Quantification, Characterisation and Impact Evaluation of Mobile IPv6 Handoff Times

Mai Banh

Centre for Advanced Internet Architectures (CAIA)
Faculty of Information and Communication Technologies
Swinburne University of Technology
Melbourne, Australia

Masters of Engineering by Research

May 2005
Abstract

There is a growing range of IP-based data and voice applications using mobile devices (e.g. 3rd, 4th generation mobile phones and PDAs) and new access technologies (e.g. Bluetooth, 802.11, GPRS, ADSL). This growth is driving a desire to support mobility at the IP level – in other words, allowing an IP host to keep on communicating with other hosts while roaming between different IP subnetworks.

Mobile IPv6 allows hosts to move their physical and topological attachment points around an IPv6 network while retaining connectivity through a single, well-known Home Address. Although Mobile IPv6 has been the subject of simulation studies, the real-world dynamic behavior of Mobile IPv6 is only gradually being experimentally characterised and analysed.

This thesis reviews the use of Mobile IPv6 to support mobility between independent 802.11b-attached IPv6 subnets, and experimentally measures and critically evaluates how long an end to end IP path is disrupted when a Mobile IPv6 node shifts from one subnetwork to another (handoff time). The thesis describes the development of an experimental testbed suitable for gathering real-world Mobile IPv6 handoff data using publicly available, standards-compliant implementations of Mobile IPv6. (An open-source Mobile IPv6 stack (the KAME release under FreeBSD) was deployed).

The component of handoff time due to 802.11b link layer handoff is measured separately to assess its impact on the overall Mobile IPv6 handoff time. Using Mobile IPv6 handoff results, the likely performance impact of Mobile IPv6 handoff on a common webcam application and a bulk TCP data transfer is also evaluated. The impact of handoff on these applications clearly shows that a default Mobile IPv6 environment would be highly disruptive to real-time and interactive applications during handoff events, even if the underlying link-layer handoff was instantaneous.
Acknowledgements

The research work described in this thesis was performed while I was a candidate for the degree of Masters by Research in the Centre for Advanced Internet Architectures (CAIA) in Faculty of Information and Communication Technologies, Swinburne University of Technology.

I would like to express my gratitude to my supervisors Associate Professor Grenville Armitage and Associate Professor Jim Lambert of CAIA, Swinburne University of Technology, who have given continuing support and guidance throughout the course of the research work. Associate Professor Armitage has introduced me to a great field of advanced Telecommunication. With his wide knowledge in the field, he has provided me with valuable and constructive comments towards my research. Associate Professor Armitage and Associate Professor Lambert have given me detailed and constructive comments towards the revisions of my thesis chapters. I also would like to thank them for their patience and encouragement.

Finally, I wish to thank Gillian Foster in CAIA, my husband, Hung, and all of my friends who gave me love and support during the time I undertake this research.
Declaration

This thesis contains no material which has been accepted for the award of any other degree or diploma, except where due reference is made in the text of the thesis. To the best of my knowledge, this thesis contains no material previously published or written by another person except where due reference is made in the text of the thesis.

Mai Banh
Melbourne
May 2005
## Contents

1 Introduction ............................................................................................................................................... 1
   1.1 Why Mobile IP? ..................................................................................................................................... 1
   1.2 Objectives of This Thesis .................................................................................................................... 4
      1.2.1 A Key Technical Issue with Mobile IPv6 ...................................................................................... 4
      1.2.2 The Research Described in This Thesis ....................................................................................... 5
      1.2.3 Publications Related to the Study Presented in This Thesis ......................................................... 6

2 Evolving from IPv4 to IPv6 ................................................................................................................... 7
   2.1 Introduction ........................................................................................................................................... 7
   2.2 IPv4 Fundamentals ............................................................................................................................... 8
      2.2.1 IPv4 Address Format ...................................................................................................................... 8
      2.2.2 IPv4 Header Format ....................................................................................................................... 10
      2.2.3 IPv4 Issues ...................................................................................................................................... 11
   2.3 IPv6 Fundamentals ............................................................................................................................. 12
      2.3.1 IPv6 Address Format ...................................................................................................................... 12
      2.3.2 IPv6 Address Prefix Reserved for Documentation .......................................................................... 14
      2.3.3 IPv6 Header Format ....................................................................................................................... 15
      2.3.4 IPv6 Neighbor Discovery ................................................................................................................ 18
      2.3.5 Improvements over Version 4 .......................................................................................................... 19
   2.4 Transition Mechanisms (IPv4 to IPv6) ............................................................................................... 19
      2.4.1 Dual IP Layer .................................................................................................................................. 20
      2.4.2 IPv6 over IPv4 Tunneling ............................................................................................................... 20
   2.5 QoS Protocols for IP Networking ...................................................................................................... 21
3 Mobile IP Development and Mobile IPv4 Issues .................................................................22
  3.1 Mobile IP Development..............................................................................................22
    3.1.1 Mobile IP ..........................................................................................................22
    3.1.2 Mobile IP Handoff............................................................................................24
    3.1.3 Mobile IP Handoff Types .................................................................................26
    3.1.4 Basic Mobile IP Requirements and Strategies..................................................28
  3.2 IP Mobility Support for IPv4 (Mobile IPv4) ..............................................................29
    3.2.1 Mobile IPv4 Operations....................................................................................30
    3.2.2 Mobile IPv4 Issues and Extensions...............................................................31
    3.2.3 Mobile IPv4 Security........................................................................................35
    3.2.4 IPv6 over Mobile IPv4......................................................................................35
  3.3 Impact of Mobile IP Handoff on Higher Protocols.....................................................35
    3.3.1 Impact of Mobile IP Handoff on the Performance of TCP...............................36
  3.4 Mobile IP QoS Issues .................................................................................................37
    3.4.1 Issues ................................................................................................................37
    3.4.2 Requirements of a Quality of Service (QoS) Solution for Mobile IP...............37

4 Mobile IPv6 – Improvements over Mobile IPv4 .................................................................39
  4.1 Mobile IPv6 Technologies..........................................................................................39
    4.1.1 Basic Mobile IPv6 Concepts ............................................................................39
    4.1.2 Mobile IPv6 vs. Mobile IPv4............................................................................41
    4.1.3 Mobile IPv6 Tunneling vs. Route Optimisation...............................................42
    4.1.4 Link Layer Information ....................................................................................44
    4.1.5 Mobile IPv6 Security........................................................................................45
  4.2 Mobile IPv6 Handoff Operations................................................................................46
    4.2.1 Movement Detection ........................................................................................47
    4.2.2 Forming New Care-of Addresses .....................................................................48
    4.2.3 Binding Management.......................................................................................49
    4.2.4 Returning Home ...............................................................................................49
    4.2.5 Summary of Mobile IPv6 Handoff Operations.................................................50
  4.3 Mobile IPv6 Issues......................................................................................................50
    4.3.1 Issues with Information in Received RA..........................................................51
    4.3.2 Delay in NUD Mechanisms..............................................................................52
    4.3.3 Hints..................................................................................................................52
  4.4 Basic Support for Network Mobility ..........................................................................52
    4.4.1 Overview of Network Mobility ........................................................................52
    4.4.2 Mobile IPv6 Network Mobility Basic Support Protocol .................................53
5 A Literature Review on Mobile IPv6 Handoff Techniques ................................................. 55
  5.1 Mobile IPv6 Movement Detection Optimisation Techniques ........................................ 55
    5.1.1 Movement Detection for Mobile IPv6 in general .................................................. 55
    5.1.2 Movement Detection for Mobile IPv6 over 802.11 Networks ............................... 56
  5.2 Fast Handoff and Smooth Handoff Support .............................................................. 57
    5.2.1 Multiple CoA Registrations in Mobile IPv6 ......................................................... 57
    5.2.2 Fast Handoff in Mobile IPv6 .............................................................................. 57
    5.2.3 Simultaneous Binding with Bi/n-casting in Mobile IPv6 ..................................... 62
  5.3 Hierarchical Mobile IPv6 (HMIPv6) .......................................................................... 63
    5.3.1 Routing Scheme for Macro Mobility Handoff in HMIPv6 ................................. 65
    5.3.2 Fast Handoff for HMIPv6 .................................................................................. 66

6 Wireless IEEE 802.11 Link Layer Technology ................................................................. 68
  6.1 Why 802.11? ............................................................................................................. 68
  6.2 Introduction to IEEE 802.11 Networks ...................................................................... 68
    6.2.1 Overview of IEEE 802.11 Architecture ............................................................... 69
    6.2.2 Security ................................................................................................................ 72
  6.3 IEEE 802.11 Link Layer Handoff .............................................................................. 73
    6.3.1 Link Layer Handoff Process ................................................................................ 73
    6.3.2 802.11 Link Layer Trigger .................................................................................. 74
7 Experimental Evaluation of 802.11b and Mobile IPv6 Handoff Times........................................76
  7.1 IEEE 802.11b Handoff Experiments ....................................................................................76
    7.1.1 Aim ..................................................................................................................................76
    7.1.2 Methodology ..................................................................................................................77
    7.1.3 Testbed ..............................................................................................................................79
    7.1.4 Trials ..................................................................................................................................80
    7.1.5 Results ...............................................................................................................................81
    7.1.6 Discussion of 802.11b Handoff Time Results .................................................................85
    7.1.7 Related 802.11b Handoff Time Research ........................................................................86
  7.2 Measured Mobile IPv6 Handoff Time over IEEE 802.11b ..................................................90
    7.2.1 Aim ..................................................................................................................................90
    7.2.2 Methodology ..................................................................................................................90
    7.2.3 Testbed ..............................................................................................................................91
    7.2.4 Trials ..................................................................................................................................92
    7.2.5 Results ...............................................................................................................................93
    7.2.6 KAME Components and Their Impact on Handoff Delay ............................................101
    7.2.7 Related Mobile IPv6 Handoff Time Research ...............................................................105
    7.2.8 Discussions of the Experimental Mobile IPv6 Handoff Time Results .........................107

8 Experimental Analysis of Implications for Networked Applications .......................................109
  8.1 Video Conference Performance over Mobile IPv6 Handoff .............................................110
    8.1.1 Methodology ...................................................................................................................110
    8.1.2 Testbed ............................................................................................................................111
    8.1.3 Results ..............................................................................................................................112
  8.2 TCP Throughput Performance over Mobile IPv6 Handoff ...............................................113
    8.2.1 Methodology ...................................................................................................................113
    8.2.2 Testbed ............................................................................................................................114
    8.2.3 Results ..............................................................................................................................114
    8.2.4 Simulation .......................................................................................................................115
    8.2.5 Discussion .........................................................................................................................117
Lists of Figures

2.1 A Network Diagram .......................................................................................................... 9
2.2 IPv4 Header Format ......................................................................................................... 10
2.3 IPv6 Header Format ......................................................................................................... 15
2.4 Examples of the Next Field in Extension Headers ............................................................. 17
3.1 Basic Mobile IP Elements ................................................................................................. 23
3.2 Format of IP packet after being encapsulated by Home Agent ........................................ 24
3.3 Mobile IPv4 Elements ..................................................................................................... 29
3.4 Basic Mobile IPv4 Handoff Procedures ............................................................................ 31
3.5 Mobile IP Routing with Reverse Tunnel ......................................................................... 32
3.6 Mobile IP Routing with Route Optimisation ..................................................................... 33
4.1 Basic Mobile IPv6 Elements ............................................................................................. 40
4.2 MN Terminated Packet Delivery in Route Optimisation .................................................. 43
4.3 MN Originated Packet Delivery in Route Optimisation ................................................... 44
4.4 Binding Update Format to CNs ...................................................................................... 44
4.5 Default Mobile IPv6 Handoff Steps ................................................................................ 46
4.6 Basic Network Mobility Elements .................................................................................. 53
4.7 Mobile IPv6 Network Mobility Basic Support Protocol ................................................ 54
5.1 Fast Handoff Operations in Predictive Mode ................................................................. 59
5.2 Fast Handoff Operation in Reactive Mode ...................................................................... 59
5.3 Access Router Based Fast Handoff Sending RA with Generated CoA ......................... 60
5.4 FHA Uses the Tunneling to Forward the Packet to Subnet B Link ................................ 62
5.5 Hierarchical Mobile IPv6 Concepts ................................................................................. 64
5.6 A Scenario of Routing Scheme for Macro Mobility Handoff in HMIPv6 ....................... 65
6.1 802.2 & 802.11 Data Link and Physical Layers ................................................................. 70
6.2 A Typical 802.11 WLAN ................................................................................................. 72
6.3 802.11b Handoff Operations ......................................................................................... 74
7.1 Time Diagram of Ping Packets between the IPv4 Host and the 802.11b Client .............. 78
7.2 Testbed for 802.11b Link Layer Handoff ..................................................................... 79
7.3 Handoff Times, Varying Access Point Power Levels, Method 1: Access Points on Same Channel (Excludes “Anomaly” Points) ..................................................... 82
Lists of Tables

7.1 Summary of 802.11b Handoff Times ................................................................. 85
7.2 Mobile IPv6 Handoff Results Including Link Layer Handoff .............................. 97
7.3 Component of Handoff due Solely to Mobile IPv6 (Trial 1 and 2) ......................... 97
7.4 Component of Handoff due Solely to Mobile IPv6 (Trial 3) .............................. 99
7.5 Comparisons of Mobile IPv6 Handoff Times from Different Studies .................. 106
8.1 Comparison of Simulated and Experimental Bulk Data Transfer Bandwidths ......... 116
1 Introduction............................................................................................................................. 1

1.1 Why Mobile IP?.................................................................................................................. 1

1.2 Objectives of This Thesis ............................................................................................... 4

1.2.1 A Key Technical Issue with Mobile IPv6................................................................. 4

1.2.2 The Research Described in This Thesis................................................................. 5

1.2.3 Publications Related to the Study Presented in This Thesis................................. 6
Chapter 1

Introduction

In the early days of the Internet’s development a decision was made that Internet protocol (IP) addresses would represent both the topological location and identity of an end-host (RFC 791) [Pos81b]. While this decision simplified the Internet’s conceptual addressing model and met the needs of early network deployments it has created difficulties for the development and deployment of truly mobile, IP-based applications and services. In the 1990s a network-layer ‘Mobile IP’ solution was first developed in the context of IP version 4 (IPv4) (RFC 3344) [Per02a]. During this same period of time, the Internet Engineering Task Force (IETF) began work on a new version of IP that has become known as IP version 6 (IPv6) (RFC 2460) [DH98]. Although IPv6 addressing still retains much of IPv4’s semantic link between location and identity, experience with Mobile IPv4 allowed the IETF to integrate better support for Mobile IP1 into IPv6.

Native IPv6 is only just beginning to see commercial deployment, North American markets are slowest to deploy IPv6, Asia’s is faster and there has been little research done into the practical performance implications of Mobile IPv6. The focus of my research has been to better understand the architectural and protocol choices made during the development of Mobile IPv6, and experimentally evaluate the consequences of key Mobile IPv6 parameters on end-to-end application performance. Based on my research I have gained an expert understanding of Mobile IPv6 and valuable experience building and operating IPv6 tools in a FreeBSD environment.

1.1 Why Mobile IP?

The growth of IP-based data and voice applications in the context of mobile devices (for example PDAs, laptops and 3rd and 4th generation mobile phones) and new access technologies (such as Bluetooth, GPRS, 802.11 Wireless LANs, ADSL, and DOCSIS cable modems) is

1 In this thesis, the term “Mobile IP” will be used where the discussion is applicable to both Mobile IPv4 and Mobile IPv6 (or the context makes it clear which is intended), otherwise I will explicitly use “Mobile IPv4” or “Mobile IPv6” as appropriate.
driving a desire to support mobility at the IP level. The goal is to allow applications on a mobile IP host to keep on communicating with other hosts while roaming between different IP networks. Roaming typically occurs when the mobile host physically moves to a new location and decides to utilise a different access link technology – this can result in the host disappearing from one point on the Internet and, topologically at least, re-appearing at another point. With standard IPv4 or IPv6 such a move would result in disruption to the mobile host’s ongoing communication. Using Mobile IP the mobile host perceives only a short disruption and then continues exchanging packets as though nothing has happened.

As indicated in the opening paragraph, Mobile IP solves the problem introduced by the fact that traditional IP addresses simultaneously represent the host’s identity and encode the host’s topological location on the IP network [Pos81b]. Moving a host’s physical attachment point to an IP network often results in the host moving to a new subnetwork with respect to the network’s IP topology. When this occurs, a new IP address must be assigned to the host in order that packets may be correctly routed to the host’s new location. (‘Subnetworks’ refer to topologically localized groupings of hosts - or potential hosts - identified by particular high-order bits in the IP address). However, since the host’s IP address is also used as a transport level end-point identity such a move breaks any transport layer connections (for example, TCP sessions) that were active at the time of the move. Packets being sent to the host’s previous IP address are simply lost, and the host’s previous peers will not know the new address to which they should now send their IP packets.

Mobile IP works around this problem by introducing two IP addresses for mobile hosts – a static ‘home address’ by which the host is known globally (for long-lived identity), and a transient ‘Care-of Address’ by which the host is temporarily known when attached to different parts of the network (for routing purposes). Dynamically managed IP-in-IP tunnels (in Mobile IPv4) and specially encoded packet-forwarding rules (in Mobile IPv6) allow the mobile host to appear accessible from its home address even when actually attached to the Internet at its foreign address. By ensuring that transport (and higher) layers only utilize the home address, Mobile IP supports network layer mobility in a manner that is transparent to all upper layers. Thus, applications designed around the assumptions of the traditional, non-mobile Internet will continue to function even in a mobile host environment.

It is not hard to imagine scenarios where Mobile IP could be attractive. For example, you are in the office and begin transferring files to your laptop over the wired LAN. The files are still downloading when you need to leave and attend a meeting elsewhere in the building. The laptop is unplugged. With traditional IP your connection would break and the file download would be disrupted (and most likely terminated, unless the application is specifically re-designed to handle such mobility events at the application level). On the other hand Mobile IP
could, for example, allow the building’s 802.11 wireless LAN to seamlessly take over as your laptop’s IP network link while away from the office – ensuring your applications stay online without restart or interruption, largely unaware of any change in the host’s IP connectivity.

Many common applications react badly to IP addresses being changed mid-session. As mentioned, it is because IP address was designed static and transport layer protocols do not handle changes of IP address while the connection is active. Furthermore host name to IP address binding is cached in DNS and clients therefore even a host name has bound to a new IP address, the host may not be reached since those caches are not updated. Any mechanism for internationalism or geographic co-location of content also fails since it occurs at binding. Those applications thus would benefit from the transparent manner in which Mobile IP enables one or both ends of the session to move around the IP network. Some examples include:

- Remote Access Virtual Private Networks (VPNs): VPN links based on IPsec (IP security encrypted tunnel) technology are intolerant of IP address changes.
- Remote File Systems: Current remote file system and ‘network drive’ mechanisms (for example, Sun’s Network File System or Microsoft’s ‘shared folders’) do not handle the client’s IP address changing in mid-session.
- Database applications: Any client server application based on ODBC drivers or similar will fail if the user changes IP-address while working.
- Voice over IP (VoIP): An enterprise VolP or IP Telephony application generally does not cope well when the underlying IP address of one or both clients changes during a call.

Rather than require each and every application designer to build mobility-awareness into their network protocol, Mobile IP solves the problem in a transport- and application-independent manner.

A key target of Mobile IP is the class of users who need to work in a geographically flexible and transient manner - whether in the office, on the road, at customer premises or at home. Hosts and users who never move their network attachment point (for example, traditional desktop PCs in a large corporate office site) do not require Mobile IP. Fortunately, both Mobile IPv4 and Mobile IPv6 can be deployed incrementally, with upgrades limited only to the mobile hosts and networks the mobile hosts will visit. Other hosts (the ‘correspondent nodes’) do not need Mobile IP extensions in order to communicate with mobile hosts.

Alternatives to Mobile IP have been proposed along the way, including rapid updates to the Domain Name System (DNS) and adding ‘host routes’ to the network’s routers. The DNS update approach makes the host’s domain name a long-term identifier. Each time the host moves, it must inform the DNS of the new IP address to which the host’s domain name should now point. Unfortunately this creates a heavy load of dynamic update traffic on the DNS, and
does nothing to avoid transport and application layer sessions dying as the IP address changes. The ‘host route’ approach requires host-specific routing and forwarding table entries to be added to every router in the network (at least in the region where mobile hosts and correspondent nodes will exist). This approach does allow transport sessions to remain active after a move - the host route essentially means that the ‘home address’ is no longer constrained to have topological significance. Unfortunately, this is precisely why host routes have not caught on – they consume router memory in proportion to the number of hosts, rather than just the number of networks or subnetworks.

1.2 Objectives of This Thesis

1.2.1 A Key Technical Issue with Mobile IPv6

Although Mobile IPv4 has been under development for at least 10 years, Mobile IPv6 is the version most likely to see substantial real-world deployment. Many of the architectural and protocol lessons learned during the development of Mobile IPv4 informed and guided the development of mechanisms in IPv6 to natively support network level mobility [Per02b]. Research into Mobile IPv4 during the late 1990s and early 2000s has been comprehensive, covering topics such as triangle routing (inefficient packet flows enforced by the need to tunnel traffic), poor detection of network change events (leading to long disruptions to IP traffic flow when a host moves) and security (authenticating a host’s movement, identity and authority to attach to a new network).

One of main benefits that IPv6 offers is solving the problem of IPv4 address shortage. Since there are more and more people using web and not many addresses left, plus the growth of new deployments (e.g. WiMax - standards-based technology that provides wireless broadband connections [WiMa]), Asia and Europe has shown a strong interest in IPv6. While China launched its largest pure IPv6 network CERNET2 over 20 cities at the end of 2004 [CHIN], other countries have been cooperating to develop applications and services using IPv6 [PN04]. Given the emerging demand for, and relevance of, IPv6 in Asian and European markets [Bar03] Mobile IPv6 is currently of strong research interest as a platform for supporting mobile, interactive and real-time services. One area that deserves closer attention is the efficacy of standard Mobile IPv6 in meeting IP service quality goals and what changes may be required to meet specified constrained service goals in the future.

I have chosen to focus on a key deployment issue in Mobile IPv6 – characterising the disruption to performance that occurs when a Mobile IPv6 host moves between networks. During these events (known as ‘handoff’, as the host’s connection to one network is handed off to another network) the flow of IPv6 packets is temporarily disrupted. This disruption may have little impact (for example, to applications such as email, web browsing or non-interactive file
transfers) or the impact could be significant (for example, to interactive and/or real-time streaming applications such as voice/video conferencing, streaming stock-tickers or online gaming).

Disruption of packet flow during handoff disrupts the network-level transparency that Mobile IPv6 attempts to provide to upper layer services. This disruption may take the form of a transient spike in the round trip time (if packets are buffered during handoff) or a burst of packet loss (if packets are discarded during handoff). The duration of this disruption depends on the interaction between the link layers (where network de-attachment and attachment are often first noticed) and link layer independent mechanisms implemented by Mobile IPv6 to detect changes in network attachment point. Some of the disruption will be due to published protocol behaviour while some will be due to issues specific to particular implementations of Mobile IPv6 and link layer devices.

1.2.2 The Research Described in This Thesis

To meet my research goals I performed the following tasks:

- Characterised and quantified the handoff latencies of Mobile IPv6.
- Analysed the characteristics of standard Mobile IPv6 independent of the link layer technology.
- Described the need for link layer handoff information to assist network layer handoff.
- Experimentally evaluated the likely performance impact of MIPv6 handoff on two common applications – bulk one-way TCP data transfer, and a basic web camera application.

This thesis begins with an introduction to IPv6 and IPv4, then continues with a detailed analysis of Mobile IPv6 in comparison to Mobile IPv4, and a review of the literature regarding various Mobile IP handoff techniques and proposals to date. Particular attention is paid to IEEE 802.11 wireless LAN (WLAN) as a link layer technology becoming common in enterprise network environments.

The thesis then describes my development of an experimental testbed suitable for gathering real-world Mobile IPv6 handoff data using publicly available, standards-compliant implementations of Mobile IPv6. First an 802.11 WLAN testbed was built and utilised to establish intrinsic handoff delays of our particular commercial WLAN implementation. This testbed was then augmented with an open-source Mobile IPv6 implementation (the KAME [KAME] release under FreeBSD [FreB]), and utilised to experimentally observe handoff delays attributable to Mobile IPv6. Finally another small testbed was designed and implemented to emulate an IPv6 network path affected by handoff occurring at various frequencies. This testbed
was used to ascertain the impact of handoff on an actual TCP bulk data transfer application and a conventional web camera application.

In addition to the experimental characterisation of Mobile IPv6 handoff delays, I document insights into the use of FreeBSD as a platform for both networking research and possible real-world deployment of Mobile IPv6 using the KAME software stack. I discovered some bugs in the KAME implementation of Mobile IPv6 and fed this information back to the KAME development community.

1.2.3 Publications Related to the Study Presented in This Thesis

Journal Article


Conference Paper


Technical Report

2 Evolving from IPv4 to IPv6 .................................................................................................................. 7
  2.1 Introduction ................................................................................................................................... 7
  2.2 IPv4 Fundamentals ......................................................................................................................... 8
    2.2.1 IPv4 Address Format ............................................................................................................. 8
    2.2.2 IPv4 Header Format ................................................................................................................ 10
    2.2.3 IPv4 Issues .............................................................................................................................. 11
  2.3 IPv6 Fundamentals ......................................................................................................................... 12
    2.3.1 IPv6 Address Format ............................................................................................................. 12
    2.3.2 IPv6 Address Prefix Reserved for Documentation ............................................................. 14
    2.3.3 IPv6 Header Format ................................................................................................................ 15
    2.3.4 IPv6 Neighbor Discovery ..................................................................................................... 18
    2.3.5 Improvements over Version 4 ............................................................................................. 19
  2.4 Transition Mechanisms (IPv4 to IPv6) ....................................................................................... 19
    2.4.1 Dual IP Layer ........................................................................................................................ 20
    2.4.2 IPv6 over IPv4 Tunneling ..................................................................................................... 20
  2.5 QoS Protocols for IP Networking ................................................................................................. 21
Chapter 2

Evolving from IPv4 to IPv6

This chapter introduces some basic concepts and terminology of IPv6 and IPv4 as a basis for the following chapters on Mobile IP. This chapter first introduces address and header format of IPv4; analyses on IPv4 limitation in networking, and then introduces new technologies and changes in address and header format of IPv6. The chapter follows up with a comparison of IPv6 and IPv4 technologies to demonstrate the improvements of IPv6 relative to IPv4 with respect to mobile application services. This chapter therefore helps to gain an understanding of IPv6 functions that would support better mobility and reasons to deploy IPv6 in the future. Also in section 2.3.2 of this chapter, IPv6 address prefix recommended to be used for documentation (RFC 3849 [HLS04]) and used later on for IPv6 and Mobile IPv6 experiments is discussed. The chapter finally covers basic theoretical information on transition mechanisms for switching IPv4 supported networks to IPv6 and QoS for IP networking.

2.1 Introduction

IPv4 has been the basis of the Internet for a substantial period of time, but although it continues to develop, it faces some challenging issues, of which the most important is the growing shortage of IPv4 addresses. Meanwhile, IPv6 has been developed as the next evolutionary step in IP networking. The Internet Engineering Task Force (IETF) introduced the latest IPv6 specification RFC 2460 in 1998. IPv6 builds upon the knowledge gained with IPv4 and provides improved features relative to IPv4. The benefits claimed for IPv6 include increased IP address size, increased addressing hierarchy support, unified addressing (global, site and local), simplified autoconfiguration of addresses (easier readdressing, DHCPv6, Neighbor Discovery instead of ARP broadcasts), streamlined header structure, improved security (security extension headers, integrated data integrity), better mobility (multi Home Agent, Care-of Address, routing extension header), and promising performance opportunities (no fragmentation by intermediate routers, no header checksum, support for flow label and traffic class in the IP header).

The following sections provide an outline of some IPv4 and IPv6 fundamentals, then comparisons of IPv6 with the current IPv4, and some current transition mechanisms for
networks to move to IPv6. The final part of this chapter outlines some IP Quality of Service (QoS) protocols, which are designed for IP and do not consider the support of mobility.

2.2 IPv4 Fundamentals

2.2.1 IPv4 Address Format

IPv4 addresses identify IP hosts on the Internet. An IPv4 address has a length of 32 bits and is commonly represented by four dotted decimal numbers, each in the range 0-255. An IPv4 address is written as x.x.x.x where each ‘x’ is the decimal value of an 8-bit byte of the address (e.g. 136.186.1.117).

According to RFC 1812 [Bak95], an IPv4 address is partitioned into two parts - a constituent network prefix (network number) and a host number on that network.

\[
\text{IP-address} = \{\text{<Network-prefix>}, \text{<Host-number>}\}
\]

The earliest classical network prefixes were the Class A, B, C, D, or E network prefix.

- 0xxx - Class A - general purpose unicast addresses with standard 8 bit prefix
- 10xx - Class B - general purpose unicast addresses with standard 16 bit prefix
- 110x - Class C - general purpose unicast addresses with standard 24 bit prefix
- 1110 - Class D - IP Multicast Addresses - 28 bit prefix, non-aggregatable
- 1111 - Class E - reserved for experimental use

Class A, B, and C address classification is no longer important; they are used as network prefixes with only historical relevance. The concept of subnet provides for a multi-level hierarchical routing structure. A subnet partitions the <Host-number> field into two parts: a subnet number, and a host number on that subnet [Bak95]

\[
\text{IP-address} = \{\text{<Network-number>}, \text{<Subnet-number>}, \text{<Host-number>}\}
\]

In a subnetted network, the internal networks use the same network prefix but different subnet numbers. Thus, only the <Network-number> part of the address is used for Internet routing outside the subnetted network. Within the subnetted network, the routers use the extended network prefix for routing \{<Network-number>, <Subnet-number>\}
In Figure 2.1, the three subnets will use the same network prefix but different subnet numbers. The local routers will use the extended network prefix. Routers within the Internet will use the <Network-number> only.

Figure 2.1: A Network Diagram

Classless Inter Domain Routing (CIDR)

Due to the explosive growth of the Internet, address assignment schemes were reviewed in 1993. Classless Inter Domain Routing (CIDR) [FLYV93] was therefore introduced and deployed, in which hosts and routers make no assumptions about the use of addressing in the Internet. By definition, CIDR comprises three elements: topologically significant address assignment, routing protocols that are capable of aggregating network layer reachability information, and a consistent forwarding algorithm. A network prefix is defined as a contiguous set of bits at the more significant end of the address that defines a set of systems [Bak95].

An IPv4 address prefix is represented by the notation: ipv4-address/prefix-length. Prefix-length is a decimal value specifying how many of the leftmost contiguous bits of the address comprise the extended network prefix.

Special Use IPv4 Addresses

Some IPv4 addresses are assigned special uses.

10.0.0.0/8 and 192.168.0.0/16 and 172.16.0.0/12: These blocks are set aside for use in private networks.

127.0.0.0/8: This block is assigned for use as the Internet host loopback address. A packet sent by a higher level protocol to an address in this block should loop back inside the host. 127.0.0.1/32 is normally implemented for loopback.
169.254.0.0/16 - This is the “link local” block. It is allocated for communication between hosts on a single link.

The “limited broadcast” destination address 255.255.255.255 should never be forwarded outside the subnet of the source.

2.2.2 IPv4 Header Format

<table>
<thead>
<tr>
<th></th>
<th>Version</th>
<th>IHL</th>
<th>Type of Service</th>
<th>Total Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flag</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fragmentation Offset</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time to Live</td>
<td></td>
<td></td>
<td>Protocol</td>
<td>Header Checksum</td>
</tr>
<tr>
<td>Source Address</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination Address</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Options</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 2.2: IPv4 Header Format*

In the IPv4 header (Figure 2.2), fields are defined as follows (RFC 791 [Pos81b]):

- Version (4 bits) indicates the version of the Internet header.
- IHL (4 bits): Internet Header Length indicates the length of the variable IPv4 header in 32 bit words.
- Type of Service (ToS) (8 bits) indicates the abstract parameters of the QoS desired.
- Total Length (16 bits) indicates the length of the datagram, measured in octets, including Internet header and data. This field allows the length of a datagram to be up to 65,535 octets. All hosts must be prepared to accept datagrams of up to 576 octets to allow a reasonable sized data block to be transmitted in addition to the required header information.
- Identification (16 bits) aids in assembling the fragments of a datagram at the destination.
- Flags (3 bits) indicate various control flags for fragmentation purposes.
- Fragment Offset (13 bits) indicates where in the datagram this fragment belongs. The fragment offset is measured in units of 8 octets (64 bits).
- Time to Live (8 bits) indicates the maximum time the datagram is allowed to remain in the Internet system. If this field contains the value zero, then the datagram must be destroyed. This field is modified in Internet header processing.
- Protocol (8 bits) indicates the next level protocol used in the data portion of the Internet datagram.
- Header Checksum (16 bits): A checksum on the header only. Since some header fields change (e.g., Time to Live), this is recomputed and verified at each point that the Internet header is processed.
- Source Address (32 bits): The source address.
- Destination Address (32 bits): The destination address.
- Options (variable length): The options may appear or not in datagrams however they are implemented by all IP modules.

2.2.3 IPv4 Issues

IPv4 has some limitations in many networking cases. These include limited address space, security issues, IP options size limitation, and routing performance. Although IPv4 has sought to address these issues, some of the solutions are complex and/or limited.

- IPv4 routers rely on transport and application level information to identify flows. IPv4 routers process data at the network layer and also require information from the transport or application protocol (i.e. socket ports) to map packets. Because of the dependency on transport or application protocol information, IP level security techniques (such as IPsec (RFC 2401, 2402, 2406) [[KA98a], [KA98b], [KA98c]]) are incompatible with some QoS protocols such as Resource Reservation Protocol (RSVP). These mechanisms may encrypt the transport header, thus hiding the port numbers of data packets from intermediate routers.

- In order to limit the uses of public IPv4 addresses, IPv4 deployments use Network Address Translator (NAT) (RFC3022) [SE01] that allows the illusion of many hosts sharing a single IP address.

- According to the IPv4 specification RFC 791 [Pos81b], each router along the transmission path must process the variable length Option Field, again, on a per packet basis even though an option might be effectively used only by the end hosts. This may place a significant processing load on each router.

- No mandatory security mechanisms: IPSec supported in IPv4 is not mandatory. Moreover, IPv4 does not encrypt its packets and a transmission in IPv4 cannot be digitally signed.

- IPv4 lacks mobility support as it was designed without mobility in mind. The change of destination address is a challenge for mobile computing based on IPv4.
2.3 IPv6 Fundamentals

IPv6 was introduced (called IPng) as the “next generation” protocol. IPv6 inherits all of the knowledge gained with IPv4 and has additional features intended to more elegantly support future mobile and wireless networking. In this section, key attributes of IPv6 are summarised. More details can be found in Appendix A.

2.3.1 IPv6 Address Format

IPv6 addresses are 128-bit identifiers for interfaces and sets of interfaces. An interface is a node’s attachment to a link. To accommodate load balancing systems, RFC 3513 “IP Version 6 Addressing Architecture” [HD03] allows multiple interfaces to use the same address as long as they appear as a single interface to the IPv6 implementation on the node. According to RFC 3513, there are 3 types of IPv6 addresses - unicast, anycast, and multicast. There are no broadcast addresses in IPv6; they are replaced by multicast addresses.

