

# 國立交通大學

資訊工程系

博士論文

VoIP 聚合環境之行動性與互用性的支援  
Mobility and Interoperability Supports for VoIP  
Converged Environments



研究生：張弘鑫

指導教授：曾建超 教授

張明峰 教授

中華民國九十五年六月

VoIP 聚合環境之行動性與互用性的支援

Mobility and Interoperability Supports for VoIP  
Converged Environments

研究生：張弘鑫

Student：Hung-Hsin Chang

指導教授：曾建超

Advisor：Chien-Chao Tseng

張明峰

Ming-Feng Chang

國立交通大學

資訊工程系

博士論文



A Dissertation

Submitted to Department of Computer Science

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

in

Computer Science and Information Engineering

June 2006

Hsinchu, Taiwan, Republic of China

中華民國九十五年六月

# 國立交通大學

## 博碩士論文全文電子檔著作權授權書

本授權書所授權之學位論文，為本人於國立交通大學 資訊工程 系所 \_\_\_\_\_ 組， 94 學年度第 2 學期取得碩士學位之論文。

論文題目：VoIP 聚合環境之行動性與互用性的支援

指導教授：曾建超、張明峰 教授

同意  不同意

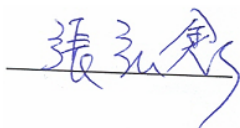
本人茲將本著作，以非專屬、無償授權國立交通大學與台灣聯合大學系統圖書館：基於推動讀者間「資源共享、互惠合作」之理念，與回饋社會與學術研究之目的，國立交通大學及台灣聯合大學系統圖書館得不限地域、時間與次數，以紙本、光碟或數位化等各種方法收錄、重製與利用；於著作權法合理使用範圍內，讀者得進行線上檢索、閱覽、下載或列印。

論文全文上載網路公開之範圍及時間：

本校及台灣聯合大學系統區域網路	<input checked="" type="checkbox"/> 中華民國 95 年 6 月 8 日公開
校外網際網路	<input checked="" type="checkbox"/> 中華民國 95 年 6 月 8 日公開

授權人：張弘鑫

親筆簽名：



中華民國 95 年 6 月 8 日

# 國立交通大學

## 博碩士紙本論文著作權授權書

本授權書所授權之學位論文，為本人在國立交通大學 資訊工程 學院 資訊工程 系所 \_\_\_\_\_ 組 94 學年度第 2 學期取得碩士學位之論文。

論文題目：VoIP 聚合環境之行動性與互用性的支援

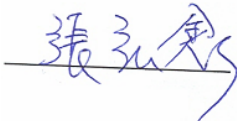
Mobility and Interoperability Supports for VoIP  
Converged Environments

指導教授：曾建超、張明峰 教授

### ■ 同意

本人茲將本著作，以非專屬、無償授權國立交通大學，基於推動讀者間「資源共享、互惠合作」之理念，與回饋社會與學術研究之目的，國立交通大學圖書館得以紙本收錄、重製與利用；於著作權法合理使用範圍內，讀者得進行閱覽或列印。

授權人：張弘鑫

親筆簽名： 

民國 95 年 6 月 8 日

國家圖書館  
博碩士論文電子檔案上網授權書

ID:GT008717806

本授權書所授權之論文為授權人在國立交通大學資訊工程學院資訊工程系所 \_\_\_\_\_ 組94學年度第2學期取得碩士學位之論文。

論文題目：VoIP 聚合環境之行動性與互用性的支援

Mobility and Interoperability Supports for VoIP  
Converged Environments

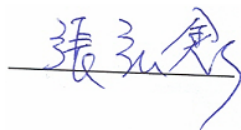
指導教授：曾建超、張明峰 教授

茲同意將授權人擁有著作權之上列論文全文（含摘要），非專屬、無償授權國家圖書館，不限地域、時間與次數，以微縮、光碟或其他各種數位化方式將上列論文重製，並得將數位化之上列論文及論文電子檔以上載網路方式，提供讀者基於個人非營利性質之線上檢索、閱覽、下載或列印。

※ 讀者基於非營利性質之線上檢索、閱覽、下載或列印上列論文，應依著作權法相關規定辦理。

授權人：張弘鑫

親筆簽名：



民國 95 年 6 月 8 日

國立交通大學  
資訊工程系博士班  
論文口試委員會審定書

本校 資 訊 工 程 系 張弘鑫 君

所提論文：VoIP聚合環境之行動性與互用性的支援

合於博士資格水準、業經本委員會評審認可。

口試委員：王祥郡 楊竹峯

向夢宏 曹孝標

曾建超 嚴力行

陳裕賢 張明峰

指導教授：曾建超 張明峰

系主任：曾文星

中華民國九十五年四月十九日

Department of Computer Science  
College of Computer Science  
National Chiao Tung University  
Hsinchu, Taiwan, R.O.C.

Date: April 19, 2006

*We have carefully read the dissertation entitled Mobility and Interoperability Supports for VoIP Converged Environments submitted by **Hung-Hsin Chang** in partial fulfillment of the requirements of the degree of Doctor of Philosophy and recommend its acceptance.*

Sheng-R. Chou

Chu-Sing Yen

Peng-Yi Tsai

Shiao-Li Tsao

Chien-Chao Tseng

Lü-Hsing Yen

Chen Yeh

Ming-Tung Chang

Thesis Advisor: Chien-Chao Tseng Ming-Tung Chang

Chairman: Stephan



## Abstract in Chinese

### VoIP 聚合環境之行動性與互用性的支援

學生：張弘鑫

指導教授： 曾建超 教授  
張明峰 教授

國立交通大學資訊工程學研究所博士班

#### 摘 要

隨著網路技術及終端設備的快速發展，未來的網路電話(VoIP)的環境將傾向於發展成爲一個支援多類網路的聚合環境。這些網路將包含公眾交換網路(PSTN)、公用陸地行動通訊網(PLMN)、有線(Wire-line)交換網路及無線(Wireless)交換網路。但在目前，要發展這樣的聚合網路，仍存在著一些問題。例如不同通訊信令(Signaling)之間的互通、穿越位址轉換器(NAT)的問題、交遞的問題、加密金鑰的傳遞問題及計費問題都仍需克服。本篇論文重點在探討互通性及行動交遞問題。

不同網路電話信令的互通是最根本需要解決的問題，不同的機構已經發展或定義出許多不同的信令規格，例如 H.323、MGCP 及 SIP。然而，在目前使用不同信令的電話設備仍無法互通。NAT 穿越問題也會阻礙同信令或不同信令之間的互通。現今許多的設備因爲缺乏實體 IP 位址(Public IP)，紛紛改採虛擬 IP 位址(Private IP)而變成隱藏在 NAT 之後。造成信令交換過程中，不管是通話發起或是接收端，會因無法得到正確的網路位址而致無法通訊。

再則，無線通訊技術的進展以及 SIP 在行動上的支援，使用者或終端設備可以在通話過程中間，改變其網路連接點而且仍能保持繼續通話。這過程稱爲交遞(Handover)，



它除了變更網路連接點(一般稱為 Access Point, AP)之外,仍包含有網路位址的改變以及通知對方更新網路位址的信令交換。這一連串的過程相當的長久,對於 VoIP 這類的及時應用會讓使用者感覺到中斷,所以必須要設計一機制來加速交遞過程。

行動通訊中有一個註冊伺服器(Registrar)來記錄使用者目前所在的位置(或網路位址)。交遞時除了通訊中的彼此需要位址更新之外,更必須要將新的位址重新登錄到註冊伺服器,以便新的通話能夠再接通。然而,這個資料庫因為會經常性的被讀取及更新,有可能會產生錯誤或損毀,如此,找不到使用者的位置以致無法建立通話。

本篇論文發表了一系列的解決方案來克服上述的諸多問題。用一個簡單又富有彈性的方法,利用 half-call model 來簡化設計並減少因新的通訊協定的加入所需的額外修改。在 NAT 穿越問題上,我們將應用層閘道器(Application level gateway, ALG)的概念從 NAT 中分離出來,所以此方法僅需更動到 Proxy,而不需修改 NAT 及使用者設備。我們也另外發展了一套利用服務網域的位置資訊進行快速網域交遞的協定,採用跨階層式的設計,大大減少交遞時所需花費的時間。最後,再針對位置資訊資料庫的毀損問題,我們分析了多種資料庫回復的方法,並比較其回存成本(Checkpoint)及通話斷訊成本(Lost-call)之間的關係。我們發現,資料庫的回存與否與此兩成本之間的比重是有相關的,若通話斷訊成本較高,並不需要使用到很複雜的回存機制,使用簡單的複製資料庫(Duplicated)的方法,就能達到最佳效益。

關鍵字: 通話狀態模型,互通性, SIP, H.323, MGCP, NAT 穿透, 交遞, AP 探索, 跨階層設計, 位置資料庫, 個人化回存, 詢呼成本

## Abstract

### Mobility and Interoperability Supports for VoIP Converged Environments

student : Hung-Hsin Chang

Advisors : Dr. Chien-Chao Tseng  
Dr. Ming-Feng Chang

Department of Computer Science  
National Chiao Tung University

## ABSTRACT

As the network and terminal technologies advance, the future Voice over IP (VoIP) environment is likely to be a converged infrastructure that consists of Public Switch Telecommunication Networks (PSTNs), Public Land Mobile Networks (PLMNs), Wire-line Packet-switched Networks, and Wireless Packet-switched Networks. However, in such VoIP converged environments, there exist several problems, such as signals interoperability, NAT traversal, handover delay, key distribution, and billing, which remain to be solved. This dissertation focuses on the mobility and interoperability supports for the VoIP converged environments.

The interoperability of different VoIP signaling protocols is one of the most important problems for the future VoIP converged environment. Several signaling protocols, such as H.323, SIP and MGCP, have been developed by different organizations to support VoIP communications. A device using a signaling protocol cannot operate with other devices using a different signaling protocol.

NAT traversal is another interoperability problem for SIP-based VoIP applications. In a VoIP converged environment, devices may situate behind an enterprise network with an NAT router due to the lack of public IP addresses and/or the administration purpose. For a device beneath an NAT router, it cannot establish, whether it initiates the communication or not, a VoIP session with another device. Previous solutions to this NAT traversal problem require

changes to the NATs and/or SIP user agents. Moreover, wireless technologies and SIP mobility make it possible for a device to change its network attachment from one point to another (henceforth referred to as handover), while retaining its VoIP session. The handoff procedure also includes sending user location update messages to both the correspondent node and the registrar for SIP-based VoIP applications. Such a handover process is considerably long and may cause serious interruption to the real-time VoIP session. Therefore a fast and smooth handover mechanism is a necessity for a VoIP converged environment.

Moreover, a registrar maintains the locations of VoIP users in a database, called user mobility database; users who wish to communicate with others should query the registrar to acquire the locations of the communication peers first. However, the user mobility database in a registrar may crash; causing call requests to fail. Therefore, failure recovery of the user location databases is another important issue for the mobility supports in VoIP converged environment. In this thesis, we present a series of solutions to the aforementioned problems. We first propose a simple, flexible framework for interworking gateway for different VoIP signaling protocols; the framework is based on a half-call model to reduce the design and implementation effort. For the NAT traversal problem, our method makes SIP proxies act like an application gateway and thus requires modification only to SIP proxies. Therefore our NAT traversal mechanism is more practical because it leaves NAT routers and SIP user agent programs intact. We also propose a novel topology-assisted cross-layer handover mechanism that can effectively reduce the overall handover delay of a VoIP session from several seconds to less than 120 ms. Finally, we study several user mobility database checkpoint methods and find that in most conditions the optimum checkpointing interval is either zero or infinity. That is to say, a user location record should either be always checkpointed at the registration, or be never checkpointed at all, depending on the weighting factor of checkpointing cost and that of lost-call cost.

Keyword: call state model, interworking, SIP, H.323, MGCP, NAT traversal, handover, AP probe, cross layer design, Mobile IP, location database, per-user checkpointing, paging cost

## **Acknowledgement**

I would like first to thank my two instructors Chien-Chao Tseng and Ming-Feng Chang. Especially for Tseng who have been guiding me from my early 20s to late 30s. Having their advice, I know how to do researches and state results formally and precisely.

