

بسم الله الرحمن الرحيم



**Sudan University of Science and Technology**  
**College of Postgraduate Studies**

**Hybrid Algorithm for Distributed Mobility Management**

**خوارزمية هجينة لإدارة التنقل الموزعة**

A Thesis submitted in fulfilment of the academic requirements for the  
degree of Doctor of Philosophy of Science

By:

TajElsir Hassan Suliman Hammad

Supervised By:

Prof. Altayeb Salih Abuelyaman

Nov, 2015

## **DECLARATION**

I hereby declare that: (1) The above thesis is my own unaided work, both in conception and execution, and that apart from the normal guidance of my supervisor, I have received no assistance apart from that stated below;(2) Except as stated below, neither the substance or any part of the thesis, has been submitted in the past, or is being, or is to be submitted for a degree in the University, or any other University.

I am now presenting the thesis for examination for the Degree of PhD in Computer Science. I also grant the University free license to reproduce the above thesis in whole or in part, for the purpose of research.

TajElsir Hassan Suliman


Nov, 27, 2015

Name

Date

As the candidate's supervisor, I have approved this dissertation for submission.

Name: Professor: Altayeb Salih Abuelyaman

Signed: 

Date: 27 Nov, 2015

آية

قال الله تعالى : "وَمَا أُوتِيتُمْ مِنَ الْعِلْمِ إِلَّا قَلِيلًا"

سورة الإسراء الآية (85)

## **DEDICATION**

To the soul of my father-may Allah forgive him

To my lovely mother (Hassona Faddallah)

To my lovely wife (Asma Kubbara) and my sons (Mohammed, Retaj and Muhana)

# ACKNOWLEDGEMENTS

First and foremost, I thank Allah for his blessings and provision.

Second, I wish to express my sincere gratitude to my family for their generous and unconditional support and love.

Third, I would like to thank my supervisor Professor Altayeb Salih Abuelyaman for his direction and continuous encouragement to improve the quality of my research.

Fourth, I am deeply grateful to my best friend Dr. Muhana Magboul. Muslam for his advice to redirect my research to the area of Distributed Mobility Management (DMM).

Fifth, I would like to thank Dr. Rashid Abdulhaleem-Deputy Dean of Scientific Research Department, Sudan University, and my true friend Dr. Adil Yousif, University of Technology, for their encouragement, consistent guidance, and support during this research work.

Sixth, I would like to express my gratitude to my colleagues at PhD lab (first batch) University of Sudan, for their initiative and neat comments.

Finally, I wish to express my deepest gratitude to my brothers and sisters for their prayers and ongoing support throughout the years.

# ABSTRACT

The Internet was originally designed for stationary (static) nodes (i.e. computers, laptops). With the advancement of mobile nodes (such as smartphones and tablets) that have wireless internet access capability, the original design of the Internet is no longer sufficient. These mobile nodes are capable of communicating while moving and changing their point of attachment (roaming) in the Internet. To maintain communication session(s) continuity for these mobile nodes, the Internet needs mobility management mechanisms.

The main mobility management protocols standardized by the Internet Engineering Task Force (IETF) are mobile IP (MIPv6), proxy MIP (PMIPv6) and Host Identity Protocol (HIP). The architectural structures of these protocols employ a centralized mobility anchor to manage the mobility of the mobile nodes in the control and data planes, respectively, however, these protocols have many limitations such as single point of failure, triangular routing problem, long handover latency and packet loss.

IETF engineers claimed that these problems can be addressed by moving to a flat architecture, adopting a Distributed Mobility Management (DMM) system, where the centralized anchor is removed and the mobility management functions are distributed to different networks elements, brought to the edge of the networks, which is closer to the mobile nodes(MN). However, to date, mobility management schemes that have been developed based on the DMM concepts are still in the preliminary stages and inherits the same problems from the centralized approach and there is no current standard in place.

The thesis proposes a novel hybrid algorithm completely built on Host Identity Protocol (HIP), hereafter called (HADMM) that combines two of leading contenders schemes proposed in [47] and [51] as an enhanced scheme proposed in [50], respectively to work in a distributed manner, in order to overcome their problems and limitations.

OMNeT++ ver 4.0 and HIPSim++ simulators are used to model HADMM to evaluate the handover performance, measuring the handover delay, packet loss and signaling overhead, the simulation results demonstrated that HADMM has better handover performance compared to aforementioned scheme [47].

Analytical analysis evidenced that HADMM has addressed all the limitations experienced by the scheme proposed in [51]. To the author's best knowledge HADMM is the first DMM algorithm completely built on HIP layer.

## المستخلص

شبكة الإنترنت صُممت أصلاً لخدمة النقاط الثابتة (static nodes) مثل أجهزة الحواسيب الشخصية والحواسيب المحمولة ولكن مع ظهور النقاط المتنقلة (Mobile Nodes) مثل (الهواتف الذكية، أجهزة اللوحات) التي لديها إمكانية الوصول إلى شبكة الإنترنت لاسلكياً أصبحت معمارية الإنترنت الحالية غير قادرة وغير مواكبة هذه النقاط المتنقلة قادرة على التواصل مع بعضها البعض أثناء التجوال والانتقال من شبكة لأخرى داخل شبكة الإنترنت وللحفاظ على إستمرارية الجلسات المفتوحة لهذه الأجهزة المتنقلة تحتاج شبكة الإنترنت لآليات إدارة التنقل.

هنالك بروتوكولات إدارة التنقل القياسية والتي تم إعتماها بواسطة منظمة (Internet Engineering Task Force-IETF) وهي منظمة تطوعية معنية بتطوير أنظمة إدارة التنقل وحل مشاكل عنونة الأجهزة على شبكة الإنترنت، أمثلة لهذه البرتوكولات (MIPv6) ، (PMIPv6) و (HIP) وكلها تعتمد في معماريتها على نقطة ربط (إرساء) مركزية لإدارة تنقل النقاط المتنقلة مهمتها التحكم في معرفة المواقع الحالية لهذه النقاط والتحكم في إرسال وإستقبال البيانات الخاصة بها على شبكة الإنترنت.

مهندسو منظمة (IETF) يرون أنه يمكن معالجة هذه المشاكل بالانتقال إلى المعمارية المسطحة (flat) والتي تعتمد على نظم إدارة تنقل موزعة تعرف ب (Distributed Mobility Management -DMM) ففي هذا النموذج يتم إزالة النقطة المركزية لإدارة التنقل ويتم توزيع نقاط الإرساء أو توزيع وظائفها إلى عناصر أخرى في الشبكة مثل الموجهات (Routers) والتي تقع في حواف (edges) الشبكة وتكون أقرب إلى النقاط المتنقلة، حتى اليوم كل الحلول المقترحة لإدارة التنقل اللامركزية (الموزعة) والتي تم بناؤها على حسب نهج DMM لا تزال في مراحلها الأولية وورثت نفس مشاكل المعمارية المركزية ولا توجد مواصفة قياسية (standard) حتى الآن. هذه الأطروحة قدمت خوارزمية (Algorithm) هجينة لإدارة التنقل الموزعة مبنية بصورة كلية على البرتوكول (HIP) وتُعرف إختصاراً ب (HADMM) وتقوم بدمج طريقتين فعاليتين تمت الإشارة إليهما في قائمة المراجع بالأرقام (47) و (51) والتي تعتبر نسخة محسنة من الطريقة المقترحة في [50] للعمل بنهج DMM والتخلص من مشاكل الطريقتين.

تم استخدام محاكي الشبكات (OMNeT++ v. 4.0) و (HIPSim++) لتصميم وإختبار معمارية HADMM لتقييم أدائها وذلك من خلال قياس كل من زمن تأخر الانتقال من شبكة لأخرى، فقدان حزم البيانات وتقليل إشارات التحكم وتبين من خلال نتائج المحاكاة أن HADMM تتفوق على الطريقة المشار إليها بالرقم (47) وعند تحليل نقاط ضعف الطريقة المشار إليها بالرقم (51) تبين أن HADMM تعالج جميع مشاكلها. حسب معرفة الباحث فإن (HADMM) تعتبر أول خوارزمية لإدارة التنقل الموزعة والمبنية بصورة كلية على طبقة البرتوكول القياسي (HIP).



# TABLE OF CONTENTS

# PAGE NO

Hybrid Algorithm for Distributed Mobility Management (HADMM).....	i
Declaration.....	ii
Ayah of Al Quran .....	iv
Dedication .....	v
Acknowledgment.....	vi
Abstract (English) .....	vii
المستخلص .....	viii
Table of Contents .....	ix-xi
List of Tables .....	xii
List of Figures .....	xiii-xiv
List of Sybmols/Abbreviations.....	xv-xvi
List of Appendices.....	xvii
Publications .....	xviii

## Chapter One: Introduction

1.1 Background .....	1-4
1.2 Problem Statement .....	4
1.3 Research Questions .....	5
1.4 Research Aim .....	5
1.5 Research Objectives .....	5
1.6 Scope of the Research .....	6
1.7 Thesis Outline .....	6

## **Chapter Two: Literature Review**

2.1 Overview.....	7
2.2 Host-based Distributed Mobility Management .....	7-9
2.3 Network-based Distributed Mobility Management .....	9-10
2.4 Comparison of Distributed Mobility Management Schemes .....	10-11
2.5 Final Remarks. ....	11

## **Chapter Three: Host-Based DMM (HDMM-TH)**

3.1 Introduction .....	12
3.2 Motivation and Design Approaches for HDMM-TH.....	12-13
3.3 Related Work.....	13
3.3.1 Architecture Overview.....	13
3.3.2 Limitations of the approach.....	14
3.4 Architecture of the Proposed HDMM-TH Scheme.. ..	15
3.4.1 Architecture Overview .....	15-16
3.4.2 Mobile Initial Registration and Data Delivery Procedure.....	16-17
3.4.3 Handover and Packet Delivery Mechanism for HDMM-TH.....	17-18
3.5 Performance Analysis.....	19-24
3.5.1 Numerical Results Analysis and Discussions .....	23-24
4.6 Summary .....	24-25

## **Chapter Four: Hybrid Algorithm for DMM (HADMM)**

4.1 Introduction and Motivation .....	26
4.2 Related Work.....	27-28
4.3 HADMM Architecture Overview .....	29-32
4.3.1 Initial MN Registration and Reachability .....	30-32

4.3.2 Establishing Security Association .....	32
4.3.3 Handover Mechanism.....	33-38
4.3.3.1 HADMM Intra-Domain Handover.....	33-35
4.3.3.2 HADMM Inter-Domain Handover.....	35-38
<b>Chapter Five: Proposed Algorithm (HADMM) Results and Discussions</b>	
5.1 Performance Analysis .....	39-49
5.1.1 HADMM Performance Analysis (Intra-Handover).....	40-47
5.1.1.1 Analytic evaluation of handover performance .....	40-42
5.1.1.2 Simulation and Results .....	43-47
5.1.1.2.1 Motivations for selecting OMneT++ Simulator.....	43
5.1.1.2.2 Simulation Setup .....	43
5.1.1.2.3 Simulation Results and Discussions.....	45-47
5.1.2 HADMM and HDMM-TH Performance Analysis (Inter-Handover).....	47-49
5.1.2.1 HADMM and HDMM-TH Analytic evaluation.....	47-49
5.2 Summary .....	49
<b>Chapter Six: Conclusion and Future Work</b>	
6.1 Conclusion.....	50-51
6.2 Future Work .....	51-52
6.3 Main Contributions.....	53
<b>Appendices (A, B,C, D) .....</b>	<b>54-82</b>
<b>References.....</b>	<b>83-91</b>

## List of Tables

## Page No.

Table 2.1 Comparison between Host-based and Network-based DMM schemes .....	11
Table 3.1 Formulas used for quantitative analysis.....	19
Table 3.2 Parameters for quantitative analysis.....	19-20
Table 3.3 Calculations of $D_{REG}$ , $S_{REG}$ and $T_{INT}$ for the three schemes.....	20
Table 3.4 Quantitative Analysis for the approaches (MIPv6, Ref [50], HDMM-TH).....	21
Table 5.1 Signalling overheads of HADMM, MHPP and PMIPv6 for intra/inter-HO.....	42
Table 5.1 Simulation Parameters under which MHPP, HADMM and DMM-LFH are examined.....	45

<b>LIST OF FIGURES</b>	<b>PAGE NO.</b>
Figure 1.1 Cisco forecast on Mobile data traffic growth.....	3
Figure 1.2 Cisco forecast on global growth of smart phones.....	3
Figure 3.1 Mobility Management in the Chan DMM approach .....	14
Figure 3.2 Mobility Management in the proposed (HDMM-TH).....	15
Figure 3.3 Registration and Packet Delivery.....	17
Figure 3.4 Registration Delay ( $D_{REG}$ ) vs. number of hops between MN and HA.....	22
Figure 3.5 Signaling Overhead ( $S_{REG}$ ) vs. Resident Time ( $n=5$ ).....	22
Figure 3.6 Traffic Intensity ( $T_{INT}$ ) vs. number of ongoing communications.....	23
Figure 4.1 MHPP Architecture.....	28
Figure 4.2 HADMM Architecture.....	29
Figure 4.3 HIP MN initial registration.....	31
Figure 4.4 Non-HIP MN initial registration.....	32
Figure 4.5 Non HIP MN intra-handover procedure.....	33
Figure 4.6 HIP MN inter-handover.....	35
Figure 4.7 Non-HIP MN inter-handover.....	37
Figure 5.1 Performance evaluation diagram.....	39
Figure 5.2 HADMM simulation topology .....	44
Figure 5.3 the first 30 handoffs for MHPP, HADMM and PMIPv6.....	46
Figure 5.4 handover-related messages of HADMM, MHPP and PMIPv6.....	46
Figure 5.5 the average packet loss of HADMM, MHPP and PMIPv6.....	47

# LIST OF SYMBOLS/ABBREVIATIONS

AR	Access Router
BA	Binding Acknowledgement
BU	Binding Update
BUL	Binding Update List
CMAG	Control Mobile Access Gateway
CN	Correspondent Node
CoA	Care-of-Address
D-PMIP	Distributed PMIP
DF-PMIP	Fully Distributed PMIP
DHCPv6	Dynamic Host Configuration Protocol for IPv6
DMM	Distributed Mobility Management
FMIPv6	Fast Mobile IP version 6
GW	Gateway Router
HA	Home Agent
FA	Foreign Agent
HIP	Host Identity Protocol
HMIPv6	Hierarchical MIPv6
HNP	Home Network Prefix
HoA	Home Address
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPv6	Internet Protocol Version 6
LMA	Local Mobility Anchor
MAG	Mobile Access Gateway
MAP	Mobile Anchor Point
MIPv6	Mobile IP version 6
MN	Mobile Node

MN-HoA MN Home Address

MN-ID MN Identifier

MNP Mobile Network Prefix

MR Mobile Router

OMneT++ OMNET Network Simulator Version 4.0

H-DMM-TH Host-based DMM with Tunneling-less Handover

CMA Control Mobility Anchor

EXT-UPDATE Extended Update Packet

HADMM Hybrid Algorithm for Distributed Mobility Management

RVS Rendezvous Server

LRVS Local Rendezvous Server

PMIPv6 Proxy Mobile IP version 6

PBA Proxy binding Acknowledgement

PBU Proxy binding update

PoA Point of Attachment

RA Router Advertisement

HOL Handover Latency

3G Third generation mobile communication networks

# **LIST OF APPENDICES**

# **PAGE NO.**

Appendix A: Extension Header (EH) of IPv6 Packet format.....	54
Appendix B: HIP Base Exchange Packets.....	55
Appendix C: Screenshots of HADMM .....	56-57
Appendix C: HADMM implementation Codes.....	58-82



## **PUBLICATIONS**

Selected peer-reviewed papers are published from this thesis listed below:

1. TajElsir H. Suliman, Altayeb S. Abuelyaman, "Distributed Mobility Management Scheme using Lightweight Fast Handover (DMM-LFH)", International Journal of Scientific & Engineering Research(IJSER), Volume 4, Issue 10, October-2013.
2. TajElsir H. Suliman, Altayeb S. Abuelyaman, "Host Based Distributed Mobility Management Scheme using Tunneling-less Handover (H-DMM-TH)", Proceedings of 11<sup>th</sup> IEEE APW2014 Conference, Taiwan, 28-29 August, 2014
3. TajElsir H. Suliman, Altayeb S. Abuelyaman, "Hybrid Algorithm for Distributed Mobility Management (HADMM)", (In Press).
- 4- TajElsir H. Suliman , Muhana A. Magboul, Rashid Abdulhalim, "HIP-based DMM", (in progress).

# CHAPTER ONE

## INTRODUCTION

### 1.1 Background

Mobile connectivity is now far from being a luxury service. Users demand Internet access while on the move, and the volume of traffic generated by mobile subscribers has been exponentially increasing during the last few years, it is expected that the data will grow by 24.3 Exabytes (1000 Petabytes) per month by 2019, nearly tenfold increase over 2014 [1], as shown in Figure 1-1. Furthermore, the number of mobile devices (Smartphone Devices) is expected to grow by 6 billion by 2019, nearly threefold increase over 2014[1], as shown in Figure 1-2.

This has been motivated by the incredible success on the development and wider introduction in the market of smart-phones, tablets and netbooks, such as Android, iOS or Windows Phone 8 based terminals which have changed not only the way users consume data services, but also the place where they do it from. Along with this, the number of available mobile applications has also exploded. Many of these applications benefit today from the use of Internet connectivity and cloud hosted functionality.

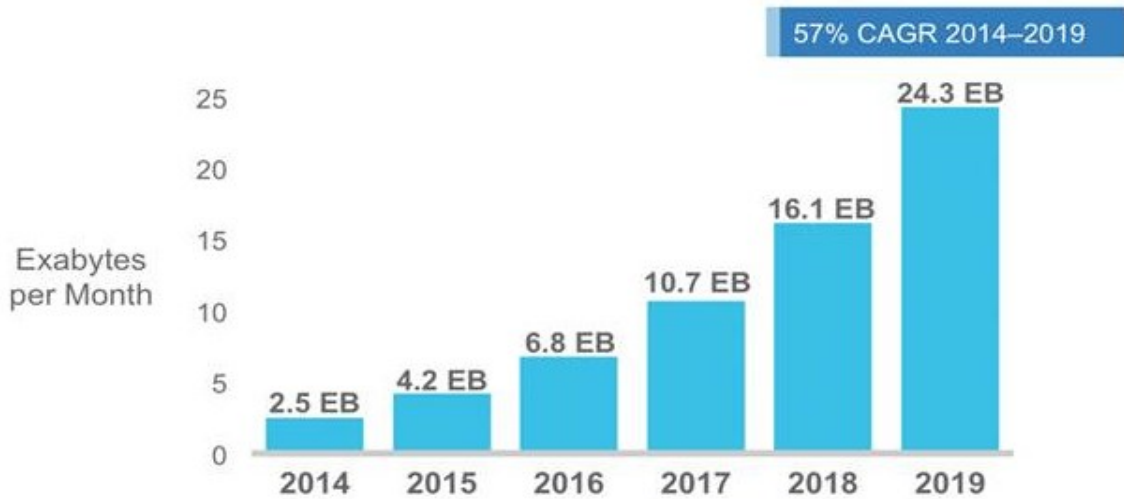
As a consequence of this paradigm shift, mobile network operators are witnessing how their networks need to cope with an increasing volume of data, saturating their access links, and triggering the need for additional access technologies to be made available to the users. As radio accesses with more capacity are deployed, and operators migrate their networks to full IP based architectures, such as the WiMAX [2] related standards or the 3GPP Evolved Packet System (EPS)[3] the load will spread between the different access networks. However, currently deployed network architectures assume that all traffic requires mobility support, which causes every packet to traverse the operator's core, leading to fall into triangular routing problem (non-optimal); and this can negatively impact the routing efficiency. Additionally, the anchor manages the mobility signaling for all MNs. Consequently, as the number of mobile nodes increases and the data traffic volume grows, the mobility anchor could find it difficult to efficiently handle such increases—due to scalability and reliability problems related to the centralized route and the mobility context management [21].

Furthermore, the centralized mobility management approaches cannot satisfactorily support the mobility in flat network architecture. This is due to limitations that include a single point of failure, traffic bottleneck, non-optimal routing path, scalability problems and long handover. Because of the new requirements imposed by mobile users' traffic operators with a large number of mobile subscribers are now looking for alternative mobility solutions that are more distributed in nature, allowing cheaper and more efficient network deployments capable of meeting their customer requirements.

In order to accommodate the data traffic growth, and to relieve the traffic load from the mobile core network, the network operators have currently adopted mechanisms, such as:

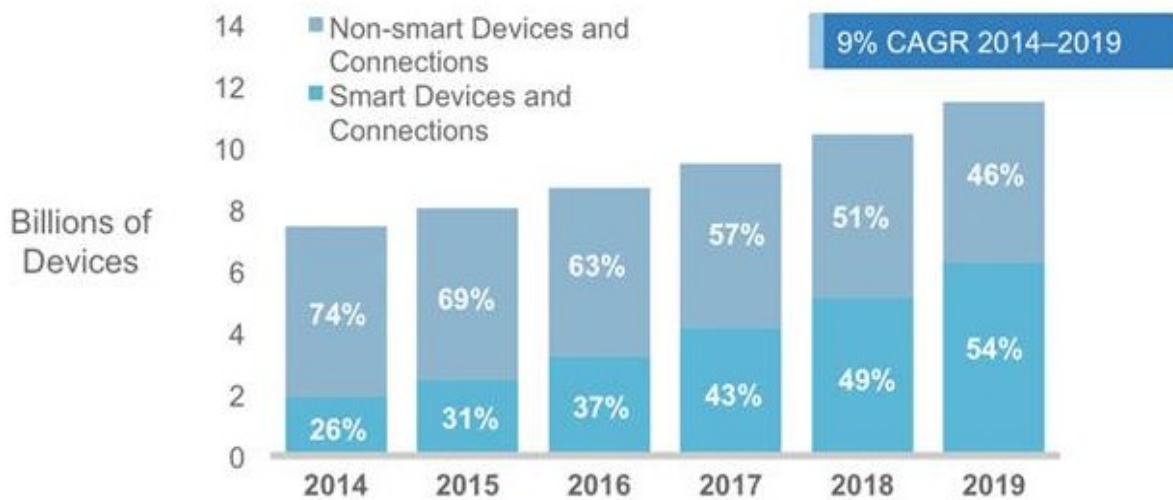
- (i) Expanding the network capacity: deploying dense cells like femto- and picocells and spectrum efficient technologies, i.e., 3GPP Long Term Evolution (LTE) and WiMAX;
- (ii) Traffic off-loading: adopting the off-loading of mobile data traffic through WiFi networks and using selective traffic off load mechanisms, such a selective IP traffic off load (SIPTO) and Local IP address (LIPA) [9]; and
- (iii) Upgrading the equipment of the core network nodes.

However, although expanding the network capacity increases the throughput of the radio access, it may lead to the development of new usages, and hence fuel the traffic growth. And while the off-loading techniques remove the traffic overload from the mobile core network, they offer limited mobility support within small localized regions [10]. Upgrading the core network is an easy approach, which seems technically and technologically possible. But mobile operators' average revenue per user (ARPU) is getting lower; and such an approach is not an economically feasible solution [11]. All these solutions are effective only in some scenarios; and they cannot effectively address the mobility demands in the mobile Internet.



Source: Cisco VNI Mobile, 2015

Figure 1.1 Cisco forecasts on mobile data traffic growth [4]



Percentages refer to device and connections share.

Source: Cisco VNI Mobile, 2015

Figure 1.2 Cisco forecasts on global growth of smart mobile devices and connections [4]

In order to overcome the shortcomings caused by the deployment of a centralized mobility anchor in mobility management, and to effectively support mobile users, the IETF has proposed a new paradigm for IP mobility management development—Distributed Mobility Management (DMM) [7]. The main concept behind DMM is to provide mobility support without relying on a centrally deployed mobility anchor. The mobility anchor as a whole, and/or mobility management functions, are distributed to different networks/elements, resulting in the mobility anchor or the mobility

management functions being brought closer to the MNs, for example, closer to the edge of the network.

Moreover, the traffic is anchored at different mobility anchors, so that the scalability and routing inefficiency issues are reduced. Furthermore, a single point of failure is eliminated. Also, the mobility support may be activated or deactivated, depending on whether the MN's session needs mobility support, or not [8]. This also reduces the amount of state information that must be maintained in various mobility agents of the mobile network.

To date, there is no standardized protocol for DMM, but various proposals are available from the IETF and the research community. These proposals will be discussed later in this thesis.

## **Significance of the Study:**

This study will help in bridging the gap between host-based and network-based distributed mobility schemes by introducing a novel hybrid algorithm that benefits from all Host Identity Protocol (HIP) features to provide network-based mobility services.

Since DMM is considered a young discipline, standard models and protocols are not standardized yet.

## **1.2 Problem Statement**

Literature review has shown that previous schemes have attempted to provide distributed mobility management (DMM) through extending MIPv6 and PMIPv6 to work in a distributed way. However, these schemes still incur some drawbacks, which include a long routing path, resulting from the MN's session(s) remaining anchored at the MN's communicating IP address-anchor point, single point of failure, a lack of route optimization for ongoing communication, and long end-to-end packet delivery delay especially for long-duration traffic.

Moreover, signaling and tunneling overhead over the wireless link still exists especially, in the host-based mobility management schemes, lack of elegant security

mechanism to protect the exchanged messages between the communications parties to resist the denial of service( DOS) and man-in-the middle attacks.

Thus, a new DMM algorithm is required to address these issues. The research presented in this thesis therefore, investigates two of cited schemes proposed in [47] and [50] which enhanced by the author and published in [51], respectively, and it develops a new hybrid algorithm that overcomes some of aforementioned limitations in an efficient way.

### **1.3 Research Questions:**

The following research questions are formulated:

- 1- How do can we overcome some of limitations experience by the selected schemes by extending the current standard Host Identity Protocol (HIP) to work in a distributed manner?
- 2- How do we can provide security for the proposed hybrid algorithm relying on HIP protocol?

These research questions are analyzed and answered in in section 6.3 in chapter six (main contributions).

### **1.4 Research Aim**

This research aims to overcome some of the limitations experienced by the current distributed mobility management schemes through introducing a hybrid algorithm based on HIP protocol.

### **1.5 Research Objectives:**

In order to achieve the aim of the research, the objectives of this research are summarized as follows:

1. To critically review, analyze and evaluate the existing IP mobility management schemes in terms of mobility function deployment;
2. To develop a global and secure hybrid algorithm for distributed mobility management fully based on Host Identity Protocol (HIP) that benefits from all HIP features as self-certifying, security, mobility, multi-homing and IPv4 and IPv6 interoperability.
3. To evaluate the performance of the developed hybrid algorithm, and to compare the performance improvement with leading counter parts using analytical analysis and simulation methods.

## **1.6 Scope of the Research**

This research focuses on enhancing two of cited schemes proposed in [47] and [50] which enhanced by the author and published in [51], respectively to function in a distributed manner. The research proposes a novel hybrid algorithm to overcome the limitations found in the mentioned contenders. Moreover, the proposed algorithm addresses the long handover latency, data packets loss, signaling overhead, single point of failure and security weaknesses experienced by the cited schemes. The performance of the proposed algorithm is evaluated by means of analytical analysis and simulations (using OMNeT++ v4.0 and HIPSim++) against leading counter parts developed by the research community. The proposed hybrid algorithm is based on HIP-layer (L 3.5).

## **1.7 Thesis Outline**

This thesis contains six chapters and is organized as follows:

Chapter 1 provides a brief introduction of the study. It covers topics on motivations, problem statement, research aim, objectives, research questions and scope.

Chapter 2 introduces a general overview of the literature review of this study, discusses the related work on distributed mobility management.

Chapter 3 describes in-depth the Host-based Distributed mobility management (HDMM-TH) that enhances the host-based DMM scheme proposed in [50], however HDMM-TH has been enhanced in chapter 4 by a novel hybrid algorithm that overcome it is shortfalls. The methodology is presented as diagrams that describe briefly the proposed architecture, registration and handover procedures and analytical models and analysis.