Unicast

A unicast address identifies a single interface. Unicast routing delivers packets addressed to a unicast address to a single interface.

Multicast

A multicast address identifies multiple interfaces. Multicast routing delivers packets addressed to a multicast address to all interfaces identified by the address. A multicast address is used for one-to-many communication, with delivery to multiple interfaces.

Anycast

An anycast address identifies multiple interfaces (as does multicast), but anycast routing only delivers to one interface (the “closest” by hop count).

IPv6 Address Representation

There are three conventional forms for representing IPv6 addresses as text strings:

1. The preferred form is x:x:x:x:x:x:x:x

   The ‘x’ s are the hexadecimal values of the eight 16-bit words of the address.

   
   or 1080:0:0:0:8:800:200C:417A

2. Compressed Form with the use of “::”

   The use of “::” is available to compress the zeros in IPv6 addresses. It indicates one or more groups of 16 bits of zeros. The “::” can only appear once in an address.
For example,
1080:0:0:8:800:200C:417A can be represented 1080::8:800:200C:417A.

3. IPv4 compatible and IPv4 mapped address format x:x:x:t:d:d:d:d

The ‘x’s are the hexadecimal values of the five high-order 16-bit words of the address, ‘t’ is the hexadecimal value of 16-bit word, and the ‘d’s are the decimal values of the four low-order 8-bit bytes of the address (standard IPv4 representation).

Two forms of IPv4 related addresses are specified by RFC 2893 [GN00] for use with a mixed environment of IPv4 and IPv6 nodes. “IPv4 compatible IPv6 address” is used by IPv6 nodes to tunnel IPv6 packets over IPv4 routing infrastructure. This carries a global IPv4 address in the low-order 32 bits. “IPv4 mapped IPv6 address” is used to represent the addresses of IPv4 hosts as IPv6 addresses.

When ‘t’ (type value) is equal to FFFF, it means “mapped”, when ‘t’ (type value) is equal to 0000, it means “compatible”. The format is then written as ::FFFF:<IPv4 address> or :::<IPv4 address>. For example ::FFFF:136.186.229.50 or ::136.186.229.50

An IPv6 address prefix is represented by the notation: ipv6-address/prefix-length. Prefix-length is a decimal value specifying how many of the leftmost contiguous bits of the address comprise the prefix.

For example, the node address 12AB:0:0:CD30:123:4567:89AB:CDEF and its subnet number 12AB:0:0:CD30::/60 can be abbreviated as

12AB:0:0:CD30:123:4567:89AB:CDEF/60

IPv6 Address Format

In IPv6, the top 64 bits of a unicast address are used for hierarchical routing. Public topology, TLA (Top Level Aggregation) and NLA (Next Level Aggregation) are specified for service providers and exchanges to provide public transit services. TLA is for global backbones and NLA is for service providers with regional scope.

Link Local Address

Link local address is written as FE80::<64-bit interface ID>. Link local address is formed by address autoconfiguration. A link-local address is formed by pre-pending the link-local prefix FE80 to interface identifiers which are typically 64-bits long and based on EUI-64 identifiers (MAC address) [TN98]. If the interface identifier has a length of N bits (for N greater than 64 to N less than or equal 118), the interface identifier replaces the right-most N zero bits of the link-local prefix. If the interface identifier is more than 118 bits in length, manual configuration is required. A link-local address is never timed out.
Global and Site Local Address

Site Local Address is written as FEC0::<subnet ID><64-bit interface ID>.

Global Address is written as <64-bit prefix><64-bit interface ID>.

Global and site-local addresses are formed by pre-pending a prefix of appropriate length to an interface identifier. Prefixes are obtained from Prefix Information options contained in Router Advertisements. A host must attempt to use stateful autoconfiguration to obtain addresses, if a link has no routers.

Address Type Identification

The type of an IPv6 address is identified by the high-order bits of the address, as follows:

<table>
<thead>
<tr>
<th>Address type</th>
<th>Binary prefix</th>
<th>IPv6 notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unspecified</td>
<td>00...0 (128 bits)</td>
<td>::/128</td>
</tr>
<tr>
<td>Loopback</td>
<td>00...1 (128 bits)</td>
<td>::1/128</td>
</tr>
<tr>
<td>Multicast</td>
<td>11111111</td>
<td>FF00::/8</td>
</tr>
<tr>
<td>Link-local unicast</td>
<td>111111110010</td>
<td>FE80::/10</td>
</tr>
<tr>
<td>Site-local unicast</td>
<td>11111111011</td>
<td>FEC0::/10</td>
</tr>
<tr>
<td>Global unicast</td>
<td>(everything else)</td>
<td></td>
</tr>
</tbody>
</table>

The IPv6 subnet prefix is associated with one link as in IPv4. Multiple subnet prefixes may be assigned to the same link [HD03].

Since each interface belongs to a single node, any of the unicast addresses for that node’s interfaces may be used as identifiers for the node.

All interfaces are required to have at least one link-local unicast address. A single interface may also have multiple IPv6 addresses of any type (unicast, anycast, and multicast) or scope. A multicast group is defined at the source. An anycast group is defined at the destination. With the appropriate anycast routing, packets addressed to an anycast address are propagated through the Internet until they reach a router that recognises them as matching an anycast group. Packets then are forwarded to the single interface that has the “nearest” address specified in the anycast group. The “nearest” interface is defined as being closest in terms of routing distance. An anycast address is used for one-to-one-of-many communication, with delivery to a single interface.

2.3.2 IPv6 Address Prefix Reserved for Documentation

RFC 3849 [HLS04] introduces the use of the IPv6 unicast address prefix 2001:DB8::/32 as a reserved prefix for use in documentation such as in RFCs, books, documentation, and the like. The use of this reserved prefix reduces the likelihood of conflict and confusion when relating documented examples to deployed systems.
When documenting IPv6 address assignments for routers and nodes on my implemented testbed, this reserved prefix 2001:DB8::/32 was used.

2.3.3 IPv6 Header Format

<table>
<thead>
<tr>
<th>Version</th>
<th>Traffic Class</th>
<th>Flow Label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>Next Header</td>
<td>Hop Limit</td>
</tr>
<tr>
<td>Source Address</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination Address</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 2.3: IPv6 Header Format*

The number of fields is reduced in the IPv6 header (Figure 2.3) compared to the IPv4 header. The IPv6 header is redesigned and simplified in order to reduce the common-case processing cost of packet handling. All optional data is moved to IPv6 Extension Headers whereas optional data is included in the IPv4 header. The IPv6 header has a fixed length of 40 bytes whereas the IPv4 header has a length of 20 to 60 bytes depending on options. In the IPv6 header format:

- Six fields of the IPv4 header are removed
  - IHL (IP Header Length) is no longer needed due to the fixed header size.
  - Type of Service, Fragmentation fields (Identification, Flag, Fragmentation Offset) and Header Checksum are no longer needed because only end-to-end fragmentation is supported in IPv6, which is handled by IPv6 Extension Headers. Header checksum is used only by lower and higher level layers.

- Three fields are renamed and slightly redefined
  - Total Length: Due to the fixed header size of IPv6, the Total Length of IPv4 is replaced by the Payload Length of IPv6.
  - Protocol: The Protocol Type field is renamed as the Next Header to reflect the new organisation of the IP packets.
  - Time to live (TTL): The field has been changed to the Hop Limit which formally specifies number of hops instead of time, to match the actual processing conditions in the routers.
- Two new fields are added:
  - Traffic Class and Flow Label: They are introduced to support QoS for IP flows.
- The option fields are replaced by the Extension Headers and arranged between the IPv6 header and transport layer data. Extension headers only need to be examined by the packet’s destination (except the Hop-by-hop Options Header). At present, the following extension headers have been defined:
  - Hop-by-hop Options Header: Defines special options that require hop-by-hop processing.
  - Routing Header: Provides extended routing, specifies intermediate nodes that must be visited (and plays the same role as the source routing option of IPv4).
  - Fragment Header: Contains end-to-end fragmentation and reassembly information for fragmented packets (almost identical to the fragmentation control parameters of IPv4).
  - Authentication Header: Provides support for packet integrity and authentication.
  - Encapsulating Security Payload: Provides support for privacy.
  - Destination Option Header: Contains transparent information processed by the final destination of the packets.

**Version**

The 4-bit Version field contains the number 6.

**Traffic Class**

The 8-bit Traffic Class field in the IPv6 header is available for use by originating nodes and/or forwarding routers to identify and distinguish between different classes or priorities of IPv6 packets.

**Flow Label**

The 20-bit Flow Label field enables the labeling of packets belonging to particular traffic ‘flows’ for which the sender requests special handling. A router can identify which end-to-end flow a packet belongs to. Nodes or routers that do not support the functions of the Flow Label field are required to set the field to zero when originating a packet, pass the field on unchanged when forwarding a packet, and ignore the field when receiving a packet.
The RFC 3513 [HD03] IPv6 states that their application is still “experimental”, but defines rules including:

- A flow is uniquely identified by the combination of a source address and a non-zero flow label.
- Hop-by-Hop Options and destination address must be the same for all packets in a flow.

Payload Length

The 16-bit Payload Length field contains the payload length - the length of the data field following the IPv6 header, in octets.

Next Header

The 8-bit Next Header field identifies the type of header immediately following the IPv6 header and located at the beginning of the data field (payload) of the IPv6 packet. The two common kinds of Next Header are TCP (next header value is 6) and UDP (next header value is 17).

Hop Limit

The 8-bit Hop Limit field is decremented by one by each node (typically a router) that forwards a packet. If the Hop Limit field is decremented to zero, the packet is discarded. The main function of this field is to identify and to discard packets that are looping because of erroneous routing information.

Source Address

The 128-bit Source Address field contains the IPv6 address of the node originating the packet.

Destination Address

The 128-bit Destination Address field contains the IPv6 address of the node recipient of the packet. If a Routing header is present, this address is not that of the ultimate recipient.

Extensions Headers

Extension headers offer the following benefits:

- More efficient forwarding
- Less stringent limits on the length of options
- Greater flexibility for introducing new options in the future

Each extension header includes a Next field, which indicates the type of the next extension header.
Figure 2.4 gives examples using the Next field in Extension headers. The first example has a TCP extension header. The second example adds a routing header. The third example adds a fragment header.

IPv6 header
Next header = TCP
Value 6
TCP header and data

IPv6 header
Next header = Routing
Value 43
Routing header
Next header = TCP
Value 6
TCP header and data

IPv6 header
Next header = Routing
Value 43
Routing header
Next header = Fragment
Value 44
Fragment header
Next header = TCP
Value 6
TCP header and data

2.3.4 IPv6 Neighbor Discovery

IPv6 introduces new Neighbor Discovery functions (specified in RFC 2461 [NNS98]). Most functions are based on and improved from IPv4 functions.

- IPv6 Router Discovery specifies ways that nodes locate routers on their link. The corresponding IPv4 functionality is ICMP Router Discovery (RFC 1256) [Dee91].
- IPv6 Prefix Discovery specifies ways that nodes determine the network prefix(es) assigned to the current link. The corresponding IPv4 functionality is DHCP (RFC 2131) [Dro97] or Manual Configuration.
- IPv6 Parameter Discovery specifies ways that nodes learn such things as the link MTU and a reasonable value to put in the Hop Limit (Time to Live) field. In IPv4, this would be manually configured.
- IPv6 Address Autoconfiguration specifies ways that nodes automatically obtain an IP address for use on an interface. The corresponding IPv4 functionality is DHCP (RFC 2131) [Dro97].
- IPv6 Neighbor Discovery for address resolution specifies ways that nodes determine the link-layer address of a neighbour whose IP address is known. The corresponding IPv4 functionality is Address Resolution Protocol (ARP) (RFC 826) [Plu82].
- IPv6 Next Hop Determination specifies ways that nodes choose a Next Hop for any outgoing packets. The corresponding IPv4 functionality is Routing Table Search.
- IPv6 Neighbor Unreachability specifies ways that nodes determine that a neighbour is no longer reachable. There is no standard IPv4 mechanism.
- IPv6 Duplicate Address Detection specifies ways that nodes determine that their respective addresses are unique. There is no standard IPv4 mechanism.

- IPv6 Redirect specifies how routers inform nodes of a better choice for a Next Hop to a destination. The corresponding IPv4 functionality is ICMP redirect (RFC 792) [Pos92].

2.3.5 Improvements over Version 4

The information below summarises the main improvements of IPv6 relative to IPv4.

- Larger address space: IPv6 128-bit addressing provides a large increase in IP addresses for the Internet relative to IPv4 32-bit addressing. IPv6 supports more levels of addressing hierarchy.

- Address Autoconfiguration: Apart from manual configuration or DHCP as used for IPv4 addresses, IPv6 addresses can be configured by address autoconfiguration.

- Efficient routing: The Destination Option header provides forwarding information. IPv6 fragmentation is not performed by routers, but only by the sending nodes (fragmentation in IPv4 may occur in both routers and the sending hosts).

- Mobility: Efficient IPv6 routing helps support the routing process in Mobile IPv6 (covered in the chapter 4).

- Simplicity: IPv6 simplifies packet processing through the use of Options and other Extension Headers. The new concept of an “ordered” linked list of Extension Headers ensures that routers process only the options necessary for correct operation.

- Built-in Security: IPv6 security offers built-in authentication (IPsec is mandatory in IPv6) and privacy mechanisms by special headers. IPv6 includes packet encryption (ESP: Encapsulated Security Payload) and source authentication (AH: Authentication Header). IPv6 packets and addresses can be encrypted.


In short, IPv6 is designed to overcome many disadvantages of IPv4. IPv6 also provides additional routing features that support routing for IP mobility. IPv6 is expected to gradually replace IPv4, with the two coexisting for a number of years during a transition period.

2.4 Transition Mechanisms (IPv4 to IPv6)

Overall, the issues for IPv6 migration and coexistence to gradually replace IPv4 can be summed up in two main categories: how to share the physical network so that IPv4 and IPv6 can be transported over the same physical network and how to handle applications that have not yet
been enhanced to support IPv6. RFC “Transition Mechanisms for IPv6 Hosts and Routers”, RFC 2893 [GN00]) defines two migration mechanisms for IPv4 to IPv6: Dual IP layer and Tunneling.

2.4.1 Dual IP Layer

The Dual IP layer technique (or Dual Stack) provides support for both IPv4 and IPv6 nodes and routers. By providing a complete IPv4 implementation, it allows IPv6 nodes to remain compatible with IPv4-only nodes. IPv6 nodes with complete IPv4 and IPv6 implementations are called “IPv6/IPv4 nodes”. IPv6/IPv4 nodes can send and receive both IPv4 and IPv6 packets. IPv6/IPv4 nodes may be operated in one of three modes: with their IPv4 stack enabled and IPv6 stack disabled, with their IPv6 stack enabled and IPv4 stack disabled, with both stacks enabled.

2.4.2 IPv6 over IPv4 Tunneling

Tunneling provides a way to utilise an existing IPv4 routing infrastructure to carry IPv6 traffic. IPv6/IPv4 hosts and routers can tunnel IPv6 packets over regions of IPv4 routing topology by encapsulating them within IPv4 packets. Tunneling can be used in many ways (RFC 2893 [GN00]):

- Host-to-Host: IPv6/IPv4 hosts that are interconnected by an IPv4 infrastructure can tunnel IPv6 packets between themselves.

RFC 2893 defines two types of tunneling. They differ primarily in how they determine the tunnel endpoint address.

Configured Tunneling of IPv6 over IPv4

IPv6 over IPv4 tunneling encapsulates IPv6 packets within IPv4 headers to carry them over IPv4 routing infrastructures. In configured tunneling, the tunnel endpoint address is determined from configuration information in the encapsulating node.

Automatic Tunneling of IPv6 over IPv4

This mechanism uses IPv4 compatible addresses to automatically tunnel IPv6 packets over IPv4 networks. It allows IPv6 only devices to communicate with IPv4 only devices by using IPv4-compatible IPv6 addresses, which have the IPv6 address format and employs embedded IPv4 addresses.
RFC 3056, “Connection of IPv6 Domains via IPv4 Clouds”, [CM01] uses automatic IPv6-over-IPv4 tunneling to interconnect IPv6 networks. It provides a mechanism for IPv6 sites to communicate with each other over the IPv4 only network via 6to4 routers, and for them to communicate with native IPv6 networks via relay 6to4 routers. 6to4 routers and 6to4 relay routers accept and decapsulate “IPv6-in-IPv4” traffic from any host in the IPv4 Internet. The IPv4 network is treated as a unicast point-to-point link layer. The method also provides a globally unique IPv6 address prefix to any site with at least one globally unique IPv4 address including the case where the address combined with an IPv4 Network Address Translator (NAT).

RFC 3056 also studied risks considering the growing number of networks using 6to4 for automatic tunnels. In these 6to4 networks, security implementations that prevent attacks against IPv4 should be added to IPv6 security implementation with regard to system efficiency when using IP security at both IPv4 and IPv6 levels.

2.5 QoS Protocols for IP Networking

Two architectures have been defined for IP QoS support: Integrated Service (IntServ) [SW97] and Differentiated Service (DiffServ) [Bla98]. QoS can also be based on a hybrid architecture of both. More details can be found in Appendix B.

IntServ is based on the signaling Resource Reservation protocol (RSVP) (RFC 2205) [BZB’97]. IntServ supports per flow QoS management and provides tight performance guarantees for high priority flows. DiffServ does not define any signaling protocol; it is based on prioritisation and is a reservation-less protocol. DiffServ supports per aggregate QoS management. It allocates a network’s bandwidth in a preferential manner according to the application’s QoS class and bandwidth management policy criteria.

IntServ lacks scalability as it handles per flow classification and queuing in core routers for thousands of flows at once. DiffServ supports only aggregated QoS for a limited number of performance classes for which only statistical differentiation is provided. These QoS architectures do not consider mobile environments.
3 Mobile IP Development and Mobile IPv4 Issues ................................................................. 22
  3.1 Mobile IP Development .............................................................................................. 22
     3.1.1 Mobile IP .......................................................................................................... 22
     3.1.2 Mobile IP Handoff ............................................................................................ 24
     3.1.3 Mobile IP Handoff Types ................................................................................. 26
     3.1.4 Basic Mobile IP Requirements and Strategies .................................................. 28
  3.2 IP Mobility Support for IPv4 (Mobile IPv4) .............................................................. 29
     3.2.1 Mobile IPv4 Operations .................................................................................... 30
     3.2.2 Mobile IPv4 Issues and Extensions .................................................................. 31
     3.2.3 Mobile IPv4 Security ........................................................................................ 35
     3.2.4 IPv6 over Mobile IPv4 ..................................................................................... 35
  3.3 Impact of Mobile IP Handoff on Higher Protocols .................................................... 35
     3.3.1 Impact of Mobile IP Handoff on the Performance of TCP ............................... 36
  3.4 Mobile IP QoS Issues ................................................................................................. 37
     3.4.1 Issues ................................................................................................................ 37
     3.4.2 Requirements of a Quality of Service (QoS) Solution for Mobile IP ............ 37
Chapter 3

Mobile IP Development and Mobile IPv4 Issues

Mobile IP has been developed as a mobility management protocol to provide uninterrupted Internet services and continuous connections. Mobile IP could be deployed in any IP environment independent of underlying link technologies (wired or wireless). This chapter first introduces Mobile IP, its terminologies and concepts, then focuses on Mobile IP handoff types and basic operations. This helps to understand the basic operations and functions of Mobile IP. The chapter is then followed on by generic Mobile IP issues (including long handoff delay and packet losses during handoff) and strategies that research communities have attempted to solve. Detailed Mobile IPv4 handoff operations and its issues and extensions (integrated into Mobile IPv6) are also described in this chapter. The chapter finally discussed impact of Mobile IP handoff (which is examined in experiments, chapter 8) and basic Mobile IP QoS issues

3.1 Mobile IP Development

3.1.1 Mobile IP

Mobile IP solves a problem introduced by the fact that traditional IP addresses simultaneously represent the host’s identity and encode the host’s topological location on the IP network [Pos81b]. Mobile IP provides IP level mobility to allow hosts to roam around different subnetworks. Applications on a Mobile IP enabled host can survive physical disconnection and reconnection while changing their points of attachment to the Internet.

3.1.1.1 Mobile IP Entities

Mobile IP entities are defined in RFC 3344, “IP Mobility Support For IPv4”.

Mobile Node (MN): A host or a router that changes its point of attachment from one network or subnetwork to another. A Mobile Node may change its location without changing its IP address; it may continue to communicate with other Internet nodes at any location using its
(constant) home IP address, assuming link-layer connectivity to a point of attachment is available.

**Home Agent (HA):** A router on a Mobile Node’s home network which tunnels datagrams for delivery to the Mobile Node when it is away from home, and maintains current location information for the Mobile Node.

**Correspondent Node (CN):** A peer with which a Mobile Node is communicating. A Correspondent Node may be either mobile or stationary.

### 3.1.1.2 Mobile IP Basic Concepts

Mobile IP enables Mobile Nodes to roam between IP subnetworks and enables applications running on an Internet host to survive physical reconnection by inserting additional features at the network (IP) layer.

![Figure 3.1: Basic Mobile IP Elements](image)

In Mobile IP, the network to which the Mobile Node is originally attached is called the Home Network, and the long-term IP address of Mobile Node on the Home Network is called the Home Address (Figure 3.1). The default router of Mobile Node at the home network is called Home Agent. Home Agent plays a very important role for Mobile Node as it manages Mobile Node identity and traffic. The network that is not Mobile Node’s Home Network is called Foreign Network. The network that Mobile Node was previously attached to is called old network or old link and the network that Mobile Node has just moved to is called new network or new link.

Mobile IP supports Mobile Nodes when they are at their original network or moving away from that network by maintaining two types of addresses, Home Address and Care-of Address (CoA). When Mobile Node is at home network, it is assigned Home Address and only uses its Home Address. This Home Address is administered in the same way as the “permanent” IP address provided to a stationary host, and is used for end-to-end communication. Mobile Node
is always identified and addressable by its Home Address regardless of its current point of attachment to the Internet. This helps Mobile IP to maintain transport and higher layer protocol sessions which depend on a static IP address.

When Mobile Node moves away from home network to a new network, Mobile Node gets a temporary, local CoA on the new network. Mobile Node registers this CoA with its Home Agent so that Home Agent can forward packets.

### 3.1.1.3 Default Mobile IP Routing

When a Mobile Node is located at home, it is co-located with its Home Agent and operates without the mobility services.

The Home Agent acts as a proxy to intercept packets destined for Mobile Node’s Home Address when Mobile Node is away from home, and forwards packets to the current point of attachment (current CoA) of Mobile Node. Because IP routing is based on the destination address, a Home Agent must encapsulate packets so that it can route them to the Mobile Node. This is done by IP-in-IP encapsulation [Per02c] that sends the original IP packet as the payload part of another IP packet. Routers use the encapsulating (outer) IP header to deliver packets from Home Agent to Mobile Node’s CoA (Figure 3.2). This kind of encapsulation creates a tunnel. One end is the Home Agent and the other end is responsible for decapsulating the packets and delivering them to the Mobile Node. The received packet is decapsulated and the inner IP header is then used in processing.

![Figure 3.2: Format of IP Packet after being encapsulated by Home Agent](image)

### 3.1.2 Mobile IP Handoff

Mobile IP handoff is defined as the process for redirecting IP packet flow destined to the MN’s old location to the MN’s current attachment point. In basic Mobile IP, when the MN moves to a new subnetwork, packets are not delivered to the MN at the new location until the CoA registration to HA is complete. The handoff interruption period may degrade Quality of Service and impact on upper layers. When a MN changes its attachment point to a new subnetwork, link layer handoff occurs first. Link layer handoff causes initial interruption to communication during the handoff period.
3.1.2.1 Mobile IP Operation

Mobile IP can involve the following sequence of steps:

- **Movement Detection**: The MN uses a range of movement detection mechanisms to detect that it has moved to a new network.

- **Registration**: When the MN detects that it has moved to a foreign network, it gets a CoA by using network address configuration mechanisms. The MN informs its HA of its current point of attachment by registering CoA to HA and requests forwarding services. Registration messages are UDP packets. The MN is thus responsible for resending the request if it does not receive a reply to a previous request packet.

- **Tunneling from the HA**: (as the default mechanism) for delivery of packets to the MN when away from the home network.

- **Binding Update**: The association between the current CoA and the Home Address of the MN is maintained by a mobility binding. Binding Update is used for renewing a registration which is due to expire, or deregistering when the MN returns home. Binding Update is also used to notify CNs about the current attachment point of the MN (optional in Mobile IPv4 and mandatory in Mobile IPv6). There is a lifetime period associated with each binding in the cache. MNs are responsible for keeping the bindings alive by reregistering before the lifetime has expired or else the binding will be deleted from the cache. Maintaining a binding cache is a good way of optimising the communication with an MN. Securing the Binding Update is also important - an authentication extension should accompany the Binding Update messages.

3.1.2.2 Components of Mobile IP Handoff Time

The handoff time in basic Mobile IP has the following main components (excluding link layer handoff):

- **Movement detection delay**: This is the time taken for a MN to detect that it has moved to a new link.

- **New address establishment delay**: This is the time for MN to configure a new address and adopt it.

- **Registration delay**: This is the time from sending Binding Update to HA from the MN to the time the MN receives the first packet on the new link. Registration delay depends on the round trip time between the MN and the HA.
3.1.3 Mobile IP Handoff Types

Mobile IP handoff can be classified in various ways, according to different kinds of criteria. The following sections introduce several of these.

3.1.3.1 Hard and Soft Handoff

Handoff can be classified based on the number of connections that a MN maintains during the handoff procedure.

Hard Handoff

In this type of handoff, the MN switches the communication from the old link to the new link. There will be a short interruption in transmission after the old link is abandoned and before the new link is established. In hard handoff, the handoff process involves both link layer and IP layer re-establishment.

Soft Handoff

The MN is connected to two links simultaneously. As it moves from one area to another, it “softly” switches from one link to another. When connected to two links, the MN can receive packets from both links and the network combines information received from 2 different routes to obtain a better quality. This is also referred to as macro diversity [VLR01].

3.1.3.2 Fast, Smooth and Seamless Handoff

Mobile IP handoff can also be classified based on its performance: Fast Handoff, Smooth Handoff and Seamless Handoff. Fast Handoff concerns the interruption time between disconnection at the old attachment point and connection to the new attachment point. Smooth Handoff concerns the rate of packet loss rather than the interruption time. Seamless Handoff combines Fast and Smooth Handoff.

Fast Handoff

Fast handoff minimises the latency due to handoffs, which is critical for real-time services. An example of fast handoff is: the MN anticipates handoff, acquires a new CoA and registers with HA before establishing a link to the new network. As soon as the MN leaves the current link, the old network’s router starts forwarding traffic to the new network’s router.

Smooth Handoff

Smooth Handoff concerns the rate of packet loss. Smooth Handoff minimises packet loss during the time that the MN is establishing its link to the new attachment. An example is: the MN sends a Binding Update containing the bindings of new CoA and old CoA to HA. Packets destined to old CoA from CNs will be intercepted by the HA and the HA will tunnel those
packets to new CoA. The MN will then send the Binding Update with its new CoA to the CN so that CN can send packets to the MN new address through direct Binding. For Smooth Handoff, the MN also may use multiple CoAs when it is in an overlap network region such as overlapping wireless cells.

**Seamless Handoff**

Seamless handoff combines both Smooth and Fast handoff. Seamless Handoff aims to sustain the service provided to a MN’s traffic during handoff.

**3.1.3.3 Handoff Control**

Handoff can be classified based on types of handoff control: mobile controlled type or network controlled type. In network controlled handoff, entities in the serving domain direct MN’s handoff to a new location. New points of attachment are determined by the network elements. In mobile controlled handoff, the MN determines its new point of attachment and directs MN’s handoff to the new attachment point [VRL01].

Four approaches to handoff control are as follows:

**Mobile Initiated (Mobile Controlled) Handoff**

The MN interprets downlink behaviour and decides when to handoff.

**Mobile Evaluated (Network Controlled) Handoff**

The MN interprets downlink behaviour but the network decides when to handoff.

**Network Initiated (Network Controlled) Handoff**

The network interprets uplink behaviour and decides when to handoff.

**Mobile Assisted (Network Controlled) Handoff**

The MN assists the network by taking measurements from the downlink, the network interprets these and decides when to handoff.

When using network initiated handoff to provide a handoff mechanism to the MN, there are often modifications and extensions to the entities in the network infrastructure such as additional signaling. For example, in 802.11 networks, approximate location and signal strength of the MN may be cached in nearby routers or base stations to force handoff of MN. This method has the advantage of offering a complete mobility management protocol for the network infrastructure, but the disadvantage of introducing greater complexity. Mobile initiated handoff involves modifications or extensions on the client side. This method has the possible disadvantage of loss of control to some extent by the network, but the advantage of simplicity and scalability.
3.1.3.4 Proactive and Reactive Handoff

Handoff also can be classified as either proactive or reactive. Proactive handoff assists the MN in detecting handoff and establishes packet flow to a target router prior to the handoff event (e.g. link layer coupling). This type of handoff is a hybrid of mobile assisted and mobile controlled handoff types. Reactive handoff uses generic movement detection schemes to detect handoff.

3.1.4 Basic Mobile IP Requirements and Strategies

Mobile IP ideally should meet the following basic requirements [WWRF01]:

- Support fast handoffs: In order to support real-time applications, the handoff delay must be minimised.
- Support smooth handoffs: In order to support loss-sensitive applications, handoff from one network to another must be accomplished with little or no loss of IP packets.
- Have functionality to minimise network signaling and physical power consumption: Bandwidth is a constrained resource especially in the case of wireless bandwidth. Another limited resource is the battery power available to a MN. Mobile IP therefore should minimise the amount of (over-the-air) signaling and allow a MN to enter power-saving mode when it is idle.
- Provide QoS support.
- Be secure, be able to work across security domains.
- Provide scalable support: The growing range of mobile devices and different access technologies requires that mobility support is scalable for a large number of MNs and mobile subnets.

The following strategies which have been proposed by research communities could help enhance Mobile IP to support seamless (fast, smooth) handoff. Details of these approaches are described in chapter 5

- Early address configuration: Mechanisms such as sending advanced handoff initiation messages to generate a new address in the new location could reduce delay
- Reduced signaling path: Frequent changes in MN’s CoA can potentially cause significant disruption in traffic. Using some hierarchical mobility support for micro mobility management could reduce the signalling delay (micro mobility is concerned with mobility within a regional network whereas macro mobility management handles mobility between regional networks).
- Reduced packet loss: Techniques such as multicast routing can be used to route packets to multiple locations to reduce packet loss.

### 3.2 IP Mobility Support for IPv4 (Mobile IPv4)

Mobile IPv4 introduces the concept of a Foreign Agent to help support Mobile IP. A Foreign Agent (FA) is a router on an MN’s visited network that provides routing services to the MN while registered with the HA. For packets sent by the MN, the FA may serve as a default router for registered MNs.

In Mobile IPv4, when a MN is away from home network, the MN is associated with a CoA provided by Foreign Agent (Foreign Agent CoA) or by external mechanisms such as DHCP (for a co-located CoA). After obtaining a new CoA, the MN then begins the registration process (registering the new CoA to HA). After HA acknowledges MN’s registration, a tunnel is established.

When the MN obtains a Foreign Agent CoA, the tunnel is established between HA and FA (Figure 3.3). The HA intercepts and encapsulates the packets and tunnels them through to the Foreign Agent. Each packet arrives at the end of the tunnel, is decapsulated and is then delivered to the MN’s CoA by the FA. When the MN obtains a co-located CoA, the tunnel is established between the HA and the MN; packets then are tunneled from the HA to the MN.

A MN sends its packets directly to the CNs using normal IP routing, which does not need encapsulation. The MN uses its Home Address as the source address of all IP datagrams that it sends. (Problems caused by reverse path filtering are discussed in section 3.2.2.2).

![Figure 3.3: Mobile IPv4 Elements](image-url)
3.2.1 Mobile IPv4 Operations

Mobile IPv4 handoff operations can be grouped as follows [Per02a]: Agent Discovery, Registration and Tunneling. These are detailed further in the following sections, and summarised in Figure 3.4.

3.2.1.1 Mobile Agent Discovery

Mobile Agents (Home and Foreign Agents) advertise their presence on each link via Agent Advertisement messages. Agent Advertisement messages contain information about the addresses of mobile agents and supported services. Advertisements have a lifetime value to show how long they are valid. The MN monitors Agent Advertisement messages and saves the received data for further processing. The MN may optionally send an Agent Solicitation message on the link of the new foreign network to find the active mobile agents quickly. If Agent Advertisement messages are sent periodically, they should be sent at an interval no longer than 1/3 of the Advertisement Lifetime given in the ICMP header. In Mobile IPv4, the recommended Agent Advertisement interval is one second. The MN then uses movement detection mechanisms to detect that it has changed link.

RFC 3344 defines two movement detection mechanisms based on Agent Advertisements.

- The first method uses the lifetime of the Agent Advertisement. If the MN has not received any Agent Advertisement message from the current Agent during this lifetime, the MN then registers to the new network from which it receives an Agent Advertisement message, and then performs handoff. This lifetime field is set in the ICMP Router Advertisement portion of Agent Advertisement and its value is triple the interval of Agent Advertisement message. If the Agent Advertisement interval is one second, advertisement lifetime will be three seconds. Therefore a MN after missing three consecutive advertisements can assume it is not reachable via the current router. Average movement detection delay is roughly 2.5 seconds

- The second method uses information from network prefixes. From the source IP address in the new Agent Advertisement messages, a MN can determine whether it has moved. The MN checks and evaluates network prefixes from new messages. If the network prefixes differ from network prefix of MN’s current network, the MN detects that it has changed its location and initiates handoff. This method requires Prefix-Length Extension in the Agent Advertisement message. Prefix-Length Extension contains the network prefix and a field for the length of the network prefix. The average movement detection delay is about 0.5 seconds.

MNs can also use other optional movement detection mechanisms, e.g. link-layer information.
3.2.1.2 Registration

The MN registers its new CoA to its HA directly (or alternatively via a FA) through exchange of Registration Request and Registration Reply messages. If the MN sends its request via a FA, the HA replies also via the same FA. The Registration Request contains a new CoA for the MN, flags specifying the requested connection type, and authentication data. The HA uses this information to update its data structures for the MN. It associates the home address of the MN to its current CoA and to the lifetime of this binding. If registration messages go through a FA, it maintains a similar mobility binding and forwards data packets to the current point of attachment of MN. The FA updates its bindings only after the HA has authenticated the Registration Request to protect the data from unauthorised changes. Registration messages are UDP packets. The MN is thus responsible for resending the request if it does not receive a reply.

3.2.1.3 Tunneling

Tunneling is the mechanism used by the HA to deliver packets to the MN when away from the home network. Tunneling is done by IP-in-IP encapsulation that sends the original IP packet as a payload part of another IP packet as described in section 3.1.1.3.

---

**Figure 3.4: Basic Mobile IPv4 Handoff Procedures**

3.2.2 Mobile IPv4 Issues and Extensions

3.2.2.1 Mobile IPv4 Issues

Specific issues in applying Mobile IPv4 have been raised:
- Private address problems: IPv4 supports private addresses inside organisations to reduce the uses of public addresses. Mobile IPv4 does not consider these private addresses since it assumes that all the entities in Mobile IP protocol have public addresses that can be reached from the Internet. This limits the environments in which Mobile IP can be used as it increases the number of the needed public addresses.