During my first three years, both economic and emotional hardships came to me. Therefore, little progress was made. My family has always been the most powerful bolster and leads me to the fulfillment.

Please allow me to deeply appreciate all my friends, they share with me in joy and sad and encourage me at any given time. Thank my colleagues in Chin-Min Institute of Technology for their help freeing me from tedious committee meetings. Special thank to Mr. Yeh, chairman of the ShinWay Co., for his financial aid easing my economic difficulty.

Respect should be phrased to my father-in-law for his full support my PhD studentship. No doubt I must acknowledge greatly to my wife for her nonstop love and patient, so I could focus myself in research, for which I am eternally grateful.

Finally, I dedicate this achievement to my father, who left me in my 5th PhD year.

## Contents

Abstract in Chinese .....	viii
Abstract .....	x
Acknowledgement .....	xii
Contents .....	xiii
List of Figures .....	xv
List of Tables .....	xvii
Chapter 1 Introduction .....	1
Chapter 2 Background .....	5
2.1 VoIP protocols .....	5
2.1.1 H.323 .....	5
2.1.2 SIP .....	7
2.1.3 MGCP .....	8
2.1.4 IN Basic Call State Model .....	10
2.1.5 Interoperation .....	11
2.2 NAT traversal .....	12
2.2.1 NAT .....	12
2.2.2 Application-aware NAT solutions .....	13
2.2.3 Application-unaware NAT solutions .....	14
2.2.4 Locating a VoIP user inside a NAT .....	15
2.3 Fast handoff .....	16
2.3.1 Handoff procedures .....	17
2.3.2 Mobile IP handoff .....	19
2.3.3 Analysis handoff delays .....	21
2.3.4 Cross-Layer topology information .....	22
2.4 User mobility database .....	25
2.4.1 Re-registration .....	25
2.4.2 Checkpointing and restoration .....	26
Chapter 3 Integrated call agent .....	28
3.1 Integrated call agent architecture .....	28
3.1.1 Interworking gateway .....	28
3.1.2 Integrated call agent .....	30
3.2 Mapping of VoIP protocol messages to the BCSM messages .....	31
3.2.1 Uniform events .....	31
3.2.2 H.323 slow-start .....	33
3.3 Implementation and result .....	35
Chapter 4 VoIP's NAT traversal .....	37
4.1 NAT traversal architecture .....	37
4.2 NAT type decision .....	38

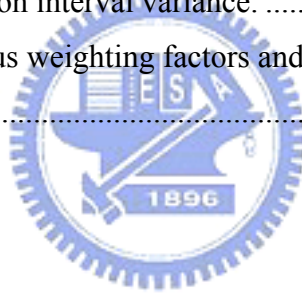
4.3 Locate an internal user .....	38
4.3.1 Without a public SIP proxy .....	39
4.3.2 With a public SIP proxy .....	39
4.4 Transport address translation .....	40
4.5 Scenarios for call establishment .....	42
Chapter 5 Topology-aided fast handoff .....	46
5.1 LAMP scheme .....	46
5.1.1 Registration and pre-allocation .....	48
5.1.2 Handoff .....	49
5.2 The integrated cross-layer fast handover .....	50
5.2.1 Predicted bi-casting .....	50
5.2.2 AP direct association .....	51
5.3 Mobile IP handoff .....	52
5.3.1 Using Foreign-Agent CoA .....	53
5.3.2 Using Co-located CoA .....	54
5.4 Implementation and result .....	56
5.4.1 SIP handoff .....	56
5.4.2 Mobile IP handoff .....	59
Chapter 6 Mobility database restoration .....	62
6.1 Three checkpointing algorithms .....	62
6.1.1 Periodically checkpointing the modified record .....	62
6.1.2 Lin's per-user checkpointing algorithm with an exponential timer .....	63
6.1.3 Lin's per-user checkpointing algorithm with a fixed checking interval .....	64
6.2 Cost function analysis .....	64
6.2.1 FIXED .....	65
6.2.2 LINEXP .....	67
6.2.3 LINFIX .....	68
6.3 Results .....	69
6.3.1 Simulation result .....	71
Chapter 7 Conclusions .....	75
7.1 Integrated call agent .....	75
7.2 VoIP's NAT traversal .....	76
7.3 Handoff .....	76
7.4 HLR .....	77
Chapter 8 Future work .....	78
Reference .....	79

## List of Figures

Figure 1. Converged VoIP communication environment. ....	1
Figure 2. A simple H.323 architecture.....	6
Figure 3. An H.323 message flow to set up a call. ....	7
Figure 4. A generic SIP architecture.....	8
Figure 5. An SIP message flow to establish a session.....	8
Figure 6. MGCP architecture. ....	9
Figure 7. An MGCP message flow to establish a call. ....	10
Figure 8. IN architecture. ....	10
Figure 9. Simplified IN BCSMs.....	11
Figure 10. Latencies of a handoff.....	17
Figure 11. Link-layer handoff procedures and delays.....	18
Figure 12. Network-layer handoff procedures and delays. ....	18
Figure 13. Application-layer handoff procedures and delays.....	19
Figure 14. Mobile IP handoff in the FA-CoA case.....	20
Figure 15. Components developed for a general VoIP gateway.....	29
Figure 16. A SIP/H.323 gateway. ....	29
Figure 17. A converged VoIP network using gateways. ....	30
Figure 18. A converged VoIP network managed by integrated call agents. ....	31
Figure 19. An example of H.323 and MGCP interworking using 2 ICAs. ....	31
Figure 20. Mapping VoIP messages to BCSM messages.....	32
Figure 21. BCSMs for the H.323 slow-start.....	34
Figure 22. Call flow of H.323 (slow-start) and SIP interworking.....	35
Figure 23. Components used in our platform.....	36
Figure 24. Call establishment delays.....	36
Figure 25. The NAT traversal for VoIP environment. ....	38
Figure 26. User registration without a public SIP proxy. ....	39
Figure 27. User registration using a public SIP proxy.....	40
Figure 28. NAT hole punching.....	42
Figure 31. LAMP architecture.....	47
Figure 32. The registration and pre-allocation procedures.....	49



Figure 33. Registration and pre-allocation procedures using a dedicated LS.....	49
Figure 34. Combined handover procedures by applying the proposed cross-layer solutions. ....	52
Figure 35. Message flow for the FA-CoA case. ....	53
Figure 36. Message flow for the CCoA case.....	56
Figure 37. The Mobile IP handoff experimental setup.....	59
Figure 38. Three per-user checkpointing algorithms. ....	63
Figure 39. Two consecutive checkpoints (FIXED).....	65
Figure 40. Two possible cases of checkpointing (LINEXP).....	67
Figure 41. Comparison of checkpointing algorithms for exponential registration interval.....	71
Figure 43. The effects of timer expiration interval. ....	73
Figure 42. Cost functions for various weighting factors (exponential registration interval). ....	71
Figure 44. The effects of registration interval variance. ....	73
Figure 45. Cost function for various weighting factors and different variance of registration interval. ....	74



## List of Tables

Table 1. An example of a network deployment table. ....	48
Table 2. Internal procedures of an SIP UA for initiation, call establishment and handover. ....	57
Table 3. Measurement results of handover delays before the proposed improvements. ....	58
Table 4. Summary of the reduced delays by using different mechanisms. ....	58
Table 5. Handover delays by applying the proposed improvements. ....	58
Table 6. Means and standard deviations of handoff delay and the number of lost packets in different settings. ....	61





## Chapter 1 Introduction

Current network technology helps information exchanging between network nodes via the well-connected Internet within a few second. This information not only includes plain texts but also multimedia data such as the digitized voice/video data that are converted by encoding and compressing methods. The high bandwidth and high transmission rate over the network enable the digitized voice data to reach the destination without much delay, so that a user can hear the reproduced voice in real-time as if the voice was just produced. One important real-time application is the telephony communication, also referred to as VoIP (Voice over IP).

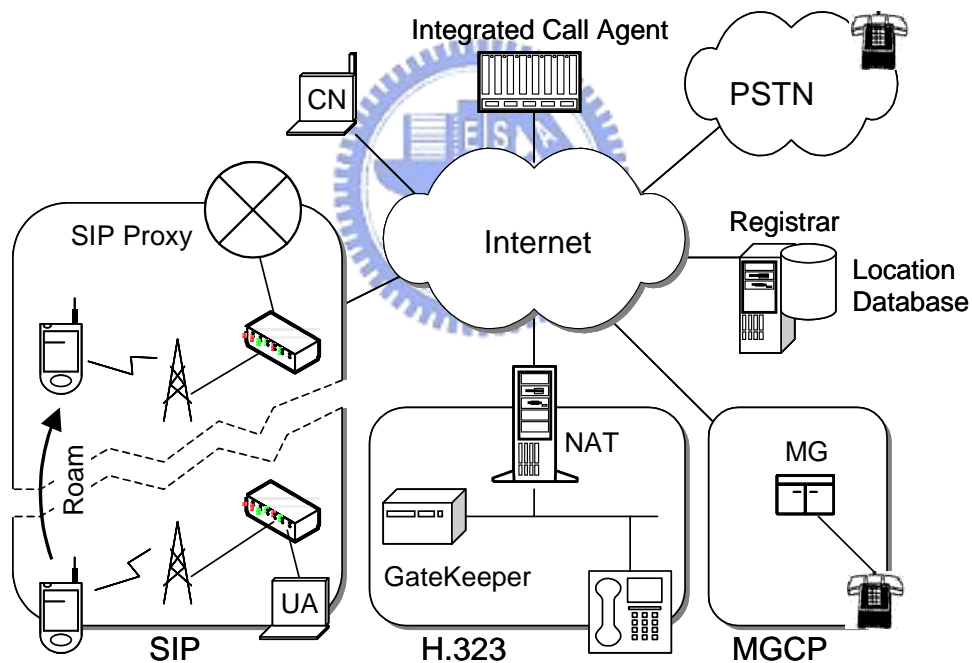


Figure 1. Converged VoIP communication environment.

To support VoIP communications, many signaling protocols such as MGCP (Media Gateway Controller Protocol), H.323 and SIP (Session Initiation Protocol) have been developed. They use sophisticated speech codecs (G.723.1, G.729, and so on.) to reduce the bandwidth required for a call and use RTP/RTCP (Real-time Transfer Protocol/Real-time Transfer Control Protocol) to transport digitized voice packets over IP networks. Using

RTP/RTCP to transmit and reproduce the voice from digitized packet, VoIP application can withstand the delay, jitter, and out-of-order arrival of the packets due to the network transmission behavior that is based on best-effort delivery.

Since these signaling protocols are expected to co-exist in the near future, users of VoIP applications are divided into several groups that cannot communicate with each other because of the incompatibility of the signaling protocols. To enable communications between different groups of VoIP application users, gateways between the different VoIP protocols are needed. The gateways have to perform two functions — signaling conversion and media conversion, because the signaling protocols are different, and the voice media are transmitted in different formats. However, traditional gateways are implemented in a brutal manner that transforms signaling from one protocol to another protocol, so that there are many mutual gateways such as H.323-to-SIP, SIP-to-MGCP and H.323-to-MGCP. It is very inconvenient to install all these gateways together to support the interworking functions of VoIP protocols. Hence, it is necessary to find an easy way to implement a gateway so that VoIP will be easy to install and can be used for supporting all existing signaling protocols.

Two VoIP users exchange their terminal capabilities (such as codecs and compression algorithm) and transport addresses (i.e., the port and IP address for receiving RTP packets) before they establish a VoIP call. Two uni-directional connections are then created to transmit digitized voices and the call is established. Yet, as the number of VoIP users booming, more and more users are hidden by NATs (Network Address Translators) and adopt private IP addresses due to the shortage of public IP addresses, they share a common public IP address on the public side of the NAT. The private IP addresses are banned by the Internet, so that any packets designated to a private address will be dropped. It poses the problem of NAT traversal that a VoIP user using a private transport address cannot receive either call request or voice packets from the Internet; their packets are abandoned and their communications are blocked.