Chapter 4 discusses the development of a hybrid algorithm for DMM (HADMM) as a combination of the previous schemes, HDMM-TH (in chapter 3) and one of the relevant scheme proposed in [47] here after called (MHPP), respectively, the chapter presents the motivation for the design of this protocol, design considerations and operational mechanisms. Chapter 5 Simulation Results are discussed in this chapter, the new algorithm has been compared with the most common and relevant schemes proposed by IETF and research community. Finally, Chapter 6 concludes the thesis by pointing-out the main contributions of the thesis and raising up some of the hot issues that require deep research in the near future.

# CHAPTER TWO

## Literature Review

### 2.1 Overview

In this sub-section, efforts for developing the Distributed Mobility Management (DMM) protocols undertaken by the research community are introduced, along with the limitations of each protocol. The schemes presented here are those that use the approach of making MIPv6 to work in a distributed way (involves the MN in managing its mobility), hence called the host-based DMM schemes; in additions to those that use the approach of making PMIPv6 to work in a distributed way (MN is not involved any mobility issues), hence called network-based DMM schemes. DMM solutions focusing on MIPv6 and PMIPv6 try to reduce the impact of the triangular routing on the overall performance.

### 2.2 Host-based Distributed Mobility Management:

Many host-based DMM schemes have been proposed by IETF and research community, in the following section we have summarized these schemes and investigated their limitations.

Fabio et al. [46] proposed a scheme based on Mobile IPv6 (MIPv6) the scheme dynamically anchors MN's traffic to appropriate access nodes (AN). Hence, during handover process, the MN's traffic remains anchored to its previous AN, However, the scheme still suffers from long routes that limit its scalability.

M.Liu et al. [53], the ADA (Asymmetric Double Agents) extension to Mobile IP is presented to optimize handover latency and communication delays. These improvements come at the cost of introducing two new entities in the network, the local mobile proxy (LMP) that takes care of the functionality of the home agent, but is located closer to the mobile node; and the correspondent mobile proxy (CMP), which is located near the correspondent node to provide an optimized route towards the LMP.

A. Nascimento et al. [61] Decouple the mobility management function of the HA in MIPv6, and distribute the routing management function, RM at access routers, while



leaving the LM centralized. If the MN undergoes handover, it gets a new CoA from the new access router to serve the newly initiated communication. Then, it registers the CoA with the location management entity, i.e., the LM. The LM then triggers the creation of the tunnel between the old and the new RMs, to enable session continuity for ongoing communication. The simulation results in [60] show that the proposed scheme reduces the packet loss and the time the MN remains unreachable during binding update procedures, when compared to MIPv6. However, the scheme performs poorly in terms of the handover delay when compared to MIPv6. In addition, the scheme proposes the MN to manually configure an IP address of its serving location management node. This may limit the deployment of the scheme, given the current increase in the number of MNs. Moreover, the scheme still requires modification to the MN's protocol stack.

A. Chan et al.[50] presents a host-based distributed mobility management scheme that extends the standard protocol MIPv6, and distributes all the mobility logical functions of the HA to the access routers, to survive the ongoing communication sessions the author implemented a bi-directional tunnel between the new and old PoAs.However the proposed scheme is suffering from non-optimal routing path, high signaling overheads to perform location update and it has to maintain  $(n-1)$  tunneling processes which adds extra overhead (40 extra packets overhead for each packet) which results in scalability problems.(this scheme has been enhanced by the author and published in [51], details in chapter 3).

R. Wakikawa et al. [54] presents the distribution of the HAs as a whole in the Internet topology. Each HA advertises the same IPv6 prefix using any casting. The traffic from the mobile node or the corresponding node is served by the closest topological HA, which reduces the communication delay. However, the signaling involved in synchronizing the mobile node Binding location can become very large, as the numbers of the MNs and the HAs increase.

H. Ali-Ahmad et al. [62] also discuss the DMM scheme, based on the MIPv6 protocol. The scheme is similar to most host-based DMM schemes that distribute all the mobility logical functions of the HA to the access routers. In contrast, the scheme proposes additional mechanisms to support an MN moving from the access routers with HA functionality to an access router without HA functionality. To achieve this, the MN

uses the MIPv6 mobility signaling to register its CoA (obtained from the classic access router) to the topologically close access router with HA functionality, which will also anchor its traffic. So, the MN does not use this new CoA to initiate a new communication. Instead, this CoA is used for routing, like the MIPv6. Consequently the new communication established at the classical access router undergoes tunneling. Nevertheless, the scheme does not optimize routes for ongoing sessions, and this may lead to long end-to-end delay for long-lasting communication, if the MN moves far away from the communication anchoring access router(s), when the topology spans a large area.

## **2.3 Network-based Distributed Mobility Management:**

In the following sub section, we have thoroughly investigated the network-based DMM schemes proposed by the research community and point out their limitations.

Koh S, et al. [45], proposed two schemes for DMM in Proxy Mobile IP (PMIPv6) based mobile networks. The schemes are: Signal-driven PMIP (S-PMIP) and Signal-driven Distributed PMIP (SD-PMIP). The former is viewed as a partially distributed mobility management approach. It enables separation of the control plane from the data plane. Nevertheless, it still suffers from packet losses during handover. Such is the case because no mechanism for setting up a new tunnel between the MAGs was considered. As such, a packet remains tunneled only to the MN's old MAG, hence the probability of losing it increases.

In [56], Chan proposes to deploy several PMIPv6-like domains forming a large mobility super-domain, and to enable inter-domain mobility by re-routing packet from one mobility anchor to another. The first domain where the MN attaches is the home network, whereas next become the visited networks. When the MN is at home, typical PMIPv6 data and control planes are used. When the MN moves, the home mobility anchor intercepts the packets destined to the MN and tunnels them to the mobility anchor where the MN currently attached. In parallel the home mobility anchor instructs the ingress node where packet for the MN are coming from to forwards such packets with a tunnel to the current visited mobility anchor. The proposed solution still maintains hierarchy levels of traffic gateways, hence a real flat architecture is not achieved.

The previous design is enriched with signaling details and a performance analysis based on simulations in [57]. Nevertheless, the signaling mechanism is based uniquely on the case in which the CN is connected to another mobility anchor of the same super-domain. More results from the same design are available in the article [58], written by the same authors. The same design is exploited to provide network mobility (NEMO) support in [59].

P.P Ernets [60] proposed PMIPv6 route optimization. In this solution, the MAGs serving the MN and CN leverage on the information stored at the LMA to establish a direct tunnel between them, so a better path can be used for the communication. This mechanism still makes use of a tunnel for the whole duration of the data session. The solution is only applicable to the case in which the CN is attached to the same PMIPv6 domain as the mobile node.

Another route optimization technique for PMIPv6 is proposed by K.Xue et al. [61]. The proposed protocol either needs the CN to be connected to the PMIPv6 domain or to be able to interpret some modified PMIPv6 signaling messages.

J. Kim et al. [58] propose three DMM approaches, partially distributed mobility control (PDMC), data-driven distributed mobility control (DDMC) and signal-driven distributed mobility control (SDMC). Both DDMC and SDMC use fully distributed architecture whereas PDMC uses partially distributed architecture. The numerical analysis revealed that the approaches outperform PMIPv6 in terms of binding update and packet delivery costs. However, the use of multicasting in DDMC and SDMC pushes unnecessary traffic into the network, which may waste network resources and the link bandwidth.

In [84] the author proposed a novel network-based and HIP solution that combines the PMIPv6 and HIP to provide secure, seamless handover, however the proposed solution prone to single point of failure since the whole system is depend on a single entity and can only be applied for a small coverage area.

## **2.4 Comparison of Distributed Mobility Management Schemes.**

Table 2.1 depicts a comparison between Host-based and Network-based DMM schemes that have been investigated in the literature from the author's point of view as

shown in the table, network-based DMM schemes are applicable to deploy since the host-based schemes require the modification of protocol stack for all registered MN's which is very difficult to change the kernel and upgrade the MN to support mobility function, in addition to the limited resources(power, memory), hereafter limits the opportunity of the deployment by network operators.

Table 2.1 Summary comparison between Host-based and Network-based DMM Schemes

Protocol Criteria	Host-based DMM Schemes	Network-based DMM Schemes
Extended Protocol	MIPv6	PMIPv6
MN managing multiple IP addresses	✓	✓
MN Protocol Modification	✓	X
Mitigating Packet overhead	✓	✓
Discovering the pre-configured IP address and Mobility Anchor-MA	X	✓
Large signaling overhead to maintain ongoing sessions during handover	✓	✓

## 2.5 Final Remarks

In this chapter, we have introduced the concept of mobility classification of mobility protocols (host-based and network-based), new paradigm of mobility management approach—the distributed mobility management – DMM as a novel algorithm to overcome the limitations and the weaknesses investigated in the centralized approaches, In addition we have reviewed the recent efforts undertaken by IETF and research community towards developing new protocols to cope with the new trends towards flat mobile network architecture.

The following chapters present the extended host-based DMM (HDMM-TH) proposed by the author as an enhancement for the relevant scheme proposed in [50], and the new hybrid DMM algorithm developed by the author, which combines the host-based DMM (HDMM-TH) proposed in [51] and a relevant scheme proposed in [47].

# **CHAPTER THREE**

## **HOST-BASED DISTRIBUTED MOBILITY MANAGEMENT (HDMM-TH)**

### **3.1 Introduction**

This chapter presents a host-based distributed mobility management support design for moving mobile node, named Host-based distributed Mobility Management approach using Tunnel-less Handover (HDMM-TH), and the proposed approach enhanced the approach developed by research community in [50] which proposed two scenarios of DMM host-based and network-based, in our research we focused in enhancing the first scenario.

The motivation for HDMM-TH design is presented in this chapter, additionally the architecture of HDMM-TH is viewed a long with the basic operations, MN initial registration and handover process.

Finally, A numerical analysis has been tackled in order to show the effectiveness of the proposed approach when compared with the standard MIPv6 and aforementioned approach in [50]; In the analysis, the same formulas and parameters used by [50] are applied to perform a realistic evaluation, performance metrics used in the numerical analysis are registration delay, signaling overhead and the traffic intensity.

### **3.2 Motivation and Design Approaches for HDMM-TH**

The Distributed Mobility Management approach tries to overcome the limitations of the traditional centralized mobility management by bringing the mobility anchor closer to the MN as mentioned in the previous chapter.

Although, there have been many efforts achieved by the IETF and the research community as mentioned in the literature to overcome aforementioned limitations [46][53][61][54][50][62], However, additional efforts are needed to enhance the proposed approaches to eliminate the encounter drawbacks such as handover latency, packet loss and scalability problem.

In this chapter, we present an enhanced host-based mobility protocol derived from both the standard Mobile IPv6 and the approach introduced in [50] in which the home agent (HA) is moved from the core to the edge of network, being deployed in the access router and default gateway of the mobile node.

This approach has been published in [51] under the name of Host-based Distributed Mobility Management approach using Tunnel-less Handover (HDMM-TH); hereafter we refer to it as HDMM-TH.

The architecture of the proposed approach and the principle operations are explained in the following sections.

### **3.3 Related Work**

As mentioned earlier, our proposed host-based DMM approach is an enhanced model of the counterpart discussed in [50] here after called Chan Model, in the following sections we will illustrate its main architecture and the drawbacks in [50] which have been solved by our enhanced approach.

#### **3.3.1 Architecture Overview:**

Figure 3.1 depicts an overview of how the introduced host-based DMM approach supports the MN's handovers in a distributed manner.

The author, used the following functional entity and signaling messages to support the MN's mobility in a distributed manner.

- **Access Mobility Anchor (AMA):** Runs on the AR. The AMA allocates a network prefix to the MN while it maintains the binding cache, which is updated by mobility signaling from the MN.
- **Access Binding Update (ABU) and Acknowledgement (ABA):** The ABU message is used for updating the MN's mobility context and requesting the bidirectional tunnel establishment between the serving AMA and the origin AMA(s). In response, the ABA message is sent from the origin AMA(s) to the serving AMA.

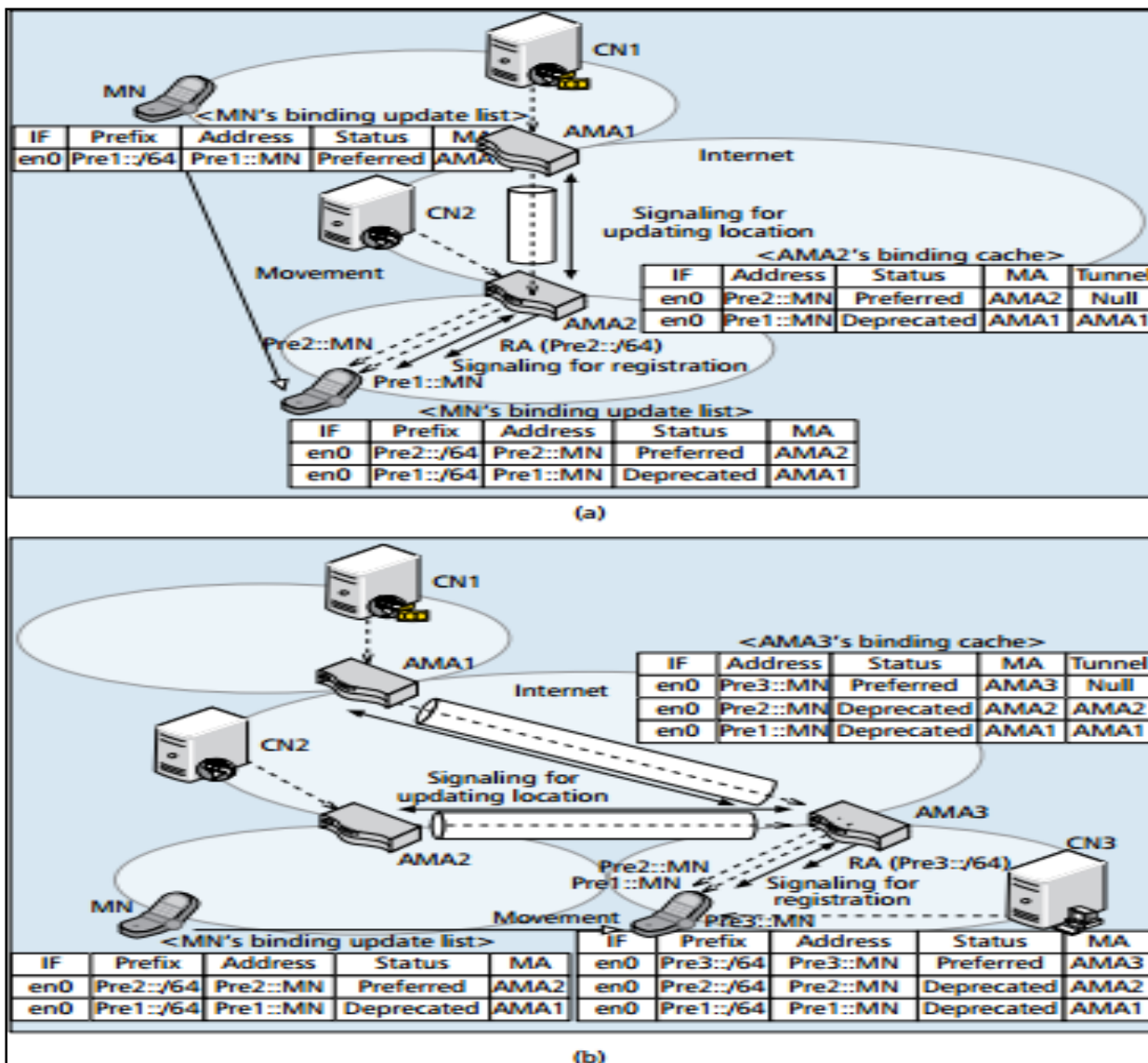


Figure 3.1 Mobility management in Chan[50] host-based DMM approach:

a) Handover from AMA1 to AMA2; b) handover from AMA2 to AMA3

### 3.3.2 Limitations of the approach

The limitations of the above approach are discussed in section 3.5.1 by using numerical analysis, which proved the superiority of our proposed approach over the standard MIPv6 and introduced approach [50] in terms of registration delay, signaling overhead and the traffic intensity.

### 3.4 Architecture of the proposed HDMM-TH

Figure 3.2 illustrates the architecture of our host-based DMM approach.

#### 3.4.1 Architecture Overview

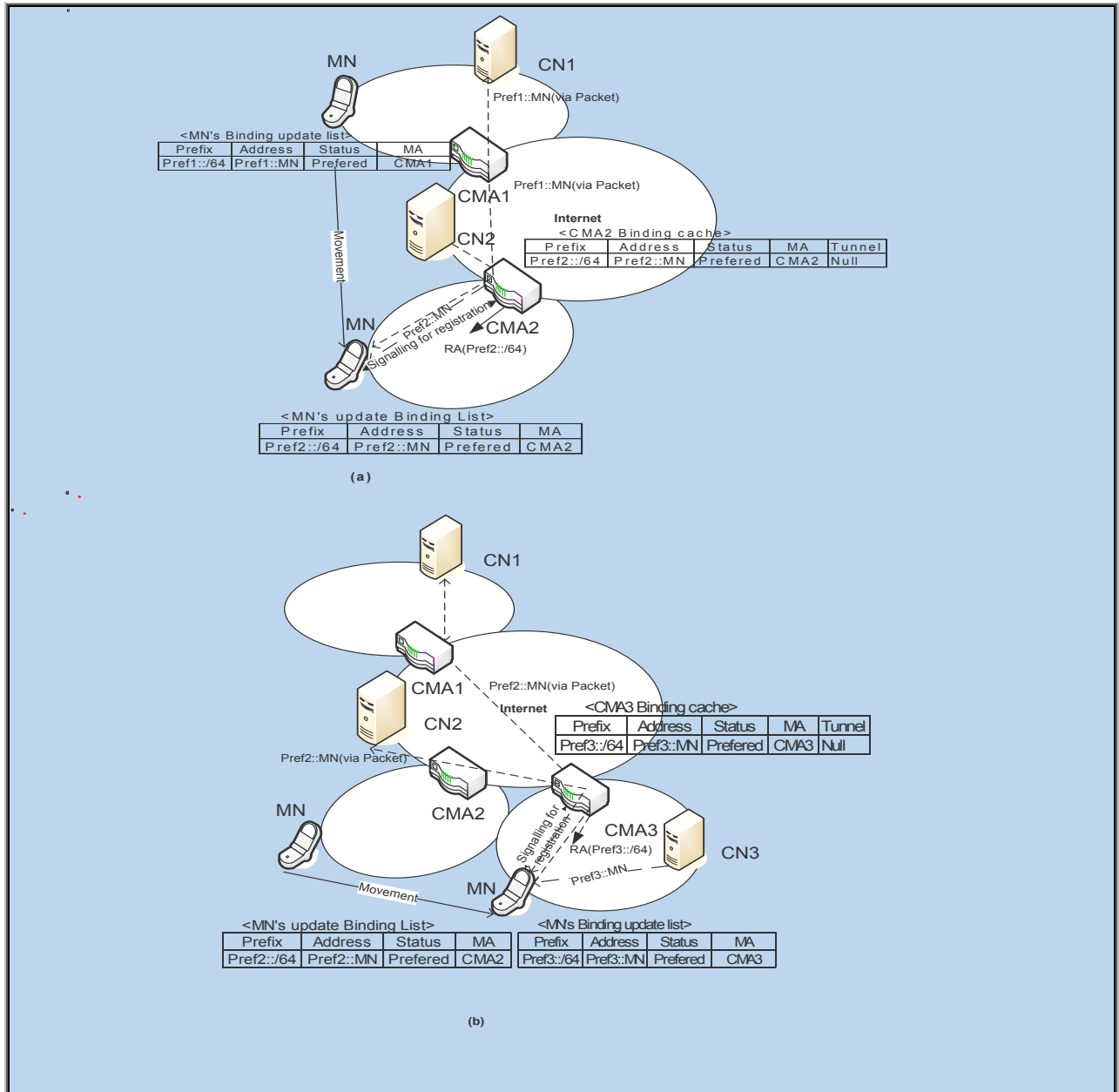


Figure 3.2: Mobility Management in the proposed Host Based DMM (HDMM\_TH) approach:  
a) handover from CMA1 to CMA2, b) handover from CMA2 to CMA3

The proposed approach HDMM-TH is an enhancement of the standard MIPv6 and the approach introduced in [50] to work in a flat and distributed manner, and a new mobility entity called Control Mobility Anchor, hereafter called CMA, is acts as a Home Agent ( HA) in



MIPv6 CMA is co-located and deployed at access routers (runs on AR's). The architecture as shown in figure 4.2, covers a wide area and is constrained to a single service provider. The HDMM-TH assigns IP addresses via its CMA and the standard MIPv6. Its mobility management functions (i.e. Location Management, Routing Management and assigning Home Network Address –HoA) are fully distributed at the CMAs to bring mobility closer to the MN.

Figures 3.3 and 3.4 demonstrate functions of the proposed architecture. Emphasis of further discussions in this following sub-section is on two operations: initial registration and handover. Each of these operations is elaborated on next. The example of HDMM-TH functional architecture in Figure 3.2 is used for purposes of discussion.

### **3.4.2 Mobile Node Initial Registration and Packet Delivery Procedures**

When the MN attaches to the first Control Mobility Anchor (i.e. CMA1) detects its presence on the link connecting both. The CMA then creates a binding entry for the MN and sends to it a Router Advertisement (RA) contains the prefix named (Pref1). Then, MN uses the prefix (Pref1) to configure an IPv6 address (e.g. Pref1:: MN) following the stateless configuration mechanism and creates a binding update list. The MN then uses the address to communicate with the CN1 without tunneling.

When CMA1 receives sent data by an MN, it mimics as standard IPv6 router and forwards the data to its destination address (CN1) without encapsulation.

Figure 3.3 illustrates the process of initial registration of the MN.

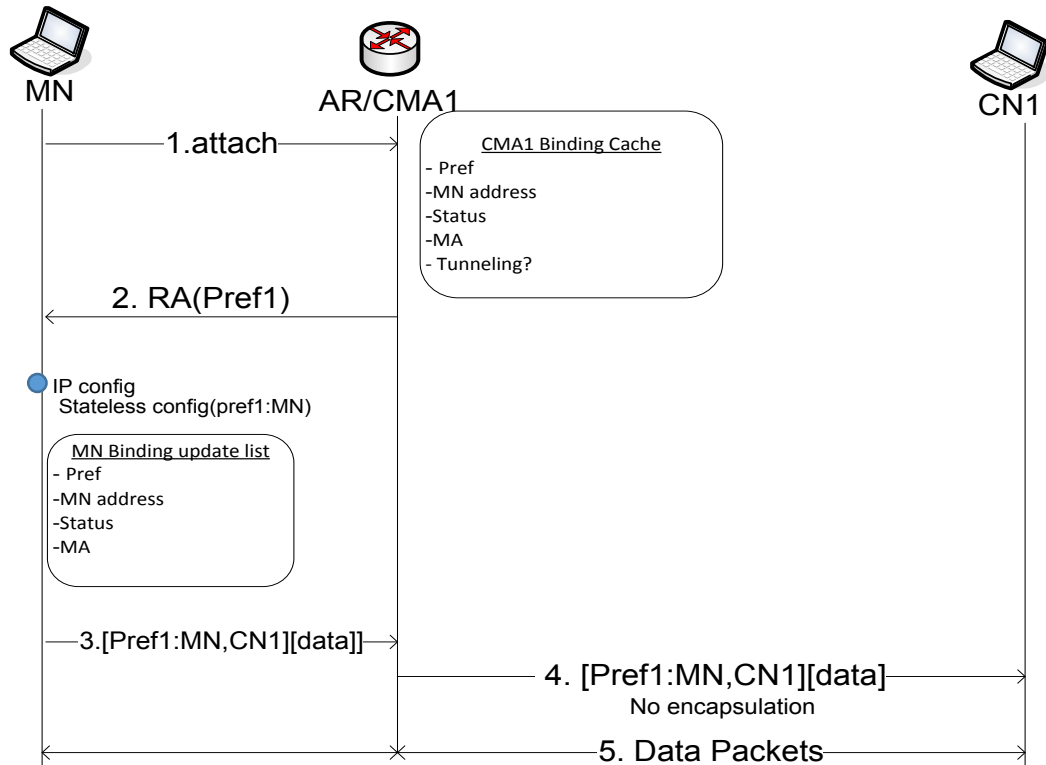


Figure 3.3: Registration and Packet Delivery

### 3.4.3 Handover and Packet Delivery Mechanism for HDMM-TH

Figure 3.2 illustrates an overview of how the MN's handovers in a distributed manner. Figure 3.2a depicts the MN's movement from the access network of CMA1 to the access network of CMA2. While the MN was staying at the access network of CMA1 in which it configured it is IP address, Pref1::MN the IP address's status was "preferred". Any communication session with Pref1::MN is established in a standard way as long as the MN stays at the access network of CMA1. When the MN moves to the access network of CMA2, it receives the router advertisement (RA) message containing the new network prefix, Pref2::/64, from CMA2.

It leads the MN to configure a new IP address, Pref2::MN Then; the MN registers it is attachment to CMA2 by sending the Binding Update (BU) message that results in updating the binding cache of CMA2.

After configuring the new IP address while communicating with CN1 (started at CMA1), The MN sends a new Packet to CMA2 that is destined to CN1. Since the new IP address (Pref2::MN) is unknown to CN1, the latter will reject delivered packets from the former. In such a case, to address this delima, we developed a novel mechanism that is inserting the

previous IP address (Pref1::MN) into the Extension Header (EH) field of the IPv6 packet (EH has variable size). Once the CN1 receives a new packet it checks the Extension Header field and accepts the packet only if it is corresponding Pref1::MN is verified. The CN1 also recognizes the move by the MN to a new CMA whose network prefix is (Pref2). Consequently, it will receive the delivered packet from MN. The CN1 then sends new packets destined to MN directly to the serving CMA (CMA2). The latter delivers the packers to the target MN acting as normal access router.

As the MN is connecting to the CMA2, the new configured IP (Pref2::MN) is used for new communication sessions since it is status is “preferred”; The CMA2 will keep only one binding entry for the MN.

Figure 3.2b illustrates the MN’s movement to the access network of CMA3 with an active sessions associated with Pref1::MN and Pref2::MN respectively. As the MN is attached to CMA3, it will configure a new IP address (Pref3::MN) using network prefix Pref3::/64 and registers to CMA3 by sending the BU message which results in updating the binding cache of CMA3 similar to the case of Figure 4.2a, since there are two active communication sessions with CN1 and CN2 respectively, the same scenario mentioned earlier will be applied and the MN will insert the previous IP address (i.e. Pref2::MN) into the Extension Header (EH) field of IPv6 Packet. Once the CN1 and CN2 receive a new packet, they check the Extension Header (EH) field of IPv6 Packet and accept the packet only if it is corresponding Pref2::MN is verified. They recognize by moving the MN to a new CMA whose network prefix is (Pref3). Consequently, they will receive the delivered packet from MN.

After getting new packets destined with the new IP address (Pref3::MN), MN will no longer keep binding entries for CN1 and CN2 in it is binding update list the new configured IP will be used for new communication sessions meanwhile the MN stays at CMA3 and it is status is “preferred”. Consider that there is no tunneling is established between the relevant CMA’s as proposed in [50] ( the use of IPv6-in-IPv6 tunneling adds 40 extra bytes to every packet, and it also requires additional processing resources for the encapsulation/de-encapsulation operations and for the tunnel management itself, the number of tunnel per MN is n-1 shared with other MNs), and CMA3 will keep only one binding entry for the MN with assigned status as “preferred”, and this is the main contribution of the introduced approach as it is so hard to maintain (n-1) number of tunneling processes simultaneously.

As mentioned earlier, the above scenario focuses on a handover process that takes place during an ongoing communication sessions. However, the second handover's which does not involve any active session follows the steps of the standard MIPv6.

### 3.5 Performance Analysis

This section presents a comparison among the standard MIPv6, hosted based DMM proposed in [50] and the HDMM-TH (introduced approach) in terms of registration delay, signaling overhead, and traffic intensity (amount of active sessions on mobility anchor).

To provide a realistic evaluation, our approach used the same parameters and formulas as used in [50]. Tables 3.1, 3.2, 3.3 and 3.4 show the formulas, parameters lists, calculations of comparing terms and quantitative analysis results respectively.