- Triangle routing: This is where all communication between CN and MN is via HA. This increases the amount of data in the Internet and the latency between a MN and the CNs with which it is communicating.

- Handoff problems: Interruption during handoff may cause jitter, delay, and packet losses for MN’s connections. There may be excessive amount of (over-the-air) signaling when a MN is moving frequently from one subnetwork to another. Especially in the case that latency between the foreign and home networks is large, the overhead of the basic Mobile IPv4 may be substantial.

- Ingress Filtering: Many IPv4 routers are nowadays using ingress filtering for all outgoing packets to defeat address spoofing [FS00]. This means that routers accept only the packets with a topologically correct source IP address (an address in the correct subnet) and they drop the packets from MNs with addresses of other subnets. Packets sent to the MN are encapsulated by the HA. When using FA decapsulation to decapsulate packets sent from HA to MN, MN uses its home address as the source IP address of the packets. When MN sends packets to CNs, it uses normal routing with source address as MN’s home address (instead of MN’s CoA). Ingress filtering routers reject these packets and thus the MN cannot maintain its connections.

3.2.2.2 Mobile IPv4 Extensions

New extensions to Mobile IPv4 standard have been suggested to overcome Mobile IPv4 deficiencies. Basic extensions by Mobile IPv4 working group are reverse tunnels, route optimisation, and foreign agent smooth handoffs.

Reverse Tunnels
RFC 3024 [Mon01] presents a solution to ingress filtering, allowing MNs to encapsulate packets in the reverse direction, using a reverse tunnel. Encapsulated packets from MN to CNs are routed via the HA. Same route is used as the packets in a forward tunnel from CNs (Figure 3.5). The outer IP header of the encapsulated packets has a topologically correct IP source address, the CoA of the MN in the visited network. Ingress filtering does not thus prevent these packets from being forwarded.

Reverse tunnels also add an extension to select different delivery styles for encapsulation. MNs can use this extension to select a mode in which they can decide on packet by packet basis whether the encapsulation via the HA is used or not. This allows MNs to use an optimised route for packets destined to the foreign network while still being able to connect to the nodes outside the ingress filtering routers.

Reverse tunneling also enables a MN with a private address to communicate with its home private network.

**Route Optimisation**

Route Optimisation [Per02b] solves Mobile IPv4 triangular routing problems. Route Optimisation sends Binding Update to inform the CN the current CoA of MN to enable the use of the optimal route for packets in both directions (Figure 3.6). This establishes direct communication between CN and MN.
In standard Mobile IPv4, HA maintains an address-mapping table that records the mapping relationship between the home addresses of all MNs and the associated CoAs. With Route Optimisation, apart from the table in the HA, each node (both the CN and the MN), must maintain a binding cache. The cache records the CoA associated with the home address of MN that the node communicates with. It is impracticable to implement a binding cache in all of the nodes over the entire Internet.

CNs use encapsulation on packets when sending packets to MN in Route Optimisation, therefore ingress-filtering routers accept them. In return, MNs may, however, still need reverse tunneling via the HA to ensure a topologically correct source address. The situation will be improved with CNs that support decapsulation - MNs can encapsulate packets directly to CNs. Basic IPv4 hosts need not support decapsulation, so this option is available only with hosts that have an extended protocol stack. Requiring that all the deployed IPv4 hosts would change their protocol stacks to support Mobile IPv4 is not feasible.

Route Optimisation is optional in Mobile IPv4 and is not often deployed because it requires CN modifications to implement binding caches and decapsulation.

**Foreign Agent Smooth Handoffs**

This operation is useful to define a smooth handoff mechanism when a MN moves from one FA to another FA. During registration with the new FA, MN sends a Binding Update message to previous FA, it will then be able to re-tunnel the MN destined packets to new MN’s CoA. This will reduce the number of packets lost and speed up the CN’s binding cache entries update process.
3.2.3 Mobile IPv4 Security

In Mobile IPv4, the security issues are handled by individual security mechanisms for each function, based on statically configured mobility security associations. In RFC 3344 [Per02a], Mobile IPv4 protocol exchanged messages apply Mobility Security Association, which is defined as a collection of security contexts between a pair of nodes. Each context indicates an authentication algorithm and mode, a secret (a shared key, or appropriate public/private key pair), and a style of replay protection in use.

Mobile IPv4 security defines identification and authentication mechanisms for registration messages between MN and HA to protect against registration replay attack. Registration messages contain identification fields. There are two identification mechanisms: using Timestamps and Nonces. The MN and HA must agree in advance which method to use as part of the mobile security association. A Nonce is a randomly chosen value inserted in a message. The MN initiates a nonce in Registration Request to send to the HA, and the HA should return the same number in Registration Reply. In timestamp method, MN generates Time-of-day. HA verifies this against its own Time-of-day. The difference of the two values should be a value that lies in the range of an agreed number (usually less than 7 seconds) [Per02a].

Authentication of registration messages between HA and MN is done using message authentication codes. The receiver of the registration message checks that this code is correct before it takes any actions based on the message. Only the authenticated nodes can thus generate correct registration messages. The default message authentication code algorithm is HMAC-MD5 [Per02a], with a key size of 128 bits. The registration extension uses a shared secret MD5 to protect data.

Authentication between HA and FA, and between MN and FA is also possible using similar security approaches.

For data integrity, an external security method such as end-to-end encryption using IP Security (IPSec) [KA98a] can be applied on top of Mobile IP.

3.2.4 IPv6 over Mobile IPv4

IPv6 over Mobile IPv4 [CEHM02] reviews Mobile IPv4 extensions that may be used by dual stack MNs to obtain IPv6 service by the use of Mobile IPv4 registrations. Mobile IPv4 registrations are used by these MNs to request IPv6 service from Mobile IPv4 Home Agents.

3.3 Impact of Mobile IP Handoff on Higher Protocols

Disruption to data flow during handoff may worsen the performance of higher layer protocols (such as Transmission Control Protocol (TCP) [Pos81a] and User Datagram Protocol (UDP) [Pos80]).
Without smooth handoff support, data packets sent by a UDP source during handoff will simply be lost.

For a TCP connection, the performance is further degraded as TCP involves exponential back-off behaviour for packet retransmissions when there is loss of packets. Furthermore, TCP misinterprets packet loss and delay during handoff as being caused by congestion, and this triggers TCP congestion control mechanisms which reduce throughput. The following subsection details TCP responses to packet loss during Mobile IP handoff.

3.3.1 Impact of Mobile IP Handoff on the Performance of TCP

TCP provides connection-oriented and reliable services between two hosts. The sender will stop transmission after all the bytes in the window have been sent. Each packet or group of packets sent has to be acknowledged regularly to guarantee reliability. During Mobile IP handoff, a TCP sender will assume that the packets it has sent are lost and continually retransmit the lost packets. This process also involves TCP exponential back-off behaviour, which further reduces TCP throughput. Consequently, TCP will assume the packet loss and delay is due to congestion, and thus invoke its congestion control mechanism (Slow Start and Congestion Avoidance Algorithm) to decrease the TCP sending rate. This misinterpretation further degrades the TCP performance over Mobile IP handoff.

Performance Degradation with Exponential Back-off Behaviour

TCP utilises exponential back-off behaviour for packet retransmissions. In the presence of packet loss, the sender retransmits unacknowledged packets after the retransmission timeout (RTO). To prevent network congestion, the RTO is doubled for each unsuccessful retransmission. The exponential back-off algorithm can result in long delays before retransmitting lost packets. Therefore after TCP receiver is ready to receive packets from the sender, there can be addition delay in packet transmission on top of the handoff delay.

Performance Degradation with Slow Start and Congestion Avoidance Algorithm

The slow start and congestion avoidance algorithm has been designed to prevent TCP from transmitting its full window size when the underlying network is congested. It is based on the assumption that if packets are lost during transmission, it is due to congestion. As a result TCP immediately reduces its current window size. The long handoff delay results in a small window size immediately after the link is restored. So it takes more time to reach a throughput similar to the one before the handoff.
3.4 Mobile IP QoS Issues

3.4.1 Issues

IP QoS protocols (such as RSVP and DiffServ) cannot be used directly in mobile environments. Using RSVP, which does not have any provision for passive reservation, a MN cannot make an advance reservation from a location where it is not currently present [TBA01]. When the MN moves to a new location, its previously reserved resources are no longer available. QoS for the MN at its new location is not guaranteed. RSVP resources must be re-reserved along the entire path each time the MN moves its attachment to a new network. This process implies a heavy load in terms of control traffic and introduces additional delays (on top of Mobile IP handoff disruptions). In addition, as the IP tunnel used by the MN is implemented by an IP-in-IP encapsulation scheme, routers inside an IP tunnel used in Mobile IP are not capable of recognising RSVP signaling messages. The original RSVP protocol number 46 is contained in the inner IP header, and so routers cannot recognise encapsulated PATH and RESV messages. Several problems also occur when applying DiffServ to Mobile IP environments. Problems are due to differences in network provisioning in mobile environments, in definition and selection of Service Level Agreements, lack of dynamic configurations, issues in mobile flow identification and billing [CK01]. As IntServ and DiffServ do not consider mobile environments, extensions to these architecture or new architectures are needed. Details of QoS schemes for Mobile IP can be found in Appendix B.

3.4.2 Requirements of a Quality of Service (QoS) Solution for Mobile IP

When a MN changes its attachment to a new location, the MN’s packet stream changes its path. After handoff, the packets belonging to the MN’s ongoing session start using a new CoA. They thus cannot be recognised by some forwarding functions. Handoff also may occur between networks that are under different administrative control. RFC 3583, “Requirements of a Quality of Service (QoS) solution for Mobile IP” [Cha03], provides steps for defining proper QoS forwarding treatment to the packets sent by or destined to a MN as they propagate along different routes in the network due to mobility. There are four steps involved in solving the QoS problem for Mobile IP [Cha03]:

1. List the requirements that Mobile IP places on the QoS mechanism
2. Evaluate current IP QoS solutions against these requirements
3. Determine if current solutions need to be extended, or if new solutions need to be defined,
4. Define new solutions or fix the old solutions
RFC 3583 also describes requirements for an IP QoS mechanism to operate satisfactorily with Mobile IP. Requirements for proper Quality of Service (QoS) forwarding treatment of the MN’s packet stream at the intermediate nodes in the network include:

**Standard Requirements**

The QoS solution for Mobile IP should satisfy standard requirements such as scalability, security, conservation of wireless bandwidth, low processing overhead on mobile terminals, providing support for authorisation and accounting, and robustness against failures of any Mobile IP specific QoS components in the network.

**Performance Requirements**

The interruption in QoS during handoff should be minimised. QoS (re)establishment to the affected parts of the packet path in the network should be localised. After handoff the QoS state along the old packet path should be released.

**Interoperability Requirements**

The QoS solution should be interoperable with mobility protocols and interoperable with heterogeneous packet paths.
4 Mobile IPv6 – Improvements over Mobile IPv4 ................................................................. 39

4.1 Mobile IPv6 Technologies .......................................................................................... 39
   4.1.1 Basic Mobile IPv6 Concepts .............................................................................. 39
   4.1.2 Mobile IPv6 vs. Mobile IPv4 .......................................................................... 41
   4.1.3 Mobile IPv6 Tunneling vs. Route Optimisation .............................................. 42
   4.1.4 Link Layer Information .................................................................................... 44
   4.1.5 Mobile IPv6 Security ....................................................................................... 45

4.2 Mobile IPv6 Handoff Operations ............................................................................ 46
   4.2.1 Movement Detection ....................................................................................... 47
   4.2.2 Forming New Care-of Addresses .................................................................... 48
   4.2.3 Binding Management ...................................................................................... 49
   4.2.4 Returning Home .............................................................................................. 49
   4.2.5 Summary of Mobile IPv6 Handoff Operations .............................................. 50

4.3 Mobile IPv6 Issues .................................................................................................. 50
   4.3.1 Issues with Information in Received RA ......................................................... 51
   4.3.2 Delay in NUD Mechanisms ............................................................................ 52
   4.3.3 Hints .............................................................................................................. 52

4.4 Basic Support for Network Mobility ....................................................................... 52
   4.4.1 Overview of Network Mobility ....................................................................... 52
   4.4.2 Mobile IPv6 Network Mobility Basic Support Protocol .............................. 53
Chapter 4

Mobile IPv6 – Improvements over Mobile IPv4

This chapter provides a deeper review of Mobile IPv6 technologies in terms of detailed operation, characterisation and security. It also addresses Mobile IPv4 issues that have been solved by Mobile IPv6. Mobile IPv6 current issues are introduced. Mobile IPv6 handoff operations and Mobile IPv6 issues are the most important in literature survey part of thesis as they provide detailed Mobile IPv6 handoff procedures and matters that are discovered in experiments described in chapter 7. Finally, the concept of network mobility, which offers advanced and extended types of Mobile IPv6 networks, is introduced.

4.1 Mobile IPv6 Technologies

The design of Mobile IP support in IPv6 [RFC 3775] [JPA04] draws benefits both from the experiences gained from the development of Mobile IP support in IPv4 and features provided by IPv6. This section first introduces some basic concepts of Mobile IPv6, followed by a comparison of Mobile IPv6 and Mobile IPv4, and an overview of Mobile IPv6 routing and security.

4.1.1 Basic Mobile IPv6 Concepts

Mobile IPv6 is designed to manage Mobile Node (MN) movements between IPv6 networks. In this chapter, the MN refers explicitly to “an IPv6 Mobile Node”. With Mobile IPv6, any MN becomes accessible to all other hosts through its home address identity.

A Home Agent (HA) is a router on the MN’s home network. The HA handles the mobility of the MN by maintaining MN’s current location information. An Access Router (AR) is defined to be the last router in the foreign network that can forward packets to MN. In Mobile IPv6, the concept of Foreign Agent (FA) does not exist. Instead, in a foreign network an AR plays a similar role to that of a FA in Mobile IPv4 except the AR does not need to assign its IP address to be the MN’s Care-of Address (CoA).
Figure 4.1 shows basic Mobile IPv6 elements.

![Figure 4.1: Basic Mobile IPv6 Elements](image)

When a MN moves to a new subnetwork, the MN gets a new CoA. The CoA is either configured using IPv6 stateless Address Autoconfiguration or stateful Address configuration (such as DHCP). The MN then registers this CoA to the HA.

There are two possible ways for the MN and Correspondent Node (CN) to communicate when the MN is away from home: direct communication (Route Optimisation) and tunneling via the HA. In Mobile IPv6, Route Optimisation is fundamental. MN sends Binding Updates to CNs for direct communication. On the other hand, if the CN is a non-Mobile IPv6 node, the CN can communicate with the MN via tunneling through the HA. Packets from a CN destined to the MN’s Home Address are intercepted by the HA and then tunnelled to the MN’s CoA, and vice versa (packets from the MN CoA are encapsulated and tunnelled to the HA and forwarded to the CN).

In Mobile IPv6, some of the basic terms from Mobile IPv4 are renamed but have the same meanings:
- Agent Advertisement is changed to Router Advertisement.
- Registration Request message is changed to Binding Request or Binding Update.
- Registration Reply is changed to Binding Acknowledgement.

The Router Advertisement (RA) beacon from an AR multicasts network identification information at regular intervals. A MN may use Router Solicitation (RS) to trigger a RA.
4.1.2 Mobile IPv6 vs. Mobile IPv4

This section summarises the main advantages of Mobile IPv6 compared to Mobile IPv4

Larger Address Space

Mobile IPv6 provides enough addresses to assign addresses to visiting MNs and also allows a MN to acquire multiple CoAs. Multiple CoAs allow for multiple concurrent base station connections in a Mobile IPv6 environment. This aids reducing packet loss during handoff.

No Foreign Agent

Mobile IPv6 eliminates the need for FAs required in Mobile IPv4. Mobile IPv6 nodes acquire CoA through stateful or stateless algorithms without the need to adopt CoA from a FA’s address.

Improved Functions

- IPv6 hosts are supported with IPv6 encapsulation therefore Route Optimisation and Reverse Tunneling are fundamental in Mobile IPv6
- Address Configuration: MN uses stateless address autoconfiguration to acquire its CoA. The MN may assign itself a Link-Local Address using Neighbor Discovery and Duplicate Address Detection (DAD), then assign a Site-Local and/or Global Address using Neighbor Discovery.
- Neighbor Discovery: In Mobile IPv6, IPv6 Neighbor Discovery optimises this process of obtaining the AR’s MAC address from the RA. The MN then is able to send a Binding Update without any delay after a handoff. In Mobile IPv4, the MN learns the MAC address of the FA through the ARP process, which introduces a delay.

Security and Authentication

While Mobile IPv4 does not have built in security, Mobile IPv6 Route Optimisation can operate securely without pre-arranged security associations using Mobile IPv6 Return Routability (Mobile IPv6 RFC 3775 [JPA04]). Mobile IPv6 also provides a scalable approach to establishing binding security associations using IP security protocol (IPsec). While Mobile IPv4 only uses IPsec for data integrity, IPsec is an integral part of Mobile IPv6 and provides sender authentication, data integrity, and replay protection. Using IPsec, Mobile IPv6 can protect routing header, destination operation header, and tunneling under mobile situations.

Automatic Home Agent Discovery

The “Dynamic Home Agent Address Discovery” mechanism described in Mobile IPv6 RFC 3775 allows the MN to dynamically discover the IP address of the HA, even when the MN is
away from home. MNs can also learn new information about home subnet prefixes through the “prefix discovery” mechanism.

Mobile IPv6 can provide support for multiple HAs and changes of HA for a MN. Only one HA will be selected as the MN’s primary HA. The Dynamic Home Agent Address Discovery mechanism in Mobile IPv6 returns a single HA address to the MN.

Support for Route Optimisation

Support for Route Optimisation is an integral part of basic Mobile IPv6.

Efficient Routing

IPv6 Destination Option Header: The IPv6 Destination Option Header makes the use of Mobile IPv6 transparent for higher layers. Mobile IPv6 MN uses the IPv6 Destination Option Header for the Home Address Option to send packets to the CN (Figure 4.2). IPv6 Home Address Option of the Destination Option Header contains transparent information only processed by the final destination of the packets. The Home Address is then examined only at the ultimate destination. IP traditionally passes the Source IP address field up to any higher-level protocol (e.g. TCP). The Home Address Option changes this behavior, so that the Home Address is passed instead, resulting in stable identification for higher-level protocols.

IPv6 Routing Header: The IPv6 Routing Header is used by CNs to send packets to a MN while the MN is in a foreign network (part of Route Optimisation).

4.1.3 Mobile IPv6 Tunneling vs. Route Optimisation

Communication between the MN and CN may be performed using two possible methods: bidirectional tunneling and Route Optimisation.

4.1.3.1 Bidirectional Tunneling

Bidirectional tunneling does not require Mobile IPv6 support at the CN. Bidirectional tunneling is available when the MN registers its current CoA to the HA.

- Packets from the CN are routed to the home network and intercepted by the HA using standard IP routing. Packets then are encapsulated and delivered via the tunnel to the MN’s primary CoA using IPv6 encapsulation

- In the reverse direction, packets from the MN to the CN are “reverse tunneled” from the MN to the HA using Home Address, and then routed to the CN using standard IP routing mechanisms.
4.1.3.2 Route Optimisation

In Route Optimisation, any host that wishes to communicate with the MN uses the CoA as a destination address for direct communication (instead of going through the HA). Route Optimisation requires Mobile IPv6 support at the CN as Mobile IPv6 Return Routability is required before Route Optimisation actually starts (covered in section 4.1.4.2). Figure 4.2 and 4.3 below demonstrate how Route Optimisation works in Mobile IPv6.

**MN Terminated Packet Delivery**

Before sending a packet to any destination, the CN checks its cached bindings for an entry matching the packet’s destination address. When a binding cache is not found in the CN, packets delivered to the MN will use the CN’s address as the source address and the MN’s Home Address as the destination address. If the cached binding for this destination address is found, the CN uses an IPv6 routing header to specify the MN’s home address and changes the destination address to the MN’s CoA. Thus packets are routed to the MN’s CoA.

![Figure 4.2: MN Terminated Packet Delivery in Route Optimisation](image)

**MN Originated Packet Delivery**

When the MN is at home, packets from the MN delivered to the CN will use the MN’s Home Address as the source address, and the CN’s address as the destination address. When the MN moves to a foreign network, after it has sent Binding Update to the CN, the MN’s Home Address is specified using the Home Address Option (in the Destination Options Header) and the source address is changed to MN’s CoA. When packets arrive at the CN, the CN uses the MN’s Home Address as the source address.
**Figure 4.3: MN Originated Packet Delivery in Route Optimisation**

**Binding Update to CN**

The Binding Update message to a CN contains header 1, header 2, header 3, and a PDU. Header 1 contains the MN’s CoA as its source address. The Source Address is thus correct for routers in the packet path which do reverse path filtering. Header 2 contains the MN’s Home Address as Home Address Option. Header 3 contains destination cache bindings. The MN maps the Home Address to its CoA.

**Figure 4.4: Binding Update Format to CNs.**

### 4.1.4 Link Layer Information

A link layer trigger is defined as link layer information that indicates the possibility of link layer change. The link layer trigger may be considered as an indication that there is a change of IP subnet. This indication also may be used to initiate IP based reachability checks. When a new link layer connection is about to be made, the link layer sends this trigger to the IP layer, or when a new link layer connection has just been made, the link layer may send a “link up” notification to the IP layer.

However, not all IP implementations at the network layer will be able to understand indications from the link layers and those indications will not always be sufficient to make a proper decision about impending handoff.
4.1.5 Mobile IPv6 Security

Mobile IPv6 uses IPsec Security Association to authenticate its control traffic between the HA and the MN. If this traffic is not protected, MNs and CNs are vulnerable to man-in-the-middle attacks, hijacking, passive wiretapping, impersonation, and denial-of-service attacks. Any third parties are also vulnerable to denial-of-service attacks, for instance if an attacker could direct the traffic flowing through the HA to an innocent third party. When authenticating the Binding Update between a MN and CN, a Return Routability procedure is used instead of Security Association.

4.1.5.1 IPsec Security Association

Before sending Binding Updates to the HA for Mobile IPv6 registration, the MN must register itself to the HA which creates an IPsec Security Association between the MN and HA. IPSec Security Association authenticates the Binding Update and Binding Acknowledgement messages between MN and HA. It is possible to either manually key IPsec Security Association or to configure Internet Key Exchange (IKE) [HC98] (specified in Mobile IPv6 RFC 3775) to automatically establish a Security Association. IKE can be used if it is supported by both HA and MN. Either Authentication Header (AH) or Encapsulated Security Payload (ESP) of IPsec can be used with an authentication algorithm. IPsec also protects ICMPv6 messages between the MN and the HA, which are used for prefix discovery or Return Routability messages (Home Test Init and Home Test).

Apart from the use of IPsec, Binding Updates also can be protected by the use of the Binding Authorisation Data option. This option employs a binding management key, which is called Kbm (Kbm stands for key binding management), which can be established through the Return Routability procedure. Correct ordering of the control traffic is ensured by a sequence number in the Binding Update and Binding Acknowledgement messages.

4.1.5.2 Return Routability Procedure

According to RFC 3775, the Return Routability (RR) procedure enables the CN to obtain some reasonable assurance that the MN is addressable at its claimed CoA as well as at its Home Address. The procedure uses cryptographic tokens. CN sends test messages as a challenge, and MN responds. After this, the MN constructs (from random data and data gathered from the procedure) a binding management key, that is used in the binding procedure. When RR succeeds, Binding Update messages are sent to the active CNs. Home Test Init (HoTI), Home Test (HoT), Care-of Test Init (CoTI), and Care-of Test (CoT) are the four messages used to ensure authorisation of those Binding Updates.
RFC 3775 states that the RR procedure requires very little processing at the CN. Home Test Init and Care-of Test Init are sent from MN to CN, if CN supports Mobile IPv6 the Home and Care-of Test messages can be returned quickly to MN.

A HoTI message is sent from the MN to the CN via the HA to acquire a home keygen token. The HoTI contains the MN’s Home Address and a home init cookie that the CN must return later. A HoT is sent to the MN at its CoA via the HA. The HoT contains a home keygen token, home init cookie and home nonce index. The home keygen token tests that the MN can receive messages sent to its Home Address. The home init cookie ensures that the message comes from a node on the route between the HA and CN. The home nonce index allows the CN to efficiently find the nonce value that it used in creating the home keygen token.

A CoTI message is sent to the CN from the MN’s CoA to acquire the Care-of keygen token. The CoTI contains the MN’s CoA and a Care-of init cookie that the CN must return later. A CoT is sent from CN to the MN’s CoA. The CoT is generated with a Care-of keygen token, Care-of init cookie and Care-of nonce index. The Care-of init cookie ensures that the message comes from a node on the route to the CN. The Care-of nonce index is provided to identify the nonce used for the Care-of keygen token.

When the MN has received both the Home and Care-of Test messages, the RR procedure is complete. As a result of the procedure, the MN has the data it needs to send a Binding Update to the CN.

### 4.2 Mobile IPv6 Handoff Operations

Mobile IPv6 operates as follows: The MN first performs movement detection in order to detect layer 3 handoff. If the MN has moved and its current IP address is invalid in the new link, the MN will perform DAD for its local address on the new link, form a new CoA and handle binding management. Figure 4.5 illustrates basic Mobile IPv6 handoff steps.
4.2.1 Movement Detection

The primary goal of movement detection is to detect layer 3 (L3) handoffs. To detect L3 movement, the MN can use the following methods (RFC 3775 [JPA04]):

Generic Movement Detection

Generic movement detection uses the facilities of IPv6 Neighbor Discovery, including Router Discovery and Neighbor Unreachability Detection (NUD) [NNS98]. Router Discovery allows the periodic multicast of RAs to nodes on an IPv6 network, and additionally allows solicitation of RAs to confirm network identity, or to speed device configuration. NUD allows solicitation of Neighbor Advertisement to confirm router reachability.

A MN can use the information in received RAs to detect subnet prefix changes and thus L3 handoffs. However the MN needs to consider the following issues in order to use this information:

- RA information that identifies a new router does not necessarily indicate L3 handoff (there may be multiple routers on the same link).
- RA information that identifies a new router with a new prefix does not necessarily indicate L3 handoff (multiple routers on the same link may advertise different prefixes).
- Link local addresses of routers in RA information are not unique and thus they are not considered to be a reliable movement indication (routers often use the same link local address on multiple interfaces). Routers may use the Router Address (R) bit (1-bit router
address flag in the Prefix Information Option of RA) to include the global address in the RA.

Generic movement detection uses NUD to detect that the default router is no longer bi-directionally reachable so that the MN must discover a new default router and proceed to handoff. However, NUD only operates when the MN has packets to send.

Hints to Aid in Movement Detection

According to RFC 3775, the MN should supplement generic movement detection with other information whenever it is available to the MN (e.g. from lower protocol layers). The MN considers the following events as indications that a L3 handoff may have occurred [JPA04].

- The Advertisement Interval Option field of the RA indicates the maximum amount of time between successive Advertisements that the MN should expect. If the MN does not receive any RA from a router within this time, at least one RA sent by the router has been lost. The MN can then implement its own policy for the number of lost Advertisements from its current default router to indicate a L3 handoff.

- A link layer (L2) trigger may be obtained from lower layer protocols or device driver software within the MN. The following should be considered when using L2 trigger [JPA04]:
  - A L2 trigger may or may not imply L2 movement and L2 movement may or may not imply L3 movement
  - After receiving a L2 trigger, the MN should verify whether the default router is still bi-directionally reachable rather than immediately multicasting a router solicitation (except that it is well-known that a L2 trigger is likely to imply L3 movement). If the default router does not respond to the Neighbor Solicitation, a Router Solicitation should be multicast.

According to RFC 3775, the MN should avoid performing a L3 handoff until it is strictly necessary due to the temporary packet flow disruption and signaling overhead involved in updating mobility bindings. If the MN detects that the current default router on the old link is still bi-directionally reachable, it should continue to use the old router on the old link rather than use a new default router.

4.2.2 Forming New Care-of Addresses

When the MN detects a L3 handoff, it performs DAD [TN98] on its link-local address on the new network (to ensure that none of its interfaces is duplicated in the new network), selects a new default router and performs Prefix Discovery with that new router (to form new CoA(s)).
RFC 2462 [TN98] specifies that, for DAD, MN should delay sending the initial Neighbor Solicitation message by a random delay between 0 and the maximum router solicitation delay (MAX_RTR_SOLICITATION_DELAY) (default 1 second). However, according to RFC 3775, since delaying DAD can result in a significant delay in configuring a new CoA when the MN moves to a new network, the MN preferably should not delay DAD.

The MN then generates a new primary CoA using normal IPv6 mechanisms - either stateless [TN98] or stateful (e.g., DHCPv6 [DBV’03]) Address Autoconfiguration. The MN can form multiple CoAs. A MN can have only one primary CoA at a time (which is registered with its HA), but it may have additional non-primary CoAs for any or all of the prefixes on its current link. A MN may form a new primary CoA at any time, but a MN must not send a Binding Update about a new CoA to its HA at more than MAX_UPDATE_RATE (default 3 times within any second). A MN may also form new non-primary CoAs at any time. A MN may have CoAs on more than one link at a time.

### 4.2.3 Binding Management

The Binding Cache maintains Binding Acknowledgement, Binding Update and Binding Request. Binding Update is transmitted periodically. Before sending a Binding Acknowledgment to a MN to acknowledge its Binding Update, a HA should perform DAD for the MN’s Home Address.

**Home Agent Performs DAD**

The HA will perform DAD for the MN’s Home Address after the MN sends a Binding Update to register the MN’s CoA to the HA. The reason for this procedure in Mobile IPv6 is that, during some time for which the MN has no binding at the HA, another node may autoconfigure the MN’s Home Address. Therefore the MN treats the creation of a new binding with the HA using an existing Home Address in the same way as it treats the creation of a new Home Address. If DAD for the MN’s Home Address fails, the HA will reply with a Binding Acknowledgement containing a status indicating that DAD failed. MN should then use other Home Addresses to continue to register the CoAs.

### 4.2.4 Returning Home

A MN detects that it has returned to its home network (home subnet prefix is again on-link) via the movement detection algorithm in use. The MN then sends a Binding Update to its HA to instruct its HA to no longer intercept or tunnel packets for it. The source address in this Binding Update is its Home Address.

When sending this Binding Update to its HA, the MN needs to take care in the way it uses Neighbor Solicitation (if needed) to learn the HA’s link layer address, since the MN is unable to
use its Home Address as the source address in the Neighbor Solicitation until the HA stops defending the Home Address.

### 4.2.5 Summary of Mobile IPv6 Handoff Operations

The list below describes basic steps involved with Mobile IPv6 layer 3 handoff when a MN travels from the current AR to the new AR using generic Movement Detection (subnet prefix changes in the received RA) (RFC 3775 [JPA04]).

1. MN sends Router Solicitations to new AR (optionally).
2. MN receives new Router Advertisements from new AR.
3. MN detects that it is on a new link and has moved away from the old link.
4. MN performs DAD on the link local address.
5. MN performs new CoA creation.
6. MN sends Binding Update to HA.
7. HA performs DAD for MN’s Home Address and sends Binding Acknowledge to MN.
8. MN completes Return Routability and sends Binding Update to CN for Route Optimisation.
9. CN returns a Binding Acknowledgement to CN and MN starts to receive packets.

### 4.3 Mobile IPv6 Issues

Mobile IPv6 solves some of the Mobile IPv4 issues (private address problems and triangle routing) and provides improved security functions. Basic Mobile IPv6 however still faces challenging issues in trying to meet the requirements of Mobile IP specified in section 3.1.4: fast and smooth handoff support, functionality to minimize network signaling and power, QoS guarantee, security and scalable support for a large number of MNs and mobile IP-subnets. For example, some of the current research topics of the IETF Mobile IPv6 working group for Mobile IPv6 security [MIP6] are: Mobile IPv6 and Firewalls, Bootstrapping Authentication, Preconfigured Binding Management Keys for Mobile IPv6, Route Optimisation Security Design Background. Some approaches of the IETF Mobile IPv6 working group for Signaling and Handoff Optimisation [MSHO] are:

- Hierarchical Mobile IPv6 mobility management: introduces extensions to Mobile IPv6 and IPv6 Neighbor Discovery to allow for local mobility handling.
- Fast Handoffs for Mobile IPv6: specifies enhancements to reduce the handoff time due to standard Mobile IPv6 procedures
Mobile IPv6 Fast Handoffs for 802.11 Networks: describes how Mobile IPv6 Fast Handoff could be implemented on link layers conforming to the 802.11 suite of specifications.

Mobile IPv6 defines basic movement detection mechanisms based on information in received RA, NUD and hints from link layer, but there are some issues when using these mechanisms to detect layer 3 handoff. The IETF working group for Detecting Network Attachment in IPv6 (DNA) [CD04] addresses these issues (that the information contained in RA messages may be inadequate to represent a link change, the NUD mechanism, and delay in link layer hints).

### 4.3.1 Issues with Information in Received RA

**Link Local Scope of Router Address**

The source address field of the RA message is a router address. Since this router address has link local scope (its uniqueness is only guaranteed on the link), a MN cannot detect if it has moved to a new location by checking the router address information in RA messages. Furthermore, when the MN receives a RA message which has the same router address as MN’s default router address, that router may be a different one which happens to have the same link-local address as its default router address.

**Omission of Prefix Information Option**

The MN can check the prefix of its current IP addresses to see if it is included in the Prefix Information Option of incoming RA messages. But an unsolicited RA message can omit some prefixes, for example to save bandwidth [NNS98]. The MN therefore cannot rely on Prefix Option Information to trigger Mobile IPv6 handoff.

**Asymmetric Reachability of Wireless Environment**

In some wireless environments, it may be possible to receive a periodically multicast RA without being able to send IP packets to the network. In this case, it is insufficient to rely on reception of unsolicited RAs as confirmation of router reachability.

**Random Delay Execution for RS/RA Exchange**

According to RFC 2461 [NNS98], it is necessary for a host to wait a random amount of time to send a RS and for a router to wait a random amount of time to reply with a RA. Therefore, before a MN sends an initial solicitation, it should delay the transmission for a random amount of time between 0 and MAX_RTR_SOLICITATION_DELAY and furthermore, any RA sent in response to a RS must be delayed by a random time between 0 and MAX_RA_DELAY_TIME (default 0.5 seconds).
4.3.2 Delay in NUD Mechanisms

When a MN examines the reachability of the current default router, a certain delay occurs if the current default router is not reachable. A MN sends a solicitation message to the default router and, upon the receipt of a reply it can assume that the router is reachable. If no reply is received to the solicitation message, in order to verify the unreachability of router the MN should allow some time to ensure that the lack of a reply is not due to some other reason (e.g. packet loss, MAC latency, or processing delay). When NUD is used for detection, movement detection takes 3 seconds to recognise that the current router is no longer reachable (as it is based on 3 consecutive misses of Neighbor Solicitations).

4.3.3 Hints

Wireless Link Layer Hints

Unlike wired environments, wireless links have unclear boundaries, as a wireless link is variable in both time and space [CD04]. Moreover reachability on a wireless link is very unstable, which may result in wrong link layer hints for movement detection.

Delay For Receiving Advertisement Interval Expiry Hints

When using the Advertisement Interval Option for a hint, [JPA04] specifies that a MN should implement its own policy to determine the number of missing RAs for a hint to detect that the MN has moved to a new network. Thus the delay associated with this mechanism depends on RA intervals.