Many researchers have devoted to the NAT traversal problem. Also, different solutions were proposed to solve different NAT types according to the NAT specification. An application-aware solution enables NATs to interpret the payloads of VoIP signaling

messages and replace the private transport addresses with a unique external location. Another approach is an application-unaware solution where the modifications are made to user application and leave the NAT unchanged. In fact, schemes of those approaches made changes to the NAT and user software may not be tangible, because they are usually under the control of the Internet service providers or enterprises. Therefore, a practical way for NAT traversal problem should not make changes to either NATs or user agent programs.

For VoIP users, there has been a trend of using handheld devices and roaming between IEEE 802.11 wireless local area networks (WLANs). Although WLANs have been widely deployed as an infrastructure providing high-speed data services to mobile users, offering VoIP services on WLANs is still considered problematic due to an inherent limitation of the IEEE 802.11 MAC protocol — the considerably long handoff process. Moreover, handoff may also involve activities at higher layers. If the handoff entails changing network domains (referred to as an inter-domain handoff), the mobile user needs to acquire a valid IP address using schemes such as DHCP (Dynamic Host Configuration Protocol) in the new network domain. If Mobile IP is used for network-layer mobility management, the mobile user should change its mobility agent and perform registration accordingly (henceforth referred to as a layer-3 handoff). These two steps cause large amount of delay.

Nevertheless, VoIP applications are time-sensitive and cannot tolerate long layer-2 plus layer-3 handoff delays. In order to maintain the desired quality of services demanded by VoIP or real-time multimedia applications, the overall handoff latency should be minimized. Many studies have made to reduce the handoff delays. Unfortunately, previous works mainly concentrate on the improvement on a single layer such as link or network layer, and lack of cross-layer considerations and enhancements.

In addition to the handoff delay problem, the handoff also poses a user location-tracking problem when a roaming user changes his location from time to time. Usually, a mobile user have to register his recent location to a service agent (such as an SIP registrar) that stores user location to a database, so that other users may know his location by querying this service agent and issue a call request to him. If the mobility database fails, calls to mobile users

cannot be set up. Henceforward, the service agent should perform user paging to locate the user and rebuild the mobility database; this leads to fairly large cost.

One way to solve the call loss problem is to enhance the reliability of the mobility database. For UMTS (Universal Mobile Telecommunications System), the mobility database is periodically checkpointed to a non-volatile storage. Several algorithms have been used to find out the optimal checkpointing interval that minimizes the total cost of checkpointing cost and lost-call cost.

In this thesis, we present a series of solutions to the aforementioned problems. A simple, flexible framework for implementing interworking gateways among several VoIP signaling protocols was proposed to achieve a converged VoIP network. This framework is based on a basic call state machine (BCSM) that is used in the intelligent network (IN). We also propose an NAT traversal method to help the VoIP application to operate in an NAT environment without any modification to NATs or VoIP user applications; this method is also applicable to different NAT types. For the handoff delay problem, a topology-aided cross-layer fast handoff mechanism is developed. By integrating the base station information and the relationship between the base stations and the service agents, our mechanism can significantly reduce the handoff delay to less than 120ms, which is considered unperceptible to real-time communications. In addition, we have analyzed several user mobility database restoration methods. Our study finds that there doesn't need a sophisticated checkpointing algorithm. A user location record should either be always checkpointed at the registration, or be never checkpointed at all, depending on the weighting factor of checkpointing cost and that of lost-call cost.

The remainder of this thesis is organized as follows. Chapter 2 gives detailed description and causes of the VoIP problems solved in the thesis. Chapter 3 discusses previous related work on those problems. Chapter 4 through chapter 7 elaborates our solutions to those problems. Chapter 8 gives the result. Conclusions and future work are in the Chapter 9.



## Chapter 2 Background

### 2.1 VoIP protocols

The PSTN (Public Switched Telephone Network) has provided reliable voice communication for decades. A telephone call to anyone in the world can be established in seconds, and the voice quality is good in general. Voice waveform transmitted in the PSTN is encoded using PCM (pulse-code modulation, G.711 A-law and u-law, both 64kbps) technique. To establish a telephone call between two parties, a dedicated link needs to be set up, and the link has to be torn down when the call terminates. This work is performed by telephone switches exchanging standard signaling, such as ISUP (Integrated Services Digital Network User Part).

As the Internet becomes overwhelmingly widespread, transporting voice communication traffic using the Internet Protocol (IP) provides advantages over the traditional PSTN. This is often referred to as IP telephony or VoIP (Voice over IP). VoIP can use sophisticated speech codecs, such as G.723.1 and G.729 [1], to reduce the bandwidth required for a call. VoIP communications use RTP/RTCP to transport voice packets over IP networks. To establish a call, the two parties involved should negotiate the codec and the RTP/RTCP [2,3] ports used for the call, as well as exchange messages to set up and terminate the call. H.323 and SIP are two existing VoIP signaling protocols. Both are designed to support the setup, communication capacity exchange and tear-down of a VoIP call.

#### 2.1.1 H.323

H.323 [4] is an ITU-T recommendation for multimedia conferencing over packet-switched networks. It is a protocol umbrella that consists of many standards, including Q.931 call control protocol, H.225 registration and administration, and H.245 media negotiation. NetMeeting of Microsoft and VoIP products from Cisco all support H.323 multimedia communications over packet-switched networks. A simple H.323 environment is depicted at

Figure 2. H.323 defines a gatekeeper as a manager and arbiter over a network. The gatekeeper is an optional entity in charge of endpoint registration, address translation, and bandwidth assignment. A call setup between two H.323 endpoints needs a large amount of signaling exchange. A gatekeeper can participate in the signaling exchange (a gatekeeper-routed call) or do not involve (a direct call). Figure 3 shows a message flow for a direct call that includes administrative messages (authentication request/confirm, *ARQ/ACF*), call setup signaling (*SETUP*, *CALLPROC*, *ALERTING* and *CONNECT*) and capability negotiation (H.245 capability exchange and logical channel opening).

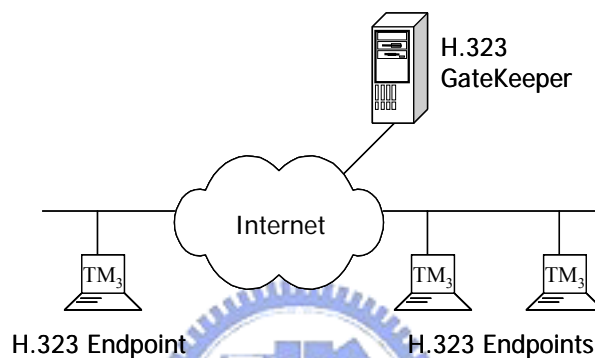


Figure 2. A simple H.323 architecture.

One of the major criticisms against H.323 is the time and complexity involved in setting up a call; H.323 version 1 uses multiple stages to exchange signaling and media capabilities. Moreover, messages are transported using TCP, which requires additional session set-up time. Recent H.323 versions can include both signaling and media capabilities in a single message transmitted by UDP [5], which eliminates the additional round-trip time for TCP handshake. However, with too many options within the signaling messages, H.323 still has compatibility problems for the products from different providers.

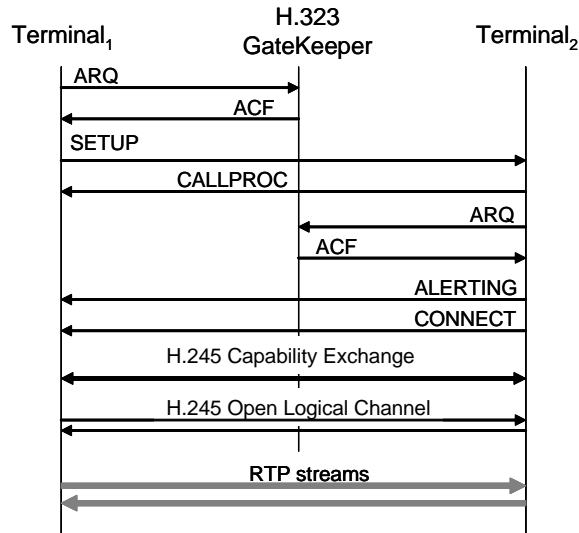


Figure 3. An H.323 message flow to set up a call.

### 2.1.2 SIP

SIP (Session Initiation Protocol, [6]) has been standardized by IETF for initiating interactive communication sessions between users. It can be used to establish Internet telephony calls. With six text-based commands: *REGISTER*, *INVITE*, *ACK*, *OK*, *INFO*, and *BYE*; SIP is a lightweight and text-based protocol and is simpler than H.323 and other signaling protocols. It has been widely adopted by communication standards such as 3GPP and 3GPP2. With early media capability exchange feature, a SIP terminal can issue only one command, *INVITE*, to set up a call. Therefore, it is faster than H.323 in call establishing and many consider SIP a powerful alternative to H.323.

The network components of an SIP application are depicted in Figure 4. An SIP user agent (UA) that is a user endpoint runs on a desktop PC or a mobile node (MN), and it issues or receives session establishment requests. An SIP UA has to register with an SIP registrar using a *REGISTER* message. The SIP registrar maintains the user profile such as the subscriber information and the user's URL location. Finally, an SIP proxy relays SIP messages on behalf of SIP UAs or other SIP proxies.

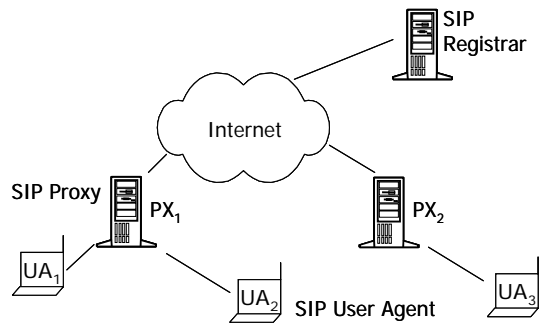


Figure 4. A generic SIP architecture.

Figure 2 shows a typical message flow for establishing a SIP session. Without loss of generality, we assume that UA<sub>2</sub> has already registered with the SIP registrar. As shown in the figure, UA<sub>1</sub> sends a *REGISTER* message registers with the SIP registrar via an SIP proxy, and the SIP registrar grants the registration by replying an *OK* message. After then, if UA<sub>1</sub> wishes to request a call to UA<sub>2</sub>, UA<sub>1</sub> issues an *INVITE* message to UA<sub>2</sub>. In the meantime, UA<sub>1</sub> prepares a RTP channel for receiving voice stream from UA<sub>2</sub>. Once UA<sub>2</sub> accepts the session request, UA<sub>2</sub> creates a RTP channel according to the capabilities described in the payload and replies an *OK* message to UA<sub>1</sub>. UA<sub>1</sub> further sends an *ACK* message to confirm the success of session establishment. Aftermath, voices are digitized into packets and sent to each other through the RTP channels.

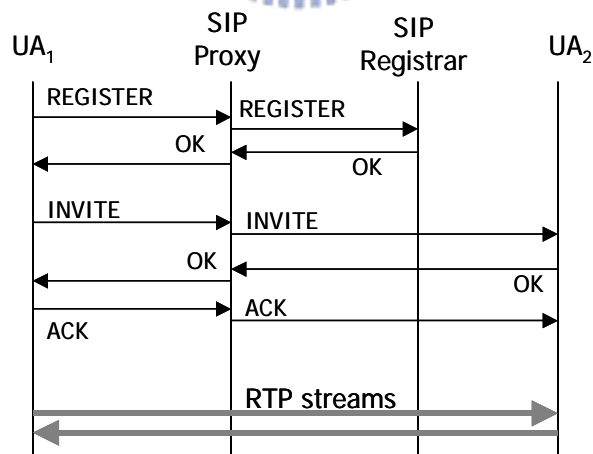


Figure 5. An SIP message flow to establish a session.

### 2.1.3 MGCP

To enable communications between VoIP users and PSTN users, gateways between the PSTN and IP-based networks are needed. Since the signaling used in the VoIP networks and

the PSTN are different, and the voice media are transmitted in different formats, the gateways have to perform two functions: signaling conversion and media conversion. The two functions can be carried out in two separated entities: MGC (Media Gateway Controller) and MG (Media Gateway). An MGC (also referred to as a CA, Call Agent) performs the signaling conversion function, and an MG performs the media conversion function. A standardized protocol can be used between an MGC and MGs. IETF proposed the MGCP (Multimedia Gateway Control Protocol, [7]) architecture depicted in Figure 6. MGCP is a master-slave protocol. A CA is a master and has control over several MGs; an MG acts as a slave and is kept simple and passive. A CA also performs the call control function as a gatekeeper in H.323, but has much tighter control. A CA instructs an MG to establish, maintain, and terminate a call between a VoIP terminal and an endpoint in the PSTN.