Table 3.1 Formulas used for quantitative analysis

<i>Metrics</i>	<i>Formula</i>
Registration Delay( $D_{REG}$ )	$D_{REG} = H_{WD} \times \left( \frac{M_{REG}}{BW_{WD}} + L_{WD} \right) + H_{WL} \times \left( \frac{M_{REG}}{BW_{WL}} + L_{WL} \right) + D_{\delta} \quad (1)$
Signaling overhead for the registration ( $S_{REG}$ )	$S_{REG} = \left[ \frac{H \times M_{REG}}{T_s} \right] + S_{\delta} \quad (2)$
Traffic Intensity to a mobility anchor ( $T_{INT}$ )	$T_{INT} = \frac{\text{amount of ongoing sessions of MN}}{\text{number of mobility anchor}} = \frac{CS \times E(S) \times Mp \times (ho+1)}{m} \quad (3)$

Table 3.2 Parameters for quantitative analysis

Symbol	Meaning	Default Value
$BW_{WD}$	Bandwidth of wired links	100 Mb/s
$BW_{WL}$	Bandwidth of wireless links	11 Mb/s
$L_{WD}$	Latency of wired links (propagation delay + link-layer delay)	0.5 ms
$L_{WL}$	Latency of wireless links (propagation delay + link-layer delay)	2 ms
$H_{MN-HA}$	Number of hops between MN and HA	10
$H_{AR-AR}$	Number of hops between neighbor ARs	1 or 3

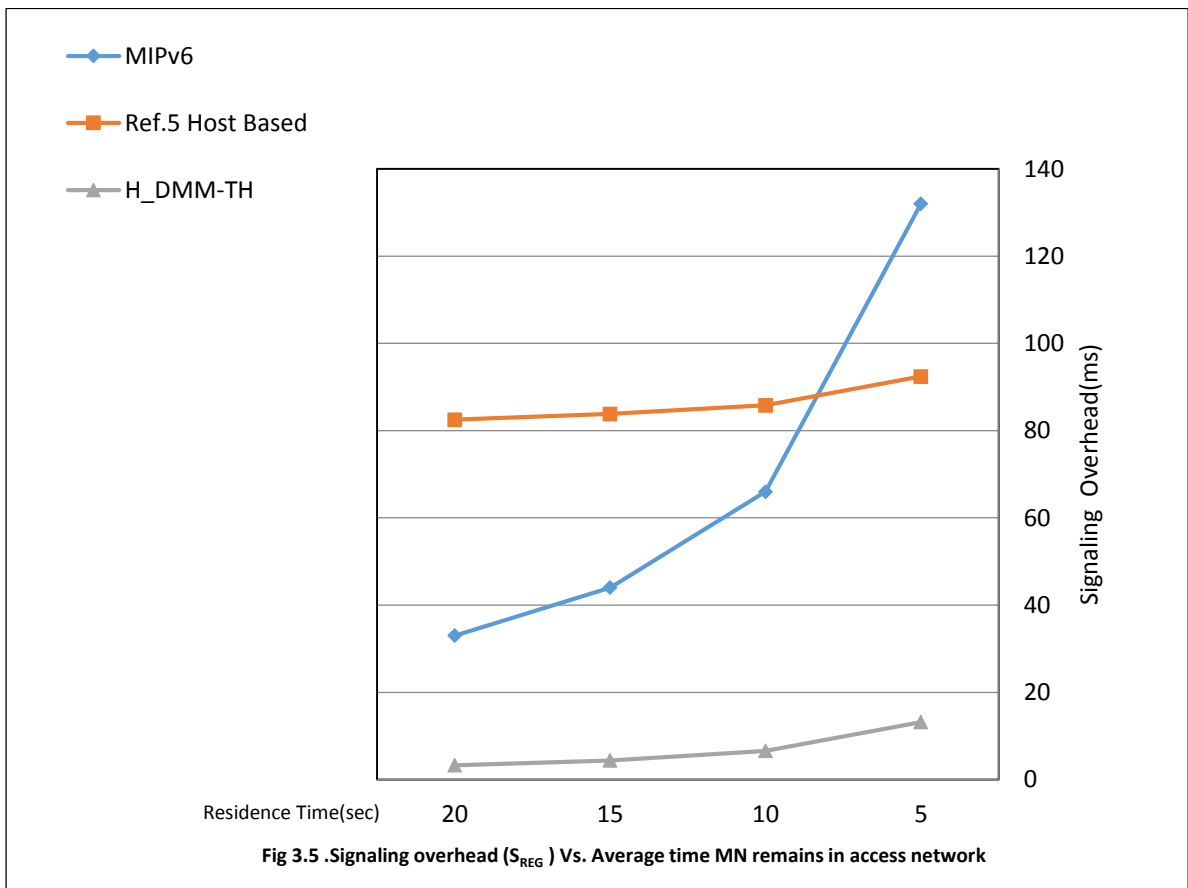
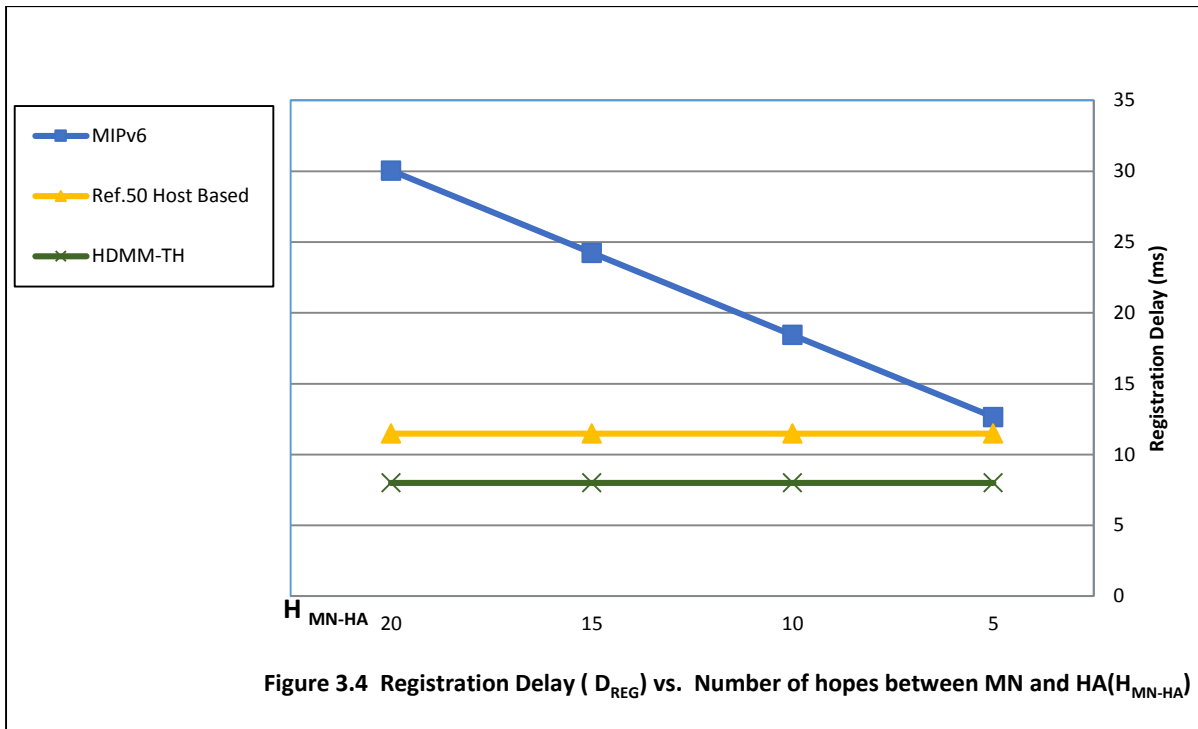
$M_{REG}$	Registration packet length	66 bytes
$M_P$	User data packet length	557 bytes
$E_{(S)}$	Average communication session length in data packets	10
$T_S$	Average time for which a MN remains in an access network	10–35 s
Symbols used in Formulas		
$H_{WD}$	Number of hops for the registration over wired links.	
$H_{WL}$	Number of hops for the registration over wireless links.	
$D_{\partial}$	Additional delay for registration to previous mobility anchor	
$H$	Number of hops required for registration. ( $H = H_{WD} + H_{WL}$ )	
$S_{\partial}$	Additional overhead for registration to previous mobility anchor.	
$M$	Number of mobility anchors Associated with MN.	
$Ho$	number of handover For <i>CMM</i> , $m=1$ For <i>DMM</i> , $m= n =ho + 1$	
$N$	Number of valid addresses configured at the MN in DMM.	<b>5</b>
$CS$	Number of ongoing sessions at the access network.	<b>5</b>

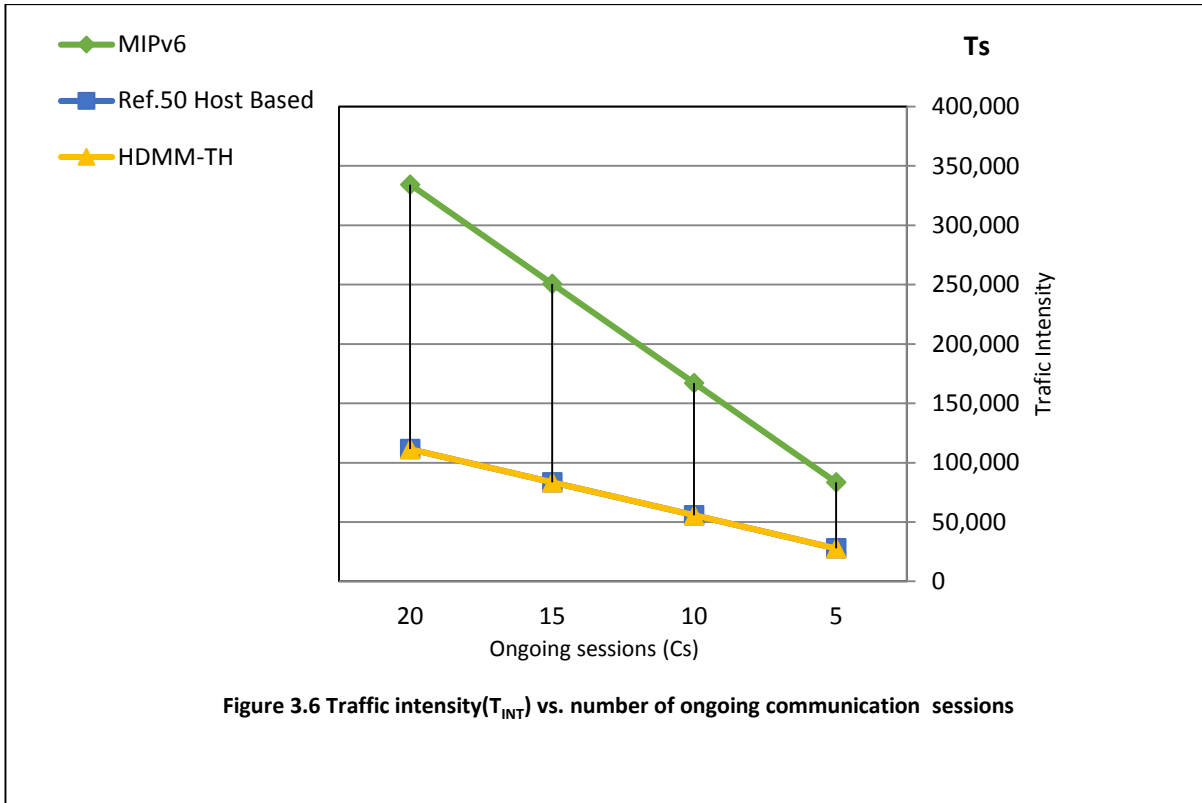
Table 3.3 Calculations of  $D_{REG}$ ,  $S_{REG}$  and  $T_{INT}$  for the three schemes

	$H_{MN-HA}$	<i>MIPv6</i>	<i>Ref.50 Host Based</i>	<i>HDMM-TH</i>
$D_{REG}$	5	12.64	11.48	8
	10	18.44	11.48	8
	15	24.24	11.48	8
	20	30.04	11.48	8
	$T_S$	<i>MIPv6</i>	<i>Ref.50 Host Based</i>	<i>HDMM-TH</i>
$S_{REG}$	5	132	92.4	13.2
	10	66	85.8	6.6
	15	44	83.6	4.4
	20	33	82.5	3.3
	$C_S$	<i>MIPv6</i>	<i>Ref.50 Host Based</i>	<i>HDMM-TH</i>
$T_{INT}$	5	83,550	27,850	27,850
	10	167,100	55,700	55,700
	15	250,650	83,550	83,550
	20	334,200	111,400	111,400

Table 3.4 quantitative Analysis for the approaches(MIPv6, Ref.(50) Host Based, Introduced approach)

Metrices	Formula	Symbols	MIPv6	Ref.(50) Host based approach	Introduced approach
registration delay( $D_{REG}$ )	$D_{REG} = H_{WD} \times \left( \frac{M_{REG} + L_{WD}}{BW_{WD}} \right) + H_{WL} \times \left( \frac{M_{REG} + L_{WL}}{BW_{WL}} \right) + D_{\delta}$ for Ref.(50) approach $D_{\delta} = H_{AR-AR} * (M_{REG}/BW_{WD}) + L_{WD}$	$H_{WD}$	9	0	0
		$H_{WL}$	1	1	1
		$D_{\delta}$	0	3.48	0
		$(M_{REG}/BW_{WD})$	0.66	0.66	0.66
		$L_{WD}$	0.5	0.5	0.5
		$(M_{REG}/BW_{WL})$	6	6	6
		$L_{WL}$	2	2	2
		<b><math>D_{REG}</math></b>	<b>18.44</b>	<b>11.48</b>	<b>8</b>
signaling overhead for the registration	$S_{REG} = H \times \left( \frac{M_{REG}}{T_s} \right) + S_{\delta}$ for Ref.(50) approach $S_{\delta} = (n-1) * (H_{AR-AR} * M_{REG}/T_s)$ n is number of addresses for MN(here n=5)	$H(H_{WD}+H_{WL})$	10	1	1
		$D_{\delta}(n=5)$	0	79.2	0
		$(M_{REG}/T_s), (T_s=10)$	6.6	6.6	6.6
		<b><math>S_{REG}</math></b>	<b>66</b>	<b>85.8</b>	<b>6.6</b>
to a mobility anchor $T_{INT}$	$T_{INT} = \frac{\text{amount of active sessions of MN}}{\text{number of mobility anchors}} = \frac{CS \times E_{(S)} \times M_p \times (ho+1)}{m}$ ho : number of handover (here ho=2) m: number of mobility anchors(for MIPv6 ,m=1 , for others m= ho+1)	$CS$	5	5	5
		$E_{(S)}$	10	10	10
		$M_p$	557	557	557
		$ho$	2	2	2
		$m$	1	3	3
		<b><math>T_{INT}</math></b>	<b>83,550</b>	<b>27,850</b>	<b>27,850</b>





### 3.5.1 Numerical Results Analysis and Discussions

Figure 3.4 shows the registration delay as a function of number of hops between the MN and HA (MIPv6 HA, AMA in [50] host based DMM, and CMA in introduced approach). As the number of hops increases, the registration delay of MIPv6 dramatically increases, except the host based DMM proposed in [50] and the introduced approach, this due to deployment of mobility anchors (serving AMA, CMA) at the access router (AR) respectively.

In this analysis, we found that the introduced approach (HDMM-TH) has the lowest registration delay, this because there is no additional overheads for registration to the previous mobility anchor.

Regarding the signaling overhead, Figure 3.5 shows that whenever the dwell time (residence time) increases, the signaling overhead of the three approaches decreases, Moreover, we have approved that the host based approach in [50] has the largest signaling overhead, this is owing to maintaining (n-1) number of tunneling processes and the signaling overhead increases dramatically with number of valid addresses configured at MN (hereafter called n).



In addition, we found that the signaling overhead of MIPv6 is in-between the ref.50 Host based DMM and our approach, and this is due to increasing number of hops required for the registration over the air.

Moreover, we have noticed that the proposed approach has the smallest signaling overhead since it doesn't require neither registration to the previous mobility anchors nor it doesn't require tunneling process.

Figure 3.6 shows that the traffic intensity for the host based DMM proposed in [50] and our approach have the same level (same growth rate), this due to distributing the mobility anchors (i.e. AMA, CMA) respectively at the access routers, to avoid the user data traffic's density that delivered to a single point, unlike the MIPv6 which has the largest traffic intensity that affected by a huge volume of packets delivered to a single static centralized mobility anchor (i.e. HA).

## 4.6 Summary

In this chapter, we presented an efficient and coordinated a host-based DMM approach that solves most of the limitations experienced by the approach developed in [50], here is the main contributions of HDMM-TH.

- Eliminating the issue of a single-point-of-failure owing to getting services only via Home Agent(HA), by deploying a new mobility entity called Control Mobility Anchor (CMA) which is an extension of HA .The CMA is distributed and co-located at the access network level (runs on AR's) and closer to the MN.
- Reducing the handover latency and packet losses by introducing a new technique that piggy bags routing information in the Extension Header (EH) field of IPv6 Packet.
- Removing the tunneling overhead over the air, which occurs during the handover process to keep the ongoing communications active by using the mentioned technique and this is main contribution of our approach.
- Solving the scalability problem by introducing the new mobility entities which are co-located at access router closer to the MN and provide the mobility services; in addition to applying the innovative technique.
- Reducing the overall signaling overhead by eliminating the unnecessary signals that used for location updates.

As a final word, since the proposed approach is a host-based that requires a modification of the mobile nodes' IP stack (Hosts must implement MIP in the kernel) indeed this is a tedious task that limits the deployment of such approach, but according to the rapid increasing in the mobile market particularly, smart phones and laptops with smart features, operators and vendors they have to re-think of deploying such schemes with a few modifications to the MN's to manage their mobility.

# CHAPTER FOUR

## HYBRID ALGORITHM FOR DISTRIBUTED MOBILITY MANAGEMENT (HADMM)

### 4.1 Introduction and Motivation

In the standard TCP/IP stack, IP is a routing address which has to change as a mobile node (MN) moves and changes its routing path in the network. However, the socket of a network session is identified by the IP address together with the port number. Therefore, the network session will not survive upon such changes of the IP address. To solve this dilemma, a new namespace is introduced to identify the network session, so that the IP address is used only as a locator. This locator/identifier split approach provides a better framework to develop solutions to support mobility multihoming, IPv4 and IPv6 interoperability, and security.

One locator/identifier split approach is the use of two separate IP addresses in Mobile IP (MIP) [3] to support host mobility in the Internet. A static IP address is used to identify the network session, a dynamic IP address is used for routing, and a mapping between these two addresses is maintained. Another locator/identifier split approach is Host Identity Protocol (HIP) [1][2] which provides secure mobility support in a simpler manner than the MIP based solutions [4][5][6]. Yet both MIP (v4/v6) and HIP are host-based protocols, requiring new functionalities in the MN protocol stack.

Network-based mobility support which is not implemented in the MN protocol stack. Since MN participation in mobility-related signaling is not needed, such network-based solutions Proxy MIPv6 (PMIPv6) [8] extends MIPv6 to provide network-based mobility support which is not implemented in the MN protocol stack. Such network-based solutions optimize handover performance in terms of handover latency and signaling overhead [8]. However, PMIPv6 lacks elegant secure mobility support.

HIP proxy attempts to provide network-based HIP communication to non-HIP hosts. Yet, it only enables fixed Non-HIP hosts to communicate with HIP hosts. In addition the HIP infrastructure does not support MIP-based mobility. Therefore, handover of a HIP session is not possible with either HIP proxy [8] or MIP. Even if the existing mobility management schemes are extended to the HIP proxy, there will be excessive handover latency and high signaling overhead.

## 4.2 Related Work

Identifier-locator separation with HIP provides better Security [10] and multihoming [11] support, and also provides a framework to design a mechanism for mobility management [11]. Yet, existing mobility support mechanisms with HIP suffer from long handover delay owing to Duplicate Address Detection (DAD), location update and other signaling overhead.

Micro-HIP [12] is a micro-mobility management solution in HIP using a local rendezvous server (LRVS) to perform Network Address Translation (NAT) in addition to other rendezvous server (RVS) [13] functions. Upon entering a visited domain, a MN registers with a LRVS and thereafter MN registers with the RVS. The RVS, in turn, registers the IP address of the MN at the Domain Name Server (DNS) [14]. Thus the MN informs the LRVS and not the Correspondent Host (CH) to redirect the data traffic to the new location at its local IP address. Yet, every time the MN changes subnets it informs the LRVS about the new location (new IP). However, this solution doesn't a void IP address configuration and re-registration at the LRVS whenever the MN moves from one subnet to another within the same domain.

The IP configuration, which requires Duplicate Address Detection (DAD) adds some delay to the handover process, while the registration takes considerable time that also contribute to the handover latency during the registration at LRVS, the LRVS continues to forward the ongoing packet that are destined to the MN to the old Access Router (AR) that was previously point of attachment (PoA) for the MN, this increases packet delay and loss.

In [47] the author proposed a generic scheme hereafter is called (MHPP) that inherits the advantages of both network-based and HIP technology. As the proposed scheme tackled mobility at HIP layer to all mobile hosts. The network-based service includes tracking mobile hosts, assigning network prefix per host identifier, securely updating the binding of mobile hosts, and providing a HIP proxy function to ensure the assignment of the same IP prefix to the mobile host during handover. The proposed scheme outperformed the PMIPv6 and standard HIP protocols in terms of handover latency and signaling overhead. However, MHPP maintains hierarchy levels (4 levels) of control communications, hence a real flat architecture is not achieved and it is not

suitable for future wireless mobile internet , in addition, it is prone to single point of failure since all the communications rely on a centralized entity called Local Rendezvous Server (LRVS) and no load balance.

In our work, we have developed a novel partial network-based distributed mobility management algorithm, hereafter called HADMM, that extends our previous scheme HDMM-TH proposed in chapter 3 and the network-based and HIP scheme (MHPP) proposed in [47]. HADMM employs the HIP layer to provide efficient, secure network-based seamless mobility and multihoming support to all MNs, with all signaling overheads on MN's interface removed. Unlike ordinary HIP proxy's solutions, our design eliminates the issue of a single-point-of-failure by deploying a cluster server that provides a reliable location management function.

The proposed architecture has been implemented using OMNeT++ v4.0 network simulator and HIPSim++ simulator, and the results are obtained in order to show the effectiveness of HADMM when compared with HDMM-TH, PMIPv6 and MHPP respectively.

The performance metrics used in the results analysis are handover latency, signaling overhead and packet loss. In addition, analytically, we analysis the limitations of the proposed HDMM-TH that have been efficiently addressed by HADMM.

Figures 4.1, 4.2 illustrate the architectures of MHPP and HADMM, respectively.

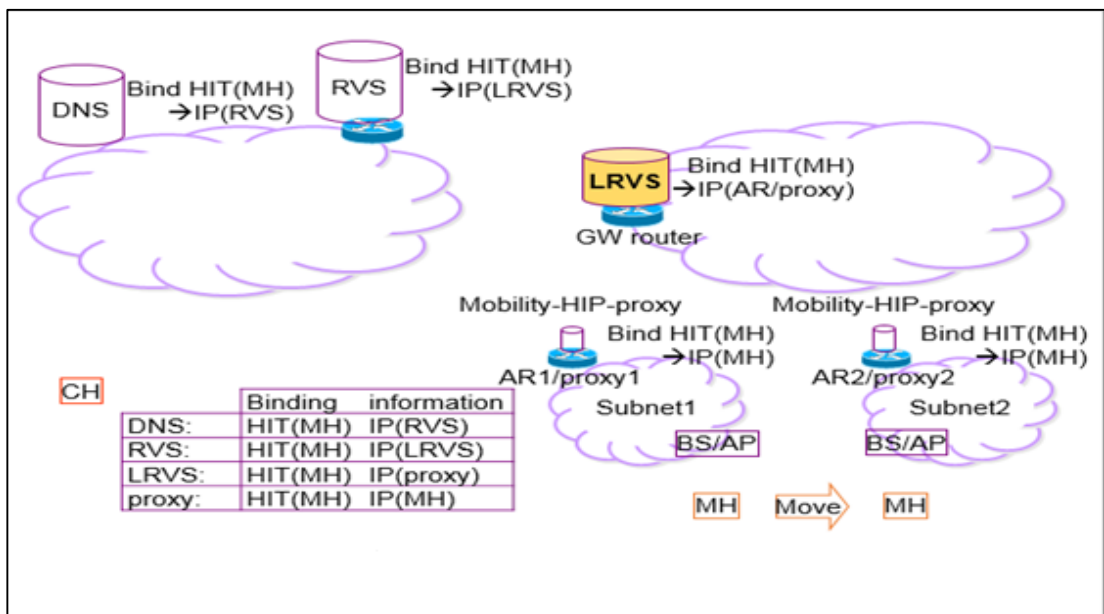


Figure 4.1 MHPP Architecture

### 4.3 HADMM Architecture Overview

Figure 4.1 depicts the proposed HADMM architecture.

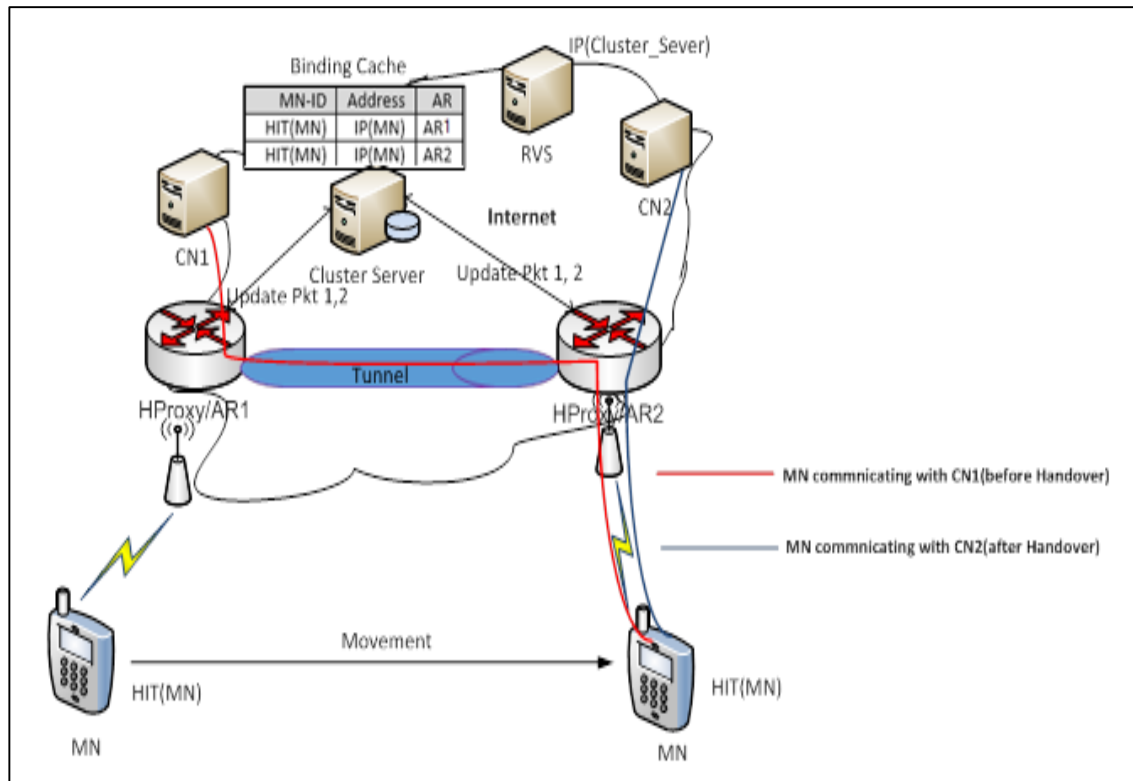


Figure 4.2 Hybrid Algorithm for DMM (HADMM) Architecture

In the proposed algorithm, which is shown in figure 4.2, two new entities are introduced as follows:

- a) Cluster Server: is implemented to provide the location management (LM) to keep track of all registered MNs by using a binding cache which stores MN information as Host Identifier (HIT), Host IP and IP address of the serving HIP proxy (HProxy) in addition it solves the single point of failure problem, since there are more than one alternative servers are stand by for any malfunction (reliability) and there is load balance among these servers. Moreover cluster server is assigned static IP address for global reachability.
- b) HIP Proxy: here after is called (HProxy) is co-located with Access Router (AR) and each HProxy has the following functionalities:
  - Assigning the Home Network Prefix (HNP) for attached MNs based on MN identifier and allows MNs to use the assigned network prefix as long as MNs move under the same cluster server (intra-handover).