Movement detection optimisations seeks to reduce the time necessary to detect layer 3 handoff. To achieve this aim, more movement detection hints should be supported or modifications should be made to Neighbor Discovery or to Mobile IPv6 elements in mobile networks.

4.4 Basic Support for Network Mobility

4.4.1 Overview of Network Mobility

IETF RFC 3963, “Network Mobility (NEMO) Basic Support Protocol” [DWPT05] defines network mobility as mobility of an entire network, which changes its point of attachment to the Internet and thus its reachability in the topology. A Mobile Network includes one or more Mobile Routers which connect that network to the global Internet (Figure 4.6). A Mobile Network can only be accessed via specific default gateway Mobile Routers that manage its movement. The IETF NEMO Working Group [NEMO] is currently working to standardise basic Network Mobility support mechanisms for Mobile IPv4 and Mobile IPv6. Initially, this working group assumes that only Mobile Routers will be aware of the network’s mobility and
Mobile Network’s movement is thus completely transparent to the other nodes inside the Mobile Network.

According to RFC 3963, a Mobile Network will not carry transit traffic. However, it could be multi-homed, either with a single Mobile Router that has multiple attachments to the Internet, or by using multiple Mobile Routers that attach the Mobile Network to the Internet.

A Mobile Network can also be comprised of multiple and nested subnets. Mobile Routers may be attached to Mobile Networks owned by different Mobile Routers and each Mobile Router is attached to another Mobile Network by a single interface. An Access Router without mobility support may be permanently attached to a Mobile Network for local routing.

4.4.2 Mobile IPv6 Network Mobility Basic Support Protocol

NEMO basic support protocol [DWPT05] provides extensions to Mobile IPv6 (Mobile IPv6) to enable support for network mobility. The extensions are backward compatible with Mobile IPv6.

NEMO Basic Support ensures session continuity and reachability for all the nodes (Mobile IPv6 MNs and non-Mobile IPv6-enabled nodes) in the Mobile Network when it moves or even when the Mobile Router changes its point of attachment to the Internet.

The Mobile Router extends the concept of a Mobile IPv6 Mobile Node, by adding routing capability between its point of attachment (Care-of Address) and a subnet that moves with the Mobile Router. Each Mobile Router has a Home Agent. The Mobile Router acquires a Care-of Address from its attachment point in the same way as a Mobile IP Mobile Node. All traffic between the nodes in the Mobile Network and Correspondent Nodes passes through the Home Agent and a bidirectional tunnel between the Mobile Router and Home Agent. This tunnel is set up when the Mobile Router sends a successful Binding Update to its Home Agent, informing the Home Agent of its current CoA (Figure 4.7). The tunnel ensures session continuity while the
Mobile Router moves. Basic NEMO does not describe route optimisation of this traffic. The operation of each Mobile Router is the same when the Mobile Router attaches to another Mobile Router and when the Mobile Router attaches to a fixed Access Router. There could be arbitrary levels of nested mobility, however each level of nesting introduces another IPv6 header encapsulation, which may introduce significant overhead on the data packets [DWPT05].

*Figure 4.7: Mobile IPv6 Network Mobility Basic Support Protocol*
5 A Literature Review on Mobile IPv6 Handoff Techniques ................................................. 55
   5.1 Mobile IPv6 Movement Detection Optimisation Techniques ........................................ 55
      5.1.1 Movement Detection for Mobile IPv6 in general .................................................. 55
      5.1.2 Movement Detection for Mobile IPv6 over 802.11 Networks ............................. 56
   5.2 Fast Handoff and Smooth Handoff Support ................................................................. 57
      5.2.1 Multiple CoA Registrations in Mobile IPv6 .......................................................... 57
      5.2.2 Fast Handoff in Mobile IPv6 ............................................................................... 57
      5.2.3 Simultaneous Binding with Bi/n-casting in Mobile IPv6 ...................................... 62
   5.3 Hierarchical Mobile IPv6 (HMIPv6) ........................................................................... 63
      5.3.1 Routing Scheme for Macro Mobility Handoff in HMIPv6 ................................. 65
      5.3.2 Fast Handoff for HMIPv6 ................................................................................... 66
Chapter 5

A Literature Review on Mobile IPv6 Handoff Techniques

Various approaches have been proposed for basic or extended Mobile IPv6 intended to provide better handoff support (for example fast handoff support, smooth handoff support, seamless handoff, network signaling minimisation). This chapter presents a review of the literature regarding various Mobile IPv6 handoff techniques and proposals to date which seek to minimise disruptions to service quality (Mobile IP handoff issues mentioned in chapter 3 of the thesis) during Mobile IPv6 handoff. This chapter provide readers with literature survey of proposed Mobile IPv6 extension techniques though they are not analysed by experimental and simulator-based work.

5.1 Mobile IPv6 Movement Detection Optimisation Techniques

5.1.1 Movement Detection for Mobile IPv6 in general

Router Solicitation on Advertisement Interval Expiry

Router Advertisements (RA) are multicast from routers randomly at a value between the minimum RA Interval and the maximum RA Interval value specified in the Advertisement Interval Option field of RA. RFC 3775 recommends the lower bound of these values be 30 and 70 ms respectively [JPA04]. In this method of movement detection, a result is achieved when the MN sends a Router Solicitation (RS) to the current router, and after one or more maximum RA Intervals, no RAs have been received with the current prefix. If no RA with current prefix arrives, the MN can detect that it has moved to a new network. As a router delays for a time between 0 and 500ms before sending a response to RS [NNS98], movement detection may take up to 710ms if the response to unsuccessful RS is based on the expiry of 3 maximum RA Intervals.

However, this mechanism has some disadvantages. RFC 2461 [NNS98] specifies an additional delay of 0-1000ms when desynchronising RS messages sent from many hosts. When a MN sends a RS, it typically sends from the link local address (unless this address is tentative)
[JPA04]. Thus sending RS messages from the MN may be delayed by DAD operations for link local address if the MN has not done so. If the MN wants to avoid this delay, RS may be sent with an unspecified address.

**Fast RA**

In Fast Router Advertisement (FastRA) [KKP04] the router unicasts an immediate RA response to a solicitation without delay.

FastRA incorporates a rate-limiting feature aimed at diminishing the potential effect of FastRA traffic on nodes which are already connected to the network. A router may transmit no more than MAX_FAST_RAS advertisements in an interval before discarding solicitations until the next unsolicited multicast RA. MAX_FAST_RAS is the maximum number of RAs returned before multicasting. By default, MAX_FAST_RAS is 10 RAs, but it should be configured based on router capacity and the expected MN solicitation load.

### 5.1.2 Movement Detection for Mobile IPv6 over 802.11 Networks

**RA caching in Link-layer Access Points**

This movement detection method [CS04] makes the Access Point responsible for sending triggered RAs. RAs are sent to the MN when the MN attaches to an Access Point associated with this network. The AP caches RAs that are recently sent from the router, and deliver a frame to the MN when it attaches to the AP. The frame is datalink-unicast to the MN and contains the most recent unsolicited RA.

When the Access Point advises the router of link layer connection, the router can send unsolicited RAs before receiving a RS from the MN. In this case, less frequent transmission of unsolicited multicast RA messages is possible. Deployment requires each AP in the network to be capable of both the caching and triggered sending operations.

**Movement Detection Using Modified ESSIDs**

A method is proposed by Tan [Tan03] to achieve fast movement detection for Mobile IPv6 handoff in IEEE 802.11 networks. In this method, Extended Service Set Identifier (ESSID) of the 802.11 network is modified such that it contains the layer 3 (L3) information of its associated Access Router (AR) or network. The ESSIDs of APs thus include the subnet prefix information of their associated ARs. When the MN receives these ESSIDs on the establishment of its layer 2 (L2) connection, based on the earlier cached configuration of the new subnet in ESSID, MN is able to quickly detect movement to a new subnet, avoiding transmission of unnecessary RSs or Router Advertisements and the associated delay.
RA Filtering

This approach to movement detection [PM03] provides a MN with the capability of filtering RAs to avoid faulty processing of handoff, by providing a RA cache in the handoff module. This enables the MN to determine its best choice of forthcoming link and reduces the MN dependence on the RA period and the RS response time. The MN receives and processes cached RAs immediately after it moves to a new network and thus movement detection time is reduced.

Two important criteria used to determine the priority of the RAs stored in the cache are:
- The link signal strength (signal quality) and the time since the RA entry was last updated.
- The number of hops to the AR on the foreign network and whether or not the AR is link local (priority is given to link local ARs since this may help the MN to detect that it is still reachable via its current IP address).

5.2 Fast Handoff and Smooth Handoff Support

The following methods are related to either fast or smooth handoff.

5.2.1 Multiple CoA Registrations in Mobile IPv6

Multiple CoAs in Mobile IPv6 [JPA04] can be used to support smooth handoff. Higher layers are unaware of multiple CoAs, which are bound to the same MN’s home address, therefore session continuity is guaranteed. The non-primary CoAs ease transition from one link to another. When the default AR becomes unreachable, the MN can use a new default AR for which it already has a CoA. For example, a MN can be reachable through multiple wireless links from physically neighbouring APs. The MN can configure a CoA for each of these APs if they are on different subnets.

When a MN uses multiple CoAs in Mobile IPv6, the MN can utilise all the active interfaces simultaneously. The MN should register all the active CoAs to the HA (because packets are dropped if the MN uses an unregistered CoA).

5.2.2 Fast Handoff in Mobile IPv6

Fast Handoff addresses the following problems: how to allow a MN to send packets as soon as it detects a new link, and how to deliver packets to a MN as soon as its attachment is detected by the new AR.

Terminology: In the Fast Handoff method [Koo04], old Access Router (oAR) is the router to which MN is currently attached, and the new Access Router (nAR) is the router to which MN is about to attach. Old CoA (oCoA) is the current CoA of the MN, and new CoA (nCoA) is the CoA that the MN is about to obtain.
5.2.2.1 Predictive and Reactive Fast Handoff for Mobile IPv6

In the predictive mode of Fast Handoff [Koo04], the MN pre-obtains a new CoA before it moves to a new network. In this mode, the oAR forwards packets to nAR. In the reactive mode of Fast Handoff, after the MN moves to a new network a bidirectional tunnel is established between the oAR and the nAR. The outgoing packets of the MN take the reverse path through the tunnel and at the oAR they are forwarded to the Internet. At some later stage, MN forms and registers a new CoA concurrently with its communications.

The Fast Handoff initiation is based on an indication mechanism that delivers a link layer trigger (L2 trigger) to the oAR and to the MN. The oAR maps the new L2 identifier into the IP address of the nAR. The scheme introduces 7 additional message types for use between ARs and the MN.

1. Router Solicitation for Proxy (RtSolPr)
2. Proxy Router Advertisement (PrRtAdv)
3. Handoff Initiation (HI)
4. Handoff Acknowledgement (HACK)
5. Fast Binding Update (FBU)
6. Fast Binding Acknowledgement (FBack)
7. Fast Neighbour Advertisement (FNA)

The MN handoff procedures are as follows (see Figure 5.1 for the predictive mode and Figure 5.2 for the reactive mode):

**Step 1:** MN sends RtSolPr to oAR. MN receives new AP identifiers through L2 trigger. MN sends RtSolPr to its oAR to resolve one or more AP Identifiers to IP subnet specific information.

**Step 2:** oAR sends PrRtAdv to MN. oAR sends back a PrRtAdv message, which contains one or more Access Point identifiers. The PrRtAdv may also be sent unsolicited.

**Step 3:** MN sends FBU to oAR. With the information provided in the PrRtAdv message, the MN forms a prospective nCoA and sends a FBU message to oAR. FBU proposes nCoA to oAR and authorises oAR to bind oCoA to nCoA.

**Step 4:** oAR sends FBack. There are two cases: oAR sends FBack to MN in the predictive mode of Fast Handoff operations, or to nAR in the reactive mode.

**“Predictive” Mode of Operation**

In this mode, FBack is sent to the MN while it is still at the oAR’s link.
Before sending FBack to the MN, oAR verifies that nAR accepts nCoA through the exchange of HI and HAck messages with nAR. HI carries the prospective nCoA, HAck returns the nCoA from nAR. The nAR can accept the proposed nCoA or assign a different nCoA. The oAR must in turn provide the assigned nCoA in the FBack message to the MN. The MN uses the nCoA returned by FBack. The oAR can forward packets to nAR at the same time it sends FBack to the MN on the oAR’s link. After connecting to nAR, the MN sends FNA immediately to nAR so that it can tunnel buffered packets to the MN.

"Reactive" Mode of Operation

In this mode, the FBack is sent to the nAR rather than to the MN as the MN has moved to the nAR’s link.
At step 3 the MN sends FBU to the oAR. The MN does not receive FBack from the oAR (either because the FBU message to oAR is lost or because the MN has left oAR’s link before receiving the FBack from oAR). The MN (re)sends a FBU to the nAR encapsulated in a FNA message with the proposed nCoA. If the nAR detects that the nCoA is already in use when processing FNA, it must discard the inner FBU packet and send a RA with “Neighbor Advertisement Acknowledge (NAACK)” option to include an alternate CoA for the MN to use. This avoids the undesirable case of address collision.

The MN uses the address in NAACK if it receives a RA with a NAACK option to send another FBU using the new CoA. Otherwise the MN obtains a new CoA using IPv6 Stateless Address Autoconfiguration and sends a FBU to the oAR. The oAR returns FBack and establishes a bidirectional tunnel between itself and the nAR and forwards traffic it receives for the MN to the nAR.

5.2.2.2 Access Router Based Fast Handoff for Mobile IPv6

In this method [HSK04] an AR will perform L3 movement detection, new CoA configuration and DAD on behalf of the MN. This happens after the AR has received L2 information. The AR sends CoA information to MN through RA messages. This is shown in Figure 5.3.

![Figure 5.3: Access Router Based Fast Handoff Sending RA with Generated CoA](image)

After the completion of the MN’s L2 handoff, the AR receives the RS message from the MN and detects the L3 movement of the MN. The AR can quickly detect the L3 movement by comparing information in neighbouring caches and the L2 information for the MN.

The details of this method are as follows.

**Step 1: Movement detection, new CoA configuration and DAD by AR**

When an AR receives a L2 trigger it compares the L2 identifier of the MN (MAC address in case of 802.11 network) with the values in neighbouring caches.

- If the L2 identifier is not found in a neighbouring cache, it means that the MN is a new arrival to the subnet area of the AR. The AR then performs CoA generation and DAD.
The AR generates a CoA for the MN by using its prefix information and L2 information included in the L2 trigger.

- If the L2 identifier is found in a neighbouring cache, it means that the MN is already included in the AR coverage. The AR does not need to perform CoA generation and DAD, and the AR must immediately inform the MN that it can continue using the existing CoA. This situation can occur when the L2 handoff does not require a change of subnet prefix. In this case, L3 handoff does not occur.

**Step 2:** Delivering a new CoA to the MN in the RA from the AR

When a MN requests a CoA through a modified RS message, an AR responds to the MN with a modified RA message which includes the new CoA. The AR waits until the completion of detection movement, CoA configuration and DAD to send the RA message. Then the AR can deliver the newly generated CoA to the MN without needing a solicited message.

For a MN to send a RS requesting a new CoA, the basic RS is modified by the addition of a single flag bit (C) indicating that the MN wants to receive a new CoA. The basic RA message is modified by adding a single flag bit C to indicate that the modified prefix information option includes the new generated CoA for the MN. In this case the prefix field contains a complete IP address as a CoA. If the IP address is the same as the existing CoA of the MN, it indicates that the MN can use its existing CoA even though there has been L2 handoff.

The benefit of this method [HSK04] is that the handoff delay from implementation is decreased independently of the period of RA messages from routers. The disadvantage of this method is the dependency on AR capability and that it explicitly requires L2 handoff triggers.

**5.2.2.3 Cellular Mobile IPv6 Using Low Latency Handoff**

The cellular Mobile IPv6 method [CCL00] uses multicasting to achieve seamless handoff in a cellular system of micro-cell and pico-cell architectures. The multicasting scheme is designed for high-speed movement of MNs among small wireless cells supporting IP data packets.

In this method, a cell is treated as a separate IP subnet. When the MN is moving through small cells at high speed, its current location will often not match the CoA registered at the CN. Packets sent to the MN therefore are lost. Thus a Foreign Home Agent (FHA) is deployed as a router for a group of subnets to forward packets to the MN. The FHA maintains a cache table of CoAs related to the MN in this group of subnets and multicasts packets to these CoAs.

The detailed operations of Cellular Mobile IPv6 are described as follows (see Figure 5.4): The FHA maintains a cache table for any MN entering the group of subnets A, B and C. When a MN enters a subnet (such as subnet A), the FHA will keep a record of the MN in the cache table (MN’s MAC address and CoA). As the MN moves into the overlap region between 2 subnets (such as the overlap region of subnet A and B) or to a new subnet, it acquires a new CoA in the
new subnet (subnet B). After the MN obtains a CoA (using stateless autoconfiguration) the FHA records details of the MN in the cache table. By checking the MAC address, the FHA will know the latest location of the MN (subnet B). Therefore, when packets sent from a CN are delivered to the MN’s old CoA via the FHA, the FHA will instead forward the packets to the new CoA (in subnet B) until Binding Updates for the new subnet are received by the HA and the CN.

The method also involves multicasting. A counter set in the MN will count the handoff frequency over some period of time. If the frequency is more than a predefined maximum update rate, the MN notifies the FHA using the “IGMP Report message” to use multicasting. The multicast group ID is the MAC address of the MN. When packets sent by the CN are delivered to the MN via the FHA, the FHA will multicast the packets to the domains of the MN’s probable migrations, according the CoA in the packet and the CoAs of the MN in the cache table of the FHA. If the radio channels are limited, the FHA will determine the migration path according to the previous behaviour of the MN. As the MN updates its CoA, the MN notifies the FHA to discard its old CoA in the multicasting member list.

This method involves frequent Binding Updates (the MN sends Binding Updates regularly when it changes FHA) and duplicate packets due to multicasting, thus increase the network load.

5.2.3 Simultaneous Binding with Bi/n-casting in Mobile IPv6

The simultaneous binding method minimises packet loss during Mobile IPv6 handoff and hence minimises service disruption due to frequent MN movements.

In the simultaneous binding method traffic redirection is via old CoAs [ES03]. When the MN moves to a new network, the CN continues to send packets to the out-of-date CoA until a Binding Update is received by the CN. Packets thus are normally lost. To minimise packets lost during this time, the MN can request old ARs to forward packets destined to MN to the new AR. This is done by the MN sending Binding Updates to the old ARs containing a mapping of
old CoA to new CoA. The old ARs on the old links intercept packets destined to the MN and forward them to the current location of the MN. When no longer needed, this temporary binding is de-registered.

However, packet loss will occur if forwarding between old AR and new AR is performed too late or too early with respect to the time at which the MN changes its attachment from the old AR to the new AR. A MN may also switch between two ARs frequently (called ping-ponging). Ping-ponging here is defined as rapid back and forth movement between two networks (e.g. due to failure of L2 handoff). For example, ping-ponging can occur if radio conditions for both the old and new APs are about equivalent and less than optimal for establishing a good, low error L2 connection. The “bi/n-casting” mechanism is to maintain simultaneous bindings for the MN with several ARs (the old AR and one or more new ARs) and to multicast packets into these ARs. Thus all the packets intended for the MN are copied to several potential locations.

Multicasting performed by the Home Agent is not considered scalable, whereas bi/n-casting (multicasting performed at the old AR) is scalable. However either multicasting approach introduces extra traffic in order to reduce packet loss for this MN, and is in conflict with objectives such as conserving wireless bandwidth and battery life of mobile devices.

The simultaneous binding method [ES03] extends the Fast Handoff protocol with the simultaneous binding function to decouple L3 handoffs from L2 handoffs and provide smooth handoff. The mechanism instructs the recipients (old ARs) of the Fast Binding Update (FBU) with the simultaneous binding flag to make multiple copies of packets destined to the MN and sends them to multiple MN’s Care-of Addresses. Simultaneous Binding flag is set in the FBU message. This eliminates the need for continuous transmission of Fast Binding Updates or continuous bidirectional tunnel setups in the Fast Mobile IPv6 Handoff.

5.3 Hierarchical Mobile IPv6 (HMIPv6)

A MN may change its point of attachment to different networks so frequently that basic Mobile IP introduces significant network overhead and increased signaling messages. HMIPv6 [SCEB04], an extension of basic Mobile IPv6, provides a localised mobility management protocol (micro-mobility support) for MN. A new conceptual entity is introduced in this scheme known as the Mobility Anchor Point (MAP) to handle mobility management. MAP is a router or a set of routers that maintains a uniquely authoritative administration to a particular domain. MAP connects the domain it serves to the Internet with its publicly routable IP address (Figure 5.5). In HMIPv6 a MN has two types of address - a regional care-of address (RCoA) and an on-link care-of address (LCoA). The RCoA is a global address that specifies a particular domain of the Internet. LCoA is a locally unique address within that domain.
When the MN moves locally between networks within a MAP domain (micro/intra-domain handoff), it changes its LCoA and only needs to register the new LCoA to a MAP on the local link. When the MN moves from one MAP domain to a new MAP domain (macro/inter-domain handoff), it changes both addresses, and therefore needs to register new local LCoA and new RCoA to the new MAP and the new MAP registers global RCoA to the MN’s HA.

When the MN moves locally within a MAP domain, it can choose between basic mode and extended mode. In basic mode, the MAP acts as the local HA to receive packets sent to MN within that domain. The MAP receives all packets destined to the RCoA and tunnels them to the corresponding LCoA of the MN. In extended mode, the RCoA is the MAP’s address. The MAP keeps a binding table with the current LCoA matched with the MN home address. When the MAP receives packets destined to a MN, it detunnels and retunnels them to the LCoA.

This hierarchical scheme can require a bi/n-casting service for its local registration. Bi/n-casting allows the MN to simultaneously register with several ARs. Unlike Mobile IPv6 where bi/n-casting is performed by the HA, which may generate delay in packet delivery, HMIPv6 bi/n-casting is performed by the MAP. When the MAP receives the request it adds a new entry for the MN (an example of simultaneous bindings). The MAP forwards the same traffic to the old and new MN network and packets are only duplicated within the domain.

HMIPv6 offers advantages over Mobile IP for local movement of MNs by localising the signaling. Firstly the signalling traffic in the broader network is reduced in the case of intra-domain handoff. Secondly the processing load on remote routers due to signaling is reduced. Thirdly the handoff delay is reduced by reducing signaling transmission delay.

However HMIPv6 appears to have disadvantages for a network whose MAP domains are small (only include a small number of ARs) and where MN inter-domain handoffs are frequent.
In this case it appears that the signaling traffic to the HA might increase relative to standard MIPv6.

### 5.3.1 Routing Scheme for Macro Mobility Handoff in HMIPv6

This method [Viv03] aims to reduce packet loss during macro handoff. It enables the MN to receive packets during its address registration with the new network (DAD check and binding management). The routing scheme adopts multicast to each MAP so that the MAP can forward packets to the new AR when the MN moves to a new network. The new AR stores these packets and forwards them to the MN during the MN’s registration.

The scheme for handoff is based on the following scenario: the MN has a regional Care-of Address RCoA1 and an on-link Care-of Address LCoA3 (Figure 5.6). When the CN sends packets to the MN, the packets will be sent via MAP1 to the MN’s LCoA3.

![Figure 5.6: A Scenario of Routing Scheme for Macro Mobility Handoff in HMIPv6](image)

When the MN is about to move from the MAP1 domain to the MAP2 domain:

- The MN sends a control message to MAP1 (while MAP1 is still forwarding packets received from the CN to the MN’s LCoA3) requesting it to build a multicast group for the MN.

- MAP1 creates a multicast group for the MN and sends a multicast group join request to all other neighbouring ARs (such as AR2 of MAP1 and AR4 of MAP2). After receiving these multicast group requests, the neighbouring ARs send response messages back to MAP1 to show they are ready to receive multicast packets from MAP1.

- MAP1 encapsulates packets from the CN and tunnels them to the multicast group members (in this case AR2 and AR4). These ARs buffer the packets. The buffer index is the MN’s unique identifier. Finally, these neighboring ARs will forward the packets (if
there is any request from the MN, otherwise the packets will be discarded after a specific time period).

When the MN travels from MAP1 domain to MAP2 domain:
- The MN first acquires a new address from the MAP2 network (RCoA2, LCoA4)
- The MN sends a Binding Update to MAP2 through AR4 and sends a message requesting AR4 to forward a multicast message. AR4 receives the request message, and then will forward the buffered packets to the MN.
- MAP2 receives the Binding Update and checks for DAD while AR4 continuously sends multicast packets to the MN. After the DAD is received, MAP2 sends a Binding Update to the MN’s Home Agent and waits for the Home Agent to send back a Binding Acknowledgement. MAP2 then sends a Binding Acknowledgement to the MN.
- The MN receives the Binding Acknowledgement and sends a Binding Update to the CN via MAP2.
- CN receives the Binding Update, changes the destination address RCoA1 to new RCoA2 and then directs the packets to MN in the new network via MAP2 and AR4.
- When AR4 receives new packets destined to the MN, it stops sending multicast packets from MAP1. MN now receives packets directly from the CN as with Hierarchical Mobile IPv6.

5.3.2 Fast Handoff for HMIPv6

Fast Handoff for Hierarchical Mobile IPv6 (HMIPv6) [PTH03] proposes a combination of HMIPv6 and Fast Handoff for Mobile IPv6.

In HMIPv6, it is preferable for the MAP rather than oAR to handle Fast Handoff, in order to reduce signaling traffic. If ARs within a MAP domain are involved in Fast Handoff, packets would travel over the MAP to oAR link twice (once when packets travel from the CN through the MAP to the oAR and once when the oAR forwards those packets to the nAR via the MAP). The Fast Handoff for HMIPv6 method proposes that the MAP should take care of the Fast Handoff process. Thus the MAP is responsible for forwarding packets to the nAR prior to handoff to eliminate the duplication of packet delivery.

In Fast Handoff for HMIPv6 (as with the Fast Handoff for MIPv6) packets are redirected from the MAP to nAR when the MAP receives a FBU message. However if the MN performs handoff immediately after sending the FBU to that the oAR’s link MAP, all the packets travelling to the oAR from the CN would be lost (until the FBU arrives at the MAP). Also during this period the MN would not receive any redirected packets. This method [PTH03] recommends that, before starting the handoff, the MN waits at the oAR’s link until either the
FBack is received or until connectivity is lost. This should minimise the loss of packets sent to the oAR. Also packets redirected to the nAR are buffered. This method aims to achieve an ideal time for handoff. Assuming that packets experience a similar delay in the path between the MAP and the ARs involved in the handoff, the FBack would indicate that new packets are already waiting or about to arrive at the nAR, and hence this is the ideal time for handoff. Thus the handoff time is minimised.
Chapter 6

Wireless IEEE 802.11 Link Layer Technology

This chapter looks at the detailed functionalities of 802.11b wireless LAN networks as a basis for the subsequent experimental evaluation of link layer handoff component of Mobile IPv6 handoff and Mobile IPv6 handoff itself. The chapter also covers an aspect of link layer trigger though it is not included in experimental work of basic, generic Mobile IPv6 handoff operation.

6.1 Why 802.11?

Mobility is a normal requirement of wireless environments, and a very common short-range wireless technology supporting IP communication is 802.11 - an entire class of standards relating to wireless LANs (WLAN). WLAN technologies only provide link layer roaming, so 802.11 clients can move around different points of attachment within the same network.

802.11 is a wireless version of Ethernet which provides high speed (802.11a/g provides up to 54 Mbps of bandwidth) and short-to-medium range (up to 100 meters) wireless network access. Bluetooth and IrDa (Infrared Data) are alternative wireless LAN network technologies which provide wireless connectivity between devices. Bluetooth supports a low bandwidth (1 Mbps) and short range (up to 10 meters) wireless data exchange and network access. IrDa provides low or medium bandwidth (115Kbps or 4 Mbps dedicated bandwidth) and very short range (up to 1 meter) wireless data exchange and network access. In practice, Bluetooth and IrDa typically provide local wireless connections (such as connecting PDAs or cell phones with PCs) rather than general purpose WLAN networking.

6.2 Introduction to IEEE 802.11 Networks

In 1997, the Institute of Electrical and Electronics Engineers (IEEE) created the first WLAN standard and called it 802.11. However, 802.11 only supported a maximum bandwidth of 2 Mbps. Modified 802.11 versions were therefore developed.

802.11b

IEEE expanded on the original 802.11 standard in July 1999, creating the 802.11b specification. 802.11b allows connections at up to 11Mbps (available bandwidths: 1, 2, 5.5, 11 Mbps).
802.11b uses the same radio signaling frequency (2.4 GHz) as the original 802.11 standard. 802.11b has the lowest cost among the 802.11 family, signal range is excellent and is not easily obstructed. However 802.11b may suffer interference (802.11b can incur interference from microwave ovens, cordless phones, and other appliances using the same 2.4 GHz range).

802.11a

At the same time that it defined 802.11b, IEEE created another extension to the original 802.11 standard called 802.11a. 802.11a supports bandwidth up to 54 Mbps and signals in a 5 GHz range. It provides higher speed, supports more simultaneous users than 802.11b, and avoids signal interference from other devices (by using higher frequencies than radio signaling frequency). 802.11a has the disadvantages of higher cost and a range limited by the propagation properties at 5 GHz (easily obstructed by walls and other objects).

802.11g

In 2002 and 2003, IEEE created a new standard called 802.11g, which attempts to combine the best of both 802.11a and 802.11b. 802.11g supports bandwidth up to 54 Mbps (as for 802.11a), and uses the 2.4 Ghz frequency (as for 802.11b). 802.11g is backwards compatible with 802.11b (802.11g Access Points will work with 802.11b wireless network adapters and vice versa). 802.11g provides the benefits of 802.11a (speed and support for simultaneous users) with superior signal range (not as easily obstructed as 802.11a), and so 802.11g has replaced 802.11a in the market. The limitations of 802.11g are that it costs more than 802.11b and appliances may interfere on the unregulated signal frequency.

6.2.1 Overview of IEEE 802.11 Architecture

6.2.1.1 Architecture Components

IEEE 802.11 is commonly used to provide a bridging service between a regular Ethernet LAN and mobile 802.11 clients, or between two Ethernet LANs. An 802.11 WLAN is based on a cellular architecture where the system is subdivided into cells.

6.2.1.2 IEEE 802.11 Layer Description

IEEE 802.11 itself defines a Medium Access Control layer (MAC) and Physical layer (PHY). The standard currently defines a single MAC which interacts with three PHYs: Frequency Hopping Spread Spectrum in the 2.4 GHz Band, Direct Sequence Spread Spectrum in the 2.4 GHz Band, and InfraRed [IEEE11].

The relationship between the 802.11 physical layer and the 802.2 data link layer is shown in Figure 7.1. The 802.11 MAC performs fragmentation, packet retransmissions, and acknowledgments.
6.2.1.3 The Basic Access Method

When several clients are connected to an Access Point, they must share the same channel. IEEE 802.11 defines two access methods: the basic protocol Distributed Coordination Function (DCF), and Point Coordination Function (PCF). The basic access mechanism of DCF is Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). With the PCF method, a point coordinator determines which transmitting client is given the right to transmit.

**CSMA/CA**

In the CSMA scheme, an 802.11 client desiring to transmit senses the medium. If the medium is busy (i.e. any other 802.11 client is transmitting), the 802.11 client will defer its transmission to a later time. If the medium is sensed free, the 802.11 client begins to transmit. CSMA allows 802.11 clients to transmit with minimum delay (when the medium is not heavily loaded). However, if the medium is heavily loaded, there is always the chance of collisions (when some clients sense the medium free and decide to transmit simultaneously). In the Ethernet case, this collision is recognised by the transmitting clients, which go to a retransmission phase based on an exponential random backoff algorithm (Collision Detection). While Collision Detection mechanisms operate well on a wired LAN, they cannot be used on a WLAN environment for two main reasons. A Collision Detection mechanism would require the implementation of Full Duplex radio, capable of transmitting and receiving at the same time, which is a costly approach. Also in a WLAN environment 802.11 clients cannot always hear each other. 802.11 uses a Collision Avoidance (CA) mechanism together with positive acknowledgement (CSMA/CA). When a destination 802.11 client receives a packet it immediately replies with a positive acknowledgement. When an 802.11 client wants to transmit, it senses the channel to determine whether another 802.11 client is transmitting. If the channel is idle, the transmitting client waits for a Distributed InterFrame Space (DIFS), and begins its transmission. At the receiving end, the destination 802.11 client checks the CRC of the received packet, waits for a Short InterFrame Space (SIFS) and then sends the ACK. ACK is sent to inform the transmitting 802.11 client that the transmission was successful. If the transmitting 802.11 client does not receive the ACK then it will retransmit the packet until it gets ACK or discard it after a given number of retransmissions. If the channel is busy, an 802.11 client defers the transmissions and
performs a backoff algorithm before it resumes channel sensing. The backoff algorithm chooses a random backoff time. The transmitting 802.11 client decreases the backoff time when the channel is idle. When the backoff time reaches zero, the 802.11 client can sense the channel to transmit.

In order to reduce the probability of two 802.11 clients colliding because they cannot hear each other, the standard defines a Virtual Carrier Sense mechanism as an extended approach to CSMA/CA. A transmitting 802.11 client willing to transmit a packet will exchange RTS/CTS frames with the destination to reserve the channel. It transmits a short control packet called RTS (Request To Send), which will include the source, destination, and the duration of the transmission. If the medium is free, the destination 802.11 client will respond with a control packet called CTS (Clear to Send), which will include the same duration information. If a collision occurs, the transmitting 802.11 client performs the backoff algorithm.

Point Coordination Function (PCF)

PCF is used to implement time bounded services (voice or video transmission). PCF uses higher priority access to issue polling requests to the clients for data transmission and control of access to the medium. In order to allow regular 802.11 clients to still access the medium, the Access Point must leave some time for Distributed Access in between the PCF accesses.

6.2.1.4 Modes of Operation

Infrastructure Networks

An infrastructure network contains one or more Access Points, which coordinate communication. In this case, all frames are relayed via the Access Point. For an 802.11 client to join such a network it must first associate with the Access Point. An Access Point and its associated 802.11 clients form a Basic Service Set (BSS) [IEEE11]. The name of each BSS is called the Standard Service Identification (SSID). An 802.11 network may be formed by one or more BSSs, where the Access Points are connected through some kind of backbone (called Distribution System or DS). Multiple BSSs connected by a backbone are collectively referred to as an Extended Service Set (ESS). The 802.11 client can move between the various BSSs in an ESS without losing connectivity, re-associating with a new Access Point if it becomes preferable to the current one.

Figure 7.2 shows a typical 802.11 WLAN, with the components described previously:
Ad-hoc Networks

A wireless ad-hoc network is a set of mobile or semi-mobile devices communicating with each other directly or indirectly with the help of intermediate nodes in the network. An ad-hoc network is also known as an Independent BSS, which is a collection of 802.11 clients which communicate with each other in a peer-to-peer manner. There is no special coordination or infrastructure (no need of an Access Point). Part of the network functionality is performed by the end user clients (like Beacon Generation or synchronisation), and clients send frames directly to the other clients in the IBSS. Ad-hoc networks are used in certain circumstances where the users desire to build up WLAN networks without an infrastructure (for example for file transfer between two notebook users).