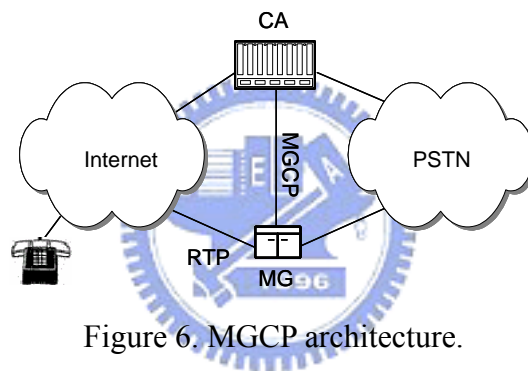


Figure 6. MGCP architecture.

The message exchange for connecting a call in MGCP is more complex than SIP and H.323 because the MG is passive. A typical message flow for establishing a call between two MGCP endpoints are depicted in Figure 7. For the detail of the message flow readers are referred to the MGCP specification. MGCP is based on a centralized network infrastructure. To serve to a wide area network, a group of CAs needs to coordinate, but MGCP does not specify how the CAs interact. Signaling used in inter-CA communications can be SIP or ISUP. MGCP describes the media capability and parameter using SDP (Session Description Protocol, [8]). Recently Megaco (or H.248, [9]), an enhanced version of MGCP, is promoted jointly by the IETF and ITU-T.

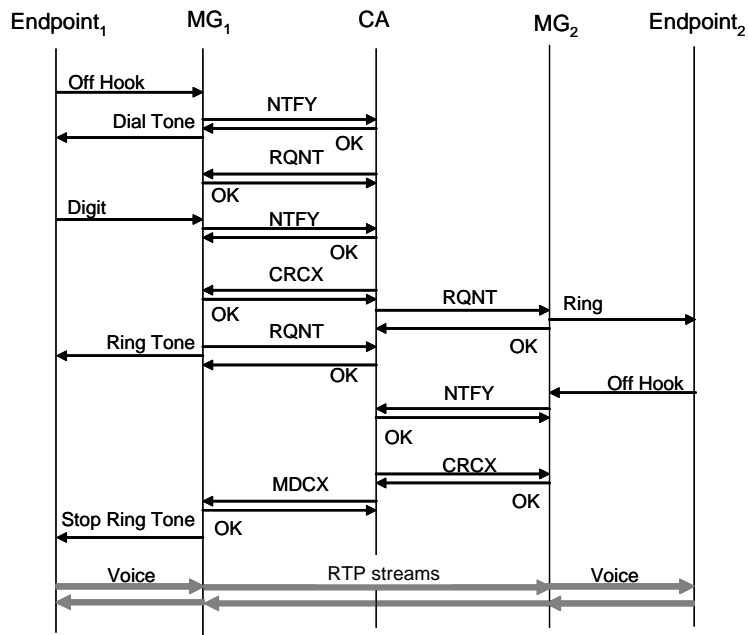


Figure 7. An MGCP message flow to establish a call.

#### 2.1.4 IN Basic Call State Model

An intelligent network (IN, [10]) separates the service logic from the switching function in the telecommunications network; the service intelligence is placed in computer nodes that are distributed throughout the network. This provides the network operators with the means to develop and control services more efficiently. New capabilities can be rapidly introduced and customized for the network. A simple IN architecture is depicted in Figure 8. A service switching point (SSP) is an IN-capable switching system dealing with the call control functions that establish, maintain, and clear a call. In addition, it detects user requests for IN-based services and queries a service control point (SCP) to determine how the call should be handled. The SCP contains the service logic to provide the IN services.

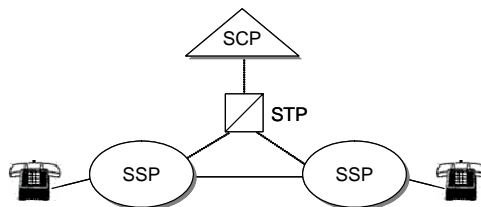


Figure 8. IN architecture.

The basic call control function of an SSP is supported by a finite state machine (FSM) called basic call state model (BCSM). The BCSM consists of point in calls (PICs), detection

points (DPs), and events. PICs represent the switching activities or states that a call goes through from call origination to termination. DPs are states at which transfer of control from the SSP to the SCP can occur. Events are messages exchanged between BCSMs, and trigger state transitions of the BCSMs.

Figure 9 shows that the BCSM is based on half-call model; a call model consists of two half-call models: the originating BCSM (O\_BCSM) and terminating BCSM (T\_BCSM). The O\_BCSM represents the states associated with the call originating party; the T\_BCSM represents the states associated with the call terminating party. Note that the originating and terminating BCSMs interact with the call parties using ISUP messages. The call control functions are performed through the exchange of events between the O\_BCSMs and T\_BCSMs. These events include *Setup*, *Alert*, *Answer*, *Disconnect*, *Busy*, *No-Answer*, and *Abandon*. For details, the readers are referred to [10].

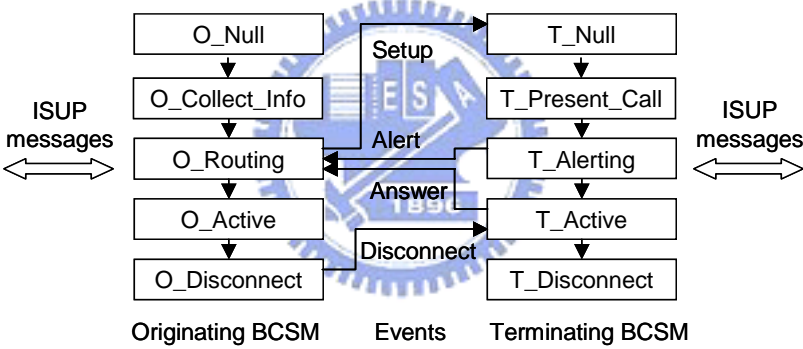


Figure 9. Simplified IN BCSMs.

2.1.5 Interoperation

Since H.323 and SIP are expected to co-exist in the near future, gateways between H.323 and SIP terminals are needed for the interworking function between H.323 and SIP. EURESCOM proposed a project for providing IN functionality for H.323 telephony calls [11]. Vemuri described an inter-operation model called SPHINX (SIP, H.323 and IN interworking) where H.323 and SIP terminals can access IN services [12]. In addition, an SIP-H323 gateway is a byproduct of this inter-operation model based on the half-call state model of the IN. Agrawal, Singh and Schulzrinne [13,14] specified the requirements for SIP-H.323 interworking. Gurbani and Rastogi [15] and Haerens [16] suggested ways to map the call



control of SIP to IN, but they did not support the H.323 slow-start call setup. Ackermann, et al., have implemented a sip-h323 gateway as a basis for supplementary service interworking [17]. They use a dedicated call model that transfers H.323 messages to SIP messages, and vice versa. Jiang et al. have presented a Columbia InterNet Extensible Multimedia Architecture (CINEMA) where MGCP and H.323 can be interworking through SIP [18]. Nevertheless, it is difficult to modify this call model to cooperate with other VoIP protocols.

## **2.2 NAT traversal**

### **2.2.1 NAT**

The NAT (Network Address Translation Protocol, [19]) provides a way to alleviate the need of the IP address space due to the Internet booming. It enables hosts beneath a NAT (internal host) to share one common IP address for creating network connections to hosts on the Internet (external host). The NAT replaces the source address and port as its IP address and a dynamically allocated port in the IP header in the outgoing packet. Therefore, the correspondent host believes that the connection is originated from the NAT. Within a NAT connections are identified by the bindings between internal hosts' source ports and NAT's outgoing ports. So, the NAT can pass the replied packet destined to the NAT to the exact internal host. The IP header of the replied packet is restored accordingly.

Though NATs are operated in port oriented, not all the inbound packets arrived at the binding port on a NAT will be redirected to the internal host. The NAT is further classified as full cone, restrict cone, port restrict, and symmetric NAT according to their port binding rules and delivery restrictions. The rule for the former three NAT types to create address binding is the same, where all requests from the same internal IP address and port are associated to the same port binding. However, delivery restrictions are different. A full cone NAT allows return packets to the internal host from any external host as long as the packet is sent to the associated port. A restrict cone NAT is similar to a full cone one except that an external host can send a packet to the internal host only if the internal host had previously sent a packet to it. A restricted cone NAT maintains a database so as to track which hosts have been visited. A

port restricted cone NAT is like a restricted cone one, but the restriction includes port numbers. A symmetric NAT is more restrictive and secure. Every connection uses a different port binding. Furthermore, only the destination host is allowed to return packets back to the internal host.

Usually, users inside a NAT employ private IPs that are reserved by the Internet Assigned Numbers Authority (IANA, [20]). Since private addresses have no global meaning, routing information about private networks will not be propagated. Thus, any packet addressed to these private IP addresses will be ignored. Consequently, connections originated from an external host to an internal host are forbidden and the NAT traversal problem is formed. The impact of the traversal problem also includes the retention of port bindings, where there is no guarantee that the NAT implementation will keep the port binding for a long time [21]. In addition, locating an internal user is difficult because internal users under different NAT may use the same private IP address. These problems hinder the deployment of VoIP applications.

### 2.2.2 Application-aware NAT solutions

An application-awareness NAT means that a NAT has the ability to interpret application payloads and create port bindings on demand. An application-aware NAT acts as a translator that translates application packets back and forth and as a router that routes packets between internal hosts and external hosts. This is also referred to as an application level gateway (ALG [22]). An ALG is application dependent, so that it requires special handling for each VoIP application.

Kuthan and Rosenberg [23] suggested decomposition of application-awareness from the NAT. An ALG becomes an intermediate device and controls address processing and associations by communicating with a NAT through a generic application-independent firewall control protocol (FCP). The FCP allocates and releases NAT port binding. Such solution is also known as Middlebox Communications (MidCom [24,25]) where the Middlebox is a device that locates between different realms or networks. Example Middleboxes are NATs. A MidCom agent (an ALG) is a logical entity that is external to a

Middlebox, handling the application specific processing and communicating with one or more Middleboxes. For example, a SIP proxy can embed a MidCom agent and may communicate with NATs or firewalls to request port bindings. Clearly, this method makes changes to the NATs and is impractical because it is impossible to upgrade the all the NATs on the Internet.

### 2.2.3 Application-unaware NAT solutions

An application-unaware NAT solution leaves NATs unchanged. Rosenberg et al. suggested such a solution called STUN (Simple Traversal of UDP through NAT, [26]). A STUN server is an external host and acts as a MidCom agent to learn the external NAT transport address of an outbound connection issued by an internal STUN client and report the learned port and address to the STUN client (this procedure is referred to as a port-learning). Two STUN servers can be used to determine what NAT type an internal STUN client resided by performs sequences of tests for the port binding rule and delivery constraint of the NAT. If the NAT is a full-cone one, the internal user can proceed the call request and use the learned external transport address as its RTP receiving address since the full-cone NAT will relay any incoming packets reached at the bound port to the internal user.

A fortified version of STUN, called TURN (Traversal Using Relay NAT, [27]), was proposed to conquer the constraints of the symmetric NAT type. It uses an external server (TURN server) to relay the packets into a NAT. An internal user may request the TURN server to allocate a pair of transport addresses on the TURN server; one is used as user's destination RTP transport address, and the other is used as a relay transport address that the user host creates an outbound connection to it and receives inbound RTP packets from it. Hereafter, the TURN server receives the inbound RTP packets coming into the destination port and pushes them out of the relay port into the NAT.

In STUN and TURN, though NATs remain intact, the user agent program should be STUN/TURN enabled and turn off the silence suppression to keep the port binding on NATs. This requires changes to the VoIP application software.