- Tracks MNs and updates their binding at the cluster server to optimize the data packets route that are destined to MN to be delivered to the current serving AR, to end that it exchanges two UPDATE messages with the Cluster Server, the UPDATE packet 1 and UPDATE packet 2 messages respectively.
- Assigning the Host Identity (HIT) for non-HIP enabled MNs and thus performs HIP signaling on behalf of it.
- Reduces the packet loss and signaling overheads.

When HIP enabled MN attachment is detected, the HProxy sends it is information (HIT, IP) to the cluster server which in turns creates a record to bind HIT of MN's with it is respective IP address. Furthermore, the HProxy assigns HITs for non-enabled HIP MNs from it is HITs pool as binds it with it is respective IP.

### **4.3.1 Initial MN Registration and Reachability**

When HIP MN enters a domain, every HIP host uses the registration mechanism described in [35] to register itself with RVS. The registration of HIP MN is shown in figure 4.3 while the registration of non-HIP MN is shown in figure 4.4.

When the registration is successfully completed, MNs becomes globally reachable from any Correspondent Node (CN) that may wish to connect with them (the registered MNs). CNs resolve HITs of MNs by requesting the IP address of RVS with which MNs are registered via the DNS.

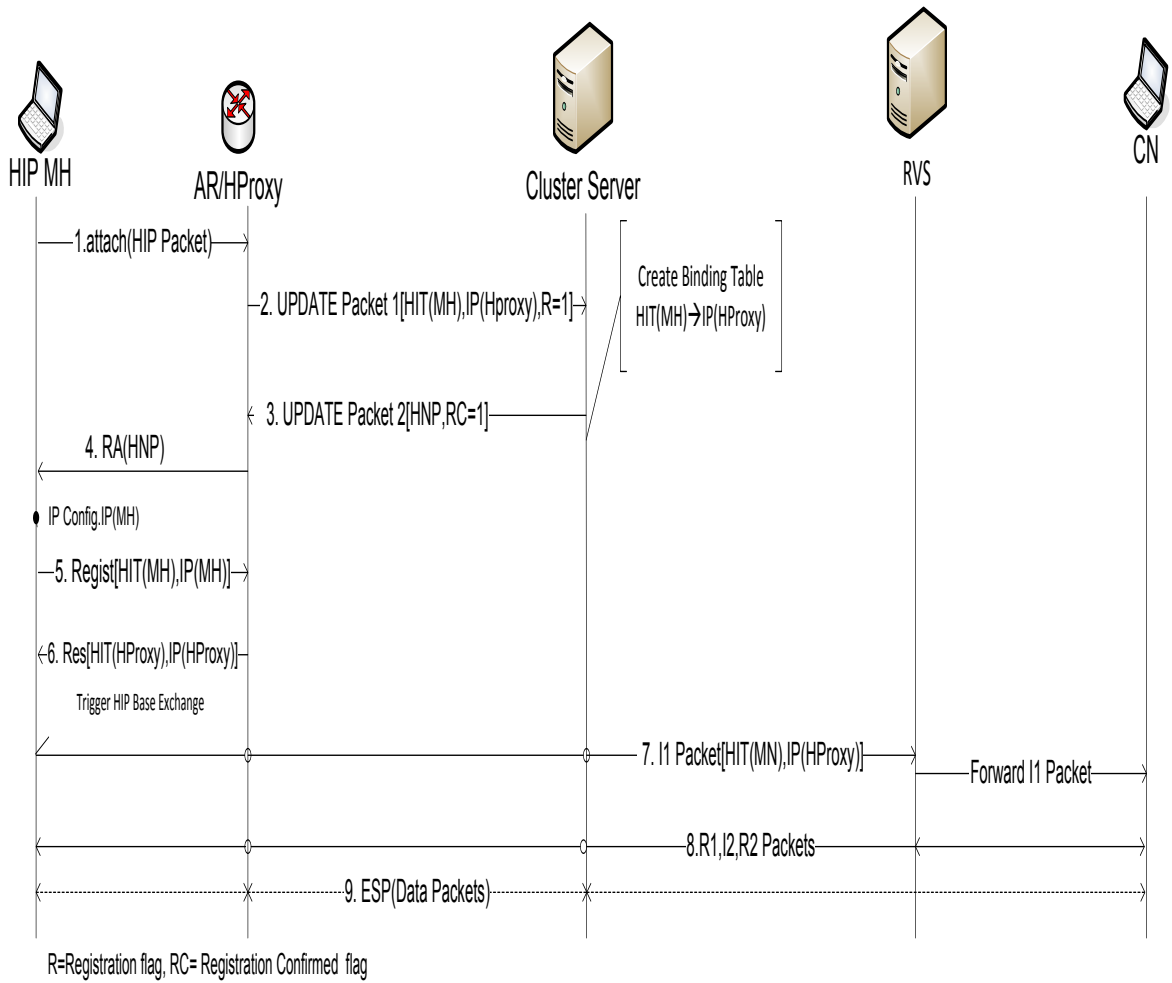


Figure 4.3: HIP enabled Mobile Node –MN Initial Registration Process



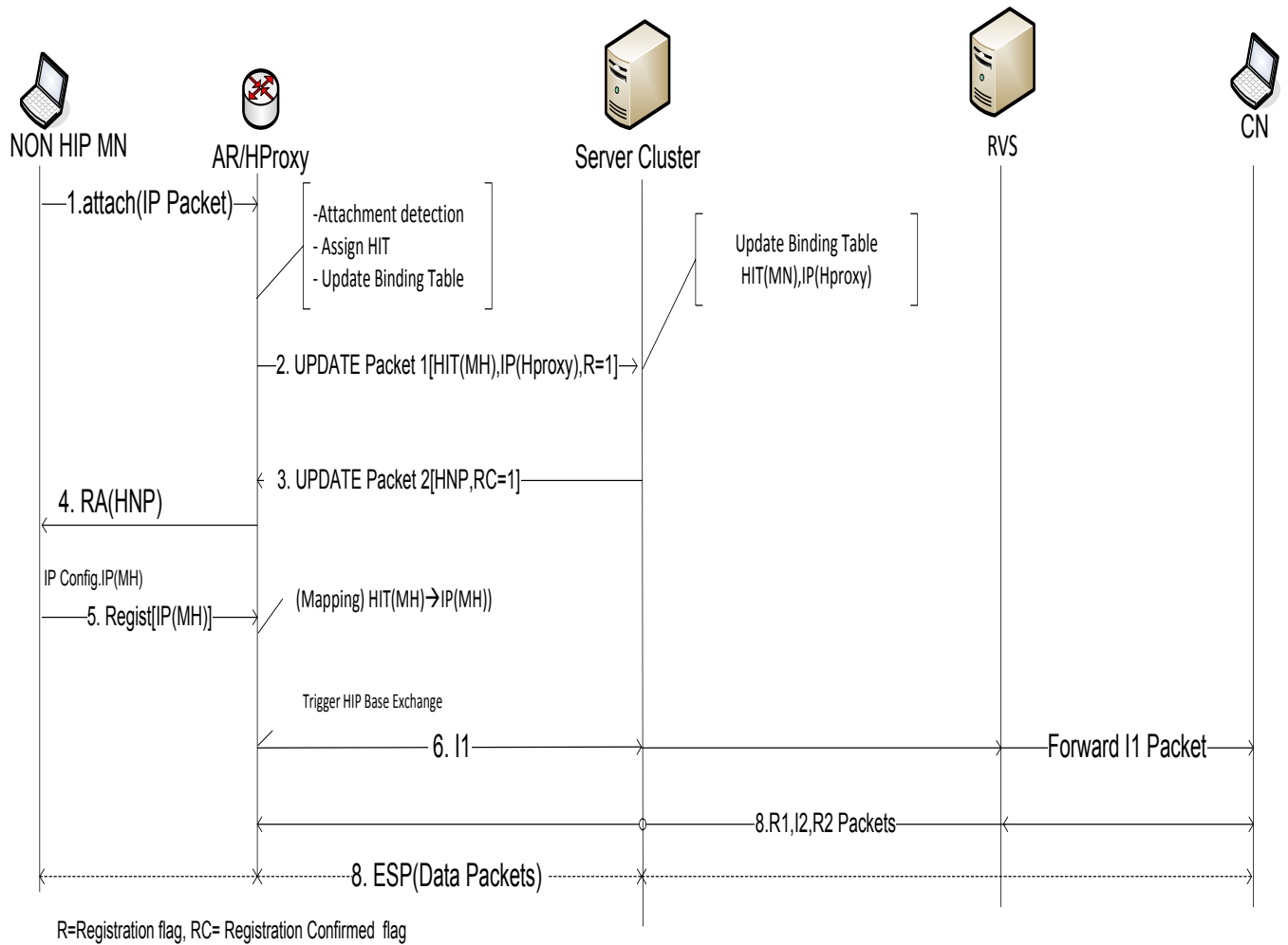


Figure 4.4: Non-HIP enabled Mobile Node -MN Initial Registration Process

### 4.3.2 Establishing Security Association (SA)

The proposed design enables the data traffic between either HIP enabled MN with HIP CN as well as non-HIP MN with HIP CN. Before data traffic starts, establishment of HIP security association is needed between HIP MN and HIP CN as shown in figure 4.3 and between HProxy and HIP CN if MN is non HIP as shown in figure 4.4.

### 4.3.3 Handover Mechanisms

HADMM domain has two point of views depend on the residence of MN and CN as follows:

- Localized domain: if both the MN and CN belong to the same domain that managed by the same cluster server. Yet, the handover process is an intra-handover.
- Global domain: if both the MN and CN belong to different domains that managed by different cluster servers. Hence, the handover process is an inter-handover.

In this following sub-sections, the mobile host's IP handover for both intra-handover (section 4.3.3.1) and inter-handover (section 4.3.3.2) scenarios are presented.

#### 4.3.3.1 HADMM intra-Domain Handover

In this section, we illustrate the intra-handover process for HADMM and takes place under the same domain.

Figure 4.5 shows a non-HIP MN moves (handoff) between two wireless networks belong to the same domain and under the same cluster server.

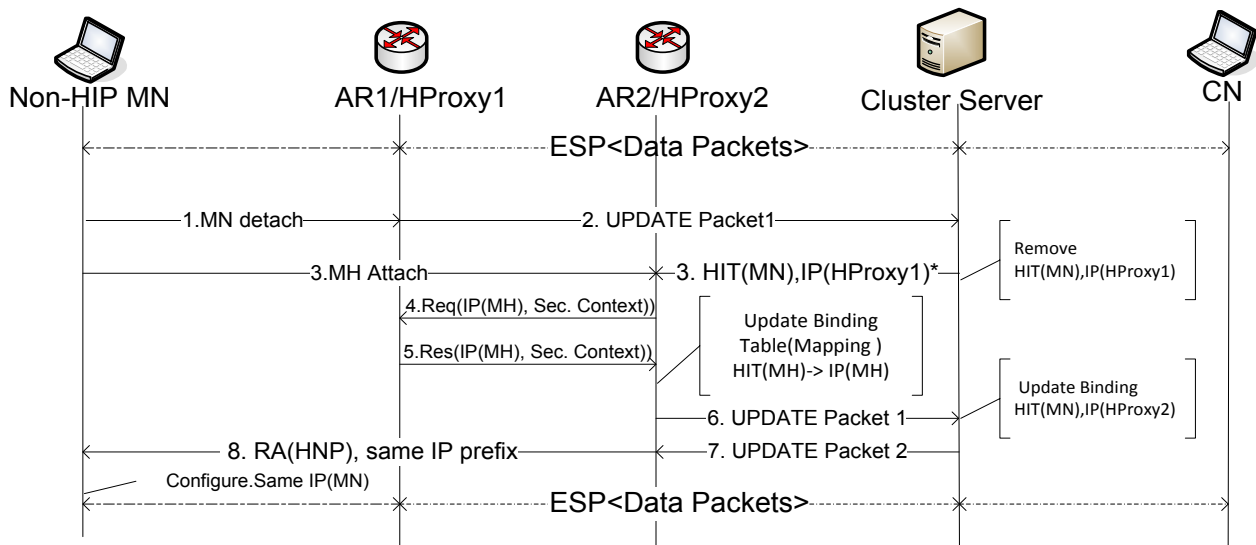


Figure 4.5 Non-HIP MN intra-Handover process (no need for BE SA) - MN, CN in the same domain

When the MN performs an intra-domain handover, HProxy1 detects the detachment and sends an UPDATE packet (packet1) to the cluster server to de-register its (HProxy1) IP address. Upon receiving the update packet from HProxy1, cluster server sends the MN binding information (i.e. HIT (MN), IP (HProxy1)-step 3) to all registered HProxies in the domain by using multicast, meanwhile HProxy2 detects the attachment of MN and acts as HIP proxy and updates the binding record of the MN at the cluster server. To do that, it needs to know the context of the HIP SA and the IP of the MN. Yet, to resolve this delima, the author proposed two new messages, HProxy2 will send a request message (Req) to HProxy1 asking for the IP of MN and it is corresponding security context, IP (HProxy1) is sent by the cluster server in the previous step, then HProxy1 will reply with response message (Res) that contains the MN IP and the required security context (SPIs) and this step occurs in a secure way.

Upon receiving the response message, HProxy2 updates it is binding by mapping the HIT(MN) and it is respective IP(MN), and sends an UPDATE packet (packet1) to the cluster server to update binding of MN. When HProxy2 receives the reply UPDATE packet (packet2) from the cluster server, it will send a Router Advertisement (RA) to the MN. The RA will have the same network prefix that the MN used to configure its IP address in the HProxy1 subnet. The MN, therefore, it retains the same IP address configuration, so that DAD is not required. This procedure significantly reduces the handover latency, signaling overheads, and packet loss.

HADMM reduces the HOL and signaling overheads by allowing both HIP-ENABLED and NON HIP-ENABLED MN to use the same IP address (to avoid the DAD process as it remains within a single domain) and sending the HIT (MN) and the previous serving HProxy IP to all registered HProxyies at an appropriate time. Then the new HProxy sends RA that contains the network prefix to the MN to retain the same IP configuration, and send the UPDATE packet 1 to the cluster server to set up a new path for the ongoing traffic. The use of the same IP address supports location privacy. Furthermore, the use of HIT in the upper layer protocol instead of IP address enables the HIP host to use the established HIP associations during and after the handover since the communication context remains the same.

Moreover, reducing the time taken to perform HIP BE can also reduce the handover delay when the re-establishment of HIP SA is required.

The number of HProxy in a domain depends on the domain's size, and each subnet is managed by HProxy which acts as the authoritative HProxy for that subnet. The number of HIP-ENABLED MNs in each subnet must not exceed the capability of the HProxy. This method can also manage the simultaneous move of the communicating parties (double jump) and multihoming in an easy and efficient way to do so.

In the following sub-section, we illustrate the inter-handover process.

### 4.3.3.2 HADMM inter-Domain Handover

In this section, we illustrate the inter-handover process for HADMM and takes place under different domains (MN and CN belong to different cluster server).

Figure 4.6 shows handover signaling between the different entities when a HIP MN moves (handoff) from one wireless network to another that belongs to a different domain and managed by another cluster server.

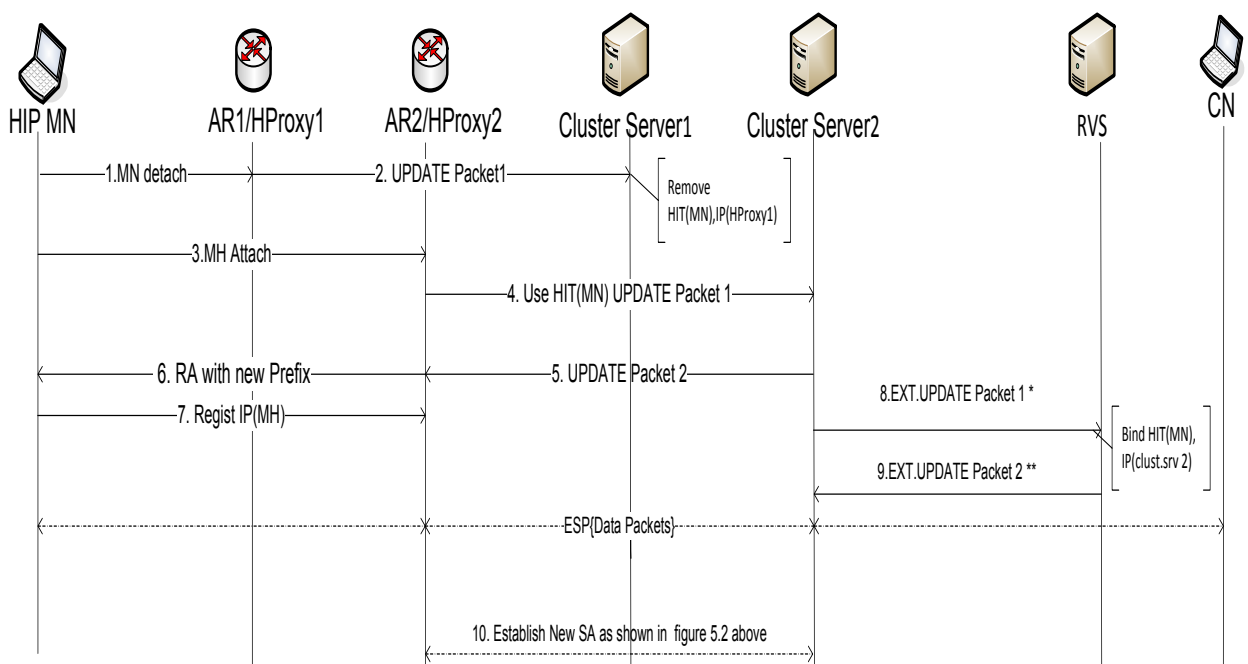


Figure 4.6 HIP-enabled MN Inter-Handover Process( MN,CN in different Domains)

\* 8. Packet 1=(IP(clust\_srv),HIT(MN),IP(clust\_srv 2))  
 \*\* 9.Packet 2=(IP(clust srv1),HIT(MN),auth)

Since the handover is not under the same cluster server, HProxy1 sends UPDATE packet1 to the old cluster server (cluster sever 1) to remove the MN's binding, whereas HProxy2 sends UPDATE packet1 to the new cluster server (cluster server 2) to create a binding record for the attached MN, Thereafter, cluster server 2 creates a temporary binding for the MN and sends a normal UPDATE packet (UPDATE pkt2) with the HNP to HProxy2. Furthermore cluster server 2 needs to determine the cluster server to which the new attached MN belongs to accomplish this, cluster server2 sends an extended update packet (Ext\_UPDATE Packet 1) to RVS, and the extended packet is used to play dual roles, that is to query about the IP address of the previous cluster server(i.e. cluster server 1), and this can be executed by using HIT(MN) as a searching key, and to bind HIT(MN) and IP(cluster server 2) for global reachability for the MN.

On receiving Ext\_UPDATE packet 1, RVS will respond with a new extended packet (Ext\_UPDATE packet 2) which contains the IP address for the old cluster server (i.e. IP (cluster server 1)) along with the received HIT (MN) and a flag called "auth" is added to indicate that the new attached MN is an authenticated MN since it has already a binding record at RVS and moved from an authenticated cluster server. (We assumed that all registered cluster servers at RVS are authenticated).

On receiving Ext\_UPDATE packet 2, cluster server 2 converts the temporary binding for the MN to a permanent binding, and instructs HProxy2 to establish a tunnel with the previous HProxy (i.e. HProxy1) which belongs to the previous cluster server1; to maintain the ongoing data traffic between the MN and its CN flows through cluster server 1, and this significantly will reduce the packet loss during the handover process. In contrast, all new communications is established directly through cluster server 2 and not via cluster server1. cluster server 2 can establish a new SA if it is necessary.

In the above sections, we explained in details the inter-handover process when the MN moves between two different domains that managed by different cluster servers.

In the following sub section, we will describe how a non-HIP MN can perform handover with different domains. Figure 4.7 depicts the inter-handover procedure (both MN and CN belong to different domains).

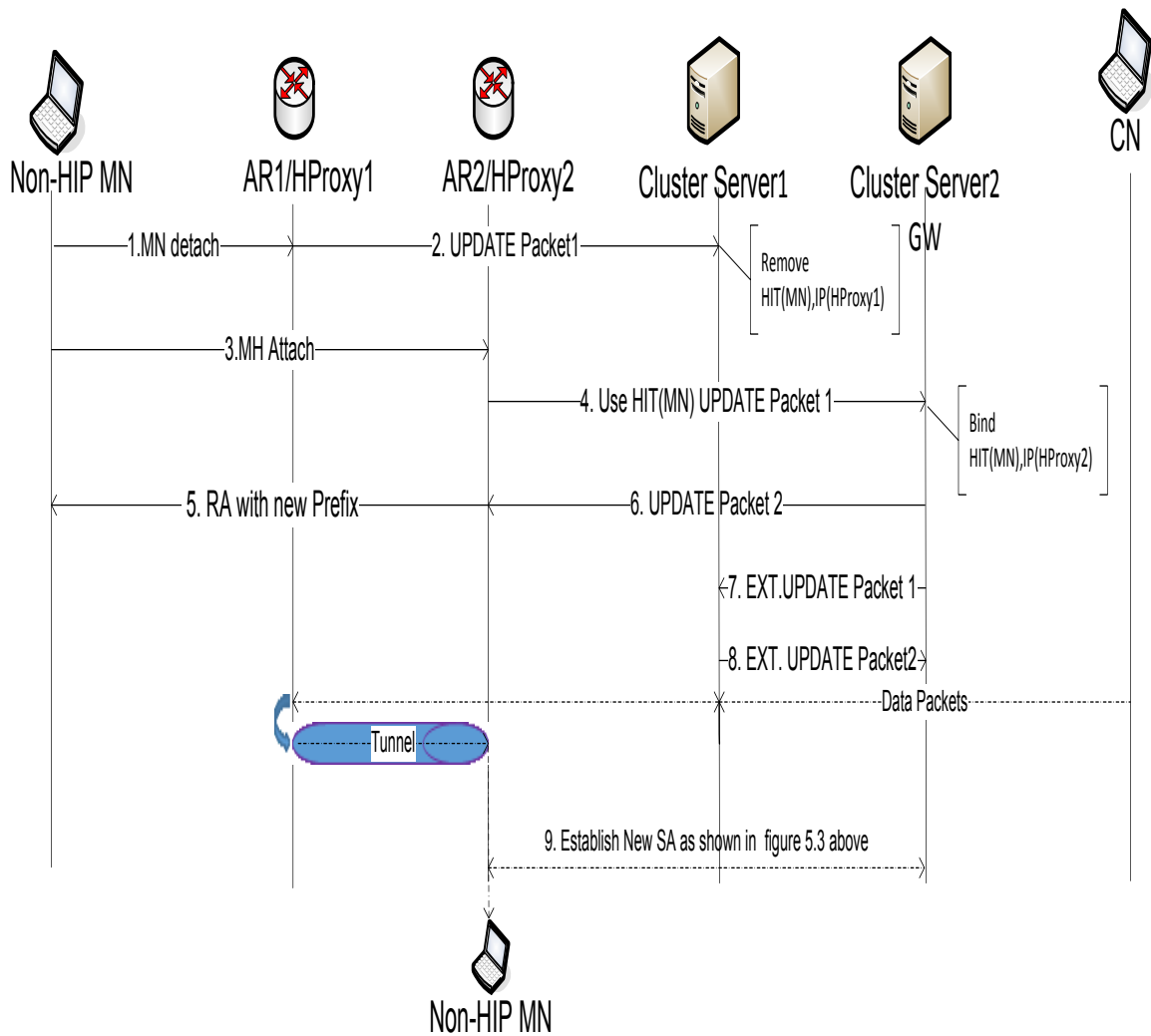


Figure 4.7 Non-HIP MN Inter-Handover Procedure (MN, CN in different Domains)

To manage the non-HIP MN inter handover, a little bit of changes to the above scenario will be carried out, since non-HIP MN hasn't HIT, HProxy 2 will act as a HIP proxy and assigns a HIT and register MN binding information with the cluster server 2 by sending the a normal UPDATE pkt1, thereafter, cluster server2 will send a normal UPDATE packet2 which contains a new network prefix (HNP) to HProxy 2. Yet, HProxy 2 sends a RA to the attached MN to a configure a new IP, in addition, HProxy 2 derives the HNP from current MN IP address.

If the extracted HNP is not belongs to the current domain (i.e. cluster server 2) means it belongs to another domain, cluster server 2 determines the cluster server to which the HNP belongs. To accomplish this, cluster server 2 indexes it is list of neighboring and authenticated cluster servers based on the received HNP. In addition, cluster server 2 creates a temporary binding for the MN and sends a normal UPDATE packet (UPDATE pkt2) with

the HNP to HProxy2. Furthermore, cluster server 2 sends an extended UPDATE packet (Ex\_UPDATE packet1) to cluster server 1. Ex\_UPDATE packet1 is created by adding a new flag (E) to the first UPDATE packet of HIP. The rest of the first UPDATE packet remains unchanged.

On receiving Ex\_UPDATE packet1, cluster server 1 uses the HNP of the MN to find the MN binding and consequently sends the MN information in an extended UPDATE packet (Ex\_UPDATE packet2) to cluster server 2. Ex\_UPDATE pkt2 is created by adding a new flag (E) to the second UPDATE packet of HIP. The rest of the second UPDATE packet remains unchanged. It is important to note that flag (E) enables the cluster servers to differentiate between UPDATE packet senders, HIP proxies or other cluster servers.

On receiving Ex\_UPDATE packet 2, cluster server 2 compares information in Ex\_UPDATE packet 2 sent by cluster server 1, against information in UPDATE packet1 sent by HProxy2. If the necessary information from cluster server 1 differs from that sent by HProxy2, cluster server 2 instructs HProxy2 to stop serving the MN and to remove the related binding (may be an attacker or vandalism). If the required information is the same, cluster server 2 converts the temporary binding for the MN to a permanent binding.

For the MN's reachability directly through cluster server 2 (i.e. not via cluster server 1), cluster server 2 updates the MN binding at the RVS. From the content of the Ex\_UPDATE packet 2, the cluster server 2 can reuse the established SA. Furthermore, cluster server 2 continues to deliver the MN data in a secure way. In contrast, cluster server 2 can establish a new SA if necessary. However, this new SA establishment adds some delay to the handover latency. To this end, the ongoing data traffic between the MN and its CH flows are tunneled through cluster server 1 (In contrast, all new communications is established directly through cluster server 2 not via cluster server 1).

It is important to note that we provide a network-based micro/macro-mobility support at HIP layer but not at IP layer as do some proposed solutions [84, 85]. Solutions [84, 85] are about using of PMIPv6 to support HIP MNs. Yet, both solutions are using IP technology to support host mobility for HIP MN. The main difference between providing mobility supports at HIP layer and at IP layer is that the first can utilize all the HIP features, which are security, multi-homing, interoperability between IPv6/IPv4, and mobility.

Results are deeply analyzed and discussed in the following chapter.

# CHAPTER FIVE

## RESULTS ANALYSIS AND DISCUSSION

### 5.1 Performance Analysis:

Since HADMM extends different types of mobility management paradigm, network-based and HIP (MHPP) and host-based DMM (HDMM-TH), respectively, the performance analysis will be tackled differently, figure 5.1 depicts the diagram used for this evaluation.

