DC	srcref, destref

For the Sending side of a connection, TPDU types that may be sent are CR, DT, and DR. States are Sent CR, Estab, and Sent DR. Retransmission information is as follows:

State	If retransmission timer fires
Sent CR	Send a duplicate CR
Estab	Resend the oldest unacknowledged DT TPDU
Sent DR	Send a duplicate DR

The transport protocol implementation can be configured to use a longer retransmission interval when in Sent CR state than for the other states. The retransmission interval is not adjusted dynamically.

State transition information for the Sending side of a connection is as follows (only the normal path is shown):

Cur. State	Input	New State	Action
[No State]	Open Request	Sent CR	Send CR
Sent CR	Receive CC	Estab	Send avail. DT until window full
Estab	New User Data	Estab	Send DT if window allows
Estab	Receive AK	Estab Estab Estab Sent DR	Send avail. DT until window full Set "all data sent" flag Send DR
new data is available			
reached connection length			
waiting for final AK			
	final AK received		
Sent DR	Receive DC	[No State]	

For the Receiving side of a connection, TPDU types that may be sent are CC, AK, and DC. States are Sent CC and Estab. No retransmission are performed. State transition information is as follows (only the normal path is shown):

Current State	Input	New State	Action
[No State]	CR	Sent CC	Send CC
Sent CC	CR	Sent CC	Re-send CC
Sent CC	DT	Estab	Send AK
Estab	DT	Estab	Send AK
Estab	DR	[No State]	Send DC

For all ESs, if a TPDU other than a CR or DC arrives on a connection (i.e., a srcref, destref pair) for which there is no state, a DC is generated. A CR causes a new connection to be established, and to prevent “DC wars,” a reply is never made to a DC.

Bibliography

- [1] ARDIS Company, Lincolnshire, Illinois. *ARDIS Network Overview*, February 1991.
- [2] ARDIS Company, Lincolnshire, Illinois. *ARDIS Network Connectivity Guide*, March 1992.
- [3] *ATN Manual (Version 2.0)*, 4 August 1993. Only chapters 5–12 were available.
- [4] Ashar Aziz and Whitfield Diffie. Privacy and authentication for wireless local area networks. *IEEE Personal Communications*, 1(1):25–31, First Quarter 1994.
- [5] S. M. Bellovin. Security problems in the TCP/IP protocol suite. *Computer Communication Review*, 19(2):32–48, April 1989.
- [6] Tom Bowen, Gita Gopal, Gary Herman, and William Mansfield. A scale database architecture for network services. *IEEE Communications Magazine*, 29(1):52–59, January 1991.
- [7] Allan Bricker, Michael Litzkow, and Miron Livny. *Condor Technical Summary (Version 4.1b)*. University of Wisconsin-Madison Computer Sciences Department, Madison, Wisconsin, 1992.
- [8] Kenneth G. Carlberg. A routing architecture that supports mobile end systems. In *Proceedings of MILCOM '92*, volume 1 (of 3), pages 6.3.1–6.3.6, San Diego, 11–14 October 1992.
- [9] CCITT Recommendation X.500 (1988). The Directory — Overview of concepts, models, and services. In *CCITT Blue Book*, volume VIII, fascicle VIII.8. ITU, Geneva, 1989.

- [10] *Cellular Digital Packet Data System Specification, Release 1.0*, 19 July 1993.
- [11] Lyman Chapin. Personal communication. BBN Communications, May 1991.
- [12] Rich Cox. Personal communication (via telephone). AT&T Bell Laboratories, October 1990.
- [13] Yogen K. Dalal and Robert S. Printis. 48-bit absolute internet and ethernet host numbers. In *Proceedings of the Seventh Data Communications Symposium*, pages 240–245, Mexico City, 27–29 October 1981.
- [14] S. E. Deering. Internetwork routing for mobile computers. Oral presentation, Symposium on Wireless Access to Distributed Computing, Columbia University Center for Telecommunications Research, 15 February 1991.
- [15] Irwin Dorros. Calling people, not places. *Communications Week*, page 12, 3 September 1990.
- [16] Irwin Dorros. Making PNC happen. *Communications Week*, page 18, 10 September 1990.
- [17] EIA/IS-41.2. Cellular radiotelecommunications intersystem operations: Intersystem handoff. Interim standard, Electronics Industries Association, Washington, D.C., February 1988.
- [18] EIA/IS-41.3. Cellular radiotelecommunications intersystem operations: Automatic roaming. Interim standard, Electronics Industries Association, Washington, D.C., February 1988.
- [19] Chip Elliott. Personal communication. BBN Systems and Technologies Division, April 1994.
- [20] James Ellis, Barbara Fraser, and Linda Pesante. Keeping Internet intruders away. *UNIX Review*, 12(9):35–44, September 1994.
- [21] Peter S. Ford, Yakov Rekhter, Mark Knopper, and Richard Colella. TUBA: CLNP as IPng. *ConneXions—The Interoperability Report*, 8(5):28–33, May 1994.