#### 2.2.4 Locating a VoIP user inside a NAT

To allow inbound calls, internal users are required to register at a registrar (such as SIP Registrar or SIP Proxy) with routable addresses, so that they can be located. J. Rosenberg et al. [28] suggested employing a STUN server to learn an external signaling address from the outbound *REGISTER* message out of a NAT as a contact address for call parties. Once the external signaling port binding on the NAT is created, a keep-alive packet should be periodically sent, or the signaling address may expire. Thus, an inbound call request can go through the NAT to the internal user. Unfortunately, not only lots of keep-alive traffic is brought in, but also this external signaling address is not worked for the internal user within a symmetric NAT due to the stern restrictions.

Davies et al. [29] mandated a SIP proxy (a TURN-like server) to allocate transport addresses for relaying both signaling and RTP packets into a NAT. They also introduced a proxy interface agent (PIA) inside a NAT to intervene the call establishment between internal users and the SIP proxy. Internal users consider as if they are communicating with the PIA. The PIA acts on behalf of one or more internal users and establishes calls with the external users. As a result, inbound RTP packets from an external user are first sent to the SIP proxy, then relayed via the NAT to the PIA, and finally delivered to the internal user. Meanwhile, outbound RTP packets also follow the same way in the reverse direction. A notable drawback is the performance deterioration due to an additional hop-count of the RTP packets via the PIA.

Sen proposed another solution that made changes to the SIP protocol [30]. First, a Behind-NAT tag is added to the *REGISTER* message indicating a user is behind a NAT. If the tag is set, the SIP proxy should take the source IP address of the *REGISTER* packet as the user' public address rather than the address specified in the message body. Second, a SIP *PING* message is introduced for the purpose of keeping the communication path alive between the internal user and the external SIP proxy. Thus, an inbound call request can be delivered to the internal user.

Sen's method does not need a broker to setup a call. Obviously, an internal user takes care of the NAT traversal problem. The internal user must use symmetric RTP, which sends and receives RTP packets at the same port, and acquire a relay transport address from the media proxy as its source and destination RTP connection. That is to say, the internal user sends RTP packets through a NAT to the media proxy, and then the media proxy relays those packets to the other call party. The symmetric RTP benefits inbound packets to travel into a NAT because the outbound packets have created a port binding for them.

The above solutions somehow change the user agent software, the signal proxy, NAT, or all of them, and mandate the coexistence of a signal proxy. Some even alters the protocol. Since those programs and servers are the productions of some companies and may be difficult to be modified at our wish, it seems impractical to implement those solutions in the near future.

### **2.3 Fast handoff**

For ubiquitous Internet accesses, a mobile device is expected to equip multiple wireless interfaces and to roam within an IEEE 802.11 wireless system and/or among heterogeneous wireless systems. The handoffs might involve a few to dozens of procedures, and introduce variant delays depending on the types of handoffs, such as inter-system or intra-system handoffs, and/or link-layer or network-layer handoffs. For instance, a mobile node (MN) moving from one access point (AP) to another AP without changing its IP address, i.e. a link-layer handoff, needs to spend around 300ms to detect the lost of the signal from the original AP, to search and then to re-associate with a new AP [31]. However, if the handoff invokes the IP change of an MN, namely a network-layer handoff, other than the link-layer procedures, the network handoff further requires an MN to acquire network resources, such as an IP address, and to inform the corresponding node (CN) to re-establish a new network connection [32]. In this case, the handoff delay might increase to several seconds.

During handoff procedures, network connections between two peer nodes are temporarily lost, and packet delays are thus introduced. For real-time applications such as voice over IP

(VoIP) and video on demand (VoD), which are sensitive to delays and delay jitters, the handoff latency influences service qualities, and poses a serious problem from a user's perspective. According to the International Telecommunication Union (ITU) specification [33], less than 150ms handoff delay is recommended for any mobile communication system.

### 2.3.1 Handoff procedures

The handoff process of wireless real-time communications consists of the aforementioned link, network, and application layer handoff procedures that involve various handoff delays, denoted as  $t_{LK}$ ,  $t_{NW}$  and  $t_{APP}$  respectively. Figure 10 shows the handoff latencies of wireless real-time communications based on DHCP (Dynamic Host Configuration Protocol, [34]) and SIP mobility.

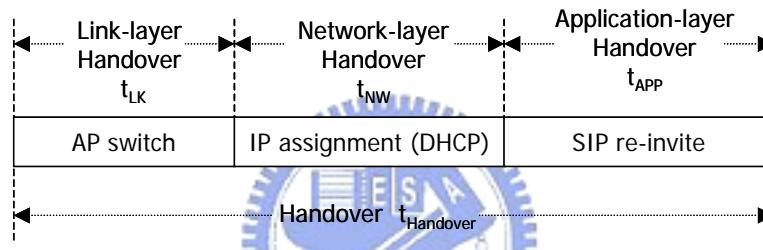


Figure 10. Latencies of a handoff.

Figure 4 shows the delay of an 802.11 link-layer handoff. The link-layer handoff delay consists of a probe, an authentication and an association delays that are denoted as  $t_{Probe}$ ,  $t_{Auth}$ , and  $t_{Assoc}$  respectively as shown in the figure. An MN starts a link-layer handoff by performing a probe phase to discover the next AP to associate with. The *probe* phase would normally introduce a considerable delay because the MN needs to scans channels to find a preferred AP, which is normally the AP with the strongest signal strength. The probe delay depends on both the number of channels the MN scans and the time the MN works with any particular channel. The time the MN works with a channel is further depending on the minimum channel time  $t_{Wmin}$  that an MN will wait for any AP response after the MN sends a probe request on the channel, and the maximum channel time  $t_{Wmax}$  that the MN will stays on the channel for other AP responses if it receives an AP response before  $t_{Wmin}$  elapses

After discovering a preferred AP, the MN performs authentication and association

procedures with the AP as specified in the IEEE 802.11 standard. Although, there are many different authentication approaches such as WPA (Wi-Fi Protected Access, 802.1x [35]) and WPA2 (802.11i [36]) can be used, these authentication approaches perform additional access-right checking through the Internet and further cause delays. Therefore, they are not considered in this paper. We assume the standard 802.11 authentication method is applied. The delay of a link-layer handoff is summarized in Figure 11.

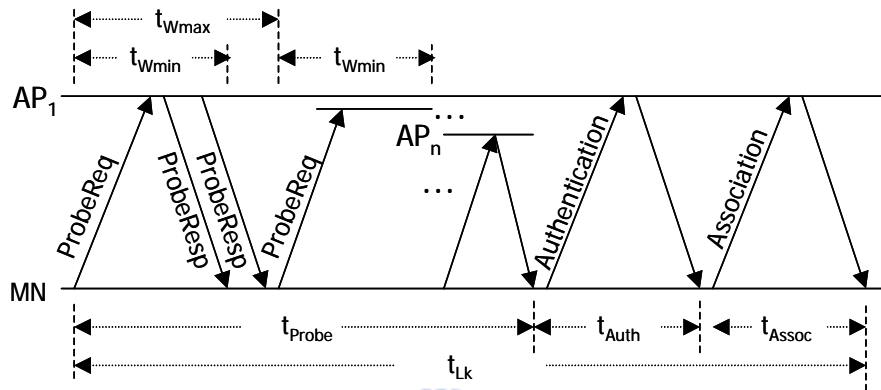


Figure 11. Link-layer handoff procedures and delays.

A handoff between two APs that belong to different sub-networks may cause an IP transition. This handoff requires further performing a network-layer handoff procedure, in addition to the link-layer handoff procedure. Once an MN detects the sub-network change, it tries to acquire a new IP address via DHCP. Therefore, the network-layer handoff latency is composed of the time for the subnet change detection ( $t_{\text{Detec}}$ ), IP acquisition ( $t_{\text{DHCP}}$ ) and IP/Gateway configuration ( $t_{\text{Conf}}$ ), and is depicted in Figure 12.

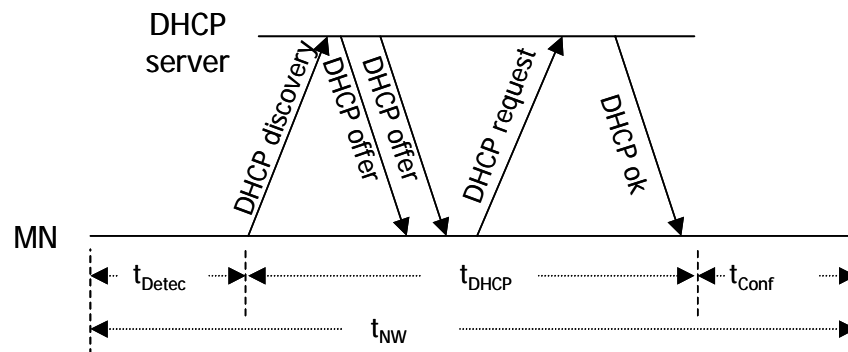


Figure 12. Network-layer handoff procedures and delays.

Afterward the MN needs to follow the procedures that are specified by different application-layer protocols to inform the CNs its new location. The SIP mobility requires



MNs to perform SIP registration and to re-establish connections to the CNs in order to receive RTP packets from the MN's new location. Figure 13 shows the procedures and their delays, i.e. SIP registration delay ( $t_{\text{Reg}}$ ), SIP re-invite delay ( $t_{\text{Update}}$ ), packet sending and receiving delays ( $t_{\text{Send}}$ ,  $t_{\text{Recv}}$ ).

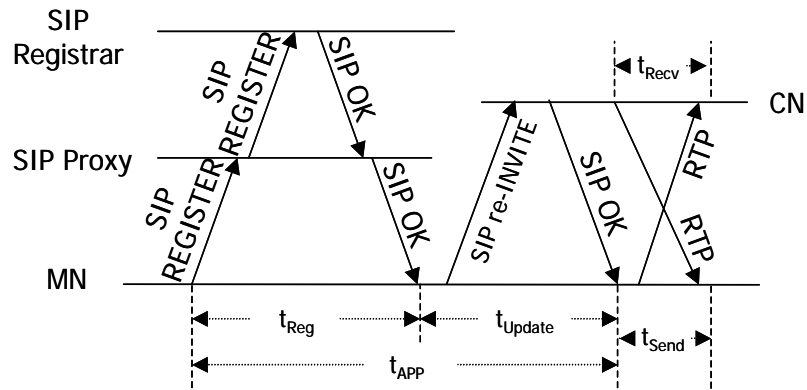


Figure 13. Application-layer handoff procedures and delays.

### 2.3.2 Mobile IP handoff

Mobile IP (MIP, [46]) is an Internet standards-track protocol that enhances the existing IP protocol to accommodate host mobility. In MIP, a special host called mobility agent (MA) maintains registration information for mobile nodes. When an MN moves away from its home network, the MA located in the MN's home network, referred to as the MN's Home Agent (HA), will tunnel packets for the MN. Tunneled packets are usually, though not always, handled by the MA on the MN's visiting network called Foreign Agent (FA). With the intervention of HA and FA, an MN away from home network can retain its connection to the Internet.

MIP offers two options on care-of address (CoA) that identifies an MN in the visited network: foreign-agent CoA (FA-CoA) and co-located CoA (CCoA). FA-CoA is usually an IP address of the FA. If an MN registers an FA-CoA with the MN's HA, the FA is responsible for intercepting all tunneled packets destined for the MN and delivering the de-tunneled packets directly to the MN. If the MN uses a CCoA, which is an IP address belonging to the visited network, the MN itself will receive and handle all tunneled packets.

When an MN detects that the current serving FA (cFA) is no longer accessible, it will

initiate a layer-3 handoff from cFA to the next FA (nFA). A layer-3 handoff consists of three phases — agent discovery, address configuration and registration phases. If FA-CoA is in use, the MN must discover nFA first and then register with the MN’s HA through nFA (Figure 14); otherwise, the MN must acquire a CCoA through some external means, such as DHCP, before it can start a registration process.