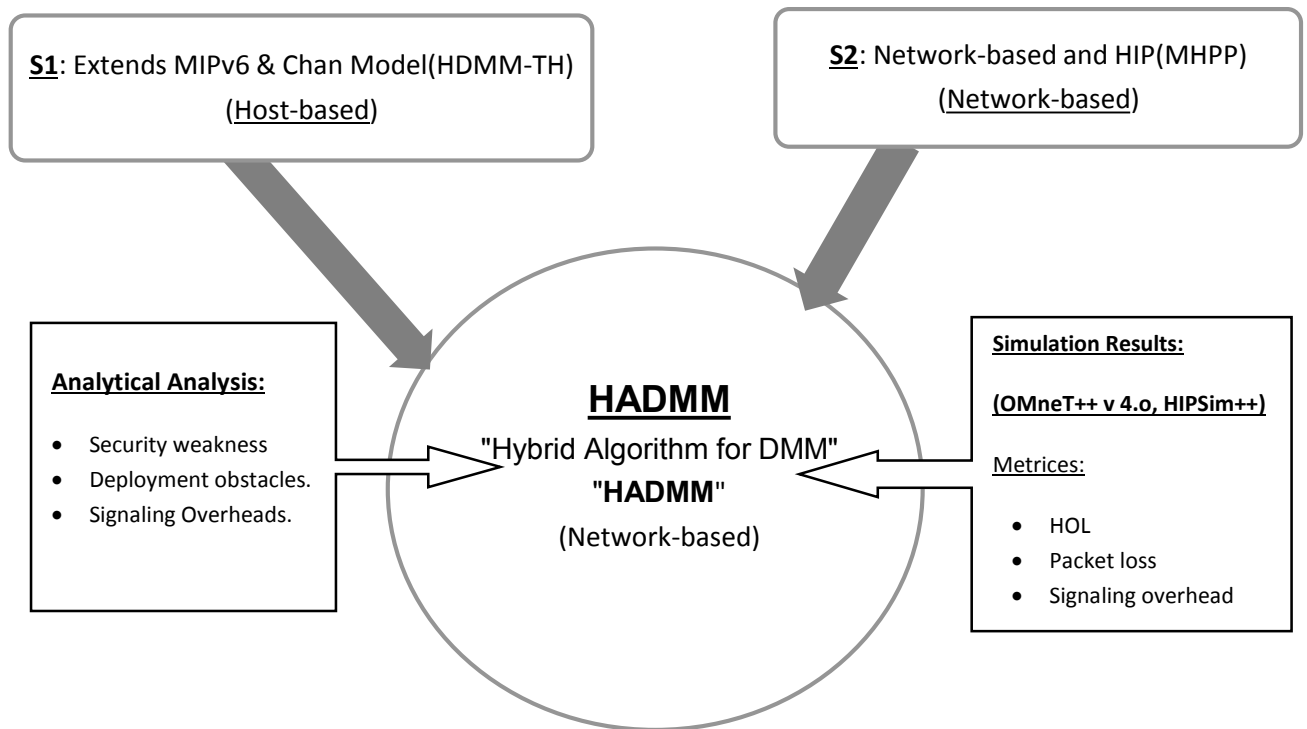


Fig 5.1 Performance Evaluation diagram

Analytical model and simulator framework are utilized to implement HADMM, and we compare the handover latency (HOL), mobility relating signals and data packet loss of HADMM (intra-handover), MHPP [47] and standard PMIPv6, respectively.

In addition, we have deeply analyzed the short comes of HDMM-TH (chapter 3) that have been addressed by HADMM.



The analytical analysis is mainly focus on the security weaknesses, deployment obstacles and signaling overheads of HDMM-TH.

## **5.1.1 HADMM Performance Analysis:**

### **5.1.1.1 Analytic evaluation of handover performance:**

The following section briefly compares the basic Handover Latency (HOL) involved in PMIPv6, MHPP [47] and our (HADMM) in intra-handover mode.

We developed an analytic model based on explanations of Figure 4.5(chapter 4) to measure HOL and mobility-related signaling overheads of PMIPv6, MHPP and HADMM, respectively. Note that CN is in the same domain. Another issue to be considered is that the receipt of the UPDATE packets depends on the distance between the MN and the respective CN. The sequence of the UPDATE packets in Figure 4.5 is one of possible exchanges that can take place in real networks.

#### **5.1.1.1.1 PMIPv6**

We assumed that the MN has ongoing communications with CN. When a MN moves from one PoA to another, the following HOL components are involved:

- The latency due to the MN's movement detection (MD) at IP layer in MN's stack,  $L_{MD}$ .
- The latency when the new MAG sends a PBU request message on behalf of the MN to the LMA in order to register the current PoA of the MN,  $L_{PBU}$ .
- The delay due to the reply message PBA sent by the LMA in response to the PBU sent by the n-MAG,  $L_{PBA}$ .
- The delay due to MN's profile acquisition hence authorization and authentication to verify eligibility of the MN to receive network-based mobility management services,  $L_{AAA-auth}$ .
- The delay due to router solicitation or any other mechanism used to enable the new MAG to detect the attachment of the MN,  $L_{RS}$ .
- The latency due to the sending of router advertisement message (RA) from the current MAG to the MN to ensure that the MN maintains the same IP address configuration,  $L_{RA}$ .

Thus, the HOL due to PMIPv6 scheme is as follows:

$$L_{PMIPv6} = L_{MD} + L_{PBU} + L_{PBA} + L_{AAA-auth} + L_{RS} + L_{RA} \dots \dots \dots (1)$$

- $L_{AAA-auth}$  is affected by the number of concurrent MN's attached to the serving MAG mobility, since the MAG has to query the AAA to verify the eligibility of the MN to receive network-based mobility management services, and to receive the response message from AAA server, moreover, LMA also has to verify the eligibility of the MN by communicating with AAA server, however, the total number of authentication messages can be expressed by this simple equation:

$$L_{AAA-auth} = 4 * n \dots \dots \dots (2)$$

Where,  $n$  is the number of concurrent MN's attached to the MAG. While **4** indicated the number of the required authentication messages that exchanged by three parties ( 2 msgs between MAG and AAA, 2 msgs between MAG and AAA server) to verify the eligibility of the attached MN.

**5.1.1.1.2 MHPP and HADMM:**

MHPP and HADMM follow similar handover management for intra-handover, and figure 4.5 (in chapter 4) shows an explanation of handover management used by both. Again, based on the same assumptions to evaluate handover performance for PMIPv6, we developed an analytic model to measure HOL and mobility-related signaling overheads while the MN has ongoing communications with CN.

MHPP and HADMM have the following components contributing to HOL:

- The latency due to the MN's attachment detection (AD) at IP layer in POA's stack, hereafter is called attachment rather than MD,  $L_{AD}$ .
- Latency due to the two-way two-way location update protocol, between HProxy/LRVS and Hproxy/cluster server,  $L_{LU1} + L_{LU2}$ , respectively.

Thus, the HOL due to MHPP and our proposed scheme HADMM is as follows:

$$L_{MHPP} = L_{AD} + L_{LU1} + L_{LU2} \dots \dots \dots (3)$$

$$L_{HADMM} = L_{AD} + L_{LU1} + L_{LU2} \dots \dots \dots (4)$$

A comparison between equations 1, 3 and 4 that describe HOL of the PMIPv6, MHPP and HADMM shows the advantages of equations 3 and 4 over equation 1 (no  $L_{AAA-auth}$ ) respectively, because MHPP and our proposed algorithm reduce LU latency and the number of the required UPDATE packets as well as eliminate messages and latency related to authenticate procedure as mentioned in equation 1.

To clearly show the differences between the PMIPv6, MHPP and HADMM, we summarized the mobility-related signaling overheads in table 5.1.

It is important to note that the number of mobility related-messages for inter-domain handover doesn't depend on the number of correspondent nodes (CNs) to which a MN has an active session.

It is also important to note that the MHPP and HADMM need not consult any third party for security purpose as they have capabilities of self-certifying because of the HIP layer. However PMIPv6 needs to consult a third party for security purposes (i.e. AAA server). This third-party security consultation incurs costs of additional signaling overheads and adds some delay to the total HOL.

In addition, MHPP and PMIPv6 are prone to single point of failure problem since they depend on a single centralized entity (i.e. LRVS, LMA) to handle all control messages however, HADMM avoid this issue by introducing a cluster server which provides high reliability due to existence of more than one stand by servers.

Table 5.1 Signaling overheads of HADMM, MHPP and PMIPv6 for intra/inter-domain handover.

Parameter	Intra-Handover Signals			Inter-Handover Signals		
	HADMM (Intra-HO)	MHPP (Intra-HO)	PMIPv6 (Intra-HO)	HADMM (Inter-HO)	MHPP (Inter-HO)	PMIPv6 Not Supported
# Of UPDATE packets per handover when communicating with n CNs?	<b>2</b>	<b>2</b>	<b>7</b>	<b>4</b>	<b>6</b>	
Signalling overheads on MN interface?	No	No	No	No	No	
Signalling overheads due to configure new IP address?	No	No	No	No	No	
Signalling overheads for consulting a third party for security purpose?	No	No	Yes	No	No	
Are there any signalling overheads due to contact with centralized mobility entity?	No	Yes	Yes	No	Yes	

## **5.1.1.2 Simulation and Results:**

### **5.1.1.2.1 Motivations for selecting OMNeT++ Simulator:**

We have looked at a number of widely used network simulators, including ns-2 and OPNET. Ns-2 is a popular network simulator among the network research community which is available for download at no costs. However, ns-2 is difficult to use and has a steep learning curve. But ns-2's split-programming model remains a barrier to many developers. OPNET is a commercial package which has a comprehensive model library, user-friendly graphical user interface (GUI), and customizable presentation of simulation results. However, OPNET is a very expensive package even though the package is offered under University academic programs. However, OPNET IT Guru is available at no costs for educational use but it has very limited functionality.

The motivations of using OMNeT++ v4.0 as a network simulator in our study are:

- It offers the combined advantages of ns-2 and OPNET.
- Free of cost (Open source).
- Our HIP simulator (HIPSim++) is only built on OMNET++ V4.0.(main motivation).
- Has a rich graphical user interface (GUI).
- Easy to use, more flexible in model development, modification and validation, and incorporates appropriate analysis of simulation output data.
- OMNeT++ has all the features of a good simulator.

### **5.1.1.2.2 Simulation Setup:**

As stated above, OMNeT++ v4.0 network simulator[75] and HIPSim++[86] simulator framework are utilized to implement the Hybrid algorithm (HADMM) design, the handover of our algorithm, MHPP[47] and PMIPv6 is each carried out in two partially overlapping IEEE 802.11b (11 Mbps peak data rate) sub-networks, the sub-networks implement MHPP[47], PMIPv6 and HADMM. In PMIPv6 the MAGS are co-located with the access routers, while in both MHPP and HADMM the mobility-HIP proxies are co-located with access routers. That is mobility in sub-network 1 and sub-network 2 is managed by MAG1 and MAG2 for PMIPv6, whereas in both MHPP [47] and HADMM it is managed by mobility-HIP proxy 1 and mobility-HIP proxy 2, respectively. The simulation topology is

typically to what is explained in figure 4.2(chapter 4), and the simulation Parameters are described in table 5.2. (We considered only the intra-handover scenario-HADMM), figure 5.2 depicts the simulation topology of HDMM developed using OMNET++ v.4.0

A HIP-enabled CN is fixed outside the access network to which the non-HIP MN is currently attached. Data is exchanged between the MN and CN at the rate of 15 Kbps and are in the form of 256-byte UDP packets. For simplicity we only consider data flow from the CN to the MN. The handover is simulated with the MN moving linearly at a constant speed 1m/s from one subnet to the other.

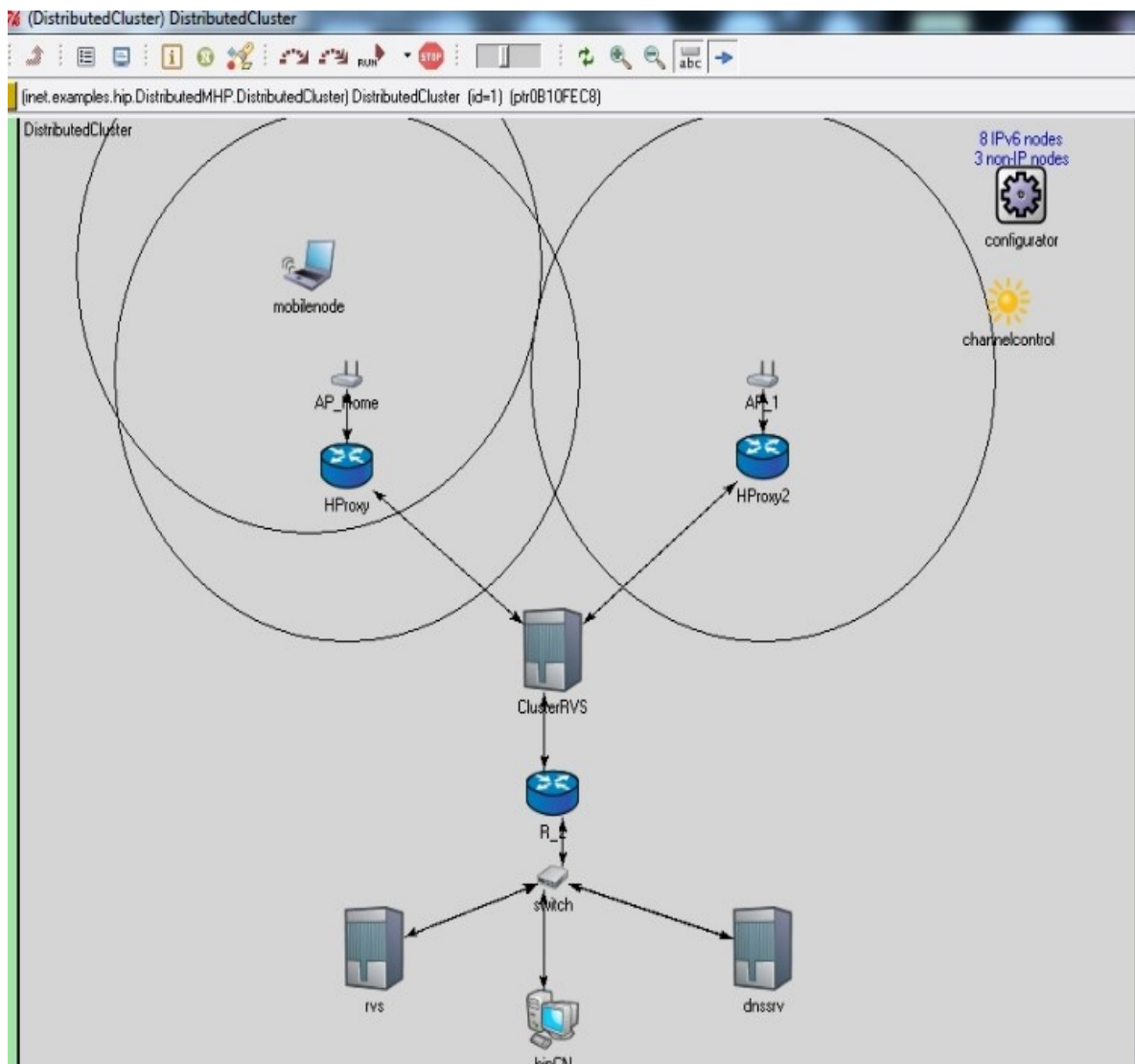


Figure 5.2 HADMM simulation topology (in OMNET++ v.4.0)

Table 5.2: Simulation Parameters under which MHPP, HADMM and PMIPv6 are examined

Parameter	Value	Parameter	Value	Parameter	Value
Speed	1 m/s	Mobility Model	Rectangle	Route Adv(RA) interval	0.3 – 0.7s
# of POA	2	Packet flow	Bi-dir CBR	AP Power	2.0 mW
# of MN	1	UDP Packet transmit Rate	0.13 s	Beacon Freq	0.1 s
Grid Size(m <sup>2</sup> )	850 x 850	Packet Size	256 byte	# of CN	1

### 5.1.1.2.3 Simulation Results:

In our investigation, we evaluated and compared the handover performance of MHPP, PMIPv6 and HADMM in terms of handover latency, signalling overhead and packet loss.

In our proposed algorithm, we defined handover latency (HOL) as the time difference between the time when the MN is able to receive packets from the new point of attachment (PoA) and the time when the MN is unable to receive packets in the old of PoA.

We also, define the packet loss as the number of lost packets in the downstream traffic (from CN to MN) during handover. In addition, signalling overhead is quantified in the terms of the number of mobility-related signalling messages per handover (in our investigation we carried out one hundred (100) handovers for each model).

As we can observed from the figure 5.3, the difference in the handover latency for the three schemes is clear over the first 5 handovers, and our HADMM has the least handover latency.

In fact, handover latencies of the MHPP, PMIPv6 and HADMM are 0.6, 2.5 and 0.4 sec, respectively.

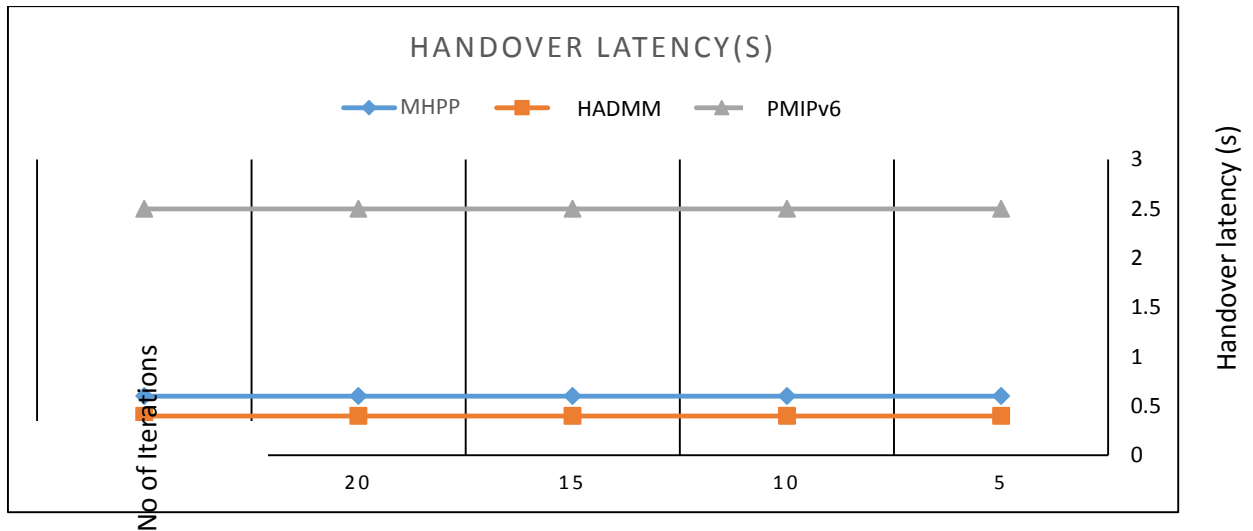


Figure 5.3 the first 30 handoffs for MHPP, HADMM and PMIPv6

This improvement in handover performance can be attributed to the following: (1) DAD and movement detection latency, (2) The distances between LRVS/Cluster server and the two mobility-HIP proxies/MAGs are the same for both(back, forth) handovers between the sub networks.(3) The attachment detection time of MN is the same in the three schemes.

Unlike MHPP and HADMM, PMIPv6 experiences additional delay due to authentication process at third party and sending extra packets to send the previous PoA address to CN.

Handover related messages in HADMM, PMIPv6 and MHPP during 15000 sec simulation time are illustrated in figure 5.4.

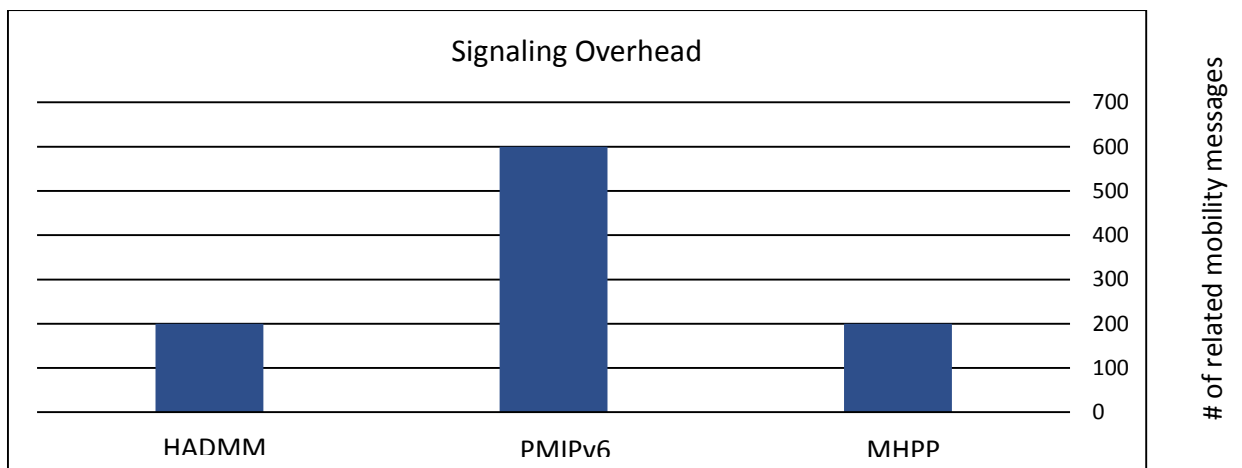


Fig 5.4 handover-related messages of HADMM, PMIPv6 and MHPP

It is obvious from figure 5.4 that HADMM and MHPP for intra-handover have the same related mobility messages, which PMIPv6 has the highest signalling overhead this because

both of the schemes use two-way location update protocol. It is important to note that both of HADMM and MHPP are not required to consult any third party as PMIPv6 to ensure secure sessions since they have capabilities of self-certifying in the HIP layer (self-certifying means by giving a HIT, computationally it is so hard to get the same HIT ).

We measured the packet loss from traffic, data packets of UDP application going between CN and MN during handover process. The inter-arrival rate of data packet was kept constant in all the cases. From the packet loss measurements, we observed that the number of packet loss is proportional to the HOL. Compared with the MHPP and PMIPv6, our proposed algorithm achieved the lowest HOL, and thus the average number of lost packets was 6 packets per 100 handovers, whereas MHPP and PMIPv6 lost 9, 38 packets, respectively. Figure 5.5 illustrates the packet loss for the three schemes.

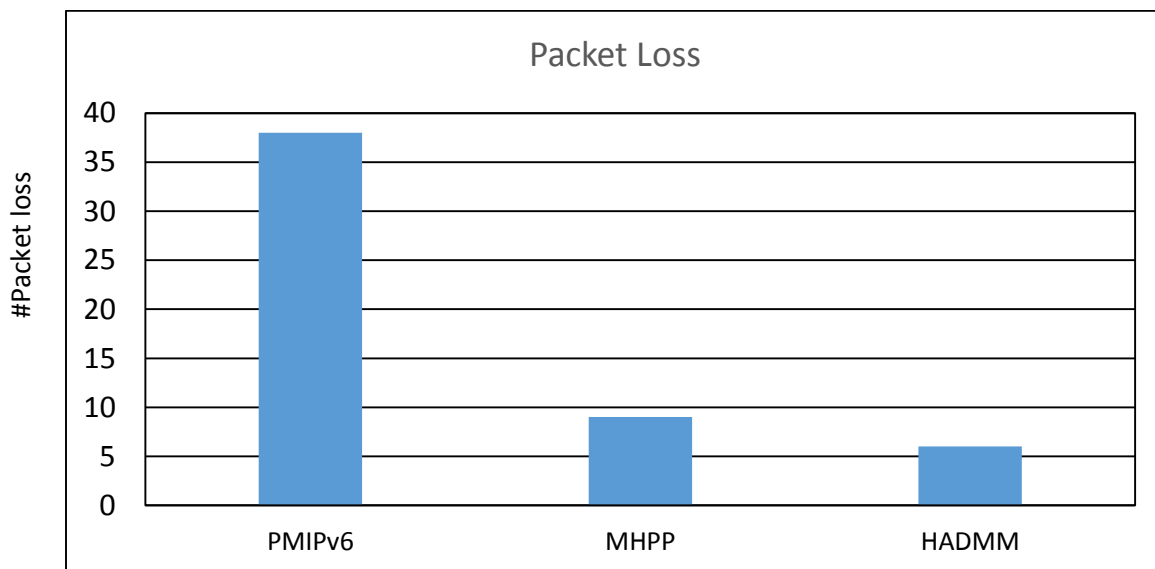


Figure 5.5 the average packet loss of PMIPv6, MHPP and HADMM (100 Ho).

## 5.1.2 HDMM-TH and HADMM Performance Analysis:

### 5.1.2.1 Analytic evaluation:

The following section deeply analysis the limitations experienced by HDMM-TH scheme that proposed in chapter 3 and have been solved by HADMM.

Our analysis is based on explanations of Figure 3.2(in chapter 3) and figure 4.6 (in chapter 4) Respectively, Note that both MN and CN are belong to different domains (inter-handover).



This section provides the analytic evaluation of HADMM. The analysis is conducted considering the following three key performance metrics i) deployment obstacles ii) security weaknesses; iii) signaling overheads.

- Since HDMM\_TH is a host-based scheme, thus it requires a modification of the mobile nodes' IP stack (Hosts must implement MIP in the kernel) to support mobility management, indeed this is a tedious task that limits the deployment of such scheme however, our hybrid algorithm is a network-based and doesn't require the involvement of the MN in any mobility issues and all signaling overheads on the MN interface are removed, moreover, the deployment of HIP proxy in HADMM provides mobility support for HIP-enabled and non-HIP enabled MNs.
- HDMM\_TH lacks elegant secure mobility support, since it depends on the security mechanism provided by the standard MIPv6, which is IPsec, that is prone to the man-in-middle and denial of service(DOS) attacks, consequently, all binding update and binding acknowledgment messages that exchanged between the three communication parties(MN, mobility anchors-MCs and CN ), respectively, to perform location update these messages must be encrypted to support location privacy, Yet, depending on IPsec security mechanism adds a significant signaling overheads, however HADMM benefits from self-certifying feature provided by HIP layer, and all upper layers applications( Transport and interworking layers ) are bound to HIT of MN instead of the routing address (IP) and MN is capable of receiving packets while it is roaming in the internet. HIP features prevent both compromising the location privacy and aforementioned attacks.
- HDMM-TH has to send  $3 * (n-1)$  data packets (see figure 3.2 in chapter 3), where n refers to number of CNs that have an active sessions with MN, and 3 indicated the number of packets that handle the previous PoA address (previous CMA IP) to notify CNs with the movement of the MN and to redirect the new packets to the serving mobility anchor (i.e.CM) address. Unlike HADMM, it needs only four additional messages (two messages exchanged between HProxy and cluster server for location update to confirm the reachability of the MN, and two messages are exchanged between the old and new cluster servers to request/send the security context associated with the MN, see figure 4.6 in chapter 4 ) consequently, it is important to note that the number of mobility related-messages for inter-domain handover doesn't depend on the number of correspondent nodes (CNs) to which a MN has an active sessions.

- Double jump problem (moving of the two communication parties simultaneously) is not addressed in HDMM-TH, but this method is managed by HADMM in a simple way.

As a final word, the proposed hybrid algorithm (HADMM) is a novel DMM algorithm that provides both micro and macro mobility management solutions, HADMM is completely built on a HIP layer and inherits all HIP features as security, seamless mobility, multihoming and IPv4 and IPv6 interoperability, Yet, the author recommends internet service providers (ISPs) and vendors to deploy the proposed algorithm in their networks in the future.

## 5.2 Summary

In this chapter, a novel hybrid algorithm for distributed management (HADMM) that combines two schemes HDMM-TH [51], which is proposed by the author in chapter 3 and the proposed scheme in [47], respectively, to overcome their limitations and to work in a distributed way.

HADMM is deployed on HIP layer to provide efficient, secure, network-based seamless mobility as well as multihoming support to all MNs, with all signaling overheads on the MN interface removed. Unlike ordinary HIP Proxy solutions, this design eliminates the issue of a single-point-of-failure due to services being received only via static HIP proxies, to end that, the author favored using of cluster server which provides a reliable services due to the presence of more than one server for any malfunction, in addition to load balance.

The network-based services include tracking mobile hosts, assigning network prefix per host identifier, securely updating the binding of mobile nodes, and providing the same IP prefix to the mobile host during the handover process in the same network domain (intra-handover).

The analytic and simulation results show that HADMM handover performance in terms of handover latency, packets loss and signaling overhead is much better compared with MHPP[47] and PMIPv6, nevertheless, analytical analysis of the HADMM shows it is effectiveness in addressing the short comes experienced by HDMM-TH(chapter 3).