6.2.2 Security

Security is one of the major concerns when decisions are made to set up or use WLANs. Two major security issues are: accessing the network resources (using similar WLAN equipment) and capturing the WLAN traffic (by eavesdropping). Access to network resources can be controlled by the use of an authentication mechanism where an 802.11 client shares knowledge of the current key. Eavesdropping was supposed to have been prevented by the use of the WEP algorithm (Wired Equivalent Privacy). WEP uses a Pseudo Random Number Generator (PRNG) initialised by a shared secret key. The WEP algorithm is a simple algorithm based on RSA’s RC4 algorithm. In this algorithm, every frame is sent with an Initialisation Vector, which restarts the PRNG for each frame. This algorithm is also self-synchronising, re-synchronising for each message. However WEP is easy to break. WEP encryption keys are static rather than dynamic, but these should be updated frequently, and the update process is manual and is difficult to keep secure. Other WLAN security solutions, such as Cisco’s Lightweight Extensible Authentication Protocol (LEAP) may be used. Cisco LEAP provides mutual and user based authentication by generating dynamic WEP keys using 802.1X (every 802.1X session timeout forces clients to reassociate to the network, and the new WEP keys are generated).
However Cisco LEAP provides limited interoperability. In most cases, client cards and Access Points must come from the same vendor. The Wireless Fidelity Alliance announced Wi-Fi Protected Access (WPA) in 2002 - a standard security mechanism that eliminates most 802.11 security weaknesses. WPA specifies the Temporal Key Integrity Protocol for distributing dynamic encryption keys and then lets the client use the authentication type of their choice. In WPA these two functions are separated from each other. WPA supports mixed vendor environments.

6.3 IEEE 802.11 Link Layer Handoff

IEEE 802.11 link layer handoff is a change of the Access Point to which a client is connected within the same network. Handoff incurs an interruption of data frame transmission. The duration of this interruption is called handoff time. Link layer handoff is also referred to as the roaming process.

The link layer handoff time is often considered in three phases: detection, search and execution (see Figure 6.3). The detection phase recognises the need for handoff. The search phase covers the search for information needed to perform the handoff (the scanning process). Execution deals with the actual handoff (the authentication and (re)association process) [IEEE11]. An 802.11 client cannot send a layer 3 packet once it has started link layer handoff.

6.3.1 Link Layer Handoff Process

Link layer handoff is the process of moving from one BSS to another without losing connection. When an 802.11 client enters an existing BSS, after idle mode, power up or moving, it needs to synchronise itself with the Access Point using information in the beacon frame of the Access Point.

The Scanning Process

An 802.11 client uses one of two scanning methods: passive scanning where it waits for signaling frames (link layer beacons) – Probe Requests periodically sent by the Access Point, or active scanning where the 802.11 client sends a Probe Request frame to solicit a Probe Response frame from the Access Point. The method should be chosen according to a power consumption/performance tradeoff.

The Authentication Process

Once the 802.11 client has found an appropriate Access Point and decided to join its BSS, it will enter the Authentication Process. This is an interchange of information between the Access Point and the 802.11 client, where each side proves that it knows a given or a null password.
The Association Process

If the 802.11 client is authenticated, the client starts an association process where the Access Point informs the 802.11 client about the transmission parameters in the BSS (e.g., the data rate and the transmission power). Once the association is completed, the client can communicate via the new Access Point.

When coverage areas of different Access Points share a common coverage zone, the 802.11 client can roam between Access Points. The 802.11 client then associates itself with the Access Point which offers the best signal or which has the minimum load.

![Diagram of 802.11 Handoff Operations](image)

**Figure 6.3: 802.11 Handoff Operations**

### 6.3.2 802.11 Link Layer Trigger

The 802.11 link layer (L2) trigger provides predictive information about the likely link layer movement of the client to a new Access Point (general L2 triggering is discussed in section 4.1.4 and its use in Mobile IPv6 movement detection is discussed in sections 4.2.1 and 4.3.1.3). At the network layer the 802.11 L2 trigger will be received by an 802.11 client (in a mobile-initiated handoff) or by the current AR (in a network-initiated handoff).
802.11 L2 triggers may be based on low Received Signal Strength (RSS) between the 802.11 client and its current Access Point (when the signal strength drops below a predefined threshold). The client monitors the RSS of the current Access Point. For successful connection support, the RSS should be above the handoff threshold.

802.11 L2 triggers may be based on the exchange of control frames in the 802.11 roaming process. For example, the first exchange of Probe Request and Probe Response can be considered as a “802.11 L2 handoff start”, indicating that the client may start L2 handoff to a new Access Point. Since the 802.11 client cannot communicate once it starts a 802.11 L2 handoff, these messages can also be considered as link down indications for the current 802.11 link.

The 802.11 L2 trigger should contain the link layer identification of the Access Point and the 802.11 client. This identification can be achieved by a prior exchange between neighbouring ARs (802.11 control frames does not contain this identification information). When an AR receives a 802.11 L2 trigger, it must be able to match the L2 identifications to an IP address.
7 Experimental Evaluation of 802.11b and Mobile IPv6 Handoff Times ............................... 76

7.1 IEEE 802.11b Handoff Experiments ............................................................................. 76

7.1.1 Aim ............................................................................................................................. 76
7.1.2 Methodology ............................................................................................................. 77
7.1.3 Testbed .................................................................................................................... 79
7.1.4 Trials ....................................................................................................................... 80
7.1.5 Results ..................................................................................................................... 81
7.1.6 Discussion of 802.11b Handoff Time Results ......................................................... 85
7.1.7 Related 802.11b Handoff Time Research ............................................................... 86

7.2 Measured Mobile IPv6 Handoff Time over IEEE 802.11b ........................................ 90

7.2.1 Aim ............................................................................................................................. 90
7.2.2 Methodology .......................................................................................................... 90
7.2.3 Testbed .................................................................................................................... 91
7.2.4 Trials ....................................................................................................................... 92
7.2.5 Results ..................................................................................................................... 93
7.2.6 KAME Components and Their Impact on Handoff Delay ..................................... 101
7.2.7 Related Mobile IPv6 Handoff Time Research ....................................................... 105
7.2.8 Discussions of the Experimental Mobile IPv6 Handoff Time Results ........ 107
Chapter 7

Experimental Evaluation of 802.11b and Mobile IPv6 Handoff Times

In order to characterise and quantify the handoff latencies of Mobile IPv6, experiments are set up to measure how long an end-to-end path is disrupted due to Mobile IPv6 and to analyse the characteristics of standard Mobile IPv6 independent of the link layer technology. This chapter details experimental work to determine Mobile IPv6 handoff time for a generic, public implementation of Mobile IPv6 over 802.11b (the KAME implementation for FreeBSD). KAME is chosen since it is generic and open-sourced and since no research literature has been found using KAME. The results of these experiments are also compared with some experimental results reported in the literature. The experiments are divided into two parts – one to evaluate the 802.11b handoff without Mobile IPv6, the other to evaluate handoff with Mobile IPv6 alone (without the 802.11b component). In order to isolate the Mobile IPv6 and 802.11b components of handoff, the 802.11b handoff time is first measured, then the total time of Mobile MIPv6 and 802.11b handoff is measured and the Mobile IPv6 handoff time itself is estimated by subtracting 802.11b handoff time.

7.1 IEEE 802.11b Handoff Experiments

7.1.1 Aim

An obvious choice for early Mobile IPv6 deployment would be to link disjoint 802.11b wireless LANs (i.e. where each 802.11b network is independent, and not linked at the Ethernet layer to form an ESS). With the IEEE 802.11b wireless network specification having obtained significant acceptance, this is chosen here as the underlying layer 2 protocol over which to run Mobile IPv6.

The aim of this first experiment is to evaluate the 802.11b handoff time independent of Mobile IPv6, as a basis for later evaluating the component of handoff time that occurs at the IP layer. In this experiment the total 802.11b handoff time is determined at IP layer.
7.1.2 Methodology

Link layer handoff in an IEEE 802.11b wireless network occurs when an 802.11b client changes its point of connection to the network, usually characterised by a move from one Access Point to another.

In order to experimentally quantify the average handoff time of an 802.11b network, a simple IP network is built, consisting of two independent 802.11b LANs linked by a regular Ethernet backbone. A single 802.11b client is forced to move back and forth between the two Access Points, creating the kind of link layer move that would trigger a Mobile IPv6 handoff event. Specific details of setting up 802.11b Clients using FreeBSD can be found in Appendix D.

There are three methods for triggering a switch between Access Points:

1. Decrease the transmission power of the Access Point with which the 802.11b client is currently associated. The 802.11b client will detect the degraded signal strength and switch to another Access Point, provided they both have the same SSID.

2. Ensure that the Access Points have different SSIDs, and change the BSS configured into the client when an Access Point switch is required. This latter method forces handoff using network control.

3. Administratively shut down the 802.11b radio interface or physically turn off the Access Point with which the 802.11b client is currently associated.

Methods 1 and 2 are used here, since method 3 would include a long time for the radio to shut down and boot up again (using visual observation, this time is found to be greater than 10 seconds for the Netgear 802.11b client used in this experiment). Method 1 and method 2 would not include these spurious factors in the measured handoff time.

Method 2 is believed to be a more consistent method of triggering handoff since handoff is forced by changing the SSID at the 802.11b client (not by recognition of power level difference). Methods 1 and 2 are used so that they can be compared for consistency. During this experiment, method 2 is found to give lower 802.11b handoff times, therefore it is used to trigger handoff in the later Mobile IPv6 experiment. Method 2 is further subdivided into method 2a (where both Access Points use the same channel) and method 2b (where the two Access Points use different channels). This subdivision is intended to explore the difference between these two conditions (when triggering handoff using method 2) for later setup of the Mobile IPv6 experiment.

In order to ‘see’ the effect of switching Access Point and measure the handoff time, IPv4 ping (ICMP) packets are generated at 10 ms intervals across the 802.11b wireless link. The 10 ms interval is intended to be short enough for reasonable accuracy and long enough to avoid
generating heavy traffic. To capture the timestamps in the ICMP echo request and reply packets, tcpdump [TCPD] (a packet sniffing tool) is used in the host communicating with the 802.11b client. tcpdump captures packets at layer 3. Handoff time is measured by finding the time difference between the last ping packet (i.e. the last ICMP echo reply packet) received by the 802.11b client before dis-associating with one Access Point and the first ping packet (i.e. the first ICMP echo request packet) received by the 802.11b client after re-associating with the other Access Point.

A script triggers handoff events of the 802.11b Client every five seconds (using method 1 and method 2). The five second period is chosen to be clearly longer than the maximum handoff time (which is found to be approximately 1 second). The number of handoff events is initially set to 100 samples.

Given the 10 ms sampling interval, the handoff estimates should have an accuracy of approximately +0/-20 ms (see Figure 7.1 and explanation below). Since the handoff time is found to be 500 ms or more, this accuracy is better than 5%.

![Figure 7.1: Time Diagram of Ping Packets between the IPv4 Host and the 802.11b Client](image)

If the last successful ping before handoff is timestamped with a value \( t_1 \) and the first successful ping after handoff is timestamped \( t_2 \) then the maximum possible handoff time is \( t_2 - t_1 \). The minimum possible handoff time is \( (t_2 - 10 \text{ ms}) - (t_1 + 10 \text{ ms}) \) (10 ms is the ping period). Thus the estimated handoff time will be \( t_2 - t_1 \), with an accuracy of +0/-20 ms and some allowance for uncertainties ping time accuracy.

The beacon interval of the Access Point is set to the default of 100 ms, because the intent is to keep the 802.11b network fairly standard. FreeBSD “wicontrol” is used to control the various settings for the Netgear wireless card, such as frequency, SSID and BSS mode.
7.1.3 Testbed

A simple bridged Ethernet network is established (Figure 7.2), with one fixed host (the sniffer box) communicating with a wireless 802.11b client that has a choice of two Access Points at any given time. Both Access Points are connected directly to a common Ethernet hub and located physically close together so that coverage areas overlap. The sniffer box is used as the IPv4 host sending ping packets to the 802.11b Client while running tcpdump to capture the timestamps of those packets.

In the testbed a VIA Mini-ITX based system running FreeBSD 4.9 [FREB] is used as the wireless client, with a Netgear MA401 PCMCIA 802.11b wireless network card [NETG] talking to two Cisco Aironet 1200 series Access Points [CISC]. Equipment details are as follows:

- A VIA Mini-ITX based system is used as the 802.11b client, which consists of:
  - VIA EPIA-V series Mini-ITX motherboard [EPIA]
  - VIA 800MHz C3 E-Series processor
  - On board 10/100 NIC (VT6102 chipset)
  - 256 Meg PC133 RAM

- The 802.11b client has a NetGear [NETG] PCMCIA wireless network card in a PCI to PCMCIA cradle inserted into its free PCI port.

- Two Cisco Aironet 1200 series wireless Access Points running Cisco IOS version 12.2(11) JA

- An Addtron Ethernet Hub 10Base-T 16 port

- A 2.64 GHz ASUS, generic Intel motherboard based host running FreeBSD 4.9 is used to sniff traffic on the wired LAN side (the ‘sniffer box’).
7.1.4 Trials

7.1.4.1 Method 1: Triggering Handoff by Configuring Access Point Power Levels

For the first method of triggering handoffs both Access Points are configured to be in the same BSS (“magicap”) and on the same channel (channel 10). By using the same BSS for both Access Points, when one Access Point is set to a low power level and the other is set to a high power level, the 802.11b client dis-associates from the low power Access Point and re-associates with the high power Access Point. The transmission power of Access Point 1 is configured to the minimum (1mW) and Access Point 2 is configured to the maximum (100mW) for the duration of the trials.

The wireless client has an adjustable sensitivity (or Access Point density) which determines the threshold at which a handoff is triggered when the signal quality of the currently associated Access Point drops. This is used in method 1 to force handoff from the low power Access Point to the high power Access Point. The Access Point density can be adjusted to be either 1 (low), or 2 (medium) or 3 (high) using the wicontrol command’s “access_point_density” option, which specifies the access point density for a given interface. When the density is 1, the 802.11b client associates with the lower power Access Point (Access Point 1). When the density is changed to 3 the 802.11b client re-associates with the highest power Access Point (Access Point 2). The handoff process is enumerated in the following list:

- Notice current Access Point gone
- Find new Access Point
- Scanning process: Receive Access Point beacons at 100 ms interval
- Authentication Process: Authentication Request and Authentication Reply
- Re-association Process: Re-association Request and Re-association Reply

A script toggles the Access Point density between 1 and 3 every five seconds, triggering handoff events of the 802.11b Client.

7.1.4.2 Method 2: Triggering Handoff by Alternating the Mobile Node’s SSID Association

The second method involves both Access Points being configured to transmit with equal signal power, but with different SSIDs - “magicap1” and “magicap2” for Access Point 1 and Access Point 2 respectively. Handoffs are triggered by switching the client’s wireless interface SSID (using FreeBSD’s “ifconfig” or “wicontrol” command).
In method 2, the set of trials is divided into 2 subsets to allow a comparison of results. In the first subset of trials (method 2a), the SSID association of the 802.11b client is alternated every 5 seconds with both Access Points set to the same channel (Channel 10 is used for this trial).

In the second subset of trials (method 2b) Access Point 1 and Access Point 2 are set to different channels (but all other parameters are as for method 2a). Access Point 1 uses channel 4 and Access Point 2 uses channel 10.

The handoff process is enumerated in the following list:

- Notice current Access Point gone
- Find new Access Point
- Scanning process: Probe Request and Probe Response
- Authentication Process: Authentication Request and Authentication Reply
- Re-association Process: Re-association Request and Re-association Reply

A script toggles the Mobile Node’s SSID association every five seconds, triggering handoff events of the 802.11b Client.

7.1.5 Results

7.1.5.1 Method 1 Triggering Handoff by Varying Access Point Power Levels

100 handoff samples using method 1 yield a mean handoff time of 951 ms. This mean includes a small fraction (5%) of handoffs longer than 2 sec (up to 2.706 sec). Occurrence of these long handoff times may be explained that during this time the running wireless card enter the radio power-saving mode (sleep state) thus lengthen the time it reconnects with other Access Point. These longer handoff times are deleted from the final estimates below because they are assumed to be “anomalies” that are not truly representative of the 802.11b standard handoff mechanism.

Excluding the data points greater than 2 sec, the mean 802.11b handoff time is 864 ms, with values ranging from 682 ms to 946 ms (Figure 7.3), a variance of 5 ms, and a standard deviation of 70 ms. Individual data points have an estimated accuracy of +0/-20 ms.
Figure 7.3: Handoff Times, Varying Access Point Power Levels, Method 1: Access Points on Same Channel (Excludes “Anomaly” Points)

7.1.5.2 Method 2: Triggering Handoff by Alternating Mobile Node’s SSID Association

Method 2a: Both Access Points on the Same Channel

Figure 7.4 shows a scatter plot of 800 handoff samples using method 2a with both Access Points on the same channel. In this trial a larger sample is taken to see what effect it has (for example its effect on the long handoff time “anomalies”). Again the samples give a small fraction of handoff times over 2 sec, but now they represent a smaller fraction (1.37%) of samples. They are again assumed to be “anomalies” that are not truly representative of the 802.11b standard handoff mechanism, because Figure 7.4 shows how rare they are, and how the other points are well clustered. Excluding the small faction (1.37%) of samples over 2 sec, the handoff time has a range from 506 ms to 781 ms, mean of 649 ms, with a variance of 2 ms and a standard deviation of 47 ms. Figure 7.5 shows a cumulative distribution of handoff times between 506 ms and 781 ms excluding the samples over 2 sec. The estimated accuracy of individual data points is +0/-20 ms, as for method 1.
Method 2b: Access Points on Different Channels

Method 2b is like method 2a, but with Access Points on different channels (Access Point 1 on channel 4 and Access Point 2 on channel 10). Figure 7.6 shows a scatter plot of 800 handoff samples using method 2b. Excluding the small faction (1.37%) of samples over 2 sec, the handoff time has a range from 508 ms to 825 ms, mean of 687 ms, with a variance of 3 ms and a standard deviation of 57 ms. Figure 7.7 shows a cumulative distribution of handoff times.
between 508 ms and 825 ms excluding the samples over 2 sec. The estimated accuracy of individual data points is +0/-20 ms as for method 1. The general form of both the scatter plot and the cumulative plot is broadly similar to those of method 2a.

**Figure 7.6: Handoff Times, Alternating SSIDs, Method 2b: Access Points on Different Channels (Scatter Plot)**

**Figure 7.7: Handoff Times, Alternating SSIDs, Method 2b: Access Points on different Channels (Cumulative Plot, Excludes “Anomaly” Points)**
7.1.5.3 A Summary of 802.11b Handoff Time Results

Table 7.1 summarises the results from the three trials: method 1 and method 2a and 2b (after excluding data points over 2 seconds).

<table>
<thead>
<tr>
<th></th>
<th>Method 1 Varying Access Point Power levels</th>
<th>Method 2 Alternating SSID handoff</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Method 2a: Both Access Points on Ch. 10</td>
<td>Method 2b: Access Points on Ch. 10 and Ch. 10 respectively</td>
</tr>
<tr>
<td>Shortest handoff</td>
<td>682 ms</td>
<td>506 ms</td>
</tr>
<tr>
<td>Longest handoff</td>
<td>946 ms</td>
<td>781 ms</td>
</tr>
<tr>
<td>Mean 802.11b handoff</td>
<td>864 ms</td>
<td>649 ms</td>
</tr>
<tr>
<td>Variance</td>
<td>5 ms</td>
<td>2 ms</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>70 ms</td>
<td>47 ms</td>
</tr>
</tbody>
</table>

Table 7.1: Summary of 802.11b Handoff Times

7.1.6 Discussion of 802.11b Handoff Time Results

Theoretically, the link layer handoff process comprises three sequential phases: detection, search and execution [IEEE11]. The detection phase refers to the realisation that a handoff operation is required. The search phase refers to the acquisition of information needed to perform the handoff. The execution phase refers to the act of carrying out the handoff procedure.

In the search phase, the 802.11b client must scan all the different radio channels to be reassociated with an Access Point.

Method 1 includes with all 3 phases of handoff - detection, search and execution. The 802.11b client must detect its movement, search for available channels to associate with, and execute the handoff.

Method 2 eliminates the detection phase, involves only search and execution, and so is expected to take less time than method 1.

Method 2a uses the same channel so that search time is minimised.

Method 2b uses different channels to make the search phase more difficult and is intended to show how much this contributes to the handoff time.

Method 1 uses the same channel as method 2a, however the 802.11b client in method 1 must detect that it needs a new Access Point before starting the search for a new Access Point. This is
consistent with the lower handoff times observed for method 2a relative to method 1 (the mean time is reduced by 215 ms (from 864 ms for method 1 to 649 ms for method 2a). Handoff triggering method therefore has an impact on 802.11b handoff times.

Method 2a and 2b are intended to indicate the difference of handoff times on two different conditions for later Mobile IPv6 experiments. The difference in measured handoff times between method 2a (Access Points on same channel) and method 2b (Access Points on different channels) ranges from 2 ms to 46 ms (see Table 7.1). The difference in mean handoff times is 34 ms. The Netgear wireless card uses random search (the “ifconfig” command can be used to observe the wireless interface behaviour) and does not appear to search if both Access Points are allocated the same channel. This is consistent with the test results – the shortest handoff times are almost identical and the longest handoff times are significantly different, suggesting that the random search may range in time from about 0 ms to 46 ms.

7.1.7 Related 802.11b Handoff Time Research

This section outlines several pieces of reported research on measurements of layer 2 handoff times in 802.11b networks. These report a range of handoff time measurements and discuss factors that influence handoff time. My primary goal is to account for the contribution of 802.11b handoff to my later Mobile IPv6 handoff measurements, so in my experimental work I do not explore 802.11b-specific methods of improving link layer handoff times beyond the default settings of the commercial equipment.

Montavont and Noel Study (2003) [MN03]

This study gives measured 802.11b handoff times for different traffic loads and network conditions in the 802.11b wireless network. They report that the handoff time in different network conditions is strongly affected by traffic load. The handoff times range from approximately 0.14 sec with no background traffic to 8.73 sec at the maximum load tested. Only broad comparisons are valid between this and my measurements because my measurements use a live testbed and commercial equipment rather than simulation, and there is little detail given regarding their measurement techniques. My study does not include varying workload, because its main purpose is to provide a basis for inferring the IP component of Mobile IPv6 handoff time.

This study simulates a 802.11b client in a IEEE 802.11b network. 802.11b clients travel between Access Points of two different networks - a home network and a visiting network. Layer 2 handoff measured in the study is the time taken to establish the new connection after disconnecting from the old Access Point. Further details of this measurement method and of the simulation setup are not reported in the study. The test involves two different cases - the “optimised” case where a 802.11b client is the only user of the 802.11b Access Point and the
“realistic” case where there are multiple users in the 802.11b network (the 802.11b client under test and another 5 static active users which are also “802.11b clients”). The raw bandwidth of the 802.11b network is also varied (1, 2, 5.5, or 11 Mbits/sec). The 802.11b client can either be an idle node (which does not send or receive data), or active node (which transmits data during 802.11b handoff) for each “optimised” or “realistic” case. An active node can either be low speed or high speed.

In the “optimised” case, layer 2 handoff time is almost constant. For all raw bandwidths, layer 2 handoff time has a mean of 0.158 sec and is bounded between 0.144 and 0.177 sec. Handoff time is usually shorter for an idle node than for an active node for all the raw bandwidths (except for a bandwidth of 1 Mbit/sec, when it is longer). Handoff time is shorter for a low speed transmitting node than for a high speed transmitting node for all the raw bandwidths. Handoff time is longer for low network bandwidth than for high network bandwidth.

In the “realistic” case, for all raw bandwidths, layer 2 handoff time has a mean of 5.511 sec and is bounded between 1.490 sec and 8.729 sec. Handoff time is shorter for an idle node than for an active node for all raw bandwidths. Contrary to the “optimised” case, the handoff time is longer for a low speed transmitting node than for high speed transmitting node for all the raw bandwidths, and higher raw bandwidths result in longer handoff times.

This study discusses the causes of long handoff at heavy traffic loads. In the “realistic” case, competition between data frames and control frames that share the same single channel cause frequent collisions and longer channel access time, thus the 802.11b client needs more time to synchronise with the target Access Point. In this case, at faster bandwidths, the error rate of exchanged 802.11b frames is higher, thus higher bandwidths result in longer handoff times. The study also finds that if the beacon interval of the Access Point is more frequent, the node will detect the new Access Point earlier (but they do not give any further detail, except that the measured results are for the default beacon interval of 100 ms). However they also observe that beacons sent out to 802.11b segments add to the load and this can delay the authentication and association procedures for 802.11b handoff.

T. Cornall, B. Pentland and K. Pang Study (2002) [CPP02]

In order to measure 802.11b handoff times the authors’ testbed consists of a fixed network device connected through an Ethernet switch and one of two 802.11b Access Points to a wireless network device, in a BSS carrying no other traffic. Layer two handoffs are manually initiated by forcing the WLAN card to change its association to a different Access Point (by changing the layer 2 address using the Wireless Extension tools for Linux). The test is carried out by using ping6 (generating ICMPv6 packets) to send independent packets every 10 ms from
the CN to the 802.11b client. The number of lost packets gives an indication of the time required for the network connection to re-establish through the new Access Point. Their results indicate that the period where loss is caused by a layer two handoff lasts consistently for an average 140 ms with a standard deviation of 23 ms.

Although the 802.11b handoff measurement methods used in this study seem to be similar to my study, the handoff times they report are substantially shorter than mine. However they give no details of their equipment (and the study reported below suggests that the handoff time is strongly affected by equipment type).

A. Mishra, M. Shin and W. Arbaugh Study (2002) [MSA02]

This study measures 802.11b handoff time using an experimental approach. To measure 802.11b handoff time, a sniffing system is used to capture 802.11b control frames (probe, authentication, and re-association frames). The study finds mean handoff times fall within the bounds between 58 and 398 ms (depending on the equipment) with standard deviations from 17 to 91 ms. They also find that the handoff search phase is the most significant contributor to the handoff time and that 802.11b handoff times depend quite strongly on the types of 802.11b equipment used. Approximate handoff time ranges from a minimum of 90 ms to a maximum of 210 ms for a Lucent 802.11b Client with Lucent Access Points; ranges from a minimum of 270 ms to a maximum of 410 ms for a Cisco 802.11b Client with Lucent Access Points; ranges from a minimum of 35 ms to a maximum of 110 ms for a Lucent 802.11b Client with Cisco Access Points; ranges from a minimum of 260 ms to a maximum of 550 ms for a ZoomAir 802.11b Client with Lucent Access Points; and ranges from a minimum of 350 ms to a maximum of 430 ms for a Cisco 802.11b Client with Cisco Access Points.

My 802.11b handoff time results are significantly longer than these. The slowest set of times reported ranges from 260 ms minimum to 550 ms maximum. My handoff times range from 506 ms to 946 ms. My handoff time measurements use different equipment and are based on a sniffing method that captures packets on layer 3 instead of layer 2 control frames, and hence include the entire time it takes for actual Ethernet level bridging to successfully resume. Sniffing at layer 3 is considered appropriate for my purpose of inferring the IP component of Mobile IPv6 handoff time. During my Mobile IPv6 tests (reported in section 7.2) the RA packet timing observed between Access Routers is consistent with my layer 2 handoff time measurements. This confirms the validity of my handoff time results.

Jon-Olov Vatn Study (2003) [Vat03]

This study presents experimental results on the performance of IEEE 802.11b handoff mechanisms. They also report the impact of handoff triggering method on handoff times. When handoff is triggered by decreasing the transmission power of the current Access Point, the
wireless client will begin the handoff before losing connectivity via the current Access Point, whereas when handoff is triggered by removing the Access Point power supply, the time to detect handoff is longer.

The Jon-Olov Vatn study finds that the actual messages sent during 802.11b handoff are not exactly the same as standard handoff procedures specified in [IEEE11]. If the 802.11b client has data packets to send, these upstream packets are buffered in the 802.11b client during the search and execution phases. Upstream data scheduled to be sent during the search phase may be flushed to the old Access Point between the search and execution phases or sent to the new Access Point after the execution phase (post handoff phase). When the 802.11b client is at the receiving end, the Access Point may buffer packets if the 802.11b client tells the Access Point that it is in power saving mode. After the search phase when the 802.11b client could notify the Access Point that it is back in active mode, the Access Point flushes buffered data to the 802.11b client before the 802.11b client enters the execution phase. Handoff times not only depend on the time intervals of different handoff phases but also the direction of the data flow and the pattern of packets that may get through during these phases. The study’s measured handoff time for upstream data is around 82 ms for a Lucent client and 213 ms for a D-Link client. Latency for downstream data is approximately 138 ms for a Lucent client and 263 ms for a D-Link client.

The Jon-Olov Vatn study also captures control frames at the data link layer (like the study of Mishra et al) to measure the handoff time.

In summary, the A. Mishra et al study [MSA02] concludes that the search phase is the most significant contributor to the handoff time. They also find that the type of wireless card firmware can have a large impact. I observe that my NetGear wireless NIC scans for channels in random order, whilst other wireless NICs can search through channels in ascending or descending order. The Jon Olov Vatn study [Vat03] finds that the actual messages sent during 802.11b handoff are sometimes not exactly the same as standard handoff procedures specified in [IEEE11]. Jon Olov Vatn also concludes that handoff times not only depend on time intervals of the different handoff phases but also the direction of the data flow and the pattern of packets that may get through during these phases.

My measured handoff time is generally longer than that reported in other literature. As discussed earlier in this section, my results appear to be valid for the equipment used, and for the later purpose of evaluating the layer 3 component of Mobile IPv6 handoff time.
7.2 Measured Mobile IPv6 Handoff Time over IEEE 802.11b

7.2.1 Aim

The aim of this section is to characterise, evaluate and quantify Mobile IPv6 handoff time using the basic (generic) Mobile IPv6 implementation from the KAME Mobile IPv6 stack (snap kit kame-20040628-freebsd49-snap.tgz) [KAME]. The results obtained from the previous section allow an estimate of the contribution of layer 2 handoff time to the total handoff time when using Mobile IPv6 over the 802.11b network.

7.2.2 Methodology

In order to experimentally quantify the average handoff time of a Mobile IPv6 network, a network including IPv6 and Mobile IPv6 is built consisting of two different IP subnetworks for the MN – one is a home network (with a Home Agent for the MN), the other is a foreign network (with an Access Router). Each network contains one Access Point. To trigger a MN handoff event between the home and foreign network, the method chosen is to alternate the MN SSID association between Access Point (as in Method 2 for measuring layer 2 handoff time). The two Access Points are assigned different channels (as in Method 2b).

A CN that communicates with a MN can be either Mobile IPv6-enabled or non-Mobile IPv6-enabled. The CN that is non-Mobile IPv6-enabled communicates with the MN through a IP tunnel via the Home Agent while the CN that is Mobile IPv6-enabled communicates directly with the MN using Route Optimisation. In this experiment, the Mobile IPv6 handoff time is measured for these two cases (Mobile IPv6-enabled CN and non-Mobile IPv6-enabled CN), to find the difference in handoff times. Later in the experiment, the RA interval is varied in order to evaluate the impact of RA interval on handoff time. The method of varying RA interval is specified in Appendix E (using the KAME Router Advertisement Daemon ‘rtadvd’, which is responsible for sending out RAs).

In order to ‘see’ the effect of switching between the two networks and to measure the handoff time, as in the 802.11b handoff experiment, ping packets are generated at 10 ms intervals while running tcpdump at both nodes - the MN and the CN. IPv6 ping (ICMPv6) packets are generated using the ping6 command in FreeBSD.

The Mobile IPv6 handoff time is determined by measuring the time between the last ICMPv6 echo reply received from the MN on the previous network and the first ICMPv6 echo response received from the MN on the new network. tcpdump captures timestamps and the destination and source of incoming and outgoing traffic. The handoff time is measured using timestamps at the CN for the last ICMPv6 packet before handoff and the first ICMPv6 packet
after handoff. The tcpdump files at the MN are useful to investigate the details of the Mobile IPv6 handoff procedure used by the KAME implementation.

Each trial of this experiment involves 100 handoff samples. A script is used to toggle the MN’s SSID association every 20 sec and generate ICMPv6 packets until 100 handoff samples are recorded. The 20 second period is chosen to be clearly longer than the maximum Mobile IPv6 handoff time (which is found to be approximately 5 sec including link layer handoff).

Given the 10 ms sampling interval, Mobile IPv6 handoff time estimates should have an accuracy of +0/-20 ms (using the analysis of section 7.1.2 - Handoff time is measured by the last ICMP echo reply packet received by sender before disconnection and the first ICMP echo request packet received by sender after reconnection). Since the Mobile IPv6 handoff time is found to be 3.6 sec or more, this accuracy is better than 0.7 %.

7.2.3 Testbed

My Mobile IPv6 testbed is shown in Figure 7.8

---

Equipment details are as follows:
- The Smartbits 2000 is a traffic generator.
- A 2.64 GHz ASUS, generic Intel motherboard based host running FreeBSD 4.9 is used to sniff traffic on the wired LAN side (the ‘sniffer box’).
- Router 1, the Home Agent (Router 2) and the Access Router (Router 3) are VIA Mini-ITX based and have an extra NIC added to their free PCI slot. The NIC uses either an Intel 1000MT (82540em chipset) or an Intel 100/Pro (82559 chipset).

- Two Cisco Aironet 1200 series wireless Access Points running Cisco IOS version 12.2(11) JA are used.

- The 802.11b client has a NetGear [NETG] PCMCIA wireless network card in a PCI to PCMCIA cradle inserted into its free PCI port.

- The switch is an Alloy NS05C 5 port Nway Mini Switch [NMS].

- The hub is an Addtron Ethernet Hub 10Base-T with 16 ports.

- All Network Interface Cards (excluding the wireless adapter) are explicitly configured to run in full duplex mode.

As shown in Figure 7.8, the network of Access Point 1 and Router 2 acts as the MN’s home network. Router 2 acts as the Home Agent. This home network has an IPv6 prefix of 2001:DB8:1:cccc, thus MN has a home address of 2001:DB8:1:cccc:230:abff:fe1c:b. Router 3 attaches to Access Point 2 and acts as the Access Router on the MN’s foreign network. The foreign network has an IPv6 prefix of 2001:DB8:1:dddd. Foreign and home networks are connected via the switch and communicate with other networks via Router 1.

Access Point 1 is configured with SSID “magicap1” on channel 10. Access Point 2 is configured with SSID “magicap2” on channel 4. When the MN moves away from its home network, the MN is associated with Access Point 2. The MN returns to the home network by being reassociated with Access Point 1.

**7.2.4 Trials**

Trial 1 and Trial 2: The first two trials use the sniffer box as a CN. In trial 1, the CN is Mobile IPv6-enabled and in trial 2 it is non-Mobile IPv6-enabled.

Trial 3: This trial is intended for calibration purposes and so uses a traffic generator to act as a CN.

Trial 4: This trial varies the interval of RAs that Home Agent and Access Router send to their networks, to test the effect of RA interval on handoff time.

The details of the four trials are as outlined below:

**7.2.4.1 Trial 1: Handoff with a Mobile IPv6-enabled Correspondent Node**

The sniffer box is used as a CN by having it send IPv6 ping traffic at 10 ms intervals to the MN. The CN and the MN monitor all inbound and outbound traffic with tcpdump. With the sniffer box acting as a Mobile IPv6-enabled node, the MN establishes route optimisation with the node
and thus avoids the overhead of relaying all traffic through the HA. After new CoA establishment and sending BUs to the HA, the MN sends a HoTI message to the HA and a CoTI message to the CN and receives HoT and CoT responses from them. The MN is then able to send BUs to the CN.