1. Agent Discovery: This concerns how an MN is made aware of the presence of nFA. Every MA can be uniquely identified by its *AgentAdvertisement* message. An MN may either listen to *AgentAdvertisement* messages broadcasted periodically by nFA or actively issue an *AgentSolicitation* message to request an advertisement.
2. Address Configuration: This concerns how an MN obtains its new CCoA, which is usually achieved by way of DHCP.
3. Registration: This informs the HA of an MN’s CoA. If an FA-CoA is in use, the MN issues a *RegistrationRequest* message to nFA, where the message is then forwarded to the HA. If a CCoA is used, the MN sends this message directly to the HA. The HA replies a *RegistrationReply* to the MN to confirm the registration. The *RegistrationRequest* will be relayed by nFA if an FA-CoA is in use.



Figure 14. Mobile IP handoff in the FA-CoA case.

The process for an MN to detect that cFA is no longer accessible is referred to as move detection. MIP specifies two move detection principles: the expiration of the advertisement lifetime and the change of the network prefix. In MIP, each *AgentAdvertisement* carries an advertisement lifetime. If the lifetime of the most recently received advertisement expires, the MN may assume that cFA is unreachable. This approach usually leads to long move detection delay, as MIP suggests that the advertisement lifetime should be long enough to tolerate three consecutive losses of advertisements. Alternatively, if the MN receives an *AgentAdvertisement* with a network prefix different from that of the MN’s current CoA, the

MN may deduce that cFA is unreachable. This also causes long move detection delay as the MN can receive nFA's advertisement only after a layer-2 handoff.

### 2.3.3 Analysis handoff delays

Each of the aforementioned delays contributes to the overall handoff. Many researches have aimed to solve those delays. Their findings and results are briefed below.

1. AP probe delay: Mishra et al. [31] have pointed out that the probe phase delay largely contributes to the layer-2 handoff latency. They suggested the use of neighbor graphs [37] to capture handoff-to relationship between APs; the MN only needs to probe the APs that are neighbors of the current AP. An AP is a neighbor of another AP only if a handoff from the latter to the former has occurred recently. Neighbor graphs thus capture only temporal handoff-to relationship.
2. Association Delay: Neighbor graph can also be used to reduce the re-association delay by caching security information before the handoff begins, where security information is needed to establish secure communication channels between APs [38].
3. 802.1x Authentication Delay: Neighbor graph was also utilized to decrease IEEE 802.1x authentication delay between an MN and an authentication server by pre-distributing key material to the candidate set of APs which the MN may potentially re-associate with [39].
4. Move Detection Delay: A cross-layer design for shortening move detection delay naturally leads to the notion of layer-2 (L2) triggers. A L2 trigger is a layer-2 signal that informs layer-3 entity of particular events before or after a layer-2 handoff [45]. Based on the timing of occurrence, two types of layer-2 triggers can be defined: *pre-handoff trigger* and *post-handoff trigger*.

A pre-handoff trigger happens before a layer-2 handoff, while a post-handoff trigger indicates the completion of a layer-2 handoff. In IEEE 802.11, a pre-handoff trigger may be conditioned on the execution of probe phase, which only takes place when an MN detects poor link performance. A candidate post-handoff trigger can be a "link up" event that occurs in an AP or MN after an MN successfully completes the re-association phase.

Wu et al. [48] utilized the post-handoff trigger in an MN as a realization of move

detection. The move detection delay is therefore eliminated.

5. Agent Discovery Delay: Wu et al. [48] also proposed the use of neighbor lists to cut off agent discovery delay. Entries of a neighbor list store IP addresses of MAs associated with neighbor APs, one for each neighbor AP. On the occurrence of a post-handoff trigger, an MN looks up its neighbor list for nFA's IP address, to which it then directly issues a registration request.<sup>1</sup>
6. Registration Delay: Handoff latency can be further improved with pre-handoff triggers instead of post-handoff triggers, as an MN could commence a layer-3 handoff even *before* the layer-2 handoff is completed. An example is the notion of pre-registration or early registration in Malki's low-latency handoff proposal for Mobile IP [40]<sup>2</sup>. While this proposal allows the use of both types of L2 triggers, pre-handoff trigger is more appropriate to pre-registration.

An MA in Malki's proposal [40] is required to acquire the advertisements of neighbor MAs prior to MN's handoffs. When a pre-handoff trigger occurs in an MN, the MN asks for nFA's advertisement by sending a *ProxyRouterSolicitation* to cFA (note that the MN does not yet have a direct link with nFA). cFA returns a *ProxyRouterAdvertisement*, i.e., nFA's Agent Advertisement. The MN then can initiate a pre-registration by sending a *RegistrationRequest* through cFA to nFA. With this pre-registration scheme, layer-3 handoff parallels layer-2 handoff and overall handoff latency is greatly reduced. Preliminary analytical results show that the pre-registration outperforms conventional MIP with route optimization [41].

### 2.3.4 Cross-Layer topology information

Previous low-latency layer-2 handoff schemes [37-39] did not utilize much topology information: usually only AP's handoff-to relationship is of interest. However, to facilitate higher-layer handoffs, the definition of topology information should be extended to incorporate cross-layer information such as the association between APs and higher-layer entities.

---

<sup>1</sup> In fact, IP address alone is not sufficient for all types of registrations. Therefore, the contents of neighbor list should be extended to include all relevant information that ought to be retrieved from Agent Advertisement's.

<sup>2</sup> This proposal addresses both pre-registration and post-registration techniques to achieve low-latency handoff.

In the pre-registration scheme mentioned above, an MN must learn of nFA's IP address before pre-registration. This implies that the following information must be available to the MN:

1. AP topology, which provides not only handoff-to relationship among APs (which neighbor graph provides) but also the physical locations of APs (could be local or global coordinates.) If AP topology information is implemented in a distributed fashion [38], the MN can acquire it from the current AP. Alternatively, the MN may request it from some designated location server [42].
2. The location and the moving direction of the MN, which is for an accurate estimate of the next AP. An MN learns of its current location and moving direction somehow, e.g. using GPS (Global Positioning System) or any indoor location technique [43].
3. Association between APs and MAs (cross-layer information for estimating the next FA to which the next AP belongs). The AP/MA association should be configured and maintained at network side, since such information is network dependent. An MN can acquire associations in a way similar to how it acquires AP topology information. It is also possible to combine AP topology information and AP/MA association. The neighbor list [48] mentioned above is an example.

Application-layer handoffs also benefit from such a topology-based cross-layer design. Specifically, we may maintain the association between APs and SIP proxy servers or AAA servers. The association information rarely changes, as network topology is nearly static, and can therefore be gathered off line. With that information, we can do more than simple pre-caching and pre-registration; pre-authentication and pre-reinvitation are also applicable.

As an example, Kwon et al. [48] discussed the use of Diameter protocol for authenticating MNs during MIP registration. They proposed Shadow Registration that can be applied to both MIP and SIP to reduce the time to process inter-domain handoff. The key idea is to establish the security association between an MN and authenticators (APs) and between the MN and foreign AAA servers in neighbor domains prior to handoffs. However, little has been addressed on how to determine the set of candidate authenticators and the associated AAA servers. Our idea applies here: with the association information, an MN or a network-side

---

But we focus on the pre-registration case here.

server can be made aware of the set of candidate authenticators and the associated AAA servers.

Previous studies mainly focus on reducing handoff delays on a specific layer. For a link-layer handoff, Mishra [31] indicates that among the delay components of a link-layer handoff, the association delay ( $t_{Assoc}$ ), and open system authentication delay ( $t_{Auth}$ ), require 10ms to 20ms each and they are about 20% of the delay of a link-layer handoff ( $t_{LK}$ ). The scan time ( $t_{Probe}$ ), contributes 80% of the  $t_{LK}$ , which varies from 100ms to 350ms. In order to shorten the scan time, Shin [44] proposes the concept of the neighbor graph to minimize the number of scanned channels from 11 to 2.2 on average, and shortens the scan time to 32ms.

For a network-layer handoff, Wu [45] investigates the detection time of network change ( $t_{Detect}$ ), and finds that it varies from 500ms to a few seconds depending on the expire timer of the agent advertisement renew (AAR) in mobile IP [46]. Dutta et al. [47] proposes the HMMP (Host Mobility Management Protocol) that installs an agent, called SIP\_EYE, between the application layer and the network layer of an SIP UA to examine headers of TCP packets. The SIP\_EYE agent maintains a connection table before and after a handoff so that the connection can be tracked. By looking up the table, the agent can identify their end-points and redirects the ongoing packets to the UA's new location without any modification on the TCP stack. Kwon [48] and Turányi [49] both note that the IP acquisition and assignment using DHCP causes considerable delays for SIP and mobile IP applications. Typically, DHCP takes about 4 to 5 seconds to complete.

Handoff latencies also introduce packet loss. Yokota [50] proposes a link-layer solution using a MAC bridge to forward packets from old AP areas to new AP areas. Therefore, packet loss problem can be eased during network handoffs. However, this approach requires MAC bridges to connect to APs that an MN might attach. Similar to Yokota's approach, Shim [51] tried a network-layer solution that an MN can receive packets that are forwarded by the new APs as soon as the completion of a link-layer handoff. By applying the above solutions, the number of lost packets can be reduced.

Alternative solutions take the advantage of the SCTP (Stream Control Transmission Protocol, [52]) to reduce handoff delays. Riegel [53] simplifies the mobility management by skipping the location update based on the SCTP. In their method, though an MN needs not to perform location update to CNs after handoffs immediately, the CN needs more time to detect that the MN's primary IP is not available, and starts to try the MN's secondary IP. Jonathan [54] presents a profile-based mobility management that introduces the concept of pre-registration and resources pre-allocation. Also, they use the MN's moving history to predict MN's next possible locations and to speed up the handoff process. Madiseti [55] proposes a similar concept to perform pre-registration and IP pre-allocation for an MN to all possible roaming areas when the MN starts up. However, this approach might occupy too much network resources if the MN does not move.

## **2.4 User mobility database**

To set up a call in time to a mobile user in a cellular network, such as GSM (Global System for Mobile Communications) and UMTS (Universal Mobile Telecommunications System), it is necessary to constantly keep track of the mobile user's location. In GSM and UMTS, user location records are stored in a two-level database that consists of HLR (Home Location Register) and VLR (Visitor Location Register) [56]. The HLR resides in the user's home network and maintains mobile users' profile information and the current visited VLRs. For each visiting user in the location areas managed by a VLR, the VLR stores the user's subscription information and current location. When a mobile user crosses a location area, the user needs to register to the VLR and/or the HLR. Thus, the mobility databases, HLRs and VLRs, are frequently modified for location tracking and queried for call delivery. If the mobility database fails, calls to mobile users cannot be set up in time because of invalid location records.

### **2.4.1 Re-registration**

Many mobility database restoration schemes have been studied. ETSI (European Telecommunications Standards Institute) recommends periodically autonomous registration



[57] where a mobile user is required to register its location with the mobility database periodically even if the user does not cross a location area. Therefore, after a location database fails, an invalid location record can be restored sooner by the autonomous registration, and the number of calls lost due to invalid location records is reduced. Haas and Lin [58] considered the tradeoff between the cost of autonomous registration and the penalty of lost calls due to invalid location records. They suggested that the autonomous registration interval should be chosen to be approximately equal to the call inter-arrival time. Fang, et al. [59] considered the same cost function, and their study concluded that the optimal choice of autonomous registration interval might not be unique. They also showed that the optimal value could be found under certain traffic conditions. In addition, Fang, et al. [60] showed that the optimal autonomous registration interval depended on the weighting ratio between the registration signaling cost and the lost-call cost. To further reduce the time to restore invalid location records, Haas and Lin [61] proposed a demand re-registration scheme where mobile users are requested to re-register after the database fails. This scheme reduces the time to restore the location database. However, since user registration requires radio contact, this demanded re-registration from a large number of mobile users may cause repeated channel collisions, and thus waste wireless resources. Lin and Lin [62] studied a similar problem, the registration interval of badge-based location tracking system. Their results indicated that the channel collisions could be reduced by using exponential registration intervals without increasing the probability of losing calls due to invalid location records.