- [22] Robert Joseph Fowler. Decentralized object finding using forwarding addresses. Technical Report 85-12-1, University of Washington Department of Computer Science, Seattle, WA, December 1985.
- [23] A. G. Fraser. A uniform naming plan for Datakit networks. Technical Memorandum (unpublished), AT&T Bell Laboratories, 27 April 1987.
- [24] John Gough. *Gardens Point Modula Language Reference Manual*. Queensland University of Technology, 16 July 1992.
- [25] Robert D. Hof and Peter Coy. Step one for Craig McCaw's national cellular network. *Business Week*, pages 108–112, October 1990.
- [26] SRI International. Network reconstitution protocol. Final Technical Report RADC-TR-87-38, Rome Air Development Center, Griffis Air Force Base, NY, June 1987.
- [27] John Ioannidis, Dan Duchamp, and Gerald Q. Maguire Jr. IP-based protocols for mobile internetworking. In *Proceedings SIGCOMM 91*, pages 235–245, Zurich, 3–6 September 1991.
- [28] John Ioannidis and Gerald Q. Maguire Jr. The design and implementation of a mobile internetworking architecture. In *Proceedings of the Winter 1993 USENIX Conference*, pages 491–502, San Diego, 25–29 January 1993.
- [29] ISO/IEC 10589. Information technology — telecommunications and information exchange between systems — intermediate system to intermediate system intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode network service (ISO 8473). International standard, International Organization for Standardization, Geneva, 1992.
- [30] ISO/IEC 10747. Information processing systems — telecommunications and information exchange between systems — protocol for exchange of inter-domain routing information among intermediate systems to support forwarding of ISO 8473 PDUs. International standard, International Organization for Standardization, Geneva, 1993.
- [31] ISO/IEC 8073. Information processing systems — open systems interconnection — connection oriented transport protocol specification. International standard, International Organization for Standardization, Geneva, 1992.

- [32] ISO/IEC 8348:1987/Addendum 2. Information processing systems — data communications — network service definition Addendum 2: Network layer addressing. International standard, International Organization for Standardization, Geneva, 1987.
- [33] ISO/IEC 8473. Information processing systems — data communications — protocol for providing the connectionless-mode network service. International standard, International Organization for Standardization, Geneva, 1988.
- [34] ISO/IEC 9542. End system to intermediate system routing exchange protocol for use in conjunction with the protocol for providing the connectionless-mode network service (ISO 8473). International standard, International Organization for Standardization, Geneva, 1988.
- [35] David B. Johnson. Mobile host internetworking using IP loose source routing. Technical Report CMU-CS-93-128, Carnegie Mellon University School of Computer Science, Pittsburgh, PA, February 1993.
- [36] John Jubin and Janet D. Tornow. The DARPA packet radio network protocols. *Proceedings of the IEEE*, 75(1):21–32, January 1987.
- [37] G. S. Lauer. Advanced protocols for the SURAN packet radio network. BBN Systems and Technologies Division.
- [38] Miron Livny. *DeNet User's Guide (Version 1.5)*. University of Wisconsin-Madison Computer Sciences Department, Madison, Wisconsin, 1990.
- [39] Rajesh Mansharamani and Miron Livny. A performance study of forwarding address schemes. Unpublished report, University of Wisconsin-Madison Computer Sciences Department, Madison, Wisconsin, 17 October 1990.
- [40] Refik Molva, Didier Samfat, and Gene Tsudik. Authentication of mobile users. *IEEE Network*, 8(2):26–34, March–April 1994.
- [41] Andrew Myles and David Skellern. Comparing four IP based mobile host protocols. *Computer Networks and ISDN Systems*, 26(3):349–355, November 1993.
- [42] Andrew Myles and David Skellern. Comparison of mobile host protocols for IP. *Internetworking: Research and Experience*, 4(4):175–194, December 1993.