During this trial, the RA interval is set to 30-70 ms (the default for current recommendations of RFC 3775 [JPA04], “Mobility Support for IP version 6”).

7.2.4.2 Trial 2: Handoff with a Non-Mobile IPv6-enabled Correspondent Node

As for trial 1, the sniffer box is used as a CN by having it send IPv6 ping traffic at 10 ms intervals to the MN. The CN and the MN monitor all inbound and outbound traffic with tcpdump. When a CN is non-Mobile IPv6-enabled, the procedure of handoff from home network to foreign network is similar, except that the CN will not reply to the MN with a CoT message when the MN sends a CoTI message. The MN does not send BUs to the CN, and a tunnel is established instead for communication between CN and MN via the HA.

As for trial 1, the RA interval is set to the default of 30-70 ms.

7.2.4.3 Trial 3: Mobile IPv6 Handoff Using a Traffic Generator as a CN

For calibration purposes, trial 3 uses a Smartbits 2000 precision traffic generator to transmit ICMPv6 ping packets to the MN at precise 10000 microsecond (10 ms) intervals. This should generate more precise timing of ping packets. The setup is otherwise similar to trial 2. As for trial 2 all inbound and outbound traffic is monitored by running tcpdump at the MN.

7.2.4.4 Trial 4: Varying RA Interval

In this trial, the RA interval is varied in order to evaluate the impact on Mobile IPv6 handoff time (relative to trial 2). RA intervals vary randomly between specified upper and lower bounds. The RA interval bounds are set in three ranges: 100-200 ms, 200-500 ms and 500-800 ms (in addition to the default range of 30-70 ms from trial 2). For each range 100 handoff samples are recorded as for the other trails.

7.2.5 Results

It was found in trial 1 (when the CN is Mobile IPv6-enabled) that the KAME Mobile IPv6 implementation has two bugs, which add three seconds to the handoff time – one in each direction (as the MN moves to the foreign network and as it returns to the home network). One is a bug in the handling of Binding Updates (which affects the movement of the MN away from home). The other is a bug in the processing of RR for Route Optimisation (which affects the movement of the MN towards home).
When the CN is non-Mobile IPv6-enabled, movement of the MN away from home triggers a similar KAME bug. This third bug adds three seconds to the measured handoff time when the MN moves to the foreign network.

These bugs are considered in detail in section 7.2.6.

For the remaining analysis, the three second bug times are subtracted from the raw measured data to generate graphs and figures, because the objective is to measure the handoff time that is intrinsic to standard Mobile IPv6 itself rather than the peculiarities of a particular KAME stack implementation.

7.2.5.1 Trial 1: Handoff with a Mobile IPv6-enabled Correspondent Node

Figure 7.9 shows a scatter plot of measured Mobile IPv6 handoff times as a series (trial 1), after correcting for the KAME bugs. The scatter plot shows that a small number (3%) of handoffs from the home network to the foreign network take an unusually long time (greater than 6 sec). It is assumed here that these are “anomalies” (using the same argument as in the 802.11b handoff time experiments). These longer handoff times are deleted from the final estimates below. The scatter plot also shows that the handoff time from home to foreign network usually is somewhat longer than the handoff time from foreign to home network.

Figure 7.10 shows a cumulative plot of handoff times excluding bug times and “anomaly” data points. This plot clearly shows the difference in handoff times for each direction. Again excluding bug times and “anomaly” results, the mean handoff time from home to foreign...
The mean handoff time from home to foreign network is 4.708 sec (with a variance of 26 ms and a standard deviation of 160 ms); whereas the mean handoff time from foreign back to home network is 3.774 sec (with a variance of 151 ms and a standard deviation of 389 ms).

*Figure 7.10: Cumulative Distribution of Handoff Times with CN as a Mobile IPv6-Enabled Node, Excludes Bug Times and “Anomaly” Points (Trial 1)*

7.2.5.2 Trial 2: Handoff with a Non-Mobile IPv6-Enabled Correspondent Node

Figure 7.11 shows a scatter plot of measured Mobile IPv6 handoff times as a series (trial 2), after correcting for the KAME bugs. As shown in Figure 7.11, a small fraction (2%) of handoffs is longer than 6 sec. As for trial 1, these longer handoff times are deleted from the final estimates below.
Figure 7.11: Handoff Times Excluding Bug Times with CN as a Non-Mobile IPv6-Enabled Node (Trial 2)

Figure 7.12 shows a cumulative plot of handoff times excluding bug times and “anomaly” data points. The mean handoff time from home to foreign network is 4.594 sec (with a variance of 61 ms and a standard deviation of 247 ms); whereas the mean handoff time from foreign back to home network is 3.638 sec (with a variance of 31 ms and a standard deviation of 175 ms).

Figure 7.12: Cumulative Distribution of Handoff Times with CN as a Non-Mobile IPv6-Enabled Node, Excludes Bug Times and “Anomaly” Points (Trial 2)
7.2.5.3 Results from Trials 1 and 2 - CN is a Mobile IPv6-Enabled Node and CN is a Non-Mobile IPv6-Enabled Node

Results of mean handoff times from trials 1 and 2 are summarised in Table 7.2.

<table>
<thead>
<tr>
<th>Mobile IPv6 handoff including link layer handoff</th>
<th>Mean handoff from home to foreign network (in sec)</th>
<th>Mean handoff from foreign back to home network (in sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trial 1: CN is a Mobile IPv6-enabled node (RA interval: 30–70 ms)</td>
<td>4.708 sec standard deviation of 160 ms</td>
<td>3.774 sec standard deviation of 389 ms</td>
</tr>
<tr>
<td>Trial 2: CN is a non-Mobile IPv6-enabled node (RA interval: 30–70 ms)</td>
<td>4.594 sec standard deviation of 289 ms</td>
<td>3.638 sec standard deviation of 175 ms</td>
</tr>
</tbody>
</table>

Table 7.2: Mobile IPv6 Handoff Results Including Link Layer Handoff

Table 7.3 shows results of Mobile IPv6 handoff times at network layer only, after removing the mean 802.11b handoff time established in section 7.1.5 method 2b (687 ms, standard deviation of 57 ms) (using the same alternation of SSID for triggering handoff, and the same scheme of Access Points on different channels).

<table>
<thead>
<tr>
<th>Mobile IPv6 handoff excluding link layer handoff</th>
<th>Mean handoff from home to foreign network (in sec)</th>
<th>Mean handoff from foreign back to home network (in sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trial 1: CN is a Mobile IPv6-enabled node (RA interval: 30–70 ms)</td>
<td>4.021 sec</td>
<td>3.087 sec</td>
</tr>
<tr>
<td>Trial 2: CN is a non-Mobile IPv6-enabled node (RA interval: 30–70 ms)</td>
<td>3.907 sec</td>
<td>2.951 sec</td>
</tr>
</tbody>
</table>

Table 7.3: Component of Handoff due Solely to Mobile IPv6 (Trial 1 and 2)

7.2.5.4 Trial 3: Mobile IPv6 Handoff Using a Traffic Generator as a CN

This trial mirrors that of trial 2 (the CN acting as a non-Mobile IPv6-enabled node), and gives almost identical results (within 3%): 4.70 sec for handoff from home to foreign network (with a variance of 41 ms and a standard deviation of 203 ms), and 3.684 sec from foreign to home network (with a variance of 4 ms and a standard deviation of 66 ms), (including 802.11b handoff and excluding bug times and “anomaly” points). The normal variability of the handoff times (both at layer 2 and 3) appears to be much more significant than any imprecision in the ping times of trials 1 and 2.

Figure 7.13 shows a scatter plot of measured Mobile IPv6 handoff times as a series (trial 3), after correcting for the KAME bugs. As shown in Figure 7.13, a small fraction (2%) of handoffs
is longer than 6 sec. As for trial 2, these longer handoff times are deleted from the final estimates.

![Figure 7.13: Handoff Times Excluding Bug Times with CN as a Traffic Generator (Trial 3)](image)

Figure 7.13 shows a cumulative plot of handoff times (including 802.11b handoff) excluding bug times and “anomaly” data points. This cumulative plot also shows that the handoff time from home to foreign network is usually longer than the handoff time from foreign to home network.
After removing the mean 802.11b handoff time (687 ms as used in section 7.2.5.3), the mean handoff time is 4.013 sec for home to foreign network and 2.997 sec for foreign to home network (summarised in Table 7.4).

Table 7.4: Component of Handoff due Solely to Mobile IPv6 (Trial 3)

<table>
<thead>
<tr>
<th>Mobile IPv6 handoff excluding link layer handoff</th>
<th>Mean handoff from home to foreign network (in sec)</th>
<th>Mean handoff from foreign back to home network (in sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trial 3: CN is a Traffic Generator (RA interval: 30 – 70 ms)</td>
<td>4.013 sec</td>
<td>2.997 sec</td>
</tr>
</tbody>
</table>

7.2.5.5 Trial 4: Varying RA Interval

This trial measures the effect of using different RA intervals. Since trials 1 and 2 show that the handoff time is very similar whether the CN is Mobile IPv6-enabled or non-Mobile IPv6-enabled, the CN in this trial is set as a non-Mobile IPv6-enabled node as trial 2 for convenience. The RA intervals are varied in four different ranges: 30-70 ms, 100-200 ms, 200-500 ms and 500-800 ms.

The results are shown in Figure 7.15, which is a plot of mean handoff time (measured for each RA case) versus median value of the RA range (where the vertical axis is handoff time
measured in seconds and horizontal axis is the median RA value in milliseconds). Results in Figure 7.15 include the 802.11b handoff time and exclude bug times and “anomaly” points.

The RA interval has an impact on the movement detection time during handoff and therefore has an impact on the overall handoff time. Under certain network conditions, a lower RA interval helps the MN to discover new network prefixes faster and therefore to handoff faster. An interesting observation is that low RA intervals have a limited impact on handoff time. Increasing the RA interval from 30-70 ms to 500-800 ms only degrades home to foreign handoff from 4.594 to 5.097 sec (11% penalty), while significantly reducing the amount of RA traffic (overhead) on the local networks.

Figure 7.15: Handoff Time (Including 802.11b Handoff) versus RA Interval

Figures 7.16 and 7.17 show a scatter plot as a series, and a cumulative plot of handoff times in both directions (home to foreign and foreign to home) (including 802.11b handoff). In both figures, the RA interval is in the range 500-800 ms (the lowest rate tested). Comparing these with the trial 2 plots (Figure 7.11 and Figure 7.12) shows a similar form, with somewhat longer times, and somewhat bigger spread of handoff times for this longer RA interval.
7.2.6 KAME Components and Their Impact on Handoff Delay

7.2.6.1 Critiquing The Kame Stack’s Mipv6 Behaviour

The aim of this section is to identify handoff steps implemented by KAME and compare these with the recommended RFC 3775 steps [JPA04]. Using the tcpdump tool at the MN allows the actual MN procedures during handoff to be examined.
An analysis of the tcpdump shows that the Mobile IPv6 implementation of KAME completes its handoff procedures with the following steps:

1. Form new CoAs and start DAD: When the MN attaches to a new foreign link, it starts to receive new RA messages from all the available interfaces of Mobile IPv6 routers. The MN then forms new CoAs using the IPv6 Address Auto-configuration mechanism. DAD is performed by the MN to ensure that MN’s link local addresses are not duplicated (MN sends a Neighbor Solicitation message containing the tentative address).

2. Movement detection: When the MN receives a new RA, it knows that it may have moved to new link. In order to detect that it needs to change its IP address, the MN uses the NUD mechanism [NNS98] to check the reachability of the current default router. NUD sends 3 probe messages and waits for a response. Movement detection and DAD for the MN’s link local address can be performed simultaneously.

3. Select a new CoA and send a BU to the HA: After NUD has completed, the MN detaches all prefixes advertised by unreachable routers. The MN then selects a new CoA from those CoAs that were formed in step 1 and sends a BU to the HA.

4. The HA receives the BU - if a MN moves from a home link to a foreign link and the HA starts to perform DAD for the home address of the MN (in case another node uses it). After DAD has completed, the HA returns a BA message to the MN. The MN receives the BA and the handoff is finished. The MN can then continue its communication session.

5. Route Optimisation (if CN is Mobile IPv6-enabled): The MN starts RR after registering the new CoA with the HA. If RR is successful, the MN sends BUs to the CN for direct communication.

The sequence of events observed for the KAME Mobile IPv6 implementation stack follows the recommendations of Mobile IPv6 RFC 3775 [JPA04].

7.2.6.2 KAME Components and Their Impact on Handoff Delay

This section discusses in detail the components of the KAME handoff and evaluates how much they contribute to the total handoff delay. Components of the handoff delay and the time difference between steps are derived from tcpdump timestamps. This section also gives details of the bugs discovered in KAME’s current Mobile IPv6 implementation (as mentioned in section 7.2.5).
Delay Component when the CN is Non-Mobile IPv6-enabled node (Trial 2)

**MN moves away from home**

*RA, Address Autoconfiguration and DAD for MN’s local link address*

After link layer handoff finishes, the MN receives RAs on its new link at times that depend on the frequency of the RA beacons from the new network’s router. With the default 30-70 ms RA interval the MN will receive its first RA at some random time up to 70 ms after the layer 2 handoff completes. About 0.45 ms after this first RA the MN completes address auto-configuration on the new link. The MN then starts to perform DAD by sending a neighbor solicitation message to see if any host is already using this auto-configured address on the new link.

*NUD detection of reachability of current router*

In order to detect that its current IP address is invalid, the MN sends three neighbor solicitations (NSs) to the old router address using its old link IP address to see if it is still reachable via the old default router. The interval of solicitations is found to be one-second ± 0.55 ms. The first NS of this NUD is sent 0.47 ms after DAD is performed. This suggests that the MN starts DAD and NUD simultaneously. The MN takes three seconds to complete the NUD trials.

*Selection of new CoA*

After NUD completes, the MN selects a new CoA (as it has detected that its default router from the previous link is now unreachable). At this point, the MN is supposed to send a BU to the HA to register its new CoA. Yet no BU to the HA can be seen, although there is a kernel log message that the KAME stack’s BU timer has started. This is where the KAME bug occurs. It takes another three seconds (± 0.55 ms) before the MN can be seen sending a new BU to register its new CoA with the HA.

Some detective work on my testbed, and discussions with KAME developers, revealed the cause. The first BU is in fact sent immediately after the MN completes its NUD process. However, due to a bug in the Neighbor Discovery cache, the BU is sent to the link layer address of the MN’s previous default router. By the time the three-second BU retransmission timer expires the Neighbor Discovery cache has been properly flushed, and the second BU is correctly transmitted and the Mobile IPv6 handoff continues.

*Registration*

After another one second (± 5 ms) the HA sends a BA to the MN’s CoA. During this time the HA performs DAD for the home address of the MN. A tunnel between the HA and the MN’s new default router, and hence a path from CN and MN, is built within 0.05 ms. The MN is then
ready to talk to the CN via the HA. The MN sends a CoTI message to the CN, but the CN will not reply to the MN with a CoT message therefore Route Optimisation does not occur.

The sum of these component times can be compared with the total handoff time measured in trial 2 (home to foreign network). Typical or representative times are chosen for the Mobile IPv6 components to allow a simple comparison with the mean time measured in trial 2.

Sum of representative Mobile IPv6 handoff time components = 687 ms (layer 2 mean handoff time) + 50 ms (median RA delay) + 3000 ms (NUD) + 3000 ms (due to KAME bug) + 1000 ms (HA performs DAD) = 7.737 sec

The total handoff time measured in trial 2 (home to foreign) is 7.594 sec (including link layer handoff and the KAME bug time, but excluding the “anomaly” points). This agrees with the above breakdown within 2%.

**MN returns to the home network**

When the MN returns to its home network, the MN first performs layer 2 handoff and waits for a short random period to receive a new RA. As soon as the MN receives the first RA with MN’s home prefix, it knows that it is in its home network. From the tcpdump data using the KAME implementation, the MN sends a BU to the HA to deregister its CoA immediately. The HA does not perform DAD for the MN’s home address (because the MN is at home). However, the MN does initiate NUD (which lasts for three seconds) to detect that its current router is unreachable.

Sum of representative Mobile IPv6 handoff time components = 687 ms (layer 2 mean handoff time) + 50 ms (median RA delay) + 3000 ms (NUD) = 3.737 sec

The total handoff time measured in trial 2 (foreign to home) is 3.638 sec (including link layer handoff, but excluding the “anomaly” points). This agrees with the above breakdown within 3%.

**Delay Component When the CN is a Mobile IPv6-enabled Node (Trial 1)**

**MN moves away from home**

In this case, the MN’s move processing is identical to the non-Mobile IPv6-enabled case until the HA sends back BA to the MN. The Mobile IPv6 handoff values in this case also include three seconds due to the KAME bug. Because the CN is a Mobile IPv6-enabled node, the Return Routability (RR) procedure for Route Optimisation is performed. RR starts only 0.05 ms after the MN receives the BA. The RR procedure starts with a CoTI message being sent from the MN to the CN and a HoTI message being sent from the MN to the HA. The CN sends back a CoT message to the MN and the HA sends back a HoT message to the MN within 0.05 ms. Less than one millisecond after the RR procedure, MN sends a BU to CN to indicate its current location. Direct communication begins between MN and CN after one more millisecond.
Sum of representative Mobile IPv6 handoff time components = 687 ms (layer 2 mean handoff time) + 50 ms (median RA delay) + 3000 ms (NUD) + 3000 ms (due to KAME bug) + 1000 ms + 2 ms (RR + BU to CN) = 7.739 sec

The total handoff time measured in trial 1 (home to foreign) is 7.708 sec (including link layer handoff and bug times, but excluding the “anomaly” points). This agrees with the above breakdown within 1%.

**MN returns to the home network**

The MN first performs layer 2 handoff and waits for a short random period to receive a new RA. When the MN receives the first RA with MN’s home prefix, the MN sends a BU to the HA to deregister its CoA immediately. Despite being home, the MN still initiates NUD (which lasts for three seconds). As the MN is still in direct communication with CN, it must perform the RR process (HoTI, CoTI, HoT and CoT) message exchange to issue a new BU to the CN. However, a bug in the KAME implementation causes the RR to start simultaneously with the MN’s home registration (the MN sends BU to the HA). This means that RR starts before home registration with DAD is complete, thus the HA does not forward the MN’s first HoTI packet to the CN for authorisation. The CN still sends packets to the MN’s CoA as RR is not completed. A second HoTI, sent after a 6 second retransmission timeout period, does successfully reach the CN. The CN then receives BU from the MN within 0.05 ms. As NUD completes in 3 seconds, handoff time consequently is only delayed a further 3 seconds by the 6 second retransmission timeout. At this point the CN replies with HoT within a 0.05 ms, and a few ms (0.005 ms) later the MN sends the appropriate BU to the CN, and the foreign to home handoff is complete.

Sum of representative Mobile IPv6 handoff time components = 687 ms (layer 2 mean handoff time) + 50 ms (median RA delay) + 6000 ms for HoTI retransmission (3000 ms NUD starts simultaneously) = 6.737 sec

The total handoff time measured in trial 1 (foreign to home) is 6.774 sec (including link layer handoff and bug time, but excluding the “anomaly” points). This agrees with the above breakdown within 1%.

**7.2.7 Related Mobile IPv6 Handoff Time Research**

T. Cornall et al (2002) [CPP02] investigate handoff times using MIPL (Mobile Implementation for Linux), a Helsinki University of Technology Mobile IPv6 for Linux program, over an 802.11b testbed. They do not provide much detail about the testbed setup nor the handoff steps implemented by their version of MIPL. They use link layer trigger mechanisms to reduce Mobile IPv6 handoff times: the MIPL shell script sends IOCTL to layer 3 software immediately after issuing the command for layer 2 handoff. When the RA interval is between 0.5 sec and 1.5 sec, their measured handoff times are 1.1 sec returning to the home network and 1.8 sec moving
to the foreign network. Related work by R. Hsieh et al (2002) [HSSE02], based on a simulation setup for an 802.11b network, gives a mean measured time of 5.487 sec for basic Mobile IPv6 handoff to a foreign network. Details of different Mobile IP implementation methods can be found in Appendix C.

Table 7.5 summarises the handoff time results from the above research in comparison with my study when MN changes its attachment to a new foreign link.

<table>
<thead>
<tr>
<th></th>
<th>My study</th>
<th>R. Hsieh et al [HSSE02]</th>
<th>T. Cornall et al [CPP02]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean handoff time</td>
<td>4.708 sec</td>
<td>5.487 sec</td>
<td>1.8 sec</td>
</tr>
<tr>
<td>Link layer handoff included</td>
<td>Yes, separate procedure</td>
<td>Yes, separate procedure</td>
<td>Yes, facilitates link layer trigger to layer 3 protocol stack</td>
</tr>
<tr>
<td>RA interval</td>
<td>30 to 70 ms</td>
<td>3 to 4 sec</td>
<td>0.5 to 1.5 sec</td>
</tr>
<tr>
<td>Method of detecting that MN is in new network</td>
<td>Using NUD process (3 sec)</td>
<td>Using RA interval (within 3 to 4 sec)</td>
<td>Using link layer trigger and RA interval (within 0.5 to 1.5 sec)</td>
</tr>
<tr>
<td>Detection of Network Attachment (Checking for need to get a new IP address)</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Configuration time (including DAD for local link)</td>
<td>(1 sec for DAD, however this happens in parallel with NUD)</td>
<td>Not mentioned</td>
<td>Not mentioned</td>
</tr>
<tr>
<td>Registration time (DAD for Home Address)</td>
<td>1 sec</td>
<td>Not mentioned</td>
<td>Not mentioned</td>
</tr>
<tr>
<td>Generality of Mobile IPv6 handoff time results</td>
<td>All types of Mobile IPv6 included. Route Optimisation is included</td>
<td>All types of Mobile IPv6 included. Route Optimisation is not mentioned</td>
<td>Non-standard (Link layer trigger required). Route Optimisation is included</td>
</tr>
<tr>
<td>Method of measurement</td>
<td>KAME Implementation (Free BSD) version - kame-20040628-freebsd49-snaptgz</td>
<td>ns simulation model, ns-allinone 2.1b7a</td>
<td>MIPL (Mobile Implementation for Linux)</td>
</tr>
</tbody>
</table>

Table 7.5: Comparison of Mobile IPv6 Handoff Times from Different Studies

D. Le et al (2003) [LGWP03] specify their Mobile IPv6 implementation in great detail for WLAN mobile networks. They also use MIPL, and observe no packet losses for Mobile IPv6 handoff in Mobile IPv6 WLAN networks while the CN is pinging the MN. Instead they see only a transient increase in the RTT from 3.95 ms to 89.23 ms. Examining their published testbed, I suspect that their MN movement only incurs layer 2 handoff since the default router is still
reachable when the MN is attached to the new link. It would appear that their trials do not trigger actual Mobile IPv6 handoff.

N. Montavont et al (2003) [MN03] report Mobile IPv6 latency values for a single MN over an 802.11b network ranging from around 300 ms to 1.7 sec when the RA interval is 50 ms. When RA interval is 1500 ms Mobile IPv6 handoff times are between 1.8 sec and 3 sec. Unfortunately, there is little detail provided regarding the tools used to measure handoff time.

7.2.8 Discussions of the Experimental Mobile IPv6 Handoff Time

Results

When the CN is a non-Mobile IPv6-enabled node (routing is done by tunneling), the mean measured handoff time is 3.907 seconds (home to foreign network, excluding link layer handoff time), while when CN is a Mobile IPv6-enabled node (using Route Optimisation), the mean measured handoff time is 4.021 seconds (home to foreign network, excluding link layer handoff time). The difference between these times (3%) is within the uncertainty limit of the handoff time measurements. The only difference between the Mobile IPv6-enabled and non-Mobile IPv6-enabled cases should be the RR process, which takes only 2 ms, which is well within the uncertainty limit. The routing method used for data delivery in Mobile IPv6 (tunneling or Route Optimisation) therefore does not have a significant impact on Mobile IPv6 handoff times.

My handoff results provide quantitative data on generic Mobile IPv6 handoff times. Adjusting for the 802.11b wireless environment, the handoff results give approximate handoff times for networks with any link layer handoff technologies.

My results are based on a KAME implementation consistent with the current Mobile IPv6 RFC 3775 [JPA04]. The NUD process accounts for most of the Mobile IPv6 handoff time in KAME as it always takes 3 seconds (in accordance with the RFC 2461 [NNS98] defaults). RFC 3775 states that “Due to the temporary packet flow disruption and signaling overhead involved in updating mobility bindings, the Mobile Node should avoid performing a layer 3 handover until it is strictly necessary”. The NUD process helps the MN to detect the need for layer 3 handoff.

Movement detection using NUD helps to check current AR reachability and validity for current CoA. Faster methods of movement detection are available (such as using information in the received RAs or using a link layer trigger e.g. using a 802.11b 100 ms Access Point beacon), but these do not necessarily guarantee the necessity for the MN to handoff. These faster methods may reduce the movement detection time to several hundred milliseconds, and so the IP level disruption due to 802.11b and Mobile IPv6 handoff together would be much reduced. In this case, the DAD delay (of 1 sec) would be dominant. With faster movement detection and an
appropriate DAD optimisation scheme, these handoff times can be reduced to a substantially lower value.

Specific optimisations are possible, both within each layer (link or network) and between each layer (using link layer state changes to expedite network layer awareness of the need for Mobile IPv6 handoff). However, the focus of my work has been to measure the intrinsic limitations of a Mobile IPv6 implementation where there are no such optimisations - indeed, my results provide further justification for tighter coupling between link layer and network layer protocols in order to significantly improve the resulting handoff times.
8 Experimental Analysis of Implications for Networked Applications ......................... 109

8.1 Video Conference Performance over Mobile IPv6 Handoff ................................. 110
  8.1.1 Methodology .................................................................................................. 110
  8.1.2 Testbed ........................................................................................................ 111
  8.1.3 Results ........................................................................................................... 112

8.2 TCP Throughput Performance over Mobile IPv6 Handoff ....................................... 113
  8.2.1 Methodology .................................................................................................. 113
  8.2.2 Testbed ........................................................................................................ 114
  8.2.3 Results ........................................................................................................... 114
  8.2.4 Simulation ...................................................................................................... 115
  8.2.5 Discussion ..................................................................................................... 117
Chapter 8

Experimental Analysis of Implications for Networked Applications

The interruption of IP packet flow during handoff can cause an impact on the performance of higher layer protocols (such as TCP and UDP) and hence on networked applications that depend on these.

Applications using UDP will often be sensitive to packet loss, and so will suffer due to packet loss during handoff. For example a real-time application may use UDP because retransmission of lost packets does not meet the real-time constraints. These applications will have limited tolerance to bursts of packet loss and will suffer disruption during handoff. Applications using TCP will sometimes be concerned with maintaining high throughput (where TCP flow control and error recovery are considered useful) and sometimes concerned with connection-oriented support. These applications will be affected by handoff because TCP suffers both delay and loss of throughput during Mobile IP handoff.

The reaction of TCP to Mobile IP handoff is discussed in section 3.3.1. Packet loss during Mobile IP handoff will not only trigger retransmission, but will also be interpreted as congestion, causing the TCP flow control to back off and reduce throughput. Also packet delay beyond a certain limit may trigger the same kind of response as packet loss. This response by TCP can strongly affect throughput and can aggravate delay.

The aim in this chapter is to experimentally assess the performance impact of Mobile IPv6 handoff on two types of TCP applications – a web camera conference and a bulk file transfer. UDP applications are not assessed because UDP itself reacts in a manner that is straightforward to understand, and so the impact of Mobile IP handoff is considered relatively obvious. The experimental approach is to use earlier measured handoff times from the previous chapter. Applications are run on two hosts while traffic between these two hosts is interrupted, timing the interrupted period to mimic the measured Mobile IPv6 handoff times. One host acts in the role of a CN and the other host in the role of the MN in motion, and traffic interruption is used
to mimic handoffs. The effect on applications running on the hosts is then observed. Details of 
the methods and results of the evaluation are explained in the rest of the chapter.

8.1 Video Conference Performance over Mobile IPv6 Handoff

This experiment evaluates the impact of Mobile IPv6 handoff on a common web camera 
application where one of the parties represents a Mobile Node in motion.

8.1.1 Methodology

A video conference session using a web camera (web camera conference) is run over a LAN 
with Yahoo Messenger (chosen because it is readily accessible and well known). Yahoo 
Messenger supports live multiple web camera conferences. It uses TCP with a typical frame rate 
of less than 1 fps (frames per second) in “standard” mode and up to 20 fps in “super” mode. 
Super mode requires broadband connections. These figures are quoted from Yahoo statistics for 
the performance of the web camera at the broadcasting end. There is a trade-off between image 
quality and frame rate.

Two computers running the web camera software are connected via a FreeBSD bridge 
machine (see Figure 8.1). One computer broadcasts the video and the other receives it. The 
bridge repeatedly blocks and unblocks IP communication between its two network interfaces, 
timing the blocked state to mimic the Mobile IPv6 handoff times measured earlier (Mobile IPv6 
handoff time including link layer handoff). Details of FreeBSD bridge setup are described in the 
subsection 8.1.2.1.

The bridge interrupts traffic once every 20 seconds. The interval of 20 seconds is found to 
be sufficient to observe the worst case disruption time of video flow for a single handoff 
sample.

The bridge first simulates 30 handoff events where each interruption is 4.7 sec (to mimic the 
mean handoff time from home to foreign network). It then simulates another 30 handoff events 
of 3.8 sec (to mimic handoff time from foreign to home network). These times include link 
layer handoff. At the receiving end, visual observation is used to identify the length of 
disruption to the smooth flow of video, using timestamps displayed in the web camera image. 
During handoff the image freezes, and the timestamp value is noted. After handoff the 
timestamp of the next frame indicates the disruption time (within the 1 sec resolution of the 
timestamp steps and the discrete frame times of $1/0.77 = 1.3$ sec).
8.1.2 Testbed

A two-party web camera conference between two Windows 2000 hosts is run through a FreeBSD 4.10-based bridge using Yahoo!Messenger version 6.0.0.1643 (as shown in Figure 8.1). The video source is a Logitech QuickCam Express web camera which is able to generate up to 30 fps with a video resolution of up to 640x480 pixels.

In order to test the direct effect of flow interruption due to handoff, a very simple web camera conference is set up in only one direction and with low frame rate (“standard” mode gives 0.77 fps/sec). Host 1 broadcasts web camera images and host 2 receives them via the bridge.

8.1.2.1 Setting up the FreeBSD bridge

The FreeBSD 4.10 bridge is set up as follows:

1. load a “bridge module” in FreeBSD using the command ‘kldload bridge’.
2. enable the bridge using the command ‘sysctl net.link.ether.bridge=1’.
   (net.link.ether.bridge has a default value of 0 after the bridge is loaded.)
3. specify 2 interfaces for the bridge using the command.
   ```
   sysctl net.link.ether.bridge_cfg= “interface_1,interface_2”
   ```

This allows IP traffic from host 1 (on interface_1) to pass through to host 2 (on interface 2).

To block IP traffic between the hosts, the bridge is disabled using the command
```
sysctl net.link.ether.bridge=0
```

To enable IP traffic between the hosts, the bridge is enable using the command
```
sysctl net.link.ether.bridge=1
```

A shell script is used to toggle the net.link.ether.bridge value between 0 and 1 every handoff time. The time is controlled using the command
```
sleep handoff_time_value
```
8.1.3 Results

Figure 8.2 and Figure 8.3 show the timestamp differences for the cases of home to foreign network (Figure 8.2) and foreign to home network (Figure 8.3) (As mentioned above the simulated interruption handoff time is obtained from the previous chapter).

For handoff from home to foreign network (always 4.7 sec handoff time) the difference in timestamps observed at the receiving web camera ranges from 5 sec to 11 sec, with the majority at 6 sec. In Figure 8.2, 73% of the results are at 6 sec or less, 3% (one result) is at 8 sec and the remaining 23% at 11 sec. This suggests that the disruption time typically lasts 1 to 2 sec (about 1 or 2 frames) longer than the handoff time, with a significant fraction of instances up to about 6 sec (about 5 frames) longer than the handoff time.

![Cumulative distribution of webcam disruption time over 30 handoff samples](image)

*Figure 8.2: Web Camera Performance over Handoffs from Home to Foreign Network*

For handoff from foreign to home network (always 3.8 sec handoff time) the disruption period observed at the receiving web camera ranges from 4 sec to 7 sec, with the majority at 6 sec. In Figure 8.3, 13% of the results are at 5 sec or less, 83% at 6 sec and only 3% (one point) at 7 sec. The disruption time for handoff from foreign to home typically lasts for some 2 sec (1 or 2 frames) longer than the handoff time.
Disruption to the actual video stream depends on the way the web camera’s transport and codec algorithms react to consecutive packet losses. These tests give an insight into the effect of Mobile IPv6 handoff on a typical, two-party web camera scenario. In this particular case, the web camera application running in standard mode (0.77 fps/sec) takes some extra time, typically ranging from around 1 or 2 seconds (1 or 2 frame times) and up to 6 seconds (5 frame times) to recover from a loss of IP connectivity, compounding the time delay introduced by Mobile IPv6 handoff itself.

8.2 TCP Throughput Performance over Mobile IPv6 Handoff

This experiment is to investigate the performance of bulk TCP transfers over an IP link that is affected by regular, emulated handoff events. The impact of Mobile IPv6 handoff rate (handoffs per minute) on TCP throughput (measured using nttcp in FreeBSD) is evaluated.

8.2.1 Methodology

A FreeBSD 4.10 based bridge (as used in the web camera trial) is used to link two FreeBSD 4.10 hosts over 100 Mbits/sec Ethernet (Figure 8.4).

As described in section 3.3.1, bulk TCP transfer will be affected by Mobile IP handoff because handoff causes both packet loss and delay. This will trigger TCP’s various congestion avoidance mechanisms, such as exponential back-off and slow start.

Running nttcp at both hosts (Figure 8.4) allows a number of single-flow, bulk TCP transfers through the bridge. nttcp output then gives a measure of the throughput of TCP data transferred over the link. As with the web camera trials, the bridge is configured to regularly interrupt IP
traffic flow in the same manner as would be experienced if the IP path operates through a Mobile IPv6 link during handoffs.

Using the setup of Figure 8.4, a set of trials is run using a range of handoff event frequencies, using the bridge to emulate real Mobile IPv6 handoff events. The handoff frequencies range from 0 through 8 handoffs per minute. The upper end of this range is the point where TCP throughput is found to fall effectively to zero. Each trial run involves the unidirectional transmission of 36,200 Mbytes of data. This is chosen so that transfer time at full line rate (about 5 minutes) is much greater than the handoff time. The 36,200 Mbytes is organised as 8,192 buffers, each of 4,6336 bytes (chosen to be an integer number of maximum packet payloads – see below). Handoff frequencies for the trials are set to 0, 1, 2, 3, 4, 5, 6, 7 and 8 handoffs per minute. The handoff rate is increased monotonically. Handoff duration is set to 4.7 sec for all cases.