#### 2.4.2 Checkpointing and restoration

Another way to solve the call loss problem is to enhance the reliability of the mobility database. Checkpointing and rollback-recovery has long been used to reduce the expected execution time of long-running computation and to enhance the reliability of a database in the presence of failures [63-66]. For UMTS, the mobility database is periodically checkpointed to a non-volatile storage [67]. After a mobility database failure, the user location information can be restored from the non-volatile storage. Checkpointing mobility database is more cost-effective than autonomous registration, because accessing a local non-volatile storage is



in general cheaper and faster than accessing a radio channel. If a user's location record is not checkpointed every time when it is updated, the restored record may be out-of-date. In this case, to set up a call, the network can page the user at the location areas around the out-of-date location. Lin [68] derived the optimal checkpointing interval to balance the checkpointing cost against the paging cost, and showed that a user record need not be checkpointed if the checkpointing frequency is higher than 10 times or lower than 0.1 times of the user's moving rate. Wang, et al. [69] proposed an aperiodic checkpointing scheme where checkpointing of location database is not performed periodically but is triggered by a threshold on the number of unchecked location records. They also showed that aperiodic checkpointing outperforms periodic checkpointing when the threshold value is not large. Lin [70] proposed a per-user checkpointing algorithm where a user record is checkpointed only if the user record is modified when the checkpointing timer for the user expires. Otherwise, checkpointing is performed when the user register for the next time. Since mobile users exhibit different characteristics in terms of registration and calling behavior, per-user checkpointing schemes can serve each user better than a whole-system scheme, but the system has to maintain a checkpointing timer for each user. This timer maintaining job seems to be a large overhead to the system, but the hashed and hierarchical timing wheels, designed by Varghese and Lauck [71], take constant ( $O(1)$ ) time to maintain  $n$  outstanding timers, i.e., the time complexity is independent of the number of timers.

In summary, per-user checkpointing schemes can serve each user best without much overhead. However, no analysis has been done on the choice of the checkpointing intervals for per-user checkpointing scheme. In this paper, we study three per-user checkpointing schemes and consider a cost function consisting of the checkpointing cost and the paging cost. Numerical analysis was used to derive the optimal checkpointing frequency when user registration interval is exponentially distributed. In addition, computer simulation was used to study a more general case where user registration interval has a gamma distribution.

## Chapter 3 Integrated call agent

Since no work has been done on interoperating all VoIP protocols in a simple and flexible framework. In this chapter we present an integrated call agent architecture that supports the interworking function of VoIP protocols (SIP, H.323, MGCP and MEGACO) using the basic call state model in Intelligent Network. The interworking function translates messages of the VOIP protocols. The translation is transparent to the call parties and kept as simple as possible.

### 3.1 Integrated call agent architecture

We proposed a simple systematic way to implement the interworking function (IWF) between VoIP protocols. There are two major functions in the IWF: call routing function and call control function. The call routing function locates the called party by querying a location database so that the call request can be delivered. How to locate a user is beyond the scope of this paper; we assume that a user location database exists. The call control function sets up and maintains the call by translating messages between two VoIP protocols.

#### 3.1.1 Interworking gateway

The call control function can be implemented using the BCSMs. For each VoIP protocols, its messages are mapped to the messages of the BCSMs. An interworking gateway of two VoIP protocols can be implemented by combining the BCSMs of the VoIP protocols. Therefore, an O\_BCSM of one VoIP protocol and a T\_BCSM of another VoIP protocol can interact through the exchange of a set of unified events based on the IN half-call model. Figure 15 shows the six BCSMs of three major VoIP protocols for a general gateway design. Note that the events between an O\_BCSM and a T\_BCSM are protocol independent. This design reduces developing efforts. To interwork  $n$  different VoIP protocols, each protocol needs only one O\_BCSM and one T\_BCSM (i.e.  $2n$  BCSMs in total), instead of hardcoding

$n^2$  messages translation modules between any two protocols.

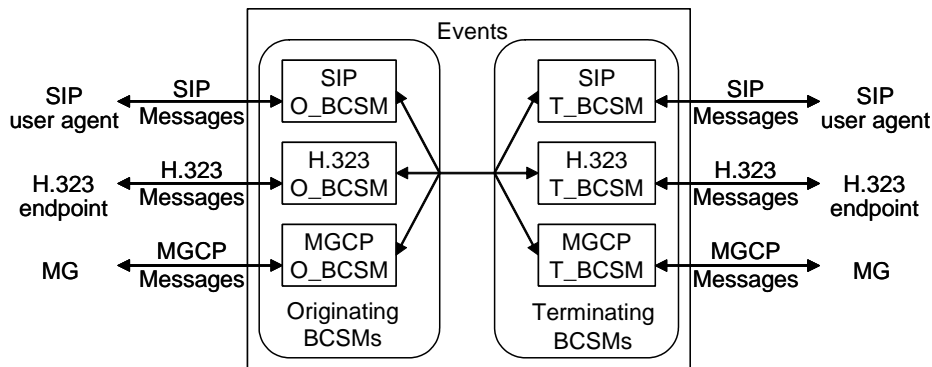


Figure 15. Components developed for a general VoIP gateway.

A combination of the BCSMs of two VoIP protocols becomes a gateway; the construction of a gateway is to design the BCSMs of the VoIP protocols. Figure 16 shows an example SIP/H.323 gateway. This gateway consists of the BCSMs of SIP and H.323. When a SIP UA originates a call to a H.323 terminal, a SIP O\_BCSM receives a SIP *INVITE* message and issues a *Setup* event to an H.323 T\_BCSM. After being notified by the *Setup* event, the H.323 T\_BCSM sends an H.323 *SETUP* message to the terminating H.323 endpoint. Subsequent SIP and H.323 messages are exchanged to set up the call through the interaction of the SIP O\_BCSM and H.323 T\_BCSM. On the other hand, an originating call from a H.323 terminal to a SIP UA can be handled by an H.323 O\_BCSM and a SIP T\_BCSM.

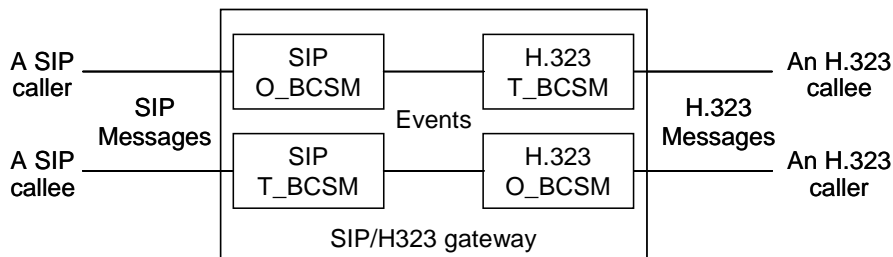


Figure 16. A SIP/H.323 gateway.

All the gateways needed in the converged VoIP network can be implemented in the same way described above. Figure 17 depicts a converged network using three gateways: SIP/H.323, SIP/MGCP, and MGCP/H.323 gateways. Note that the gateways can share the same BCSMs developed. Moreover, when a new VoIP protocol, such as MEGACO, is added to the converged network, the gateways to MEGACO network use the same BCSMs developed for existing VoIP protocols; only the BCSMs of MEGACO need to be

implemented.

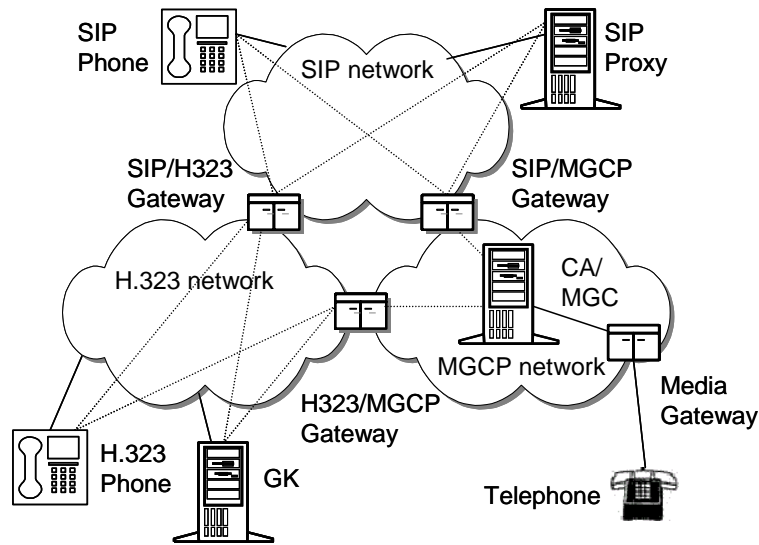


Figure 17. A converged VoIP network using gateways.

### 3.1.2 Integrated call agent

To route an originating call, the location of the callee and the signaling protocol that the callee supports must be determined so that a correct T\_BCSM can be invoked for messages translation. Solutions to this call routing function are beyond the scope of this paper; we assume that a solution is available. We integrate the call routing function and the BCSMs of all VoIP protocols in an entity called integrated call agent (ICA). The ICA can not only translate messages between VoIP protocols but also act as a H.323 gatekeeper, a SIP proxy/registrar, and a MGCP call agent.

The design is extensible. Figure 18 shows the converged architecture partitioned into two zones managed by two ICAs. Calls between entities of two zones can be set up by the ICAs exchanging SIP messages through two SIP BCSMs. Figure 19 depicts an example call between an H.323 terminal and an MGCP phone, and two ICAs (ICA 1 and ICA 2) are involved. When an H.323 terminal in Zone 1 initiates a call to an MGCP phone in Zone 2, ICA 1 invokes a SIP T\_BCSM and sends a SIP *INVITE* message to ICA 2. ICA 2 initializes a SIP O\_BCSM to handle this call request. Since ICA 2 knows that the callee is an MGCP phone, an MGCP T\_BCSM was invoked to set up this call connection.

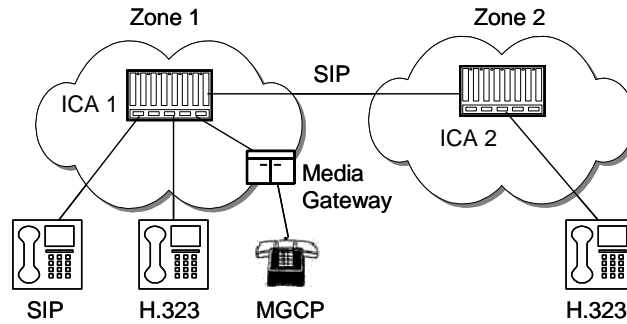


Figure 18. A converged VoIP network managed by integrated call agents.

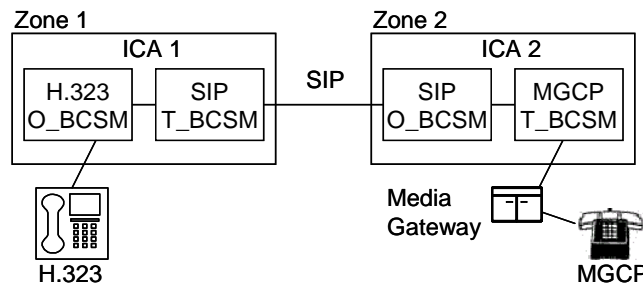


Figure 19. An example of H.323 and MGCP interworking using 2 ICAs.

### 3.2 Mapping of VoIP protocol messages to the BCSM messages

#### 3.2.1 Uniform events

The implementation of half-call BCSM for each VoIP protocol is to map the VoIP call signaling messages to the IN BCSM messages. We focus on the relationship between the BCSM states and the VoIP messages. This message mapping is straightforward except for the slow-start version of H.323. Figure 20 depicts the mapping for H.323, SIP, and MGCP.

When a call request message arrives (e.g., *INVITE* for SIP, *SETUP* for H.323, or *NTFY* for MGCP) from a calling party, the corresponding O\_BCSM becomes active and initialize itself to state *O\_Null*. The O\_BCSM proceeds to state *O\_Collect\_Info* where a called number or address is collected from the calling party. For SIP and H.323, the *INVITE* and *SETUP* messages carry the identifier of the called party, media descriptor, and signaling transport address. Hence the O\_BCSM simply skips state *O\_Collect\_Info* and go to state *O\_Routing* directly. For MGCP, however, a *RQNT* message should be sent to the calling party to collect the dialed number. Once the calling party collects the number dialed, it sends a *NTFY* with the dialed number to the O\_BCSM. Then, the O\_BCSM transits to state *O\_Routing*.