- [43] Charles Perkins. Providing continuous network access to mobile hosts using TCP/IP. *Computer Networks and ISDN Systems*, 26(3):357–369, November 1993.
- [44] Charles E. Perkins. Mobile IP as seen by the IETF. *ConneXions—The Interoperability Report*, 8(3):2–20, March 1994.
- [45] James S. Plank. Jgraph — a filter for plotting graphs in PostScript. In *Proceedings of the Winter 1993 USENIX Conference*, pages 61–66, San Diego, 25–29 January 1993.
- [46] J. B. Postel. Internet protocol. Internet Working Group Request for Comments 791, DARPA, September 1981.
- [47] J. B. Postel. Transmission control protocol. Internet Working Group Request for Comments 793, DARPA, September 1981.
- [48] Michael L. Powell and Barton P. Miller. Process migration in DEMOS/MP. In *Proceedings 9th ACM Symposium on Operating Systems Principles*, pages 110–119, Bretton Woods, NH, October 1983.
- [49] RAM Mobile Data, Woodbridge, New Jersey. *RAM Mobile Data System Overview (Release 4.1)*, 30 September 1993.
- [50] Y. Rekhter and Richard Colella. TUBA mobility support. Network Working Group Internet Draft draft-ietf-tuba-mobility-00.txt, IBM T. J. Watson Research Center, 16 May 1994. Working Draft; expires 16 November 1994.
- [51] R. Rivest. The MD5 message-digest algorithm. Internet Working Group Request for Comments 1321, MIT Laboratory for Computer Science and RSA Data Security, Inc., April 1992.
- [52] Dave Sanford. Air/Ground routing protocol options. Aeronautical Radio, Inc., 27 February 1991.
- [53] D. Sheinbein and R. P. Weber. 800 Service using SPC network capability. *The Bell System Technical Journal*, 61(7):1737–1744, September 1982.
- [54] W. A. Simpson. IP mobility support. Network Working Group Internet Draft draft-ietf-mobileip-protocol-04.txt, Computer Systems Consulting Services, Madison Heights, Michigan, June 1994. Working Draft; expires December 1994.

- [55] W. David Sincoskie and Charles J. Cotton. Extended bridge algorithms for large networks. *IEEE Network*, 2(1):16–24, January 1988.
- [56] C. A. Sunshine and J. B. Postel. Addressing mobile hosts in the ARPA Internet environment. IEN 135, University of Southern California Information Sciences Institute, Marina del Rey, California, March 1980.
- [57] Rieko Tanaka and Masahiko Tsukamoto. A CLNP-based protocol for mobile End Systems within an area. In *Proceedings of the First International Conference on Network Protocols (ICNP 93)*, pages 64–71, October 1993.
- [58] Andrew S. Tanenbaum. *Computer Networks*. Prentice Hall, Englewood Cliffs, New Jersey, second edition, 1988.
- [59] Fumio Teraoka. VIP: a protocol providing host migration transparency. *Internetworking: Research and Experience*, 4(4):195–221, December 1993.
- [60] Fumio Teraoka, Kim Claffy, and Mario Tokoro. Design, implementation, and evaluation of virtual internet protocol. In *Proceedings of the 12th International Conference on Distributed Computing Systems (ICDCS '92)*, pages 170–177, Yokohama, 9–12 June 1992.
- [61] Fumio Teraoka and Mario Tokoro. Host migration transparency in IP networks: The VIP approach. *Computer Communication Review*, 23(1):45–65, January 1993.
- [62] Fumio Teraoka, Yasuhiko Yokote, and Mario Tokoro. IP-based protocols for mobile internetworking. In *Proceedings SIGCOMM 91*, pages 209–220, Zurich, 3–6 September 1991.
- [63] Keisuke Uehara, Fumio Teraoka, Hideki Sunahara, and Jun Murai. Enhancement of VIP and its evaluation. In *Proceedings of INET '93*, pages BEA-1—BEA-10, San Francisco, 17–20 August 1993.
- [64] Hiromi Wada, Takashi Yozawa, Tatsuya Ohnishi, and Yasunori Tanaka. Mobile computing environment based on internet packet forwarding. In *Proceedings of the Winter 1993 USENIX Conference*, pages 503–517, San Diego, 25–29 January 1993.
- [65] Jan-Michael Wyckoff. Personal communication (via telephone). ARDIS Company, Lincolnshire, Illinois, 6 July 1994.