To find the maximum packet payload size, Ethereal network analyser and tcpdump files were used. The Maximum Transmission Unit (MTU) is 1500 bytes. The packet overhead is made up as follows: Ethernet header (14 bytes), IP header (20 bytes), TCP header (32 bytes including the timestamp option). Hence the packet payload (MSS) is 1448 bytes.

For maximum TCP performance, the receiving window size should be set at no less than the RTT bandwidth product. Mean RTT between host 1 and host 2 is 0.292 ms and bandwidth is 100 Mbits/sec, so the receiving window size should be no less than 5 Kbytes. Hence the nttcp window size is left at its default of 16 Kbytes (receiver buffer size).

8.2.2 Testbed

The testbed is as shown in Figure 8.4. The method of setting up the FreeBSD bridge is the same as that used for the previous experiment in section 8.1.2.1. The impact of Mobile IPv6 handoff rate is evaluated using the throughput as reported by nttcp.

8.2.3 Results

With no handoff events between the two hosts, nttcp throughput has a mean value of 94.08 Mbits/sec (almost the full line rate of 100 Mbits/sec) and the bulk transfer takes approximately 5 minutes 23 seconds. As the handoff rate is increased, the nttcp throughput decreases (and bulk
transfer time increases) as shown in Figure 8.5. The data label on each plot specifies the measured throughput and the time taken by nttcp to complete the data transfer for that handoff rate case. Each point represents the mean of three tests. The decrease of throughput with increasing handoff rate is approximately linear. The final experiment at 8 handoffs per minute did not complete in 3 hours, so the throughput is taken to be zero.

![Bulk Data Transfer over Mobile IPv6 handoffs](image)

Figure 8.5: nttcp Performance over Mobile IPv6 Handoffs

It is clear that handoff disruptions have a significant impact on TCP throughput, and that the throughput continues to fall as the handoff rate increases.

### 8.2.4 Simulation

The aim of this section is to use simulation to try to replicate the experimental results of 8.2.3 (the effect of repeated handoffs on a bulk TCP transfer). The simulation uses ns2 [NS2], and the model network is intended to approximate the performance of nttcp bulk TCP transfers over an IP link that is affected by regular, emulated handoff events.

Two ns2 nodes are connected with a duplex link and a DropTail queue (see details below).

To represent the conditions of the nttcp trial, each simulated trial run involves the unidirectional transmission of 36200 Mbytes using FTP over a TCP connection while emulating handoff events. FTP is expected to achieve high throughput by approaching the full line rate after each emulated handoff. It is not known how well this will approximate the bulk TCP transfers of nttcp. Handoff is emulated by using “link down” and “link up” statements in the ns2 script. The times of emulated handoff events are chosen to give handoff frequencies of 1, 2, 3, 4, 5, 6 and 7 handoffs per minute. Handoff duration is set to 4.7 sec for all cases and nttcp window size is always set to 16 Kbytes (16384 bytes).
Ns2 parameters used in the simulation are as follows (simulation codes are provided in Appendix F):

- Link bandwidth is 100 Mbits/sec.
- Link delay $0.292/2 = 0.146$ ms (half the mean RTT).
- Interruption time is 4.7 sec, which represents the handoff duration.
- Frequency of interruption is 0, 1, 2, 3, 4, 5, 6 and 7 handoffs per minute. The frequency is increased monotonically.
- For the TCP connection
  
  o Window size is 16384 bytes.
  
  o IP packet Maximum Segment Size (MSS) is 1448.
  
  o TCP timestamp option is set to true and its size is 12 bytes

Simulated and real bulk data transfer bandwidth results over Mobile IPv6 handoff are presented in the Table 8.1

<table>
<thead>
<tr>
<th>Frequency of handoff per min</th>
<th>Simulated total time taken to complete data transmission (in sec)</th>
<th>Simulated Bandwidth</th>
<th>Experimental total time taken to complete data transmission (in sec)</th>
<th>Experimental nttcp Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>322.79</td>
<td>96.22</td>
<td>315.60</td>
<td>94.08</td>
</tr>
<tr>
<td>1</td>
<td>368.34</td>
<td>84.92</td>
<td>357.61</td>
<td>82.44</td>
</tr>
<tr>
<td>2</td>
<td>429.14</td>
<td>73.42</td>
<td>413.61</td>
<td>70.76</td>
</tr>
<tr>
<td>3</td>
<td>520.26</td>
<td>62.79</td>
<td>483.61</td>
<td>58.37</td>
</tr>
<tr>
<td>4</td>
<td>649.46</td>
<td>51.59</td>
<td>588.63</td>
<td>46.76</td>
</tr>
<tr>
<td>5</td>
<td>862.16</td>
<td>40.51</td>
<td>749.64</td>
<td>35.22</td>
</tr>
<tr>
<td>6</td>
<td>1309.30</td>
<td>29.10</td>
<td>1043.65</td>
<td>23.19</td>
</tr>
<tr>
<td>7</td>
<td>2682.60</td>
<td>17.77</td>
<td>1708.69</td>
<td>11.32</td>
</tr>
</tbody>
</table>

*Table 8.1: Comparison of Simulated and Experimental Bulk Data Transfer Bandwidths*
Figure 8.6 compares the earlier experimental results and these simulated results.

![Figure 8.6: Comparison of Simulated and Experimental Bulk Data Transfer Performance during Handoff](image)

### 8.2.5 Discussion

As can be seen from Figure 8.6, the simulated results have a similar form to the earlier experimental results. They both show an almost linear decrease of bandwidth as the handoff frequency increases.

The FTP simulated transfer gives a maximum throughput (with no handoffs) at almost the full line rate (allowing for header overhead of approximately 3.5%). `nttcp` gives a very similar (but 2% lower) maximum throughput on the “real” network. It appears likely that this is explained by the multiple slow starts required to transfer the multiple buffers used in `nttcp`. Each packet transmission takes 0.120 ms (for packet size of 1500 byte and line rate of 100 Mbits/sec), so for a RTT of about 0.3 ms, 2.5 packets can be transmitted. If the slow start algorithm begins with a window size of 2 packets, then only 0.06 ms will be wasted in the first RTT of slow start. For each buffer transferred by `nttcp` the total time spent in slow start would be $81920 \times 0.06 \text{ ms} = 4.9 \text{ sec}$. This accounts for a 1.5% difference (compared with the 2% observed in Table 8.1).

The general form of the decrease in throughput with increasing handoff frequency is consistent with the handoffs causing breaks in data transmission, but the rate of decrease of throughput is surprisingly high. For example Figure 8.5 shows that at a handoff frequency of 8 handoffs per minute the throughput falls to approximately zero. At this handoff frequency the handoff period is 7.5 sec. Each handoff time of 4.7 sec is thus followed by 2.8 sec of time in which the link is available, but TCP is not able to achieve significant throughput. That is, TCP
is not able to achieve significant throughput in about 9,300 RTTs following reconnection after handoff. It seems likely that this is due to a large RTO increase associated with the block of lost packets during handoff.

The general linear form of Figure 8.5 is consistent with each handoff causing a loss of transmission for a fixed time (although this time is substantially greater than the handoff time itself). For example at 7 handoffs per minute, the handoff period is 8.57 sec, of which 4.7 sec is the handoff time, 1.01 sec is occupied with transmission (assuming this is at full line rate), leaving 2.86 sec of time where the link is available but there is no throughput.

At the highest rates of handoff FTP achieves almost 60% more throughput than nttcp (see Table 8.1 at a handoff frequency of 7 per minute). Using the same kind of analysis the time where the link is available but there is no throughput would be 2.29 sec (compared with 2.86 sec for the nttcp experiment). The reason for this difference is not explored further here.
9 Concluding Remarks and Directions for Future Research .......................................................... 119
Chapter 9

Concluding Remarks and Directions for Future Research

Interruption of packet flow during handoff disrupts the network-level transparency that Mobile IPv6 attempts to provide to upper layer services. This disruption may take the form of transient spikes in the round trip time (if packets are buffered during handoff) or a burst of packet loss (if packets are discarded during handoff). The duration of this disruption depends on the interaction between the link layers (where network disconnection and attachment are often first noticed) and link layer independent mechanisms implemented by Mobile IPv6 to detect changes in network attachment point. In this thesis, experimentally triggered handoff events are used to measure the time period during which connectivity is lost using KAME Mobile IPv6 stack in FreeBSD. This is chosen to represent a standard Mobile IPv6 implementation. The characteristics of standard Mobile IPv6 independent of the link layer technology are also analysed in the thesis.

Findings are as follows:

- Real-world 802.11b handoffs are typically completed in less than 700 ms. The handoff triggering method has an impact on 802.11b handoff times.

- The IP level disruption due to Mobile IPv6 handoff together lasts significantly longer - around 4.0 sec with a standard deviation of 103 ms (for home to foreign network handoff) or 3.1 sec with standard deviation of 332 ms (for foreign to home network handoff).

- The choice of different Mobile IPv6 routing methods (tunneling method or Route Optimisation) does not have significant impacts on Mobile IPv6 handoff times.

- When evaluating each component of the Mobile IPv6 handoff times (detection, configuration and registration times), detection time is found to have accounted for most of the Mobile IPv6 handoff time. Detection (in KAME Mobile IPv6 stack) is the NUD process which always takes 3 seconds.
- Tuning the router advertisement (RA) intervals from 30-70 ms (the default) to 500-800 ms does not significantly degrade these handoff times. This suggests that short RA intervals may, in practice, not be worth the transmission overhead, particularly for resource-constrained environments (e.g. limited bandwidth or battery power).

- Providing mobility support at the network layer has the advantages of being independent of the link layer used in an access system. However information from the link layer is essential for more effective and efficient support of seamless handoff. Indeed, our results provide further justification for tighter coupling between link layer and network layer protocols in order to significantly improve the resulting handoff times.

- Simple implementation bugs can cause substantial increases in the handoff latencies, regardless of the actual Mobile IPv6 protocol itself. Such bugs were found in the KAME Mobile IPv6 stack used here.

- The impact of handoff on applications clearly shows that a default Mobile IPv6 environment would be highly disruptive to real-time and interactive applications during handoff events, even if the underlying link-layer handoff was instantaneous.

Care is required in setting up experiments for handoff evaluation and for assessing the sources of experimental error involved (for example by calibrating equipments or by running multiple trials (refer to chapter 8)). In this work some of the problems in such experiments have been identified, and the methods have been documented (refer to chapter 8, chapter 9, appendix D and E) so that others can evaluate and build upon the results.

Further research is desirable to investigate and what factors can be used to reduce each component of Mobile IPv6 handoff time. This means a more comprehensive study of nearest neighbour discovery protocol is also included in future work. Further work is also required to investigate packet loss and jitter during mobile IP handoff operations. Benchmarking some of the available methods for Mobile Node fast movement detection e.g. predictive and non-predictive handoff optimisations using RAs, fast handovers for Mobile IP over WLANs using layer 2 triggers etc. would also provide useful information for those interested in Mobile IPv6 and its applications.
Appendix A

IPv6 Addressing

More details of IPv6 addressing are covered in this appendix.

A.1 IPv6 Address Format

A.1.1 Address Allocation

IPv6 globally routable unicast addresses have the following format

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>FP</td>
<td>TLA ID</td>
<td>RES</td>
<td>NLA ID</td>
<td>SLA ID</td>
<td>Interface ID</td>
</tr>
</tbody>
</table>

- FP: Format Prefix
- TLA ID: Top Level Aggregation ID
- RES: Reserved
- NLA ID: Next Level Aggregation ID
- SLA ID: Site Level Aggregation ID
- Interface ID is generated using interface’s link layer address which is typically the lower 48 bit MAC address

For example if FP = 001, TLA = 010000001001, NLA = FECE08 and SLA = 010E, the IPv6 address will be written as 2813:00FE:CE08:010E:<interface ID>

RFC 3177 [IAIE01], “IAB/IESG Recommendations on IPv6 Address Allocations to Sites”, recommends IPv6 Address Allocations to sites as follows:

- A /48 prefix (e.g. a unique TLA/NLA prefix combination) is commonly assigned to most IPv6 sites. The /48 prefix is assigned to IPv6 sites by default, however large sites can be assigned a /47 or multiple /48 prefixes (multiple NLAs) contiguously.
- A globally unique unicast address is assigned a /128 prefix.
- A single subnet site is assigned a /64 prefix (e.g. a unique TLA/NLA/SLA prefix combination).

A.2 IPv6 Address Autoconfiguration

Using IPv6 address autoconfiguration, an IPv6 node automatically obtains an IP address for use on an interface [TN98]. Autoconfiguration is performed on multicast capable interfaces. IPv6 address autoconfiguration forms an IPv6 address by pre-pending the prefix to interface
identifiers of appropriate length. These addresses are checked using Duplicate Address Detection to ensure the uniqueness of the address before they are assigned to an interface.

**A.2.1 Link local address**

Link local address is formed by address autoconfiguration whenever an interface becomes enabled [TN98]. The interface becomes enabled after the following events:

- It is initialised at system startup time
- It attaches to a link for the first time
- It is administratively disabled or failed and then administratively enabled

**A.2.2 Duplicate Address Detection**

Duplicate Address Detection (DAD) is performed on each unicast address before it is assigned an interface regardless of whether it is obtained through stateful, stateless or manual configuration. DAD ensures addresses are not duplicated on a link.

RFC 2462 [TN98], “Stateless Address Autoconfiguration” describes the DAD process for stateless address autoconfiguration. In this case DAD is determined by the interface identifier. The link local address must be tested for uniqueness with DAD. If the uniqueness of the link local address is guaranteed, an implementation may choose to skip DAD for other addresses formed from the same interface identifier.

The unicast address that is formed before DAD has completed successfully (before assigning to an interface) is called a tentative address. The IPv6 node should log a system management error if this tentative address is found to be duplicated. In this case the interface should be either disabled or be assigned a new identifier, or all IP addresses for the interface will need to be manually configured.

DAD detects duplicate addresses using Neighbor Solicitations and Advertisements containing the tentative address in the Target Address field. The interface must accept only Neighbor Solicitation and Advertisement messages using this tentative address. It must silently discard all other packets with a target address that matches the tentative address.

DupAddrDetectTransmits (with default value 1) is the number of consecutive Neighbor Solicitation messages sent for Duplicate Address Detection on a tentative address. DupAddrDetectTransmits Neighbor Solicitation is sent within RetransTimer milliseconds (default value 1000 milliseconds) to detect if the address is unique. A valid Neighbor Advertisement will tell nodes whether the target address is tentative or matches a unicast or any address assigned to the interface. The address is not unique if the target address is tentative. Otherwise it is unique and can be assigned to an interface.

**A.3 Transitions from IPv4 to IPv6**

RFC 2893 [GN00], “Transition mechanisms for IPv6 hosts and routers” specifies IPv4 compatibility functions that can be implemented by IPv6 hosts and routers, which include DNS and IPv4 ICMP error handling.
A.3.1 DNS

When both IPv4 and IPv6 protocols are supported in the network, a Domain Name System (DNS) is used that supports both IPv4 and IPv6. This DNS provides resolving capabilities to deal with IPv4 “A” records and IPv6 “A6” and “AAAA” records to map between IP addresses and host names.

A.3.2 Handling IPv4 ICMP errors

In IPv6 over IPv4 tunneling, the encapsulating node might receive IPv4 ICMP error messages from IPv4 routers because the encapsulated packets come from an IPv4 source. For example, based on IPv4 Path MTU Discovery, ICMP “packet too big” error messages may be generated, and this will be recorded in the path MTU at the IPv4 layer. Using the recorded path MTU, the IPv6 node is then able to determine if an IPv6 ICMP “packet too big” error must be generated.

According to RFC 2893, the handling of other types of ICMP error message depends on the amount of information included in the “packet in error” field of the encapsulated packet that caused the error. If the packet in error field does not include enough space for the IPv6 header, the IPv6 ICMP message cannot be generated. If the offending packet includes enough data, the encapsulating node can generate an IPv6 ICMP message for sending to the originating IPv6 node.

A.3.3 Scenarios and Analyses for Introducing IPv6 into ISP Networks

RFC 4029 [LKS’05], “Scenarios and analyses for introducing IPv6 into ISP networks” is for ISPs running IPv4 networks aiming to provide IPv6 connectivity to customers without interrupting existing IPv4 services. RFC 4029 discusses and evaluates these scenarios and also identifies challenges for ISPs. RFC 4029 also introduces and analyses the existing transition mechanisms from IPv4 to IPv6 from the ISP’s point of view.

The RFC includes actions required for backbone transition, for customer connection transition and for network and service operation transition. Overall, an IPv4 backbone and IPv4 customer connection network is transformed to a dual-stack network. Dual-stack customer connection networks are connected to other IPv6 networks through an IPv4 backbone. IPv6 customers are connected to an IPv6 backbone through an IPv4 network. Security for network and service operation is also required to be implemented.
Appendix B

QoS Protocols

B.1 IP QoS Protocols

B.1.1 Resource Reservation Protocol (RSVP)

RSVP [BZB’97] is a signaling protocol which provides mechanisms to specify data flows and to reserve resources along the flow communication path before actual data transfer takes place. RSVP is a receiver-oriented protocol, in which the receiver requests reservation of resources, and the senders tell about their flows. RSVP reserves resources for both unicast and multicast flows.

RSVP signaling messages are path (Path), reservation (Resv), error (Err), reservation confirmation (ResvConf), and teardown (Tear). Path and Resv message types are used to setup resource reservation states on the routers (or nodes) between a sender and a receiver. Data flows may go from sender to receiver with Path message type (Path, PathTear, ResvErr, ResvConf), which sets up the path state and return to sender with Resv message type (Resv, ResvTear and PathErr), which sets up reservation state.

An RSVP sender sends out a Path message hop by hop to RSVP node/router(s) to find the path to the receiver for a specific flow. When the Path message reaches the receiver, the receiver returns a Resv message, which travels a reverse path identical to the originating path to ask for resources. Upon receiving a Resv message, each node/router on the path will allocate the resources specified if sufficient resources are available and pass the message back toward the sender. When the Resv message reaches the sender the reservation is complete and data transfer begins, with the presumption that each RSVP node/router along the path has guaranteed the resources.

Each RSVP message carries a Session object, which identifies the destination of the flow. The Session object contains the destination address of the flow, Protocol ID and the destination port number. RSVP messages are as follows:

- Path message: Path messages are sent with the same source and destination address as the data, so that they will be routed correctly through non-RSVP networks. A RSVP node/router creates Path state for each sender by processing a Path message. It also periodically records all these four objects for each sender and scans their Path states to generate new Path messages and forward them toward the receiver.

- Resv message: The Resv message includes Flow_spec, Filter_spec and Scope.
- Error messages (PathErr and ResvErr): These indicate errors to the sender of the message.
  - PathErr is sent upstream to the sender.
  - ResvErr is sent to the receiver if the reservation is rejected at any router along the upstream path.
- ResvConf message: This message is sent upstream to the receiver to acknowledge the reservation request.
- Teardown message (PathTear and ResvTear): These are used to immediately remove the path or reservation state respectively. ResvTear is sent upstream to the sender and PathTear is sent downstream to the receiver.

The RSVP protocol contains soft state (to manage the reservation), local repair (for fast adaptation to routing changes), and merging (to reduce message overhead) features. Soft states are periodically refreshed by Path or Resv messages.

**B.1.2 Integrated Service (IntServ)**

IntServ [SW97] is a flow-based QoS architecture, which provides each individual flow with certain service guarantees based on reservation. IntServ relies on RSVP to set up and tear down resource reservations along the intended path of a flow of packets using a dynamic signaling protocol, admission control, packet classification, and intelligent scheduling to achieve the desired QoS. In IntServ, RSVP is used to request a specific type of QoS and make a resource reservation for the flow.

**B.1.3 Differentiated Service (DiffServ)**

The DiffServ [Bla98] architecture is reservation-less. It classifies packets into a number of service types and uses priority mechanisms to provide differentiated services (QoS) to the traffic. The service classifications are based on the typical requirements of different kinds of applications. QoS will be assured by dropping traffic in excess of allowable levels and supplying the needed bandwidth by priority queuing mechanisms. DiffServ aims to solve the RSVP scaling issues when dealing with large numbers of flows.

The DiffServ defines an 8-bit DSCP (DiffServ Code Point), which corresponds to service differentiation by DiffServ-enabled routers using the 8-bit ToS field in IPv4 headers and the 8-bit Traffic Class field in IPv6 headers. Therefore DiffServ-based QoS occurs at the IP layer, regardless of the physical and data link protocols. The DSCP byte restructures the ToS field to carry information about IP packet service requirements and traffic behaviour controls.

DiffServ routers use a marking method to examine each packet. Packets are classified via source destination address, port number, and protocol and marked with a DSCP. Packets are then treated based on DSCP mapping where the DSCP indicates per hop behaviour, which corresponds to the type of handling. DiffServ domains consist of DiffServ routers that agree on a DSCP mapping. Thus packets can be passed from one DiffServ domain to another.

The performance of DiffServ relies on sufficient provisioning of network resources in the backbone. DiffServ assumes that resources in access networks (between a host and the backbone) are over-provisioned.
B.2 QoS Schemes for Mobile IP

When a MN moves from one location to another, the active data flow path changes. The bandwidth, jitter, throughput and delay may change due to changes in the path length and the difference of router congestion levels. In addition, the MN suffers from packet loss and delay during Mobile IP handoff. Thus the MN perceives a different QoS relative to the previous location. In the worst case, the MN may drop its connection, as it cannot achieve its minimum QoS requirement.

Zhigang et al [ZJJJ01] lists issues to be considered when defining a QoS solution for Mobile IP:
- Handoff and roaming in heterogeneous QoS domains
- Roaming between different media
- No advanced resource reservation
- No QoS negotiation/signaling for heterogeneous domains
- Duplicate Signaling for InServ Mobile IP
- Mobile IP’s packet loss and delay during handoff

As covered in chapter 3 of the thesis, IP QoS protocols (such as RSVP and DiffServ) cannot be used directly in mobile environments. The subsections below describe some approaches defining extensions to basic IP QoS protocols.

B.2.1 Extensions to RSVP to Support Mobile IP

There have been several designs for extensions to RSVP to allow for seamless mobility.

B.2.1.1 RSVP Tunnel

Terzis et al. [TSZ99] defines RSVP tunnel mechanisms that guarantee QoS inside the RSVP tunnel. RSVP sessions between the tunnel entry and exit points are established by adding an extra pair messages - ‘tunnel Path message’ and ‘tunnel Resv message’.

![RSVP Tunnel Diagram](image)

Figure B.1: RSVP Tunnel

The original end-to-end RSVP Path message header is tagged with the RSVP protocol number 46 and records the addresses of the sender and the receiver. When this Path message is delivered to the tunnel entry point (Figure B.1), it is encapsulated with a new IP header. The new IP header is tagged with the Mobile IP protocol number 4 and records the addresses of the tunnel entry and exit points. The tunnel entry point router, after sending the encapsulated Path message to routers, issues a new tunnel Path message, which records the addresses of the tunnel entry and exit points and is tagged with the RSVP protocol number 46. On receiving the Path message, each router on the path of the tunnel forwards the message downstream to the exit
point. On receiving the ‘tunnel Path message’, each router performs the path finding function. On arriving at the tunnel exit point, the Path message will be decapsulated and forwarded to the receiver. The tunnel Path message will be processed only by the exit point and not be forwarded to the receiver.

On receiving the Path message, the receiver returns a Resv message to the sender. When Resv message arrives at the tunnel exit point, it will be encapsulated with an IP header (Mobile IP protocol number 4) and then be forwarded to the sender. The tunnel exit point issues a new ‘tunnel Resv message’ and delivers it to the tunnel entry point. Thus all routers on the path of the tunnel can reserve the desired available resources for the receiver.

The RSVP Tunnel can resolve the RSVP signaling invisibility problem in Mobile IP. However this method does not support seamless handoff due to the lack of advanced resource reservations.

### B.2.1.2 Advanced Resource Reservations

The challenging issue when performing reservation in advance in Mobile IP is how to anticipate MN movement so that reservation can be done in advance. MN movement prediction should aim to provide topological information about the new network and advanced warning of the handoff event.

Mobile RSVP (MRSVP)

Talukdar et al defines Mobile RSVP (MRSVP) [TBA01], which is an extension of standard RSVP to provide advanced resource reservation for MNs. MRSVP provides resource reservations to a set of locations which the MN could visit in the future. These locations are defined in MSPEC. MSPEC can be specified by the network or by a MN. When a MN initiates reservation, MRSVP assumes that the MN has acquired its MSPEC either from the network or from its mobility profile.

In MRSVP, proxy agents are introduced to acquire resource reservations for the MN. A local proxy agent is located at the current location of the MN. Remote proxy agents will be located at the other locations specified in the MSPEC. The local proxy agent makes an active reservation from the sender to the MN (or from the MN to the receiver). The remote proxy agents make passive reservations at the other locations of MSPEC. When the MN moves to a new location, MRSVP changes the passive reservation of the new location into an active state, and the original active reservation is changed to the passive state [TBA01]. The MN also needs to search all of the proxy agents in its neighbourhood when at its new location and then update MSPEC with the proxy agents found by the MN. The updated MSPEC then sends a message to the sender, which tells the sender about the MN’s possible locations so that it can properly initialise flows to the MN. The MN also sends this message to all remote proxy agents recorded in the updated MSPEC. These remote proxy agents thus are able to retrieve the QoS guaranteed parameters for any flows between sender and MN. In MRSVP the MN is responsible for supplying its mobility information (i.e., a list of CoAs in the foreign networks it may visit).

The MRSVP model solves the timing delay for QoS re-establishment, but the protocol is complex and expensive since it wastes bandwidth. Proxy agents and their communication protocol increase the complexity of the network and may degrade the network performance.
Hierarchical Mobile RSVP (HMRSVP)

HMRSVP attempts to manage QoS for the MN when it moves to a new location (as does MRSVP) but makes less advanced resource reservations. HMRSVP adopts the hierarchical concept of Hierarchical Mobile IP and makes advanced resource reservations for the MN only when the MN is in an overlap area of two domains.

When the MN moves within the regional domain, it only needs to make a Mobile IP regional registration. HMRSVP sets up active resource reservation (similar to MRSVP) along the path from the sender to the MN. When the MN moves between regional domains and in the overlap area of two domains, HMRSVP just establishes one passive resource reservation along the path from the sender to the overlapped domain that the MN is about to move to. This saves the MN from making excessive passive reservations compared with MRSVP. However HMRSVP is still a complex and expensive protocol which involves the use of proxy agents.

B.2.1.3 Dynamic RSVP (DRSVP) for Mobile IPv6

Kuo et al [KK00] introduces the concept of a DRSVP tunnel to support resource clearing and resource reservation when the MN hands off to a new location. DRSVP tunnel resources in common intermediate routers (components of DRSVP that are shared by both old and new paths of MN) are able to be reused (Figure B.2).

![Figure B.2: DRSVP Network Elements](image)

When the MN and the CN are communicating, a DRSVP Reservation Tunnel is established. When the MN moves to a new location, the DRSVP Reservation Tunnel is re-established. The MN reserves resources at the new routers (in this case Rd2 and Rj) of the DRSVP Reservation Tunnel (Resource Reservation process). The reserved resources of components on the old DRSVP Tunnel (in this case Rd1 and Rl) are released (Resource Clear process).

In DRSVP, as shown in Figure B.2, after the MN obtains a new CoA, the MN initiates Resource Clear and Resource Re-reservation Request simultaneously to a new Router Rj. The DRSVP Tunnel is re-established to determine allowable reserved resources and bandwidth. Rc, the last common router, which is defined as the critical common intermediate router, will initiate a Resource Clear request to Rd1 to release reserved resources and bandwidth. Rd1 will transmit this request downstream to Rl. Rc initiates Resource Reservation Acknowledgement to Rd2 and Rd2 sends this acknowledgement to the CN. Resources are thus reserved along new communication path of the MN without re-establishing resource reservations for the whole path.

DRSVP still does not solve the scalability problem.
B.2.1.4 Others

Indulksa and Cook [IC98] presents an extension of RSVP and characterises the static reservation process used in RSVP to support QoS in a Mobile IP environment.

Hadjiefthymiades et al [HPFM98] proposes a Path Extension scheme. In this scheme the RSVP protocol is modified so that existing reservation is preserved and reservation extension is performed locally from the old to the new Access Router. However in this scheme, modifications are required in the network components and the related protocols.

Das et al [DJKS99] solves the issue of MRSVP that relies on the MN to supply its mobility information (such as a list of CoAs in the foreign networks it may visit). It introduces two new protocols - Neighbor Mobility Agent Discovery Protocol and Mobile Reservation Update Protocol.

Chen and Huang [CH00] proposes an extension of RSVP based on multicast IP to support a Mobile IP environment. The mobility of the MN is modeled as a changeover in a multicast group membership. When a MN moves to a new network, the multicast tree is modified dynamically. Flow re-routing is thus reduced, however this approach introduces the need for background processing for multicast packets.
Appendix C

Mobile IP Implementation

This appendix provides information on available tools for implementing Mobile IP networks or simulating Mobile IP networks. Information is obtained from the website of the Mobile IP4 working group [IMPL].

C.1 Implementation Tool

C.1.1 Mobile IPv4

(As of March 2005)

<table>
<thead>
<tr>
<th>Open source</th>
<th>Commercial</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monarch</td>
<td>Monarch</td>
</tr>
<tr>
<td>NetBSD 1, FreeBSD 2.2.2</td>
<td>NetBSD 1, FreeBSD 2.2.2</td>
</tr>
<tr>
<td>Rich University</td>
<td>SEGCO</td>
</tr>
<tr>
<td>Linux FreeBSD 2.2.8, 4.6, 4.8, 4.9, 5.2</td>
<td><a href="http://www.seego.com/">http://www.seego.com/</a></td>
</tr>
<tr>
<td>Secure Mobile Net</td>
<td>Windows</td>
</tr>
<tr>
<td><a href="http://www.cs.pdx.edu/research/SMN/">http://www.cs.pdx.edu/research/SMN/</a></td>
<td>Linux</td>
</tr>
<tr>
<td>Actively updated</td>
<td>Windows</td>
</tr>
<tr>
<td>Dynamics Helsinki University of Technology</td>
<td><a href="http://www.ipunplugged.com/">http://www.ipunplugged.com/</a></td>
</tr>
<tr>
<td><a href="http://dynamics.sourceforge.net/">http://dynamics.sourceforge.net/</a></td>
<td>Birdstep</td>
</tr>
<tr>
<td>Linux Windows</td>
<td><a href="http://www.birdstep.com/">http://www.birdstep.com/</a></td>
</tr>
<tr>
<td></td>
<td>Windows embedded systems</td>
</tr>
<tr>
<td>UoB-zNOMAD</td>
<td>Userland</td>
</tr>
<tr>
<td>Linux SPL, based on NOMADv4</td>
<td>index.html</td>
</tr>
<tr>
<td>MosquitoNet</td>
<td>Platform independent, Linux 2.0.3</td>
</tr>
<tr>
<td><a href="http://mosquitonet.stanford.edu/MobileIP/">http://mosquitonet.stanford.edu/MobileIP/</a></td>
<td>Ecutel.com</td>
</tr>
<tr>
<td>Linux kernel 2.2.16</td>
<td><a href="http://www.mip4.org/2004/mip-implementations.html">http://www.mip4.org/2004/mip-implementations.html</a></td>
</tr>
</tbody>
</table>
C.1.2 Mobile IPv6

(As of March 2005)

<table>
<thead>
<tr>
<th>Open source</th>
<th>Commercial</th>
</tr>
</thead>
<tbody>
<tr>
<td>KAME</td>
<td>Elmic Systems (Treck Inc.)</td>
</tr>
<tr>
<td>FreeBSD 4.9, NetBSD 1.6.1, OpenBSD 3.3</td>
<td>OS independent</td>
</tr>
<tr>
<td>Stable, actively updated</td>
<td></td>
</tr>
<tr>
<td>Mobile IPL</td>
<td>Cisco Systems</td>
</tr>
<tr>
<td>GPL, stable, actively updated</td>
<td>OS: Cisco IOS</td>
</tr>
<tr>
<td>HMobile IPv6</td>
<td>Mobile IPv4 commercial, Mobile IPv6 technology preview</td>
</tr>
<tr>
<td>Linux</td>
<td></td>
</tr>
<tr>
<td>Actively updated</td>
<td></td>
</tr>
<tr>
<td>HMobile IPv6</td>
<td>INRIA’s</td>
</tr>
<tr>
<td>FreeBSD 3.4</td>
<td>Old</td>
</tr>
</tbody>
</table>

C.2 Simulation Tool

C.2.1 Mobile IPv4: Wireless and Mobility Extensions to ns-2

“Wireless and Mobility Extensions to ns-2” is provided by the Rice Monarch Project [RIMO]. The project has made extensions to the ns-2 network simulator that enable it to simulate MNs connected by wireless network interfaces, including the ability to simulate multi-hop wireless ad hoc networks.

C.2.2 Mobile IPv6: Mobiwan over ns2

MobiWan [MOBI] is a public simulation tool based on ns2 (on version ns-2.1b6 only) to simulate Mobile IPv6 under large Wide-Area Networks (both local-area mobility and global-area mobility). MobiWan comprises extensions to simulate Mobile IPv6, and extensions to manipulate and configure large network topologies (TOPOMAN / TOPOGEN). Mobiwan has been developed by Motorola Labs Paris in collaboration with the INRIA Planete Team. “Mobiwan for ns-2.26” is developed by Thierry Ernst, which is an extended version of MobiWan (as MobiWan operates on ns-2.1b6 only). Mobiwan for ns-2.26 works with ns2 version ns-2.26. It maintains the same code structure (is an embedded extension) and may therefore conflict with other extensions [MOBI2].

C.3 SHISA - The New KAME Mobile IPv6/NEMO Stack

At the time writing this thesis, there are some changes to KAME Mobile IPv6 development. The following information is from the KAME newsletter 21/12/2004 [NEKE] and is included here to help clarify any difference between Mobile IPv6 code used in this thesis and the newly implemented Mobile IPv6 code.
The old KAME (KAME stacks up to 27/12/04 are referred in this discussion, but the stack used for experimental work in this thesis is the release of 28/06/2004) enlarges the kernel size since the most of code is implemented in the kernel. The large kernel makes it difficult to debug and also to add further extensions. The new Mobile IP6 KAME code is named SHISA. SHISA implements the Mobile IPv6 stack on userland. The current SHISA is based on RFC3775 [JPA04], RFC3776 [ADD04], RFC 3963 [DWPT05] and draft-wakikawa-mobileip-multiplecoa-03 [WEN04]. It works on FreeBSD 4.10, FreeBSD 5.3 and NetBSD 1.6.2. Changes in SHISA (relative to the old KAME) are:

- Userland: Most functions (especially signaling) are moved to userland.
- Home address assignment: Home addresses are not assigned automatically. They will be specified explicitly.
- The NEMO basic support protocol is implemented by default.
- Getting control information: ‘Mip6control’ is no longer supported. telnet can be used to get mobility status or information.

It is not clear whether implementation in userland instead of the kernel will have an impact on handoff time. It may slow the signaling required for handoff.