To the author's best knowledge, HADMM is the first partially distributed mobility management algorithm completely built on HIP layer.

# CHAPTER SIX

## CONCLUSION AND FUTURE WORKS

This chapter concludes the thesis and gives some recommendations for future works in the promising area distributed mobility management (DMM).

### 6.1 Conclusions

We conclude that distributed mobility management (DMM) paradigm can be used to solve the current mobile internet network limitations, and can play a vital role in introducing the next generation network (All-IP network) in near future.

Hybrid Algorithm for Distributed Mobility Management (HADMM) can provide mobility solution for both HIP enabled and non-HIP enabled mobile nodes at HIP layer.

Cluster server can be deployed to eliminate the single point of failure issue, bottle neck and to perform location update for roaming mobile nodes, moreover, HIP proxy can be used to track the mobile node and update it is current location at the cluster server for global reachability, and assigning host identity tag (HIT) for none HIP mobile nodes from it is pool.

Simulation results show that the new hybrid algorithm (HADMM) performs better than MHPP and PMIPv6, in terms of handover delay, signaling overhead and packet loss. Simulation results achieved a 0.4, 0.6 and 2.5 handover delay for the three algorithms. Results for signaling overhead delay are 200, 2000 and 600 and results for packet loss are 6, 9 and 38 for the three algorithms, respectively.

Analytic analysis proved the evidence that the proposed hybrid algorithm HADMM has addressed all the limitations experienced by HDMM-TH (chapter 3), which include the security weaknesses, deployment obstacles and signaling overheads.

The proposed hybrid algorithm (HADMM) was developed and evaluated using OMNET++ ver 4.0 and HIPSIm++ simulators, respectively, and the obtained results have been investigated and analyzed deeply.

We hope that our success in applying Host Identity Protocol (HIP) in developing a distributed mobility management algorithm might inspire others to adopt similar algorithms and thus make a positive effort towards standardizing the DMM protocol.

To the author's best knowledge, HADMM is the first partially distributed mobility management algorithm completely built on HIP layer.

## 6.2 Future Works

Presently, no standard protocol has been introduced by IETF for DMM, and only individual submissions are discussed, however, there are still some areas that could be extended before coming at a full solution. There are some open points that are currently under discussion as:

a) Application-based mobility management: from the comparison of centralized and distributed mobility management protocols, we have noticed that in some cases a centralized approach is favored, and in some cases the DMM approach is fits best, hence the author suggests developing a hybrid deployment where DMM and CMM co-exist switching between the two approaches depends on the specific needs. Specifically, these needs are determined by the user's applications, those applications that can stay alive after an IP address changes do not need mobility support at all, so a DMM approach is preferable, however, those applications that generate long flows for which a CMM solution is preferable, and those that generate short flows, for which a DMM approach is suitable.

The research challenge in this scenario is, how do we identify the application behavior (short flow or long flow), and if this information is available, how to switch between the two schemes.

b) The performance evaluation and analysis of the proposed algorithm has been done by using analytical modelling and simulation. However, it will be stimulating to test the algorithm with a real-life implementation or test-bed, so as to disclose the real-world performance of the algorithm.

c) Improving the security of the proposed algorithm without relying on HIP mechanism is required, since the authentication and authenticity process defined by HIP protocol adds considerable delay which is not preferable for real time applications, hence further studies are required to improve the security of the proposed algorithm.

d) Our proposed algorithm, HADMM is deployed as a partial DMM algorithm, a further research can focus on re-designing the same algorithm with some modifications to work as a full distributed DMM, and this can be carried out by finding a mechanism to find out the information of the previous point of attachment (PoA) of the roaming Mobile Node (MN) (i.e. IP of the previous cluster server), moreover, deploying of a new version of HIP protocol since the current version is too slow.

## 6.3 Main Contributions

The major contribution of this study can be summarized as follows:

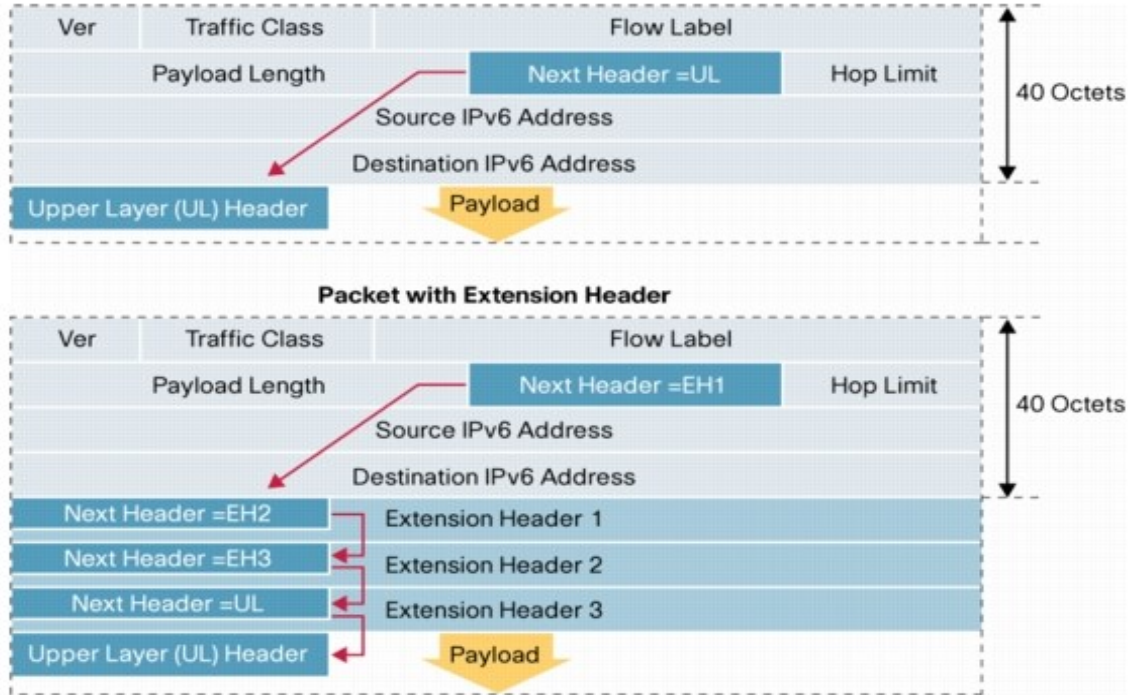
- 1- Developing of a novel hybrid algorithm for distributed mobility management (HADMM) based on Host Identity Protocol (HIP) to provide mobility services for both HIP enabled and non-HIP enabled mobile hosts at HIP layer. HADMM benefits from all HIP features such as self-certifying, security seamless mobility, multihoming and IPv4 and IPv6 interoperability.
- 2- Deploying of HIP proxy to provide mobility support for HIP enabled and non-HIP enabled (legacy) mobile hosts which will motivate the operators to adopt the proposed hybrid algorithm.
- 3- The proposed hybrid algorithm has solved the single point of failure and bottle neck problem by deploying the cluster server which increases the reliability and offers load balance for the whole network.
- 4- The proposed hybrid algorithm provides both micro and macro mobility supports for all roaming mobile nodes in a secure and efficient way.
- 5- Better handover performance in terms of handover latency, signaling overheard and data packet loss is achieved when comparing with one of leading contenders, MHPP [47], and the analytical analysis proved the effectiveness of HADMM in addressing the limitations experienced by HDMM-TH which is proposed and published by the author as an enhanced scheme proposed in [50].The results discussed and summarized in chapter 5.

To the author's best knowledge, HADMM is the first partially distributed mobility management algorithm completely built on HIP.

# APPENDICESE

## Appendix A

### Extension Header (EH) of IPv6 Packet



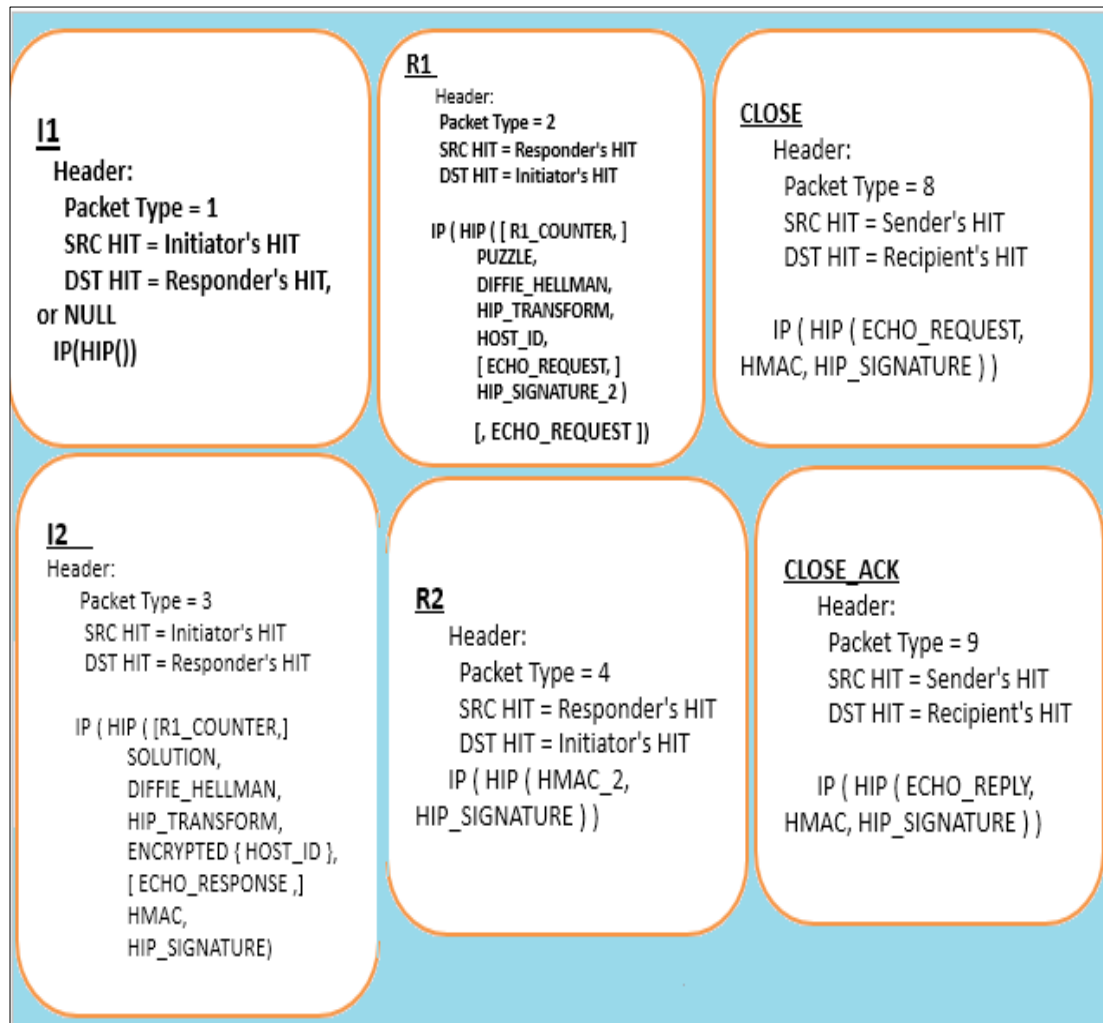
RFC2460 defines the extension headers as shown in the following table along with the Next Header values assigned to them:

**Table 1.** IPv6 Extension Headers and their Recommended Order in a Packet

Order	Header Type	Next Header Code
1	Basic IPv6 Header	-
2	Hop-by-Hop Options	0
3	Destination Options (with Routing Options)	60
4	Routing Header	43
5	Fragment Header	44
6	Authentication Header	51
7	Encapsulation Security Payload Header	50
8	Destination Options	60
9	Mobility Header	135
	No next header	59
Upper Layer	TCP	6
Upper Layer	UDP	17
Upper Layer	ICMPv6	58

## Appendix B

### HIP Base Exchange Packets(mutual authentication and integrity)







```

Simulation - DHMPsolution/src/hip/RvsHIP.h - OMNeT++ IDE
File Edit Refactor Source Navigate Search Project Run Window Help

Project Explorer
hip
├── FSMBaseMsg_m.cc
├── FSMBaseMsg_m.h
├── HIP.cc
├── HIP.h
├── HipFsm.cc
├── HipFsm.h
├── HipFsmBase.cc
├── HipFsmBase.h
├── HipMessages_m.cc
├── HipMessages_m.h
├── LrvsHIP.cc
├── LrvsHIP.h
├── RvsHIP.cc
├── RvsHIP.h
├── sara.cc
├── SrvsHIP.cc
├── SrvsHIP.h
├── cppfiles.lst
├── FSMBaseMsg.msg
├── HIP.ned
├── HipFsm.ned
├── HipMessages.msg
├── LrvsHIP.ned
├── Makefile.vc
├── msgfiles.lst
├── nedfiles.lst
├── RvsHIP.ned
└── SrvsHIP.ned

RvsHIP.h
//*****
// File:      RvsHIP.h
//
// Authors:   Laszlo Tamas Zeke, Levente Mihalyi, Laszlo Bokor
//
// Copyright: (C) 2008-2009 BME-HT (Department of Telecommunications,
// Budapest University of Technology and Economics), Budapest, Hungary
//
// This program is free software; you can redistribute it and/or
// modify it under the terms of the GNU General Public License
// as published by the Free Software Foundation; either version 2
// of the License, or (at your option) any later version.
//
// This program is distributed in the hope that it will be useful,
// but WITHOUT ANY WARRANTY; without even the implied warranty of
// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
// GNU General Public License for more details.
//
// You should have received a copy of the GNU General Public License
// along with this program; if not, write to the Free Software
// Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
//
//*****
// Part of: HIPSIM++ Host Identity Protocol Simulation Framework developed by BME-HT
//*****

#ifndef _RVSHIP_H_
#define _RVSHIP_H_

#include <HIP.h>
#include "IPv6Address.h"

```

```

Simulation
Project Explorer
includes
src
├── applications
├── base
├── hip
├── FSMBaseMsg_m.cc
├── FSMBaseMsg_m.h
├── HIP.cc
├── HIP.h
├── HipFsm.cc
├── HipFsm.h
├── HipFsmBase.cc
├── HipFsmBase.h
├── HipMessages_m.cc
├── HipMessages_m.h
├── LrvsHIP.cc
├── LrvsHIP.h
├── RvsHIP.cc
├── RvsHIP.h
├── SrvsHIP.cc
├── SrvsHIP.h
├── cppfiles.lst
├── FSMBaseMsg.msg
├── HIP.ned
├── HipFsm.ned
├── HipMessages.msg
├── LrvsHIP.ned
├── Makefile.vc
├── msgfiles.lst
├── nedfiles.lst
├── RvsHIP.ned
└── SrvsHIP.ned

HIP.cc
//*****
// Part of: HIPSIM++ Host Identity Protocol Simulation Framework developed by BME-HT
//*****

#include "HIP.h"
#include "IPv6ControlInfo.h"
#include "IPv6ExtensionHeaders_m.h"
#include "IPv6Datagram.h"
#include "DNSBaseMsg_m.h"
#include "DNSRegRvsMsg_m.h"
#include "UDPPacket.h"
#include "UDPControlInfo_m.h"
#include "IPAddressResolver.h"
#include "InterfaceTableAccess.h"
#include "InterfaceTable.h"
#include "InterfaceEntry.h"
#include "IPv6InterfaceData.h"
#include "NotificationBoard.h"
Define_Module(HIP);
HIP::HIP() {};
// Default destructor
HIP::~HIP() {
    if(!listHITtoTriggerDNS.empty())
        for(listHITtoTriggerDNSIt = listHITtoTriggerDNS.begin(); listHITtoTriggerDNSIt != listHITtoTriggerDNS.end()
            delete listHITtoTriggerDNSIt->second;
};

// Initializing the module's basic parameters
void HIP::initialize()
{
    ev << "Initializing HIP...\n";

    expectingDnsResp = false;
    expectingRvsDnsResp = false;

```

## Appendix D

### HADMM Implementation Codes

**File : HIP.h**

```
#ifndef __HIP_H__
#define __HIP_H__

#include <omnetpp.h>
#include "IPv6Address.h"
#include "HipMessages_m.h"
#include "InterfaceEntry.h"
#include "INotifiable.h"
#include "INETDefs.h"

#define HIP_START_CHECK_TIMER 4
#define HIP_CHECK_TIMER 0.1

class INET_API HIP : public cSimpleModule, public INotifiable
{
protected:
    // Connection attempts waiting for DNS response
    typedef std::map<IPv6Address, cMessage *> ListHITtoTriggerDNS;
    ListHITtoTriggerDNS listHITtoTriggerDNS;
    ListHITtoTriggerDNS::iterator listHITtoTriggerDNSIt;

    bool firstUpdate;
    bool expectingDnsResp;
    bool expectingRvsDnsResp;
    int hipMsgSent;
    cOutVector hipVector;
    int currentIfId;

public:
    //constructor/destructor
    HIP();
    virtual ~HIP();

    void incHipMsgCounter();

    std::map<InterfaceEntry *, IPv6Address> mapIfaceToIP;
    std::map<InterfaceEntry *, bool> mapIfaceToConnected;

    //HIT - IP struct and mapping
    typedef struct HitToIpMapEntry{
        std::list<IPv6Address> addr;
        int fsmId;
    } HitToIpMapEntry;

    typedef std::map<IPv6Address, HitToIpMapEntry *> HitToIpMap;
    HitToIpMap hitToIpMap;
    HitToIpMap::iterator hitToIpMapIt;

    //HIT - IP struct and mapping for HIP proxy, Taj
    /* typedef struct ProxyHitToIpMapEntry{
```

```

        std::list<IPv6Address> ProxyAddr; // IP of non-HIP
host
        std::list<IPv6Address> ProxyHIT; // HIT of non-HIP
host's partner
        std::list<IPv6Address> PartnerAddr; // IP of non-
HIP host's partner
        int fsmId;
        } ProxyHitToIpMapEntry;

        typedef std::map<IPv6Address,ProxyHitToIpMapEntry *>
ProxyHitToIpMap;
        ProxyHitToIpMap ProxyhitToIpMap;
        ProxyHitToIpMap::iterator ProxyhitToIpMapIt; */

//pointer of FSM instance
cModuleType *fsmType;

//Address of the RVS
IPv6Address rvsIPAddress;

// Address of the LRVS
IPv6Address lrvsIPAddress; // added by Taj

// Address of the SRVS
IPv6Address srvsIPAddress; // added by Taj

//working variables
IPv6Address srcWorkAddress;
std::vector<std::string> address;
IPv6Address partnerHITwork;
int dstSPIwork;
int fsmIDwork;
IPv6Address partnerHIT;
HitToIpMapEntry * hitToIpMapEntryWork;

//returns the partner HIT
IPv6Address getPartnerHIT();

//returns if the HIP module is an RVS module
bool isRvs();
//returns the RVS address
IPv6Address* getRvsAddress();
//returns the working variable for source address
IPv6Address* getSrcWorkAddress();
//returns the RVS HIT
IPv6Address* getRvsHit();
//returns if there was a HIT-IP swap
//bool isAddressSwitched();
//Begin of the code added by Taj
//returns if the HIP module is an LRVS module
bool isLrvs();
//returns the LRVS address
IPv6Address* getLrvsAddress();
//returns the LRVS HIT
IPv6Address* getLrvsHit();

// SRVS
//returns if the HIP module is an SRVS module
bool isSrvs();
//returns the SRVS address
IPv6Address* getSrvsAddress();

```

```

        //returns the SRVS HIT
        IPv6Address* getSrvsHit();

        IPv6Address regdHostHIT;
        int distinguisher; // to distinguish B/W R2 packets
        int pktdirection; //0 means in and 1 means packet goes
outside the domain

        IPv6Address inProgMNregHIT; // to save the HIT of the MN
that the SRVS is registering at LRVs
        IPv6Address inProgMNregIPAddr; // to save the IP address
of the MN that the SRVS is registering at LRVs

// end of the code added by Taj

        ///void receiveChangeNotification(int category, const cObject
* details);
        virtual void receiveChangeNotification(int category, const
cPolymorphic *details); // Taj

protected:
        virtual void initialize();
        virtual void specInitialize();
        virtual void handleMessage(cMessage *msg);
        virtual void finish();

        virtual void handleMessageFromTransport(cMessage *msg);
        virtual void handleMessageFromNetwork(cMessage *msg);
        virtual void handleRvsRegistration(cMessage *msg);
        virtual void handleLrvsRegistration(cMessage *msg); // added by
Taj
        virtual void handleAddressChange();
        virtual void handleAddressChangeR(); // added by Taj
        virtual void handleAddressChangeR(MACAddress MacMN); // Taj,

        //creates a new FSM daemon, returns its pointer
        virtual cModule* createStateMachine(IPv6Address ipAddress,
IPv6Address &HIT);

        //searches for a FSM daemon by its module id
        virtual cModule* findStateMachine(int fsmID);

        bool _isRvs;
        IPv6Address _rvsHit;
        bool _isLrvs; // added by Taj
        IPv6Address _lrvsHit; // added by Taj
        bool _isSrvs; // added by Taj
        IPv6Address _srvsHit; // added by Taj

};

#endif
*****
*****

File: HIP.cc

#include "HIP.h"
#include "IPv6ControlInfo.h"

```

```

#include "IPv6ExtensionHeaders_m.h"
#include "IPv6Datagram.h"
#include "DNSBaseMsg_m.h"
#include "DNSRegRvsMsg_m.h"
#include "UDPPacket.h"
#include "UDPControlInfo_m.h"
#include "IPAddressResolver.h"
#include "InterfaceTableAccess.h"
#include "InterfaceTable.h"
#include "InterfaceEntry.h"
#include "IPv6InterfaceData.h"
#include "NotificationBoard.h"

Define_Module(HIP);

HIP::HIP() {};
// Default destructor
HIP::~HIP() {
    if(!listHITtoTriggerDNS.empty())
        for(listHITtoTriggerDNSIt = listHITtoTriggerDNS.begin();
listHITtoTriggerDNSIt != listHITtoTriggerDNS.end();
listHITtoTriggerDNSIt++)
            delete listHITtoTriggerDNSIt->second;
};

// Initializing the module's basic parameters
void HIP::initialize()
{
    ev << "Initializing HIP...\n";

    expectingDnsResp = false;
    expectingRvsDnsResp = false;
    fsmType = cModuleType::get("inet.hip.HipFsm");
    hipMsgSent = 0;
    hipVector.setName("HIP_DNS_msgs");
    currentIfId = -1;

    WATCH_MAP(mapIfaceToIP);
    WATCH_MAP(mapIfaceToConnected);
    WATCH_MAP(hitToIpMap);
    WATCH(firstUpdate);
    WATCH(expectingDnsResp);
    WATCH(expectingRvsDnsResp);
    WATCH(currentIfId);

    NotificationBoard * nb = NotificationBoardAccess().get();
    nb->subscribe(this, NF_L2 DISSOCIATED);
    nb->subscribe(this, NF_L2 ASSOCIATED_OLDAP);
    nb->subscribe(this, NF_L2 ASSOCIATED_NEWAP);
    nb->subscribe(this, NF_IPv6_HANDOVER_OCCURRED);
    nb->subscribe(this, NF_HIP_HO_INITIATION); // Taj,
// nb->subscribe(this, NF_L2 ASSOCIATED); // by Taj

    // nb->subscribe(this, NF_INTERFACE_IPv6CONFIG_CHANGED); //
added by Taj
    specInitialize();
}

void HIP::specInitialize()
{

```

```

    _isRvs = false;
    _isLrvs = false; // added by Taj
    _isSrvs = false; // added by Taj

    //partnerHIT = -1; //will get from DNS
    if ( (int)this->par("registerAtRVS") == 1)
    {
        ev <<"Hello Taj\n";
        ev << "hip init3\n";
        ev << "k" << this->par("RVSAddr").getName() << "k" <<
endl;
        //if rvsipaddress.. get rvsip from dns... scheduleAt
        scheduleAt(this->par("REG_StartTime"),new
cPacket("HIP_REGISTER_AT_RVS"));
    }
    // partener = -1; // get from source code (hard coded) added
by Taj

    firstUpdate = true;
    distinguisher = 0; // The first R2 packet to trigger RVS
registration
    pktdirection =0; // 0 means the packet goes to Host inside the
domain
}

// Changes in interface states and addresses handled here with the
help of the NotificationBoard object
//void HIP::receiveChangeNotification(int category, const cObject *
details) {
void HIP::receiveChangeNotification(int category, const
cPolymorphic *details){ // Taj
    Enter_Method_Silent();
    printNotificationBanner(category, details);

//if (isSrvs) {ev<<"Yes can be treated here"<<endl;}
/*
// Begin of the code added by Taj

    if (category==NF_INTERFACE_IPv6CONFIG_CHANGED)
    {
        ev<<"Tommorow is nice\n";
    }
// end of the code added by Taj
else
*/
// OLD AP, or disassociating, update is needed

    if (category==NF_L2 DISSOCIATED ||
category==NF_L2_ASSOCIATED_OLDAP)
    {
        InterfaceTable* ift =
(InterfaceTable*)InterfaceTableAccess().get();
        for (int i=0; i<ift->getNumInterfaces(); i++)
        {
            InterfaceEntry *ie = ift->getInterface(i);
            // IF the interface is usable
            if(!(ie->isLoopback()) && !(ie->isDown())) {
                if(mapIfaceToConnected.find(ie) ==
mapIfaceToConnected.end() || mapIfaceToConnected[ie] != ie-
>isConnected()){

```

```

mapIfaceToConnected[ie] = ie-
>isConnected();
was up
interface, start update
>getNumInterfaces(); j++){
>getInterface(j);
!(ie2->isDown()) && mapIfaceToConnected[ie2] == true) {
    handleAddressChange();
    //scheduleAt(simTime() + HIP_CHECK_TIMER, interfaceCheck = new
cPacket("HIP_UPDATE_CHECK"));
    return;
}
}
else {
    // Iface becomes connected and
it is an old AP, the address will not change
    handleAddressChange();
    return;
}
}
}
// Iface is connected to a new AP, wait for address change and
then update
else if (category==NF_L2_ASSOCIATED_NEWAP) {
    InterfaceTable* ift = (InterfaceTable*)
InterfaceTableAccess().get();
    for (int i=0; i<ift->getNumInterfaces(); i++)
    {
        InterfaceEntry *ie = ift->getInterface(i);
        // IF the interface is usable
        if(!(ie->isLoopback()) && !(ie->isDown())) {
            if(mapIfaceToConnected.find(ie) ==
mapIfaceToConnected.end() || mapIfaceToConnected[ie] != ie-
>isConnected())
                mapIfaceToConnected[ie] = ie-
>isConnected();
            handleAddressChange(); // added by Taj
            return; // added by Taj
        }
    }
}
else if (category==NF_IPv6_HANDOVER_OCCURRED) {
    // ev<<"Tomorrow is nice\n"; // added by Taj
    InterfaceTable* ift =
(InterfaceTable*)InterfaceTableAccess().get();
    std::list<IPv6Address> addrList;
    for (int i=0; i<ift->getNumInterfaces(); i++)
    {
        InterfaceEntry *ie = ift->getInterface(i);