C.4 Mobile IPv6 Implementation for MIPL and KAME

C.4.1 MIPL implementation

MIPL [MIPL] implements Mobile IPv6 handoff on a Linux platform using the following processes [CPP02] (version 09):

- Layer 2 handoff: MIPL shell script sends IOCTL to layer 3 software immediately after issuing a command to perform layer 2 handoff. IOCTL is a function that prepares the RS-send-function then calls the RS-send function and sets the current router state to unreachable

- Layer 3 handoff:

  MN sends RS: within local scope (transmission takes time of the order of ms).
  Router processes RS and then waits for a random time up to 500 ms.
  Router sends RA: within local scope. MN then processes RA, calculates its new CoA, prepares to send a BU to the HA.
  MN sends BU to HA: Transmission time depends on distance from MN to HA.
  HA processes the BU and prepares an acknowledgement.
  HA sends BA to MN: Transmission time depends on distance from MN to HA.
  MN waits for random time up to 500 ms for another packet transmission that it can use to send the BU to the CN.
  MN sends BU to CN: Transmission time depends on distance from MN to CN.
  CN processes BU and prepares an acknowledgement
  CN sends BA to MN: Transmission time depends on distance from MN to CN.
C.4.2 KAME implementation

The KAME implementation is discussed in Chapter 7 of the main body of the thesis. The brief outline here is for comparison with the MIPL outline above.

When the MN moves from the Home Link (HA (HA)) to a Foreign Link (Router 1 (R1)), KAME executes Mobile IPv6 handoff procedures using the following basic processes (for snap kit kame-20040628-freebsd49-snap.tgz):

- HA sends RA (HA -> all HA’s nodes); R1 sends RA (R1 -> all R1’s nodes)
- NUD: MN sends Neighbor Solicitations to HA (3 Neighbor Solicitations)
- HA receives BU from MN; HA sends BA to MN
- MN sends BU to CN; CN sends BA to MN
Appendix D

Setting up 802.11 Clients using FreeBSD

FreeBSD allows the configuration of 802.11 clients, which are machines using PCI- or ISA- to-PCMCIA adapters and PCMCIA clients. This appendix documents steps for setting up 802.11 Clients in a FreeBSD environment. These procedures are used to set up 802.11b networks used for the experiments reported in this thesis.

D.1 Modifying The Kernel

The kernel (any version of FreeBSD) is re-configured by adding supporting kernel options for the wireless card

- ‘device <wi|an|awi>’ is added where “wi” is for Orinoco cards, “an” is for Aironet cards, and “awi” is for PRISM cards.
- Only in the case of using a PCI adapter, the following three lines should be added to support PCMCIA
  
  device card

  device pcic0 at isa? irq 0 port 0x3e0 iomem 0xd0000

  device pcic1 at isa? irq 0 port 0x3e2 iomem 0xd4000 disable

- IPsec-related options for security should be added (mandatory for IPv6), “options IPSEC”, “options IPSEC_ESP” and “options IPSEC_DEBUG”.

D.2 Modifying the File System: rc.conf

In /etc/rc.conf file, ‘pccard_enable=“YES” ’ must be added to enable the PCMCIA card.

D.3 Wireless Configuration

There are a number of parameters that must be configured for wireless connection (e.g. frequency, operating mode and station name). In FreeBSD a script such as /usr/local/sbin/wireless.sh could be created which will change all of the parameters at once. Below are the basic commands in the wireless.sh shell scripts for wireless configuration using Orinoco cards.

#!/bin/sh (run via the sh shell)

  wicontrol -f 10

  #Sets the wireless card’s frequency.
wirecontrol -p 1
#BSS mode, meaning clients must associate with the server, and cannot directly connect
to one another.
wirecontrol -c 1
#IBSS mode, an extention of BSS mode.
wirecontrol -s “servername”
#Sets the station name.
wirecontrol -k “12345”
#The WEP key
wirecontrol -e 1
#Enables WEP security.
ifconfig wi0 ssid “magicap”
# Set the name of the wireless network.
ifconfig wi0 inet IP_address netmask netmask_address
# Set the IP address and netmask of the wireless device
route add default IP_address_of_Access_Point
# Set default gateway

D.4 Modifying the File System: pccard.conf

The file pccard.conf (directory: /etc/defaults/pccard.conf) defines default values for the wireless
cards, e.g. headers of the card like: “Lucent Technologies”, “WaveLAN/IEEE”, or “Aironet”. If
rc.conf (directory: /etc/rc.conf) defines the pccard_conf variable, that value will be used.
Otherwise, the default in pccard.conf applies.

D.5 Troubleshooting

In FreeBSD, if a PCI adapter is used and there is trouble (such as FreeBSD hanging on boot, or
the wireless card is not working or cannot be recognised), the following two lines should be
added to /boot/loader.conf

    hw.pcic.intr_path=“1”
    hw.pcic.irq=“0”

If the machine hangs when the wireless card runs, it may be because the card takes an IRQ
that is already in use. Remove the card, reboot the machine, and find a free IRQ from the log
message’s output. Change the section’s config line of the pccard.conf to the free IRQ number.
This will force pccard to give the wireless card the free IRQ.
Appendix E

Technical Report – Implementing an IPv6 and Mobile IPv6 testbed using FreeBSD 4.9 and KAME
Implementing an IPv6 and Mobile IPv6 testbed using FreeBSD 4.9 and KAME

Lawrence Stewart  Mai Banh  Grenville Armitage
Centre for Advanced Internet Architectures. Technical Report 040401A
Swinburne University of Technology
Melbourne, Australia
lastewart@swin.edu.au  mbanh@swin.edu.au  garmitage@swin.edu.au

Abstract-This technical report aims to explain and document the process of implementing a mobile IPv6 testbed using VIA EPIA-V Mini-ITX based machines running FreeBSD 4.9 and the development code from the KAME project. The steps taken to verify the functionality of the testbed have also been documented.

Keywords- IPv6, Mobile IPv6, KAME, FreeBSD

I. INTRODUCTION

The exponential growth of the Internet in recent years has posed a number of problems for engineers to solve. Every node that expects to be publicly visible and reachable on the Internet must have an IP (Internet Protocol) address that is unique among all the other publicly visible and reachable nodes on the Internet [1]. When IP was conceived and developed in the late 70s, the designers allocated 32 bits to represent an IP address, which translates to just under 4.3 billion addresses. No one expected the Internet to gain as much popularity as it has, and 32 bits was therefore thought to be far more than necessary at the time of inception.

As a result of IPv4's hierarchical address allocation scheme (subdividing the IPv4 address space into networks and subnetworks) and phenomenal growth in popularity and scope of the Internet [2], the Internet Engineering Task Force (IETF) began work on a successor to IPv4 in the early 1990s - which became IP version 6 (IPv6) [3][4]. Although IPv6 retains the same hierarchical address assignment approach, its expanded 128bit address fields provide substantial breathing room in the face of further growth in the numbers of Internet attached devices.

Accompanying the Internet boom has been a huge increase in the demand for mobile communications and services. Mobile phones, for example, are now not only used to make calls, but send messages, pictures, video content and even check email.

The inclusion of support for Mobile IP in the base IPv6 specification and development effort is another improvement [5]. Mobile IPv6 grew out of experiences with Mobile IPv4 [6], itself an attempt to enable IP attached devices to migrate between physical networks without having to change the publicly visible IP address by which they were uniquely known to the rest of the Internet.

The potential for mobile IPv6 in applications such as 4G generation mobile phones and other mobile devices is huge, and as a result of this, a large number of companies are putting money into research and development. The KAME project, which is the source of the mobile IPv6 code used to build our testbed, is a joint venture between 6 different companies in Japan.

We found the setting up of an experimental mobile IPv6 testbed to be quite challenging. However, with our documented steps, some basic FreeBSD knowledge and some hardware, you too can have your own testbed running in a couple of hours.

II. TESTBED LAYOUT

Figure 1 shows the physical layout for our testbed and the interface/address information for each host.

![MAGIC Testbed Layout](image)

**Figure 1. Mobile IPv6 testbed layout**
III. TESTBED HARDWARE AND SOFTWARE

VIA Mini-ITX based systems were used for the various devices in the testbed and consisted of:

- VIA EPIA-V series Mini-ITX motherboard [7]
- VIA 800MHz C3 E-Series processor
- On board 10/100 NIC (VT6102 chipset)
- 80 Gig Seagate Barracuda 7200 RPM HDD
- 256 Meg PC133 RAM

The mobile node had a VIA Aironet 350 Series PCMCIA wireless network card in a PCI to PCMCIA cradle inserted into its free PCI port.

Two VIA Aironet 1200 series wireless access points were used in the testbed.

The switch used was an Alloy NS05C 5 port Nway Mini Switch [8].

All NICs (excluding the wireless adapter) were explicitly configured to run at 10Mbps in full duplex mode.

All systems ran FreeBSD 4.9 [9] as the operating system. To provide the mobile IPv6 features, we used the KAME [10] code snap kit dated 22/03/2004 for FreeBSD 4.x series [11].

It should be noted that the KAME software is still under heavy development and often has features that are not completely implemented, buggy or being changed from snap kit to snap kit. The KAME newsletter [12] and mailing lists [13] can provide some help and information.

IV. CONFIGURATION

A. The KAME snap kit

Extracting the KAME snap kit from the archive results in the creation of the top level directory “kame” in which all snap kit related files are placed. The “INSTALL” file within this directory is of most interest, as it outlines the necessary steps that need to be taken to prepare the system.

As we were building the code for a new installation of FreeBSD 4.9, the only command we needed to issue was “make TARGET=freebsd4 prepare”.

This creates all the necessary symbolic links to build a kernel from the FreeBSD and KAME source trees.

The newly created directory named “freebsd4” contains an “INSTALL” file that walks you through the rest of the process, which includes rebuilding the FreeBSD kernel and building the KAME userland applications.

B. FreeBSD 4.9 kernel

We rebuilt the FreeBSD kernel using a modified version of the default GENERIC.KAME configuration file that came with the snap kit.

As the VIA processors used were i686 equivalents, we removed kernel support for i386, i486 and i586 processors, as it was suggested in the kernel configuration options file “LINT” that doing this would make some parts of the system run faster. We also added the kernel options “options MIP6”, “options MIP6_DEBUG”, “device acpica” and “options HZ=1000” for all machines.

The home agent (HA) was compiled with the added option “options MIP6_HOME_AGENT”.

The mobile node (MN) was compiled with the added options “options MIP6_MOBILE_NODE” and “options hid 1”.

Interestingly, the KAME source code would automatically configure the IPv6 address of the hif0 interface to use the onboard NIC’s MAC address instead of the Cisco Wireless Card’s MAC address, even though the NIC had not been activated or configured. As a result of this finding, we had to remove kernel support for the on board VIA ethernet NIC on the MN. A loadable kernel module for the VIA NIC is provided in /modules/if_vr.ko and can be explicitly loaded using the “kldload” command after the boot sequence if support for the on board NIC is required. Following this procedure will eliminate the observed problem.

C. FreeBSD rc files

We backed up /etc/rc.network6 and copied the KAME supplied version of this script and rc.mobileip6 from /<path_to_unziped_kame_snapshot>/kame/freebsd4/etc to /etc. The contents of the rc.conf files for the different machines are shown in Appendix I. Note that adding the “PATH=”...” statement is important to ensure that KAME binaries located in /usr/local/v6/{bin, sbin} are used in preference to any other system binaries with the same name.

We discovered two errors in the rc.mobileip6 file supplied as part of the snap kit. Line 101 reads: prefix='expr "$home_prefix" : ".\(.*\)::/\(.*\)"' where home_prefix contains the list of home prefixes given in the rc.conf option “ipv6_mobile_home_prefixes”. This does not function as it was supposed to and needed to be changed to: prefix='expr "$home_prefix" : ".\(.*\)\"'.

Line 103 reads: ifconfig ${ipv6_mobile_home_link} inet6 ${prefix}: fddf:ffff:ffff:fffe prefixlen 64 anycast alias

As prefix already contains the ::; the line should be changed to: ifconfig ${ipv6_mobile_home_link} inet6 ${prefix}:fddf:ffff:ffff:fffe prefixlen 64 anycast alias.

D. The Router Advertisement Daemon: rtadvd

Rtadvd is responsible for sending out router advertisements to aid in the automatic address configuration processes implemented as part of IPv6. Rtadvd sends router advertisements at 200-600 second intervals by default. For a wired, fairly static LAN, this
is completely acceptable. However, mobile nodes use router advertisements to realise that they have moved from one network to another. As such, the speed with which they get advertisements will affect the amount of time a mobile node is unreachable because of expired routing information.

As a result of this, we decreased the router advertisement time period to 4-6 seconds as recommended in the KAME newsletter dated 20031007 [14]. We created the file /usr/local/etc/rtadvd.conf and added the lines:

```
vr0:\
: maxint#6: minint#4:
```

The HA rc.mobileip6 needed line 107 which reads:
```
rtadvd -m ${ipv6_mobile_home_link} changed to:
rtadvd -c /usr/local/etc/rtadvd.conf -m ${ipv6_mobile_home_link}.
```

The FA required an executable script placed in /usr/local/etc/rc.d/<scriptname>.sh with the line:
```
rtadvd -c /usr/local/etc/rtadvd.conf -m vr0
```

added to it.

E. IPsec

Initially, we attempted to use the IP Security protocol for HA to MN tunneling and vice versa. It functioned correctly where conversation between the HA and MN was concerned. However, when an outside source attempted to ping the MN when it was not at home, the echo reply would arrive from the MN to the HA in encrypted form and then be sent to the corresponding node with the encryption still in place. This resulted in the corresponding node being unable to “see” the echo reply and therefore resulted in unsuccessful pinging.

Owing to time considerations, we did not pursue the cause of this behaviour. We suspect that IPsec needed to be configured on the corresponding node, as IPsec is supposed to be used end to end which means traffic form the MN to the CN should be encrypted as well as traffic between the HA and MN. However, we will briefly document the steps involved in getting the IPsec tunneling working, which came from a KAME newsletter article [14].

After creating the directories /usr/local/v6/etc/mobileip6 and /usr/local/v6/mobileip6/<node_name>, create a file named config in the /usr/local/v6/mobileip6/<node_name>/ directory. Appendix 2 includes a copy of our config file. Note that the hmac-sha1 authentication algorithm uses a 160 bit keylength i.e. 20 characters x 8 bits per character.

Issuing the command makeconfig.sh <node_name> should result in a number of files being created in the working directory.

Modifying the /etc/rc.conf file by replacing the line
```
ipv6_mobile_security_enable="NO"
```
with
```
ipv6_mobile_config_dir="/usr/local/v6/etc/mobileip6"
```
will result in the use of the generated IPsec files from the previous step.

F. Cisco Aironet 1200 Series Wireless Access Point

A rollover cable and RJ-45 to DB9 connector were used to connect each AP to the serial port of a host machine on the network. By using the “tip” command e.g. tip com1, were able to access the console interface of the AP and configure it from our FreeBSD host. The output of the show run command from each of the APs is given in Appendix 3.

V. Verifying the Testbed

There are a number of KAME userland utilities that are provided to configure and examine the state of IPv6 and mobile IPv6. These tools are located in /usr/local/v6/bin and /usr/local/v6/sbin.

A. Mip6control

This tool resides in /usr/local/v6/sbin and is probably the most useful for examining what the mobile IP code is actually doing. “man mip6control” will list the available command line options (although we found one option listed in the man page to have been removed from the binary, which goes to show that the documentation is often behind the code). For verifying mobile node functions, the -a and -b options show the list of home/foreign agents list and binding update list.

For verifying the functions of all other nodes, the -c option shows the binding cache list.

B. Mip6stat

This tool resides in /usr/local/v6/sbin and is useful to simply verify that mobility related packets are being sent and received by the node in question.

C. Ping6

This tool resides in /usr/local/v6/sbin and can be used to test the end to end network connectivity between two IPv6 hosts.

D. Ifconfig

This tool resides in /usr/local/v6/sbin and is used to view the local network interface configuration settings.

E. Useful tests

The most obvious question is how to trigger a mobility event to occur, such as the MN moving from the home network to the foreign network. We found the simplest way to do this was by issuing the command:
```
ifconfig an0 ssid "magicap2"
```
on the mobile e node. This resulted in a binding update event occurring once a router solicitation from the foreign router had been received.

Once the transition to the new network has been made, issuing the command ifconfig on the mobile node should show 2 changes from when the node was on the home network. Hif0 should have the mobile nodes home IPv6 address (that an0 had when the mobile node was at home) and an0 should now have the same host address as before but with the prefix of the foreign network. Mip6control -a should now show an entry for the home agent marked “Home” and an entry for the
no ip route-cache
!
ip http server
ip http help-path
  cdp timer 5
  no cdp run
  bridge 1 route ip
  !
  !
    line con 0
      password 7 13261E010803
      login
    line vty 0 4
      password 7 047802150C2E
      login
    line vty 5 15
      password 7 096F471A1A0A
      login
    !
end
APPENDIX 2

Contents of /usr/local/v6/etc/mobileip6/HA1/config:

```plaintext
mobile_node="cccc::28ff:fe46:4ec9"
home_agent="cccc::1"
transport_spi_mn_to_ha=2000
transport_spi_ha_to_mn=2001
transport_protocol=esp
transport_esp_algorithm=blowfish-cbc
transport_esp_secret="magic"
transport_auth_algorithm=hmac-sha1
transport_auth_secret="abcdefghijklmnopqrst"
tunnel_spi_mn_to_ha=2002
tunnel_spi_ha_to_mn=2003
```
tunnel_uid_mn_to_ha=2002
tunnel_uid_ha_to_mn=2003
tunnel esp_algorithm=blowfish-cbc
tunnel esp_secret="magic"
tunnel auth_algorithm=hmac-sha1
tunnel auth_secret="abcdefghijklmnopqrst"

APPENDIX 3

API show run output:

Building configuration...

Current configuration : 1662 bytes
!
version 12.2
no service pad
service timestamps debug datetime msec
service timestamps log datetime msec
service password-encryption
!
hostname magicap1
!
enable secret 5 $1$K6fT$9ZlUUnozsLVASyVh4r3Aw0
!
username Cisco password 7 047802150C2E
ip subnet-zero
!
!
bridge irb
!
!
interface Dot11Radio0
no ip address
no ip route-cache
!
ssid magicap1
  authentication open
  guest-mode
!
speed basic-6.0 9.0 basic-12.0 18.0 basic-24.0
  36.0 48.0 54.0
  rts threshold 2312
  station-role root
  bridge-group 1
  bridge-group 1 subscriber-loop-control
  bridge-group 1 block-unknown-source
  no bridge-group 1 source-learning
  no bridge-group 1 unicast-flooding
  bridge-group 1 spanning-disabled
!
interface Dot11Radio1
no ip address
no ip route-cache
shutdown
!
ssid magicap1
  authentication open
  guest-mode
!
speed basic-6.0 9.0 basic-12.0 18.0 basic-24.0
  36.0 48.0 54.0
  rts threshold 2312
  station-role root
  bridge-group 1
  bridge-group 1 subscriber-loop-control
  bridge-group 1 block-unknown-source
  no bridge-group 1 source-learning
  no bridge-group 1 unicast-flooding
  bridge-group 1 spanning-disabled
!
interface FastEthernet0
no ip address
no ip route-cache
speed 10
full-duplex
!
bridge-group 1
no bridge-group 1 source-learning
bridge-group 1 spanning-disabled
!
interface BVI1
ip address 192.168.100.1 255.255.255.0
no ip route-cache
!
ip service pad
ip subnet-zero
ip http server
ip http help-path
cdp timer 5
no cdp run
bridge 1 route ip
!
!
line con 0
password 7 13261E010803
login
line vty 0 4
password 7 13261E010803
login
line vty 5 15
password 7 062506324F41
login
!
end

AP2 show run output:

Building configuration...

Current configuration : 1657 bytes
!
version 12.2
no service pad
service timestamps debug datetime msec
service timestamps log datetime msec
service password-encryption
!
hostname magicap2
!
enable secret 5 $1$zLEF$mZs5.d.n1teyILMkzmR420
!
username Cisco password 7 00271A150754
ip subnet-zero
!
!
bridge irb
!
!
interface Dot11Radio0
no ip address
no ip route-cache
!
SSID magicap2
authentication open
guest-mode
!
speed basic-1.0 basic-2.0 basic-5.5 basic-11.0
rts threshold 2312
channel 2412
station-role root
bridge-group 1
bridge-group 1 subscriber-loop-control
bridge-group 1 block-unknown-source
no bridge-group 1 source-learning
no bridge-group 1 unicast-flooding
bridge-group 1 spanning-disabled
!
interface Dot11Radio1
no ip address
no ip route-cache
shutdown
!
SSID magicap2
authentication open
guest-mode
!
speed basic-6.0 9.0 basic-12.0 18.0 basic-24.0 36.0 48.0 54.0
rts threshold 2312
channel 2412
station-role root
bridge-group 1
bridge-group 1 subscriber-loop-control
bridge-group 1 block-unknown-source
no bridge-group 1 source-learning
no bridge-group 1 unicast-flooding
bridge-group 1 spanning-disabled
!
interface FastEthernet0
no ip address
no ip route-cache
speed 10
full-duplex
bridge-group 1
no bridge-group 1 source-learning
bridge-group 1 spanning-disabled
!
interface BVI1
ip address 10.0.0.1 255.255.255.0
no ip route-cache
!
ip http server
ip http help-path
cdp timer 5
no cdp run
bridge 1 route ip
!
!
line con 0
password 7 13261B010803
login
line vty 0 4
password 7 047802150C2E
login
line vty 5 15
password 7 096F471A1A0A
login
!
end
Appendix F

ns2 Simulation Codes
# Author: Mai Banh
# This file is to measure data only throughput (payload) when transferring 3620 Mbytes
# over a link while there is handoff interruption at frequency of 0,1,2,3,4,5,6,7
# handoffs per minute
# using FTP over a TCP connection

# Create a simulator object
set ns [new Simulator]

# Define some variables
# Interrupted duration 4.7s or disable this variable if simulating no handoff
set time_int 4.7

# Frequency of Interruption 1,2,3,4,5,6,7 per minute or disable this variable if
# simulating no handoff
set freq_int 1.00

# Time to end simulation, varies depending on handoff frequency - must transfer at least
# 3620 MBytes
set time_finish 1800.00

# Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

# Open the Trace file
set tracefile [open out.tr w]
$ns trace-all $tracefile

# Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf

# Define a 'finish' procedure
proc finish {trace} {
    puts "Traffic Volume = [set bytes]
    global ns tracefile nf
    $ns flush-trace
    #Close the Trace file
    close $tracefile
    #Close the NAM trace file
    close $nf
    #Execute NAM on the trace file
    exec nam out.nam &
    exit 0
}

# Create 2 nodes
set n0 [$ns node]
set n1 [$ns node]

# Create link between the nodes
$ns duplex-link $n0 $n1 100Mb 0.146ms DropTail

# Give node position (for NAM)
$ns duplex-link-op $n0 $n1 orient right

# Monitor the queue for link (n0-n1). (for NAM)
$ns duplex-link-op $n0 $n1 queuePos 0.5
# Trace data and count number of bytes transferred in total
Class TraceApp -superclass Application

TraceApp instproc init {args} { 
    $self set bytes_ 0.0 
    eval $self next $args 
}

TraceApp instproc recv {byte} { 
    $self instvar bytes_ 
    set bytes_ [expr $bytes_ + $byte] 
    return $bytes_ 
}

# Setup a TCP connection
set tcp [new Agent/TCP] 
$tcp set class_ 2 
$tcp set window_ 16384 
$tcp set packetSize_ 1448 \ # MSS is 1448 
$tcp set timestamps_ true \ # Trigger TCP timestamp option 
$tcp set ts_option_size_ 12 \ # Set TCP timestamp option to 12 bytes 

$ns attach-agent $n0 $tcp 

# Setup a FTP over TCP connection
set ftp [new Application/FTP] 
$ftp attach-agent $tcp 
$ftp set type_ FTP 

#Trace data transferred over the link at TCP/IP layer
set sink [new Agent/TCPSink] 
set traceapp [new TraceApp] 
$traceapp attach-agent $sink 

$ns attach-agent $n1 $sink 
$ns connect $tcp $sink 
$tcp set fid_ 1 

#Schedule events for the FTP agents
$ns at 0.00 "$traceapp start" 
$ns at 0.00 "$ftp start" 

#Set up handoff interruption for link n0 and n1 at frequency of freq_int 
for {set i 0} {$i < 245} {incr i} { 
    set t_down [expr 60/$freq_int/2+60.0/$freq_int*$i] 
    set t_up [expr $t_down + $time_int] 
    $ns rtmodel-at $t_down down $n0 $n1 
    $ns rtmodel-at $t_up up $n0 $n1 
}

# 3620 Mbytes (81920 buffers of 46336 Bytes each) 
# 3620Mbytes = 3795845120 bytes – amount of data to be sent 

#Tell FTP to stop transfer of data at a specified time
$ns at time_finish "$ftp stop" 

#Detach tcp and sink agents (not really necessary) 
$ns at time_finish "$ns detach-agent $n0 $tcp ; $ns detach-agent $n1 $sink" 

#Call the finish procedure after simulation time
$ns at time_finish "finish $traceapp"
# Author Mai Banh
# This file is written in Perl script. It scans through the ns trace file resulted from
# ns_thput_script.tcl to determine the time at which the node receives 3620 Mbytes
# (3795845120 bytes)
# type: perl throughput.pl <trace_file> <required_node> <granularity> <output_file>

$infl=$ARGV[0];

# We compute how many bytes were transmitted during time interval specified
# by granularity parameter in seconds
# Define sum (used to accumulate the total payload data)
# – to identify when the node receives 3620 Mbytes
$sum=0;

# Open the trace file
open (DATA,"<$infl")
  || die "Can't open $infl $!";

while (<DATA>)
{
  @x = split(' ');

  if ($sum >= 3795845108)   #=(3795845120-12) since the first TCP SYN has
    # a header of only 40 bytes
  {
    $time = $x[1];    # The second column of trace file is timestamp
    print STDOUT "$time 
";  # which is the transfer time for 3620 Mbytes
    $throughput = 3795845120*8/($time)/1000000;
    print STDOUT "Throughput = $throughput Mbits/sec
";
    close DATA;
    exit (0);
  }
  else
  {
    #checking if the event corresponds to a reception
    if ($x[0] eq 'r')
    {
      #checking if the packet type is TCP
      if ($x[4] eq 'tcp')
      {
        $sum=$sum+$x[5]-52; # The total payload transferred (not including
        # 52 byte TCP/IP header in each packet)
        # (and allowing for the first 40 byte TCP SYN)
      }
      }
# List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACK</td>
<td>Acknowledgement</td>
</tr>
<tr>
<td>AH</td>
<td>Authentication Header</td>
</tr>
<tr>
<td>AP</td>
<td>Access Point</td>
</tr>
<tr>
<td>AR</td>
<td>Access Router</td>
</tr>
<tr>
<td>ARP</td>
<td>Address Resolution Protocol</td>
</tr>
<tr>
<td>BSS</td>
<td>Basic Service Set</td>
</tr>
<tr>
<td>CA</td>
<td>Collision Avoidance</td>
</tr>
<tr>
<td>Care-of init cookie</td>
<td>A cookie sent to the Correspondent Node</td>
</tr>
<tr>
<td>Care-of keygen token</td>
<td>A keygen token sent by the Correspondent Node in the Care-of Test message.</td>
</tr>
<tr>
<td>CCOA</td>
<td>Co-located Care-of Address</td>
</tr>
<tr>
<td>CN</td>
<td>Correspondent Node</td>
</tr>
<tr>
<td>CoA</td>
<td>Care-of Address</td>
</tr>
<tr>
<td>Cookie</td>
<td>A random number</td>
</tr>
<tr>
<td>CSMA</td>
<td>Carrier Sense Multiple Access</td>
</tr>
<tr>
<td>CTS</td>
<td>Clear To Send</td>
</tr>
<tr>
<td>DAD</td>
<td>Duplicate Address Detection</td>
</tr>
<tr>
<td>DCF</td>
<td>Distributed Coordination Function</td>
</tr>
<tr>
<td>DHCP</td>
<td>Dynamic Host Configuration Protocol</td>
</tr>
<tr>
<td>DiffServ</td>
<td>Differentiated Service</td>
</tr>
<tr>
<td>DIFS</td>
<td>Distributed InterFrame Space</td>
</tr>
<tr>
<td>DNA</td>
<td>Detecting Network Attachment</td>
</tr>
<tr>
<td>DoS</td>
<td>Denial of Service</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>EDGE</td>
<td>Enhanced Data for Global Evolution</td>
</tr>
<tr>
<td>ESP</td>
<td>Encapsulated Security Payload</td>
</tr>
<tr>
<td>ESS</td>
<td>Extended Service Set</td>
</tr>
<tr>
<td>ESS</td>
<td>Extended Service Set</td>
</tr>
<tr>
<td>ESSID</td>
<td>Extended Service Set Identifier</td>
</tr>
<tr>
<td>FA</td>
<td>Foreign Agent</td>
</tr>
<tr>
<td>FMIPv6</td>
<td>Mobile IPv6 Fast Handoff</td>
</tr>
<tr>
<td>HA</td>
<td>Home Agent</td>
</tr>
<tr>
<td>HMIPv6</td>
<td>Hierarchical Mobile IPv6</td>
</tr>
<tr>
<td>Home address</td>
<td>A unicast routable address assigned to a Mobile Node</td>
</tr>
<tr>
<td>Home init cookie</td>
<td>A cookie sent to the correspondent node in the Home Test Init message, to be returned in the Home Test message.</td>
</tr>
<tr>
<td>Home keygen token</td>
<td>A keygen token sent by the correspondent node in the Home Test message.</td>
</tr>
<tr>
<td>Home link</td>
<td>The link on which a mobile node's home subnet prefix is defined.</td>
</tr>
<tr>
<td>Home registration</td>
<td>A registration between the mobile node and its home agent, authorized by the use of IPsec.</td>
</tr>
<tr>
<td>Home subnet prefix</td>
<td>The IP subnet prefix corresponding to a mobile node's home address.</td>
</tr>
<tr>
<td>IBSS</td>
<td>Independent Basic Service Set</td>
</tr>
<tr>
<td>ICMP</td>
<td>Internet Control Message Protocol</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>IKE</td>
<td>Internet Key Exchange</td>
</tr>
<tr>
<td>Interface ID</td>
<td>Interface Identifier</td>
</tr>
<tr>
<td>IntServ</td>
<td>Integrated Service</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IPIP</td>
<td>IP-within-IP (encapsulation)</td>
</tr>
<tr>
<td>IPSec</td>
<td>IP Security</td>
</tr>
<tr>
<td>IPv4</td>
<td>IP version 4</td>
</tr>
<tr>
<td>IPv6</td>
<td>IP version 6</td>
</tr>
<tr>
<td>IrDA</td>
<td>Infrared Data</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider</td>
</tr>
<tr>
<td>Term</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>KAME</td>
<td>Mobile IPv6 implementation for FreeBSD</td>
</tr>
<tr>
<td>Kbm</td>
<td>A binding management key (Kbm)</td>
</tr>
<tr>
<td>Keygen token</td>
<td>A keygen token is a number supplied by a correspondent node</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>L2</td>
<td>Layer 2</td>
</tr>
<tr>
<td>L2 handoff</td>
<td>Layer 2 handoff</td>
</tr>
<tr>
<td>L2 trigger</td>
<td>Information from L2 that informs L3 of particular events before and after L2 handoff.</td>
</tr>
<tr>
<td>L3</td>
<td>Layer 3</td>
</tr>
<tr>
<td>L3 handoff</td>
<td>Layer 3 handoff</td>
</tr>
<tr>
<td>LEAP</td>
<td>Lightweight Extensible Authentication Protocol</td>
</tr>
<tr>
<td>MAC</td>
<td>Media Access Control</td>
</tr>
<tr>
<td>MAP</td>
<td>Mobility Anchor Point</td>
</tr>
<tr>
<td>MIPL</td>
<td>Mobile Implementation for Linux (for Mobile IPv6)</td>
</tr>
<tr>
<td>MN</td>
<td>Mobile Node</td>
</tr>
<tr>
<td>Mobile IPv4</td>
<td>Mobility Support in IPv4</td>
</tr>
<tr>
<td>Mobile IPv6</td>
<td>Mobility Support in IPv6</td>
</tr>
<tr>
<td>Mobile-initiated handoff</td>
<td>L3 handoff in which the Mobile Node initiates the handoff</td>
</tr>
<tr>
<td>Mobility Message</td>
<td>A message containing a Mobility Header</td>
</tr>
<tr>
<td>Movement</td>
<td>A change in a Mobile Node’s point of attachment to the Internet</td>
</tr>
<tr>
<td>MTU</td>
<td>Maximum Transmission Unit</td>
</tr>
<tr>
<td>NA</td>
<td>Neighbor Advertisement</td>
</tr>
<tr>
<td>nAR</td>
<td>New Access Router</td>
</tr>
<tr>
<td>NAT</td>
<td>Network Address Translation</td>
</tr>
<tr>
<td>nCoA</td>
<td>New Care-of Address</td>
</tr>
<tr>
<td>NEMO</td>
<td>Network Mobility</td>
</tr>
<tr>
<td>Network-initiated handoff</td>
<td>L3 handoff in which oFA or nFA initiates the handoff.</td>
</tr>
<tr>
<td>nFA</td>
<td>New Foreign Agent</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Nonce</td>
<td>Nonces are random numbers used internally by the correspondent node</td>
</tr>
<tr>
<td>Nonce index</td>
<td>A nonce index is used to indicate which nonces have been used</td>
</tr>
<tr>
<td>NS</td>
<td>Neighbor Solicitation</td>
</tr>
<tr>
<td>NUD</td>
<td>Neighbor Unreachability Detection</td>
</tr>
<tr>
<td>oAR</td>
<td>Old Access Router</td>
</tr>
<tr>
<td>oCoA</td>
<td>Old (previous) Care-of Address</td>
</tr>
<tr>
<td>oFA</td>
<td>Old Foreign Agent</td>
</tr>
<tr>
<td>PCF</td>
<td>Point Coordination Function</td>
</tr>
<tr>
<td>Ping-ponging</td>
<td>Rapid back and forth movement between two points</td>
</tr>
<tr>
<td>RA</td>
<td>Router Advertisement</td>
</tr>
<tr>
<td>RFC</td>
<td>Request For Comments</td>
</tr>
<tr>
<td>RR</td>
<td>Return Routability</td>
</tr>
<tr>
<td>RS</td>
<td>Router Solicitation</td>
</tr>
<tr>
<td>RSS</td>
<td>Received Signal Strength</td>
</tr>
<tr>
<td>RSVP</td>
<td>Resource Reservation Protocol</td>
</tr>
<tr>
<td>RTS</td>
<td>Request To Send</td>
</tr>
<tr>
<td>Seamless handoff</td>
<td>L3 handoff that is both low latency and low loss</td>
</tr>
<tr>
<td>SIFS</td>
<td>Short InterFrame Space</td>
</tr>
<tr>
<td>SSID</td>
<td>Standard Service Set</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>ToS</td>
<td>Type of Service</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>VoIP</td>
<td>Voice over IP</td>
</tr>
<tr>
<td>VPN</td>
<td>Virtual Private Network</td>
</tr>
<tr>
<td>WEP</td>
<td>Wired Equivalent Privacy</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
<tr>
<td>WPA</td>
<td>Wi-Fi Protected Access</td>
</tr>
</tbody>
</table>

152
Bibliography


[FLYV93] V. Fuller, T. Li, J. Yu, and K. Varadhan, “Classless Inter-Domain Routing (CIDR) an


[WEN04] R. Wakikawa, T. Ernst, and K. Nagami, “Multiple Care-of Addresses Registration”, IETF Internet Draft, draft-wakikawa-mobileip-multiplecoa-03.txt, June 2004, expired