```



```

        // IF the interface is usable
        if (!(ie->isLoopback()) && !(ie->isDown())) {
            if(mapIfaceToConnected.find(ie) ==
mapIfaceToConnected.end() || mapIfaceToConnected[ie] == true) {
                if(mapIfaceToIP.find(ie) ==
mapIfaceToIP.end() || mapIfaceToIP[ie] != ie->ipv6Data()-
>getPreferredAddress()) {
                    mapIfaceToIP[ie] = ie-
>ipv6Data()->getPreferredAddress();
                    mapIfaceToConnected[ie] = ie-
>isConnected();
                    handleAddressChange();
                    //scheduleAt(simTime() +
HIP_CHECK_TIMER, interfaceCheck = new cPacket("HIP_UPDATE_CHECK"));
                    return;
                }
            }
        }
    }
}
else if
((category==NF_L2_ASSOCIATED)&&!(category==NF_IPv6_HANDBOVER_OCCURRED
)) {
    ev << "Srvs only subscribed to this event " << endl;
    handleAddressChanger();
}
else if ((isSrvs())&&(category==NF_HIP_HO_INITIATION)) {
    ev << "Srvs only subscribed to NF_HIP_HO_INITIATION " <<
endl;
    mnMAC *mnmac = check_and_cast<mnMAC*>(details);
    MACAddress MacMN = mnmac->getMNMAC();
    ev << "From HIP, MN's MAC address is " << mnmac->getMNMAC()<<
endl; // Taj
    handleAddressChanger(MacMN);
}
}

// LRVS registration's
void HIP::handleLrvsRegistration(cMessage *msg)
{
    if (msg->isName("HIP_REGISTER_AT_LRVS"))
    {
        //delete msg;
        ev << "lrvs selfmsg arrived\n";

        if (lrvsIPAddress.isUnspecified())
        {
            ev << "LRVS's IP address is not specified\n";
            char LHIT[150] = ""; // to get the LRVS's HIT from
omnetpp.ini
            char LIP[150] = ""; // to get the LRVS's IP from
omnetpp.ini
            strcat(LHIT, par("LRVSHIT"));
            strcat(LIP, par("LRVSAddr"));
            lrvsIPAddress = LIP;
            _lrvsHit = LHIT;
            ev << "but its HIT is "<< _lrvsHit << "\n";
            ev << "Surprised!!! "<<lrvsIPAddress << endl;
        }
    }
}

```

```

        // create FSM
        sendDirect(msg, createStateMachine(lrvsIPAddress,
_lrvsHit), "localIn");
    }
    else
    {
        ev << "LRVS's IP address is specified\n";
        //sendDirect(msg, createStateMachine(rvsIPAddress,
_lrvsHit), "localIn");
    }
}

}

// RVS registration's first DNS lookup handled here
void HIP::handleRvsRegistration(cMessage *msg)
{
    if (msg->isName("HIP_REGISTER_AT_RVS"))
    {
        delete msg;
        ev << "rvs selfmsg arrived\n";
        if (rvsIPAddress.isUnspecified())
        {
            ev << "Starting dns request for " <<
par("RVSAAddr").getName() << endl;

            // Generating the DNS message
            DNSBaseMsg* dnsReqMsg = new DNSBaseMsg("DNS
Request"); //the request
            dnsReqMsg->setId(this->getId());
            char reqstring[50] = "hip-";
            strcat(reqstring, par("RVSAAddr"));
            dnsReqMsg->setData(reqstring);
            ev <<"We are asking for the " << reqstring <<
endl;

            //UDP controlinfo
            UDPControlInfo *ctrl = new UDPControlInfo();
            ctrl->setSrcPort(0);
            InterfaceTable* ift = (InterfaceTable*)
InterfaceTableAccess().get();
            ///ev<<"currentIfId1 "<<currentIfId<<endl;// must
be removed
            ctrl->setSrcAddr(ift->getInterface(currentIfId) -
>ipv6Data()->getPreferredAddress());
            ctrl-
>setDestAddr(IPAddressResolver().resolve(par("dnsAddress")));
            ctrl->setDestPort(23);
            ctrl->setInterfaceId(currentIfId);
            dnsReqMsg->setControlInfo(ctrl);

            UDPPacket *udpPacket = new UDPPacket(dnsReqMsg-
>getName());

            //TODO UDP_HEADER_BYTES
            udpPacket->setByteLength(8);
            udpPacket->encapsulate(dnsReqMsg);

            // set source and destination port
            udpPacket->setSourcePort(ctrl->getSrcPort());
            udpPacket->setDestinationPort(ctrl-
>getDestPort());

```

```

        /// ev<<"currentIfId "<<currentIfId<<endl;///
must be removed

        IPv6ControlInfo *ipControlInfo = new
IPv6ControlInfo();
        ipControlInfo->setProtocol(IP_PROT_UDP);
        ipControlInfo->setSrcAddr(ctrl-
>getSrcAddr().get6());
        ipControlInfo->setDestAddr(ctrl-
>getDestAddr().get6());
        ipControlInfo->setInterfaceId(currentIfId);
        // begin of the code added by Taj
        //if (_isLrvs == true)
        if (_isSrvs == true)
        { // if's begin
            ev << "Srvs: registers non-HIP MN at RVS\n
";
            /// ev<<"currentIfId
"<<currentIfId<<endl;/// must be removed
            ev << "The source address is " <<
ipControlInfo->getSrcAddr ()<<endl;
            ev << "The interface id is " <<
ipControlInfo->getInterfaceId ()<<endl;

            } // if's end
            // end of the code added by Taj
            udpPacket->setControlInfo(ipControlInfo);

            send(udpPacket,"udp6Out");

            hipVector.record(1);
            expectingRvsDnsResp = true; // DNS response to
resolve IP of RVS
        }
        else
        {
            sendDirect(msg,createStateMachine(rvsIPAddress,
_rvsHit), "localIn");
        }
    }
    else if(msg->isName("DNS Response"))
    {
        // Handling the DNS response
        IPv6ControlInfo *networkControlInfo =
check_and_cast<IPv6ControlInfo*>(msg->removeControlInfo());
        srcWorkAddress = networkControlInfo->getDestAddr();
        //set rvs ip, rvs HIT
        ev << "Getting partner HIT from dns resp \n";
        DNSBaseMsg* dnsMsg = check_and_cast<DNSBaseMsg
*>(check_and_cast<cPacket *>(msg)->decapsulate());
        ev << "The HIT is " << dnsMsg->data() <<endl;
        _rvsHit.set(dnsMsg->data());
        rvsIPAddress = dnsMsg->addrData().get6();
        ev << _rvsHit << endl;
        delete msg;
        delete dnsMsg;
        expectingRvsDnsResp = false;
        sendDirect(new cPacket("HIP_REGISTER_AT_RVS"),
createStateMachine(rvsIPAddress, _rvsHit), "localIn");
    }
}

```

```

//send the message to specific handle functions
void HIP::handleMessage(cMessage *msg)
{
    //*****REGISTER AT RVS*****
    if (msg->isSelfMessage() && msg->isName("HIP_REGISTER_AT_RVS"))
    {
        handleRvsRegistration(msg);
    }
    //*****REGISTER AT LRVS added By Taj*****
    // if (msg->isSelfMessage() && msg->isName("HIP_REGISTER_AT_LRVS"))
    // {
    //     handleLrvsRegistration(msg);
    // }
    //*****DNS*****
    else if (msg->arrivedOn("udpIn") && strcmp(msg->getName(), "DNS Request")==0)
    {
        expectingDnsResp = true;
        send(msg, "udp6Out");
    }
    else if (msg->arrivedOn("udp6In") && strcmp(msg->getName(), "DNS Response")==0)
    {
        if(expectingDnsResp || expectingRvsDnsResp) {
            if (expectingDnsResp)
            {
                DNSBaseMsg* dnsMsg =
                check_and_cast<DNSBaseMsg*>(check_and_cast<cPacket*>(msg)->decapsulate());
                partnerHIT.set(dnsMsg->data());
                if(listHITtoTriggerDNS.count(partnerHIT) > 0)
                {
                    ev << "DNS response recieved, starting FSM\n";
                    // if (partnerHIT.isUnspecified())
                    partnerHIT.set(this->par("PARTNER_HIT"));

                    sendDirect(listHITtoTriggerDNS.find(partnerHIT)->second,
                    createStateMachine( dnsMsg->addrData().get6(), partnerHIT,
                    "localIn"));

                    listHITtoTriggerDNS.erase(partnerHIT);
                    if(listHITtoTriggerDNS.empty())
                        expectingDnsResp = false;
                }
                delete msg;
                delete dnsMsg;
            }
            if (expectingRvsDnsResp)
                handleRvsRegistration(msg);
        }
        else {
            ev << "DNS response when not expected, dropping...\n";
            delete msg;
        }
    }
}

```

```

}

//*****"normal message"
else
{
    //msg from transport layer
    if(msg->arrivedOn("tcpIn") || msg->arrivedOn("udpIn"))
        handleMsgFromTransport(msg);
    //msg from hip daemon to transport
    else if (msg->arrivedOn("fromFsmIn"))
        if ((check_and_cast<IPv6ControlInfo *>(msg-
>getControlInfo()))->getProtocol() != IP_PROT_UDP)
            send(msg, "tcpOut");
        else
            send(msg, "udpOut");
    //msg from hip daemon to network
    else if(msg->arrivedOn("fromFsmOut"))
        if ((check_and_cast<IPv6ControlInfo *>(msg-
>getControlInfo()))->getProtocol() != IP_PROT_UDP)
            {// if's begin added by Taj
                // begin of the code added by Taj
                if (dynamic_cast<HIPHeaderMessage *>(msg))
                    {// inside if's begin if it is HIP header
                        HIPHeaderMessage *hipHeader =
check_and_cast<HIPHeaderMessage *>(msg);
                        if ( _isLrvs == true && hipHeader->getKind()
== R2 && distinguisher ==0)
                            { // inside if's begin if R2
                                send(msg, "tcp6Out");
                                ev <<"Good Taj This is BIG step.. this
R2 in the LRVS.. send on tcp6Out\n ";
                                currentIfId = 3;
                                scheduleAt(simTime(),new
cPacket("HIP_REGISTER_AT_RVS"));
                            } // inside if's end if R2
                                // end of the code added by Taj
                                else
                                    send(msg, "tcp6Out");
                                }// inside if's end if it is HIP header
                                else
                                    send(msg, "tcp6Out");
                            } // if's end added by Taj
                        else
                            {// else's begin added by Taj
                                // begin of the code added by Taj
                                if (dynamic_cast<HIPHeaderMessage *>(msg))
                                    {// inside if's begin if it is HIP header
                                        HIPHeaderMessage *hipHeader =
check_and_cast<HIPHeaderMessage *>(msg);
                                        if ( _isLrvs == true && hipHeader->getKind()
== R2 && distinguisher ==0)
                                            { // inside if's begin if R2
                                                send(msg, "udp6Out");
                                                ev <<"Good Taj This is BIG step.. this R2 in the
LRVS.. send on udp6Out\n ";
                                                currentIfId = 3;
                                                scheduleAt(simTime(),new
cPacket("HIP_REGISTER_AT_RVS"));
                                            } // inside if's end if R2
                                                // end of the code added by Taj
                                                else

```

```

        send(msg, "udp6Out");
        } // inside if's end if it is HIP header
        else
            send(msg, "udp6Out");
    } // else's end added by Taj

    //msg from network
    else if (msg->arrivedOn("tcp6In") || msg-
>arrivedOn("udp6In"))
    {
        ev << "handleMsgFromNetwork invoke\n";
        handleMsgFromNetwork(msg);
    }
    else if(msg->arrivedOn("fromFsmTrigger")) //process
previously received message which was trigger
    {
        handleMsgFromTransport(msg);
    }
}

// Handles message from transport layer
void HIP::handleMsgFromTransport(cMessage *msg)
{
    IPv6ControlInfo *networkControlInfo =
check_and_cast<IPv6ControlInfo*>(msg->removeControlInfo());

    //if the destination address is in the IP-HIT mapping
    bool exists = false;

    //try to receive HIT from dest address
    ev << "Registered dest addr (HIT): " << networkControlInfo-
>getDestAddr() << endl;;
    //original destination address
    IPv6Address originalHIT = networkControlInfo->getDestAddr();

    //if hip association already exists, find its hip daemon and
forward to it
    for(hitToIpMapIt = hitToIpMap.begin(); hitToIpMapIt !=
hitToIpMap.end(); hitToIpMapIt++)
    {
        if( originalHIT == hitToIpMapIt->first )
        {
            ev << "Existing HIP association found...sending
msg to it....FROM_TRANSPORT\n";
            msg->setControlInfo(networkControlInfo);
            sendDirect(msg, findStateMachine(hitToIpMapIt-
>second->fsmId), "localIn");
            exists = true;
            distinguisher =1; // to prevent the triggering of
RVS registration
            break;
        }
    }
    //if not exists start the DNS sequence if it is not started
before
    if(!exists && listHITtoTriggerDNS.count(originalHIT) == 0)
    {
        // Need IP address for target HIT

```

```

        // Sending DNS
        ev << "Host not associated, need IP address, sending out
DNS...\n";
        msg->setControlInfo(networkControlInfo);
        listHITtoTriggerDNS[originalHIT] = msg;

        DNSBaseMsg* dnsReqMsg = new DNSBaseMsg("DNS Request");
//the request
        dnsReqMsg->setId(this->getId());
        dnsReqMsg->setAddrData(originalHIT);
        //UDP controlinfo
        UDPControlInfo *ctrl = new UDPControlInfo();
        ctrl->setSrcPort(0);
        InterfaceTable* ift =
(InterfaceTable*)InterfaceTableAccess().get();

        ctrl->setSrcAddr(ift->getInterface(currentIfId)-
>ipv6Data()->getPreferredAddress());
        ctrl-
>setDestAddr(IPAddressResolver().resolve(par("dnsAddress")));
        ctrl->setDestPort(23);
        ctrl->setInterfaceId(currentIfId);
        dnsReqMsg->setControlInfo(ctrl);

        UDPPacket *udpPacket = new UDPPacket(dnsReqMsg-
>getName());
        //TODO UDP_HEADER_BYTES
        udpPacket->setByteLength(8);
        udpPacket->encapsulate(dnsReqMsg);

        // set source and destination port
        udpPacket->setSourcePort(ctrl->getSrcPort());
        udpPacket->setDestinationPort(ctrl->getDestPort());

        IPv6ControlInfo *ipControlInfo = new IPv6ControlInfo();
        ipControlInfo->setProtocol(IP_PROT_UDP);
        ipControlInfo->setSrcAddr(ctrl->getSrcAddr().get6());
        ipControlInfo->setDestAddr(ctrl->getDestAddr().get6());
        ipControlInfo->setInterfaceId(currentIfId); //FIXME
extend IPv6 with this!!!
        udpPacket->setControlInfo(ipControlInfo);

        send(udpPacket, "udp6Out");
        hipVector.record(1);
        expectingDnsResp = true;
    }
}
// Handle message from ip(v6) layer
void HIP::handleMsgFromNetwork(cMessage *msg)
{
    IPv6ControlInfo *networkControlInfo =
check_and_cast<IPv6ControlInfo*>(msg->getControlInfo());

    if (dynamic_cast<HIPHeaderMessage *>(msg))
    {
        HIPHeaderMessage *hipHeader =
check_and_cast<HIPHeaderMessage *>(msg);
        ev << "Its a hip header\n";
    }
}

```

```

        //if dest locator's SPI is unknown -it is an I1 msg
        int fsmId = -1;
        if(hitToIpMap.find(hipHeader->getSrcHIT()) !=
hitToIpMap.end())
        fsmId = hitToIpMap.find(hipHeader->getSrcHIT())-
>second->fsmId;
        if(fsmId < 0)
        {
            IPv6Address realSrc;
            //check if its via rvs
            if (!hipHeader->getFrom_i().isUnspecified())
            {
                realSrc = hipHeader->getFrom_i();
            }
            else
            {
                realSrc = networkControlInfo->getSrcAddr();
            }
            //create new daemon and fw to it

            sendDirect(msg, createStateMachine( realSrc,
hipHeader->getSrcHIT()), "remoteIn");
        }
        else
        {
            //spi is known, find daemon and fw to it
            sendDirect(msg,
findStateMachine(fsmId), "remoteIn");
        }
    }
    //msg without hipheader, is it ESP?
    else
    {
        //msg->removeget);
        ESPMessage *espHeader = check_and_cast<ESPMessage
*>(msg);
        //TODO check if esp header exists
        //which spi?
        ev << "Got esp head: " << espHeader->getEsp()->getSpi()
<< endl;
        int spi = espHeader->getEsp()->getSpi();
        //decapsulating the msg
        //cPacket *transportPacket = msg->decapsulate();
        //transportPacket->setnetworkControlInfo);

        sendDirect(msg, findStateMachine(spi), "remoteIn");
        //delete msg;
    }
}

// If an address is changed on an interface
// initiate an update mechanism on the fsms
void HIP::handleAddressChange() {

    InterfaceTable* ift =
(InterfaceTable*)InterfaceTableAccess().get();
    for (int i=0; i<ift->getNumInterfaces(); i++)
    {
        InterfaceEntry *ie = ift->getInterface(i);

```



```

        if(!(ie->isLoopback()) && !(ie->isDown())) {
            if(mapIfaceToConnected.find(ie) !=
mapIfaceToConnected.end() && mapIfaceToConnected[ie] == true) {
                currentIfId = ie->getInterfaceId() - 100;
//WTF?
                break;
            }
        }
    }
    for(hitToIpMapIt = hitToIpMap.begin(); hitToIpMapIt !=
hitToIpMap.end();hitToIpMapIt++)
        sendDirect(new cPacket("ADDRESS_CHANGED"),
findStateMachine(hitToIpMapIt->second->fsmId), "HIPinfo");
}
void HIP::handleAddressChangeR() {
    return;
}
// Taj
void HIP::handleAddressChangeR(MACAddress MacMN) {
    return;
}

// Creates a new FSM and returns its pointer
cModule* HIP::createStateMachine(IPv6Address ipAddress, IPv6Address
&HIT)
{
    cModule *newFsm = fsmType->createScheduleInit("HipFsm", this);
    hitToIpMapEntryWork = new HitToIpMapEntry();

    hitToIpMapEntryWork->addr.push_front(ipAddress);
    hitToIpMapEntryWork->fsmId = newFsm->getId();
    hitToIpMap[HIT] = hitToIpMapEntryWork;

    return newFsm;
}

// Returns a pointer to the FSM with the specified ID
cModule* HIP::findStateMachine(int fsmID)
{
    cModule *fsm = simulation.getModule(fsmID);
    return fsm;
}

// Returns if the node is an RVS
bool HIP::isRvs()
{
    return _isRvs;
}

// Returns the nodes assigned RVS's HIT
IPv6Address* HIP::getRvsAddress()
{
    return &rvsIPaddress;
}

// The most recent DNS lookup's address (needed by FSM) TODO:
Nicer...
IPv6Address* HIP::getSrcWorkAddress()
{
    return &srcWorkAddress;
}

```

```

// Returns the nodes assigned RVS's HIT
IPv6Address* HIP::getRvsHit()
{
    return &_rvsHit;
}

IPv6Address HIP::getPartnerHIT() { return partnerHIT; };

// Counts the HIP messages sent
void HIP::incHipMsgCounter() {
    hipMsgSent++;
    hipVector.record(1);
}

// Outputs the statistics gathered
void HIP::finish() {
    recordScalar("HIP msg counter", hipMsgSent);
}
// addedby Taj
// Returns if the node is an LRVS
bool HIP::isLrvs()
{
    return _isLrvs;
}

// Returns the nodes assigned LRVS's HIT
IPv6Address* HIP::getLrvsAddress()
{
    return &lrvsIPAddress;
}

// Returns the nodes assigned LRVS's HIT
IPv6Address* HIP::getLrvsHit()
{
    return &_lrvsHit;
}

// SRVS
// Returns if the node is an SRVS
bool HIP::isSrvs()
{
    return _isSrvs;
}
// Returns the nodes assigned SRVS's HIT
IPv6Address* HIP::getSrvsAddress()
{
    return &srvsIPAddress;
}
// Returns the nodes assigned SRVS's HIT
IPv6Address* HIP::getSrvsHit()
{
    return &_srvsHit;
}
// ends of functons added by Taj
*****
*****
File: RVSHIP.h

#ifndef __RVSHIP_H__

```

```

#define __RVSHIP_H__

#include <HIP.h>
#include "IPv6Address.h"

class INET_API RvsHIP : public HIP
{
private:
    //bool exists;

    //work variables
    IPv6Address dstHITwork;
    IPv6Address ownHIT;
public:
    RvsHIP();
    virtual ~RvsHIP();
protected:
    virtual void handleMsgFromNetwork(cMessage *msg);
    virtual void specInitialize();
    virtual void alterHipPacketAndSend(HIPHeaderMessage* hipHead,
IPv6Address &rvsIP, IPv6Address &destIP, IPv6Address &fromIP);
    virtual void handleMsgFromTransport(cMessage *msg);
    virtual void handleAddressChange();

};

#endif

```

.....

**File: RVSHIP.cc**

```

#include "RvsHIP.h"
#include "IPv6ControlInfo.h"
#include "HipMessages_m.h"
#include "IPv6Datagram.h"

Define_Module(RvsHIP);

RvsHIP::RvsHIP() {}
RvsHIP::~RvsHIP() {}
void RvsHIP::specInitialize()
{
    _isRvs = true; //is on
    // _isLrvs = false; // added by Taj
    // _isSrvs = false; // added by Taj
    ownHIT.set(this->par("OWN_HIT")); //get RVS HIT
}

// Handles incoming I1 messages
void RvsHIP::handleMsgFromNetwork(cMessage *msg)
{
    if(dynamic_cast<HIPHeaderMessage *>(msg) != NULL)
    {
        HIPHeaderMessage *hipHeader =
check_and_cast<HIPHeaderMessage *>(msg);
        //HIP msg
        if (hipHeader != NULL)
        {
            IPv6ControlInfo *networkControlInfo =
check_and_cast<IPv6ControlInfo*>(msg->removeControlInfo());

            //dest is RVS's HIT, it's a registration

```

```

        ev << "destHIT ? ownHIT " << hipHeader-
>getDestHIT() << " ? " << ownHIT << endl;
        if (hipHeader->getDestHIT() == ownHIT)//if the dest HIT
is the same RVS HIT, it is registration
        {
            ev << "true" << endl;
            int fsmId = -1;//finite state machine init
            if(hitToIpMap.find(hipHeader->getSrcHIT())
!= hitToIpMap.end())//source HIT is registered at RVS
                fsmId = hitToIpMap.find(hipHeader-
>getSrcHIT())->second->fsmId;//get sender's HIT and store in fsmId
            if(fsmId < 0)//src HIT not registered at RVS
                {
                    //create new daemon and fw to it//
daemon means a record in RVS
                    ev << "RVS received I1 " << endl;
                    ev << "I1 source is (SPI not exist) "
<< networkControlInfo->getSrcAddr() << endl; // added by Taj , get
source's IP

                    msg-
>setControlInfo(networkControlInfo);
                    sendDirect(msg,
createStateMachine(networkControlInfo->getSrcAddr(), hipHeader-
>getSrcHIT()), "remoteIn");//create binding for source by storing
it's source ip and source HIT
                }
                else
                {
                    //spi is known, find daemon and fw to
it
                    ev << "RVS received SPI (possible
UPDATE, BE) " << hipHeader->getLocator(1).locatorESP_SPI <<
endl;//find the source by searching with spi
                    ev << "I1 source is (SPI exist) " <<
networkControlInfo->getSrcAddr() << endl; // added by Taj, get
source's ip

                    msg-
>setControlInfo(networkControlInfo);
                    sendDirect(msg,
findStateMachine(fsmId), "remoteIn");//mapping src ip and spi
                }
                //not registration, somebody's is trying to
talk HIP with destHIT, fw I1 to it
                else
                {
                    ev <<"Taj : you did well \n";
                    hitToIpMapIt = hitToIpMap.find(hipHeader-
>getDestHIT());//find dest ip and and assign to hitToIpMapIt
                    //alter the HIP header set src ip is
sender's ip, and dest ip is the dest ip found in RVS
                    alterHipPacketAndSend(hipHeader,
networkControlInfo->getDestAddr(), hitToIpMapIt->second-
>addr.front(), networkControlInfo->getSrcAddr());//get the senders
IP
                }
            }
        }
    }
    else
    {
        delete msg;
    }

```

```

    }
}

// Forwards I1 messages to the registered host's IP address
void RvsHIP::alterHipPacketAndSend(HIPHeaderMessage* hipHead,
IPv6Address &rvsIP, IPv6Address &destIP, IPv6Address &fromIP)
{
    //create new dest locator
    HipLocator *destLoc = new HipLocator();
    destLoc->locatorESP_SPI = -1;
    destLoc->locatorIPv6addr = destIP;
    hipHead->setLocator(1,*destLoc);
    //set HIPHeader's FROM field
    hipHead->setFrom_i(fromIP); //senders ip
    hipHead->setRvs_mac(1);
    IPv6ControlInfo *networkControlInfo = new
IPv6ControlInfo(); //object
    networkControlInfo->setDestAddr(destIP); //R or I IP from
received I1
    networkControlInfo->setSrcAddr(rvsIP); //via RVS
    networkControlInfo->setProtocol(6); //IPv6
    hipHead->setControlInfo(networkControlInfo);
    send(hipHead, "tcp6Out"); //send I1 from RVS to the dest I or
R , packet type is tcp
}

```

```

// Overridden, RVS doesn't functions as a host
void RvsHIP::handleMsgFromTransport(cMessage *msg) { delete msg; }
// And hopefully doesn't changes IP addresses either
void RvsHIP::handleAddressChange() { return; }
.....

```

**File: LrvsHIP.h(used to act as Cluster Server)**

```

/*
 * LrvsHIP.h
 *
 #ifndef LRVSHIP_H_
 #define LRVSHIP_H_
 #include <HIP.h>
 #include "IPv6Address.h"

 class INET_API LrvsHIP : public HIP
 {
 private:
     //bool exists;
 //work variables
     IPv6Address dstHITwork;
     IPv6Address ownHIT;//LRVS HIT
 public:
     LrvsHIP();
     virtual ~LrvsHIP();
 protected:
     virtual void handleMsgFromNetwork(cMessage *msg);
     virtual void specInitialize();
     virtual void alterHipPacketAndSend(HIPHeaderMessage* hipHead,
IPv6Address &lrvsIP, IPv6Address &destIP, IPv6Address &fromIP);
     virtual void handleMsgFromTransport(cMessage *msg);
     virtual void handleAddressChange();
 };

 #endif /* LRVSHIP_H_ */

```

```
*****
*****
```

**File: LRVSHIP.cc**

```
//header files
#include "LrvsHIP.h"
#include "IPv6ControlInfo.h"
#include "HipMessages_m.h"
#include "IPv6Datagram.h"

Define_Module(LrvsHIP);

LrvsHIP::LrvsHIP() {}
LrvsHIP::~LrvsHIP() {}
void LrvsHIP::specInitialize()
{
    _isLrvs = true; // added by Taj//LRVS is on
    // _isSrvs = false; // added by taj
    // _isRvs = false; // added by taj

    ownHIT.set(this->par("OWN_HIT")); //Lrvs HIT
}

// Handles incoming I1 messages
void LrvsHIP::handleMsgFromNetwork(cMessage *msg)
{ // function's begin
    if(dynamic_cast<HIPHeaderMessage *>(msg) != NULL)
    {
        HIPHeaderMessage *hipHeader =
check_and_cast<HIPHeaderMessage *>(msg);
        //HIP msg
        if (hipHeader != NULL)
        {
            int d = 0; // added by Taj
            if (hipHeader->getKind() == I1)//if the received packet
is I1 type
            { // if's begin
                ev << "Salaam Taj: this is a third step Congratulation
\n";

                } // if's end
            else if (hipHeader->getKind() == I2)//if the received
packet is I2 type
            { // if's begin
                ev << "Salaam taj this is a fifth step
Congratulation \n";
                ev <<"Congratulation taj\n";
                regdHostHIT = hipHeader->getSrcHIT();//derive the
source HIT from the msg and assign it to regdHostHIT
                ev<<"The new source HIT is " << regdHostHIT
<<endl;

                distinguisher = 0; // to allow RVS registration
                //scheduleAt(simTime(),new
cPacket("HIP_REGISTER_AT_RVS"));
            } // if' s end
            else if (hipHeader->getKind() == R1 || hipHeader-
>getKind() == R2)//if the received packet is R1 or R2 type
            { // else's begin
                d=1;//if d=0 then the packet type is I1 or I2, and
if d=1 then the packet type is R1 or R2
```

```

        ev << "the value of d is " << d << endl; //show the
value of d(packet type)
    } // else's end

    IPv6ControlInfo *networkControlInfo =
check_and_cast<IPv6ControlInfo*>(msg->removeControlInfo());

    //dest is LRVS's HIT, it's a registration to the LRVS
    ev << "destHIT ? ownHIT of LRVS " << hipHeader-
>getDestHIT() << " ? " << ownHIT << endl; //check whether the dest
HIT is Lrvs HIT or another
    if ((hipHeader->getDestHIT() == ownHIT) || (hipHeader-
>getDestHIT() == regdHostHIT && d==1)) //dest is HIT(I) and the sender
is R, and the packet type is R1 or R2
    {
        ev << "true" << endl;
        d=0; // added by taj
        int fsmId = -1;
        if(hitToIpMap.find(hipHeader->getSrcHIT())
!= hitToIpMap.end()) //check whether the source HIT is found
            fsmId = hitToIpMap.find(hipHeader-
>getSrcHIT()->second->fsmId); //find source HIT and store in fsmId
        if(fsmId < 0) //not found
        {
            //create new daemon and forward to it
            ev << "LRVS received I1" << endl;
            msg-
>setControlInfo(networkControlInfo);
            ev << "Taj be patient we are going to
create FSM at remotein \n";
            sendDirect(msg,
createStateMachine(networkControlInfo->getSrcAddr(), hipHeader-
>getSrcHIT()), "remoteIn");
        }
        else
        {
            //spi is known, find daemon and
forward to it
            ev << "LRVS received SPI (possible
UPDATE, BE) " << hipHeader->getLocator(1).locatorESP_SPI << endl;
            msg-
>setControlInfo(networkControlInfo);
            sendDirect(msg,
findStateMachine(fsmId), "remoteIn");
        }
    }
    //not registration, somebody's trying to
reach destHIT, forward I1 to it
    else
    {
        ev << "Taj: not registration, somebody's trying to
reach destHIT, forward I1\n ";
        int fsmId = -1;
        if(hitToIpMap.find(hipHeader->getDestHIT()) !=
hitToIpMap.end())
            fsmId = hitToIpMap.find(hipHeader->getSrcHIT())-
>second->fsmId;
        if(fsmId < 0)
        { // inside if's begin locate by Source
means from inside domain to inside

```

```

//      pktdirection =0; // 0 means the packet
goes to Host inside the domain
hitToIpMapIt =
hitToIpMap.find(hipHeader->getSrcHIT());
ev <<"The source HIT is " <<
hipHeader->getSrcHIT() <<endl;
IPv6Address *srcIP = &hitToIpMapIt-
>second->addr.front();//source address
ev << "and the fsmid is " <<
hitToIpMapIt->second->fsmId << endl;
// if (networkControlInfo->getSrcAddr()==
"aaaa:0:1:0:aaa:ff:fe00:7" )
//{
//      networkControlInfo-
>setDestAddr("aaaa::aaa:ff:fe00:1");
// }
// else if (networkControlInfo-
>getSrcAddr()== "aaaa:1:1:0:aaa:ff:fe00:7")
// {
//      networkControlInfo-
>setDestAddr("aaaa:1::aaa:ff:fe00:3");
// }
ev <<"datagram destination " <<
networkControlInfo->getDestAddr() << endl;// added by Taj
// IPv6ControlInfo *networkControlInfo
= new IPv6ControlInfo();
networkControlInfo-
>setProtocol(6);//ipv6
msg-
>setControlInfo(networkControlInfo);
sendDirect(msg,
findStateMachine(hitToIpMapIt->second->fsmId),"remoteIn");
} // inside if's end
else
{ // else's begin Locate by Destination
means from outside domain to inside
// if (strcmp(hipHeader-
>getSrcHIT(),"2001:0db8:85a3:0000:0000:8a2e:0370:0290")==0) // if
it is from LRVS2 do the following
if (hipHeader-
>getSrcHIT()=="2001:0db8:85a3:0000:0000:8a2e:0370:0290")//from Lrv2
address
{
//from LRVS2
ev <<"At the LRVS update packet
from LRVS2"<< endl;
hitToIpMapIt =
hitToIpMap.find(hipHeader->getDestHIT());//getting the dest HIT
IPv6Address destIP =
"aaaa:1::aaa:ff:fe00:3";//Ip for dest static
ev <<"It's IP addr is " <<
destIP<< endl;//show dest IP
ev << "and the fsmid is " <<
hitToIpMapIt->second->fsmId << endl;
networkControlInfo-
>setDestAddr(destIP);
msg-
>setControlInfo(networkControlInfo);
sendDirect(msg,
findStateMachine(hitToIpMapIt->second->fsmId),"remoteIn");
}

```



```

    }
    else
    { // else's begin when it's not LRVS2
      //pktdirection =0; // 0 means the
packet goes to Host inside the domain
      hitToIpMapIt =
hitToIpMap.find(hipHeader->getDestHIT()); //get dest HIT from hip
header
      ev <<"The source HIT is " <<
hipHeader->getSrcHIT() <<endl;
      ev <<"The destination HIT is " <<
hipHeader->getDestHIT() <<endl;
      IPv6Address *destIP = &hitToIpMapIt-
>second->addr.front(); //find the dest IP depend on dest HIT
      ev <<"It's IP addr is " << *destIP<<
endl;
      ev << "and the fsmid is " <<
hitToIpMapIt->second->fsmId << endl;
      // IPv6ControlInfo *networkControlInfo =
new IPv6ControlInfo();
      networkControlInfo-
>setDestAddr(*destIP); //handle the dest ip and now ready for
delivery to the dest
      // networkControlInfo-
>setSrcAddr(lrvsIP);
      // networkControlInfo->setProtocol(6);
if (hipHeader->getKind() ==
UPDATE) //if it is update packet
    {
      // for Update packet from Srvs
      IPv6Address destIP =
"aaaa::aaa:ff:fe00:1"; // srvs ip
      ev <<"It's IP addr is " <<
destIP<< endl; //showing
      networkControlInfo-
>setDestAddr(destIP); //setting as dest address
    }

    msg-
>setControlInfo(networkControlInfo);
      sendDirect(msg,
findStateMachine(hitToIpMapIt->second->fsmId), "remoteIn"); //sending
to the corresponding host
      // To set the regdHostHIT variable to
Null to allow R2 go to its destination
      if (hipHeader->getKind() == I2) //if
type is I2 packet
    { // if's begin
      regdHostHIT = hipHeader-
>getDestHIT(); //get the dest HIT and assign it to regdHostHIT
      distinguisher =1; // to prevent
the triggering of RVS registration

      } // if's end
    } // else's end when it's not LRVS2
  } // else's end
  //

```

```

        //create new dest locator
        // HipLocator *destLoc = new HipLocator();
        // destLoc->locatorESP_SPI = -1;
        // destLoc->locatorIPv6addr = *dstIPwork;
        // hipHead->setLocator(1,*destLoc);

        // alterHipPacketAndSend(hipHeader,
networkControlInfo->getDestAddr(), hitToIpMapIt->second-
>addr.front(), networkControlInfo->getSrcAddr());
    }

    }
else
    {
        delete msg;//discard the msg
    }
}
else
{ // else's begin if ESP packet
ev << "Taj salaaaaaaaaaaaaaaaaaaaaam This is ESP\n";
// pktdirection =0;
ESPMessage *espHeader = check_and_cast<ESPMessage *>(msg);
ev << "Got esp head: " << espHeader->getEsp()->getSpi() <<
endl;//getting spi
// ev << "The spi of ESP is " << espHeader->getSpi() <<endl;
IPv6ControlInfo *networkControlInfo =
check_and_cast<IPv6ControlInfo*>(msg->removeControlInfo());
ev <<"datagram Source " << networkControlInfo->getSrcAddr()
<< endl; // added by Taj,getting source ip address
ev <<"datagram destination " << networkControlInfo-
>getDestAddr() << endl; // added by Taj,//getting dest ip address
ev <<"The registered HIT is " <<regdHostHIT <<endl;// here to
find the SA to send ESP packet
hitToIpMapIt = hitToIpMap.find(regdHostHIT);//Retrieving the
corresponding ip
IPv6Address *regdIP = &hitToIpMapIt->second-
>addr.front();//Retrieving the reg ip
// if (networkControlInfo->getSrcAddr()==*regdIP) // packet
going outside the domain //
if (networkControlInfo->getSrcAddr()==
"aaaa:0:1:0:aaa:ff:fe00:7" || networkControlInfo->getSrcAddr()==
"aaaa:1:1:0:aaa:ff:fe00:7" || networkControlInfo-
>getSrcAddr()=="aaaa:aaa:ff:fe00:1"||networkControlInfo-
>getSrcAddr()=="aaaa:1::aaa:ff:fe00:3")//we have four host with the
mentioned ips
{
    ev <<"This packet will go outside the domain\n";
}
else // packet going inside the domain
{
    ev <<"This packet will go inside the domain\n";
networkControlInfo->setDestAddr(*regdIP);//store in the
dest of the packet
ev <<"the new datagram destination " <<
networkControlInfo->getDestAddr() << endl; // added by Taj
}
}
}

```

```

        msg->setControlInfo(networkControlInfo);
        sendDirect(msg, findStateMachine(hitToIpMapIt->second-
>fsmId), "remoteIn");

    } // else's end if ESP packet
} // function's end

// Forwards I1 messages to the registered host's IP address
void LrvsHIP::alterHipPacketAndSend(HIPHeaderMessage* hipHead,
IPv6Address &lrvsIP, IPv6Address &destIP, IPv6Address &fromIP)
{
    //create new dest locator
    HipLocator *destLoc = new HipLocator();
    destLoc->locatorESP_SPI = -1;
    destLoc->locatorIPv6addr = destIP;
    hipHead->setLocator(1, *destLoc);
    //set HIPHeader's FROM field
    hipHead->setFrom_i(fromIP);
    hipHead->setRvs_mac(1);
    IPv6ControlInfo *networkControlInfo = new
IPv6ControlInfo();//object of IPv6ControlInfo
        networkControlInfo->setDestAddr(destIP);//dest ip
        networkControlInfo->setSrcAddr(lrvsIP);//packet goes
from lrvs to dest ip
        networkControlInfo->setProtocol(6);//ipv6 type
        hipHead->setControlInfo(networkControlInfo);//hip header
para
        send(hipHead, "tcp6Out");//sending the packet
    }
void LrvsHIP::handleMsgFromTransport(cMessage *msg) { delete msg; }
void LrvsHIP::handleAddressChange() { return; }//do nothing

```

\*\*\*\*\*missing files\*\*\*\*\*

- HipFsm.h
- HipFsm.cc
- HipMessages\_m.h
- HipMessages\_m.cc
- HipFsmBase.h
- HipFsmBase.cc
- FsmBaseMsg\_m.h
- FsmBaseMsg\_m.cc

**Hint:** These files are included in the HIPSim++ simulator, available in reference [86]

## REFERENCES

[1] Cisco visual networking index: Global mobile data traffic forecast update, 2014-2019. White Paper, March 30, 2015. Available:

[http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/wHITE\\_PAPER\\_C11-520862.html](http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/wHITE_PAPER_C11-520862.html), viewed on 10/12/2011

[2] WiMAX forum. <http://www.wimaxforum.org>, viewed on 12/12/2011

[3] 3GPP. <http://www.3gpp.org>, viewed on 12/12/2011

[4] H. Chan, "Requirements of distributed mobility management," Internet-Draft (work in progress), Nov. 2012.

[5] R. Koodli, "IP address location privacy and mobile IPv6: Problem statement," IETF RFC 4882, May 2007.

[6] The Internet Engineering Task Force (IETF). <http://www.ietf.org>, viewed on 01/01/2012

[7] IETF Distributed Mobility Management (DMM) WG.

<http://datatracker.ietf.org/wg/dmm>, viewed on 01/02/2012

[8] P. Bertin, S. Bonjour and J.-M. Bonnin, "An evaluation of dynamic mobility anchoring," in Proc. IEEE 70th Vehicular Technology Conference Fall (VTC 2009-Fall), 2009, pp.1-5.

[9] 3GPP TS 23.829, "Local IP access and selected IP traffic offload (LIPA-SIPTO)," V10.0.1, October 2011.

[10] K. Lee, J. Lee, Y. Yi, I. Rhee and S. Chong, "Mobile Data Offloading: How Much Can WiFi Deliver?" IEEE/ACM Transactions on Networking, vol. 21, no.2, April 2013, pp.536-550.

[11] H. Chan, D. Liu, P. Seite, H. Yokota and J. Korhonen, "Requirements for distributed mobility management," IETF draft-ietf-dmm-requirements-11 (work in progress), November 2013.

- [12] H. A. Chan, H. Yokota, J. Xie, P. Seite and D. Liu, "Distributed and Dynamic Mobility Management in Mobile Internet: Current Approaches and Issues," *Journal of 151 Communications*, vol. 6, no. 1, February 2010, pp. 4-15.
- [13] C. Perkins, "IP Mobility Support for IPv4," RFC 3344, IETF, August 2002.
- [14] V. Chandrasekhar, J. G. Andrews and A. Gatherer, "Femtocell networks: a survey," *IEEE Communications Magazine*, vol. 46, no. 9, September 2008, pp. 59-67.
- [15] K. Pentikousis, Q. Zhou and H. Wang, "Design considerations for mobility management in future infrastructure networks," in *Proc. Telecom World (ITU WT), Technical Symposium at ITU*, 2011, pp. 87-92.
- [16] D.-H. Shin, D. Moses, M. Venkatachalam and S. Bagchi, "Distributed mobility management for efficient video delivery over all-IP mobile networks: Competing approaches," *IEEE Network*, vol. 27, no. 2, April 2013, pp. 28-33.
- [17] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury and B. Patil, "Proxy mobileIPv6," IETF RFC 5213, August 2008.
- [18] H. Soliman, editor, "Mobile IPv6 support for dual stack hosts and routers," IETF RFC 5555, June 2009.
- [19] C. Perkins, D. Jonson and J. Arkko, "Mobility Support in IPv6," IETF RFC 6275, July 2011.
- [20] P. Bertin, S. Bonjour and J.-M. Bonnin, "An evaluation of dynamic mobility anchoring," in *Proc. IEEE 70th Vehicular Technology Conference Fall (VTC 2009-Fall)*, 2009, pp. 1-5.
- [21] P. P. Ernest, H. A. Chan and O. E. Falowo, "Distributed mobility management scheme with mobility routing function at the gateways," in *Proc. IEEE GLOBECOM*, December 2012, pp. 5254-5259.
- [22] V. Devarapalli, R. Wakikawa, A. Petrescu and P. Thubert, "Network mobility (NEMO) basic support protocol," IETF RFC 3963, January 2005.

- [23] A. De La Oliva, C. J. Bernardos, M. Calderon, T. Melia and J. C. Zuniga, "IP flow mobility: smart traffic offload for future wireless networks," IEEE Communications Magazine, vol. 49, no.10, October 2011, pp.124-132.
- [24] L. Bokor, Z. Faigl and S. Imre, "Flat architectures: Towards scalable future internet mobility," Future Internet Assembly, LNCS 6656, 2011, pp.35-50.
- [25] C. J. Bernardos, J. C. Zunniga and A. Reznik, "Towards flat and distributed mobility management: A 3GPP evolved network design," in Proc. IEEE ICC, June 10-15, 2012, pp.6855-6861.
- [26] P. Bertin, S. Bonjour and J.-M. Bonnin, "Distributed or centralized mobility?" in Proc. IEEE GLOBECOM, 2009, pp.1-6.
- [27] R. Koodli, editor, "Mobile IPv6 fast handovers," IETF RFC 5568, July 2009.
- [28] H. Soliman, C. Castelluccia, K. ElMalki and L. Bellier, "Hierarchical mobile IPv6 (HMIPv6) mobility management," IETF RFC 5380, October 2008.
- [29] D. Johnson, C. Perkins, and J. Arkko, "Mobility Support in IPv6," RFC 3775, IETF, June 2004.
- [30] C. Perkins, D. Johnson, and J. Arkko, "Mobility Support in IPv6," RFC 6275, IETF, July 2011.
- [31] T. Narten, E. Nordmark, W. Simpson, and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," RFC 4861, IETF, September 2007.
- [32] S. Thomson, T. Narten, and T. Jinmei, "IPv6 Stateless Address Autoconfiguration," RFC 4862, IETF, September 2007.
- [33] J. Arkko, V. Devarapalli, and F. Dupont, "Using IPsec to Protect Mobile IPv6 Signaling Between Mobile Nodes and Home Agents," RFC 3776, IETF, June 2004.
- [34] P. Nikander, J. Arkko, T. Aura, G. Montenegro, and E. Nordmark, "Mobile IP Version 6 Route Optimization Security Design Background" RFC 4225, IETF, December 2005.
- [35] R. Moskowitz, P. Jokela, P. Nikander and T. Henderson, "Host identity protocol," Rfc-6253, April 2008.

- [36] R Moskowitz, P Nikander, Host identity protocol (HIP) architecture. IETF, RFC4423 (2006)
- [37] <http://datatracker.ietf.org/doc/search/?name=PMIPv6>, viewed on 30/12/2012
- [38] T. T. Nguyen and C. Bonnet, "DMM-based inter-domain mobility support for proxy mobile IPv6," in Proc. IEEE WCNC, Shanghai, China, April 2013, pp. 1998-2003.
- [39] H. A. Chan, H. Yokota, J. Xie, P. Seite and D. Liu, "Distributed and Dynamic Mobility Management in Mobile Internet: Current Approaches and Issues," Journal of 151 Communications, vol. 6, no. 1, February 2010, pp. 4-15.
- [40] Ibrahim Al-Surmi - Mohamed Othman - Borhanuddin Mohd Ali , "Mobility management for IP-based next generation mobile networks: Review, challenge and perspective", Journal of Network and Computer Applications - Vol. 35 - Issue 1 - 2012 - pp. 295-315.
- [41] J. C. Zuniga, C. J. Bernardos, T. Melia, and C. Perkins, "Mobility Practices and DMM Gap Analysis," Internet-Draft (work in progress), Dec. 2012.
- [42] B. Sarikaya, "Distributed Mobile IPv6," Internet-Draft (work in progress), Feb. 2012.
- [43] F. Giust, A. de la Oliva, and C. J. Bernardos, "Flat Access and Mobility Architecture: an IPv6 Distributed Client Mobility Management Solution," in 3rd IEEE International Workshop on Mobility Management in the Networks of the Future World (Mobiworld 2011), in conjunction with IEEE INFOCOM 2011, Apr. 2011.
- [44] <https://tools.ietf.org/id/draft-yokota-dmm-scenario-00.txt>, viewed on 10/1/2013
- [45] Koh S., Kim J., Jung H., and Han Y., "Use of Proxy mobile IPv6 for Distributed Mobility Control" IETF draft-sjkoh-mext-pmip-dmc-01 (work in progress), March 2011.
- [46] Fabio G., Antonio O., Carlos J. B., "Flat Access and Mobility Architecture: an IPv6 Distributed Client Mobility Management Solution", Mobility Management in the Network of the Future World (MobiWorld), and IEEE 2011.

- [47] Muhana M. Muslam, H. Anthony Chan, Linoh A. Magagula, Neco Ventura " Network-based mobility and Host Identity Protocol", 2012 IEEE Wireless Communications and Networking Conference (WCNC), 2012, 2395 – 2400.
- [48] TajElsir H. Suliman, Altayeb S. Abuelyaman, " Distributed Mobility Management Scheme using Lightweight Fast Handover (DMM-LFH)", International Journal of Scientific & Engineering Research(IJSER), Volume 4, Issue 10, October-2013.
- [49] Seonggeun Ryu, Gye-Young Kim, Byunggi Kim Youngsong Mun, "A Scheme to Reduce Packet Loss during PMIPv6 Handover considering Authentication", ICCSA 2008.
- [50] Jong-Hyouk, Jean-Marie, Pierrick Seite, H. Chan, "Distributed IP Mobility Management the perspective of the IETF: Motivations, Requirements, Approaches, Comparison and Challenges", IEEE Wireless Communications, October 2013.
- [51]TajElsir H. Suliman, Altayeb S. Abuelyaman, "Host Based Distributed Mobility Management Scheme using Tunneling-less Handover (H-DMM-TH)", Proceedings of IEEE APW2014 Conference, Taiwan, 28-29 August, 2014.
- [52] P. Bertin, S. Bonjour and J.-M. Bonnin, "A distributed dynamic mobility management scheme designed for flat IP architectures," in Proc. 3rd International Conference on New Technologies, Mobility and Security( NTMS),November 2008, pp. 1-5.
- [53] M. Liu, X. Guo, A. Zhou, S. Wang, Z. Li, and E. Dutkiewicz, "Low latency IP mobility management: protocol and analysis," EURASIP Journal on Wireless Communications and Networking, Springer, vol. 2011, no. 1, pp. 1–16, 2011.
- [54] R. Wakikawa, G. Valadon, and J. Murai, "Migrating home agents towards internet-scale mobility deployments," in emerging Networking EXperiments and Technologies (CoNEXT), 2006 ACM International Conference on, December 2006.
- [55] F. Giust, A. de la Oliva and C. J. Bernardos, "Flat access and mobility architecture: An IPv6 distributed client mobility management solution," in Proc. IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), April 2011, pp. 361-366.



- [56] H. A. Chan, "Proxy Mobile IP with distributed mobility anchors," in GLOBECOM Work-shops (GC Wkshps), 2010 IEEE, December 2010, pp. 16–20.
- [57] P. P. Ernest, H. A. Chan, and O. E. Falowo, "Distributed mobility management scheme with mobility routing function at the gateways," in Global Communications Conference (GLOBECOM), 2012 IEEE, December 2012, pp. 5254–5259.
- [58] P. P. Ernest, H. A. Chan, J. Xie, and O. E. Falowo, "Mobility management with distributed mobility routing functions," *Telecommunication Systems*, Springer, 2015, to appear.
- [57] F. Giust, A. De La Oliva, C. J. Bernardos and R. P. F. Da Costa, "A network-based localized mobility solution for distributed mobility management," in Proc. 14th International Symposium on Wireless Personal Multimedia Communications (WPMC), October 2011, pp. 1-5.
- [58] J.-I. Kim, H. Jung and S. J. Koh, "Distributed mobility control for mobile-oriented future internet environments," in Proc. International Conference on ICT Convergence (ICTC), September 2011, pp. 342-347.
- [59] P. P. Ernest, H. A. Chan, O. E. Falowo, L. A. Magagula, and S. Cespedes, "Network-based distributed mobility management for network mobility," in Consumer Communications and Networking Conference (CCNC), 2014 11th IEEE, 2014, pp. 417–425.
- [60] P. P. Ernest and H. A. Chan, "Enhanced handover support and routing path optimization with distributed mobility management in flattened wireless networks," in *Wireless Personal Multimedia Communications (WPMC)*, 2011 14th IEEE International Symposium on, October 2011, pp. 1–5. REFERENCES 165.
- [61] A. Nascimento, R. Sofia, T. Condeixa and S. Sargento, "A decoupling approach for distributed mobility management," in Proc. 21st International Conference on Computer Communications and Networks (ICCCN), 2012, pp. 1-6.
- [62] H. Ali-Ahmad, M. Ouzzif, P. Bertin and X. Lagrange, "Distributed dynamic mobile IPv6: Design and evaluation," in Proc. IEEE WCNC, April 7-10, 2013, pp. 2166-2171.

- [63] S. Yan, C. Jiayin and C. Shanzhi, "A Mobile IPv6 based Distributed Mobility Management Mechanism of Mobile Internet," *Physics Procedia*, vol. 25, 2012, pp. 2249-2256.
- [64] M. Fischer, F.-U. Andersen, A. Kopsel, G. Schafer and M. Schlager, "A distributed IP mobility approach for 3G SAE," in *Proc. IEEE PIMRC*, September 2008, pp. 1-6.
- [65] L. Yi, H. Zhou, D. Huang and H. Zhang, "D-PMIPv6: A distributed mobility management scheme supported by data and control plane separation," *Mathematical and Computer Modelling*, vol. 58, no. 5-6, September 2013, pp. 1415-1426.
- [66] R. Costa, T. Melia, D. Munaretto and M. Zorzi, "When mobile networks meet content delivery networks: Challenges and opportunities," in *Proc. Seventh ACM International Workshop on Mobility in the Evolving Internet Architecture*, 2012, pp. 11-16.
- [67] M. Muslam, H. A. Chan and N. Ventura, "Host identity protocol extension supporting localized mobility management," in *Consumer Communications and Networking Conference(CCNC)*, 2011 IEEE, 2011, pp. 106-110.
- [68] M. M. Muslam, H. A. Chan and N. Ventura, "Inter-Subnet Localized Mobility Support of Host Identity Protocol," *EURASIP Journal on Wireless Communications and Networking* 2011, 4 August 2011, doi:10.1186/1687-1499-2011-55.
- [69] Yinghui Qiu, "HIP based mobility management for heterogeneous networks," in *Wireless Communications, Networking and Mobile Computing (WiCOM)*, 2011 7th International Conference on, 2011, pp.1-4.
- [70] G. Iapichino and C. Bonnet, "Host identity protocol and proxy mobile IPv6: A secure global and localized mobility management scheme for multihomed mobile nodes," in *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, 2009, pp. 1-6.
- [71] J. Y. H. So and Jidong Wang, "Micro-HIP A HIP-based micro-mobility solution," in *Communications Workshops, 2008. ICC Workshops '08. IEEE International Conference on*, 2008, pp. 430-435.

- [72] A. Gurtov, "Host Identity Protocol (HIP): Towards the Secure Mobile Internet", Wiley and Sons, 2008.
- [73] Ji-In Kim, Seok-Joo Koh and Nam-Seok Ko, "B-PMIPv6: PMIPv6 with bicasting for soft handover," in *Advanced Communication Technology, 2009. ICACT 2009. 11th International Conference on*, 2009, pp. 218-221.
- [74] J. Laganier and L. Eggert, "Host identity protocol (HIP) rendezvous extension," *Rfc-5204*, April 2008.
- [75] OMNet++ open source network simulator. "Official website: <http://www.omnetpp.org>", viewed on 12/08/2014.
- [76] M. M. Muslam, H. A. Chan and N. Ventura, "Inter-Subnet Localized Mobility Support of Host Identity Protocol," *EURASIP Journal on Wireless Communications and Networking* 2011, 4 August 2011, doi:10.1186/1687-1499-2011-55.
- [77] M. Muslam, H. A. Chan and N. Ventura, "Host identity protocol extension supporting localized mobility management," in *Consumer Communications and Networking Conference (CCNC), 2011 IEEE*, 2011, pp. 106-110.
- [78] P. P. Ernest, O. E. Falowo, H. A. Chan and L. A. Magagula, "Fast route optimization considering mitigating packet loss for PMIPv6 with coordinating MAG" in *Proc.Southern Africa Telecommunication Networks and Applications Conference*, Stellenbosch, South Africa, September 1-4, 2013.
- [79] A. Nascimento, R. Sofia, T. Condeixa and S. Sargento, "A decoupling approach for distributed mobility management," in *Proc. 21st International Conference on Computer Communications and Networks (ICCCN)*, 2012, pp. 1-6.
- [80] I. F. Akyildiz, J. S. M. Ho, and Y.-B. Lin, "Movement-based location update and selective paging for PCS networks," *Networking, IEEE/ACM Transactions on*, vol. 4, no. 4, pp.629–638, August 1996.
- [81] Ali-Ahmad, Ouzzif M, Bertin P, Lagrange X, "Distributed Mobility Management: Approaches and analysis," in *Communications Workshops (ICC), 2013 IEEE International Conference on*, June 2013, pp. 1297–1302.

[82] OPNET Technologies, "Official homepage: <http://www.opnet.com>," visited on 20/04/2015.

[83] 3GPP Official homepage," <http://www.3gpp.org>", viewed on 01/9/2014

[84] M Muslam, HA Chan, N Ventura, LA Magagula, Hybrid HIP and PMIPv6 (HIPPMIP) mobility management for handover performance optimization, in 6th International Conference on Wireless and Mobile Communications, 2010, ICWMC, pp. 232–237 (2010).

[85] G Iapichino, C Bonnet, Host identity protocol and proxy mobile IPv6: a secure global and localized mobility management scheme for multihomed mobile nodes, in Global Telecommunications Conference, 2009. GLOBECOM 2009, IEEE, pp. 1–6 (November 30 2009 - December 4 2009).

[86] L. Bokor, S. Novaczki, L.T Zeke and G. Jeney, " Design and Evaluation of Host Identity Protocol (HIP) simulation framework for INET/OMnet++", in the proceedings of the 12<sup>th</sup> ACM international conference on modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM 2009), Spain, Oct,2009.