State *O\_Routing* responds to the calling party with the call progressing message (e.g., *100 Trying* for SIP or *CALLPROC* for H.323). In addition, the VoIP protocol used by the called party is determined, and a *Setup* event is issued. Consequently, a T\_BCSM for the called party is initialized at state *T\_Null* to handle this *Setup* event. The O\_BCSM stays in state *O\_Routing* waiting for the *Alert* and *Answer* events sent from the T\_BCSM. If both events are received, the call has been accepted by the called party and the O\_BCSM responds with a call connect message (such as *CONNECT* for H.323, *OK* for SIP, or *MDCX* for MGCP) to the calling party indicating that the call has been accepted. The O\_BCSM stays at state *O\_Active* until the call is terminated.

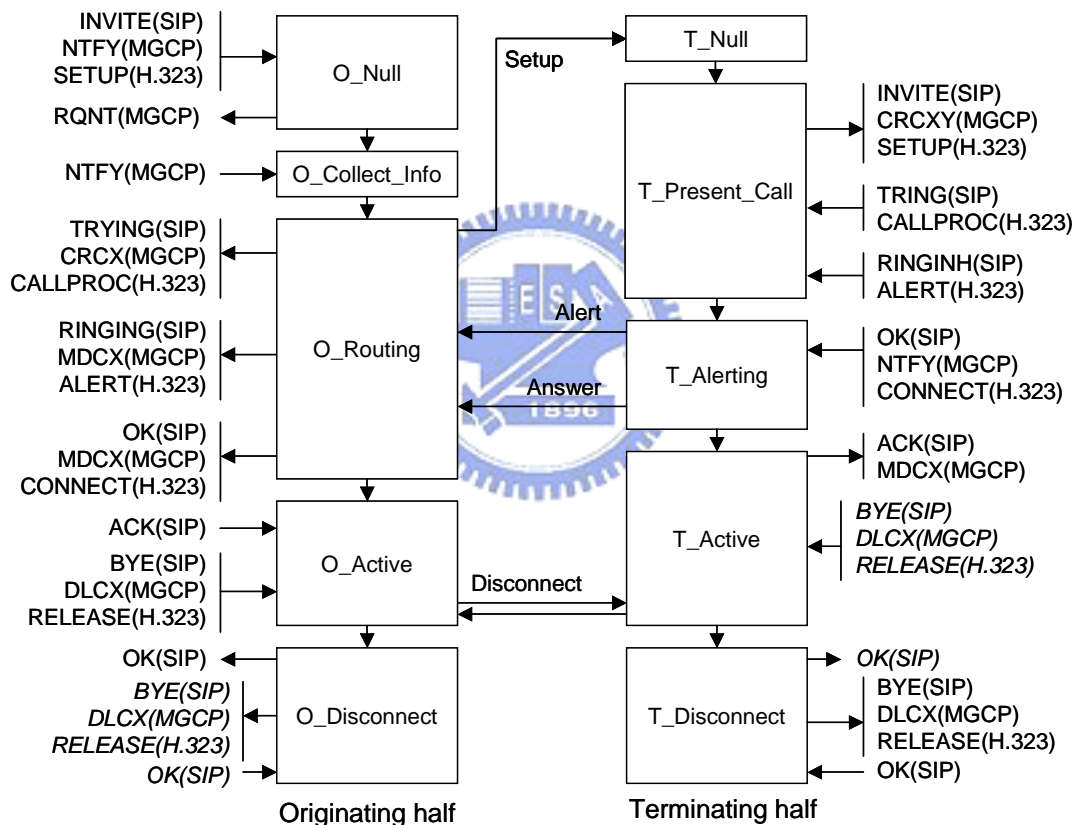


Figure 20. Mapping VoIP messages to BCSM messages.

When a T\_BCSM is activated by a *Setup* event from an O\_BCSM, it goes to state *T\_Present\_Call* and checks the validation of the called number. If the number is valid, the T\_BCSM sends a call request message (e.g., *INVITE* for SIP, *SETUP* for H.323, or *CRCX* for MGCP) to the called party and waits for the response messages such as trying, altering, and answered from the called party. Upon receiving the alerting message (e.g., *180 Ringing* for

SIP or *ALERT* for H.323), the T\_BCSM informs the O\_BCSM with an *Alert* event and proceeds to state *T\_Alert*.

In state *T\_Alert*, the T\_BCSM waits for the call connect message (e.g., *OK* for SIP, *CONNECT* for H.323, or *NTFY* for MGCP). Once the message is received, the call has been accepted by the called party. The T\_BCSM activates an *Answer* event to the O\_BCSM and transits to state *T\_Active*. In this state, an acknowledgement message (e.g. *ACK* for SIP or *MDCX* for MGCP) is sent back to the called party. Thus far, a basic two party call connection is established.

If a disconnect message (e.g., *BYE* for SIP, *RELEASE* for H.323, or *DLCX* for MGCP) is received by either the originating or the terminating BCSM, the BCSM will issue a *Disconnect* event to the corresponding BCSM of this call. Both BCSMs proceeds to state *Disconnect*, and the call is terminated.

### 3.2.2 H.323 slow-start

For the H.323 slow-start, the mapping is quite different and those events described above are inadequate to exchange the media capability. For the BCSMs described above, after the media capability of the calling party becomes available at state *O\_Null*, the O\_BCSM issues a *Setup* event. Moreover, after the capability of the called party becomes available at state *T\_Present\_Call*, the T\_BCSM issues an *Answer* event respectively. However, for H.323 slow-start, this capabilities will not be determined until the H.245 negotiation after the *CONNECT* messages. As a result, the BCSMs do not get media capability when the *Setup* or *Answer* event is issued. In addition, there is no event reporting the media capability to the calling and called party after the *Answer* event. Thus, we made two modifications to the H.323 BCSMs. First, the *Answer* event is postponed to state *T\_Alert* where the T\_BCSM receives an H.245 open logical channel message (*OLC*) which carries the media capability for the called party. Second, a new event, media capability ready (*MediaReady*), is introduced at the end of state *O\_Routing* to notify the T\_BCSM the media capability of the calling party. Figure 21 shows the modifications to the mapping for the H.323 slow-start.



To interwork with the H.323 BCSMs that supports slow-start feature, the BCSMs of SIP and MGCP were modified to send a *MediaReady* event anyway at the end of state *O\_Routing*. i.e., all O\_BCSMs should issue a *MediaReady* event at state *O\_Routing*. Although the called party of SIP or MGCP expects to receive the media capability when it receives the *INVITE* or the *CRCX* message, this capability exchange can be delayed till when the *ACK* or the *MDCX* message is received. Figure 22 shows the call flow of H323 and SIP interworking where the H.323 terminal is in slow-start mode. The call flow of a slow-start H323 terminal and an MGCP phone can be done in a similar manner. If the calling party is an H.323 phone using the slow-start feature, the T\_BCSM issues an *INVITE* message without specifying the caller's media capability to the SIP callee. The media capability is available at state *O\_Routing* after the H.245 *OLC* message is received. Consequently, the O\_BCSM issues a *MediaReady* event to the T\_BCSM at the state *O\_Routing*. Therefore, the T\_BCSM encloses this capability in the *ACK* message sent to the called party.

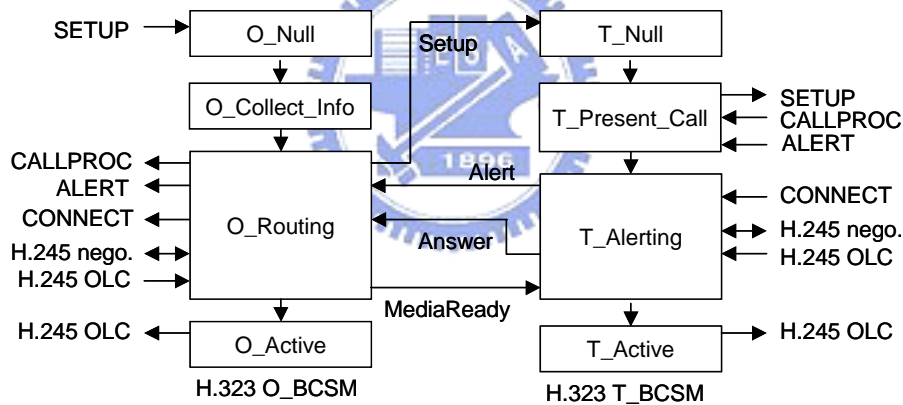


Figure 21. BCSMs for the H.323 slow-start.



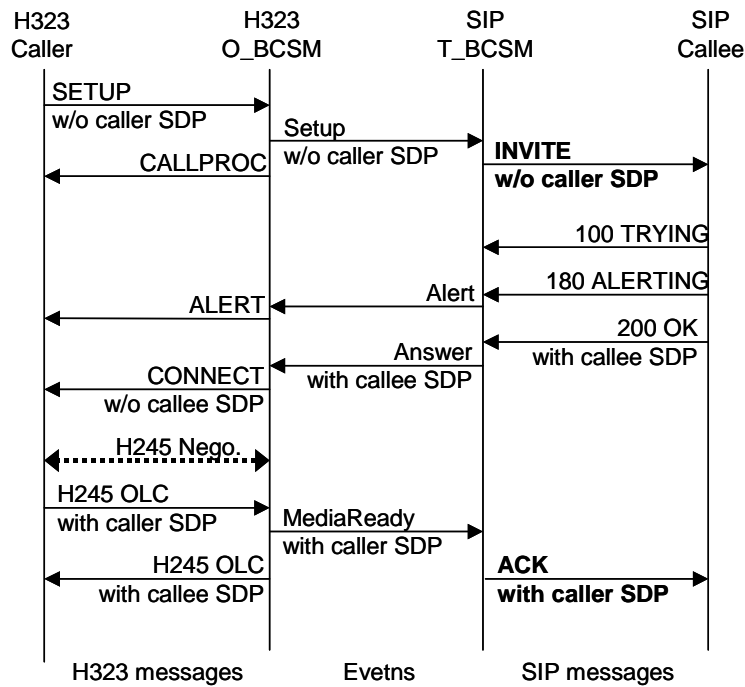


Figure 22. Call flow of H.323 (slow-start) and SIP interworking.

### 3.3 Implementation and result

To reduce the effort in developing a VoIP gateway, we use existing, well-developed protocol stacks and endpoints. In our implementation, we use the MGCP and SIP protocol stacks developed by CCL/ITRI, Taiwan and the open-source H.323 protocol stack developed by OpenH323. In addition, our experiment platform includes two residential gateways (RGWs) which are also developed by ITRI using D/41E and D/41ESC cards from Dialogic Corp. The RGWs support MGCP and can connect up to 16 telephones. The ICA platform also supports Microsoft NetMeeting (using H.323) and SIP user agent. The H.323 BCSMs are modified from the source code of the OpenH323's OpenGate that supports registration administration status (RAS) messages and gatekeeper-routed call signaling. We have also developed a SIP proxy/registrar based on the ITRI SIP protocol stack. Figure 23 summarizes the components used in our platform.

In our experiments, a call can be successfully set up between any two VoIP phones. The Microsoft NetMeeting currently supports only H.323 slow-start version; we use an open-source OpenPhone (with both slow-start and fast-start capabilities) to test the cases of

slow-start version. In addition, a Vocal sip proxy, developed by Vovida, was used to test calls between SIP UAs. Since an ICA acts as both SIP proxy and H.323 gatekeeper, an ICA can initiate a call to the phones that are under the control of a SIP proxy or an H.323 gatekeeper without further modification.

The comparison of the delays in establishing a call between various types of phones by OpenGate H.323 gateway, our ICA, and Vocal SIP proxy is depicted at Figure 24. No result of inter-protocol calls through OpenGate and Vocal is listed, because they do not convert the messages of different protocols. Figure 24.a shows call establishment delays for the calls initiated from various types of phones to NetMeeting, which only equipped with slow-start mode, and Figure 24.b shows those for calls to OpenPhone in H.323 fast-start Mode. Although our ICA supports signaling conversion for different VoIP protocols, the results indicates that the ICA sets up calls faster than the OpenGate in all cases except for calls between two NetMeeting users.

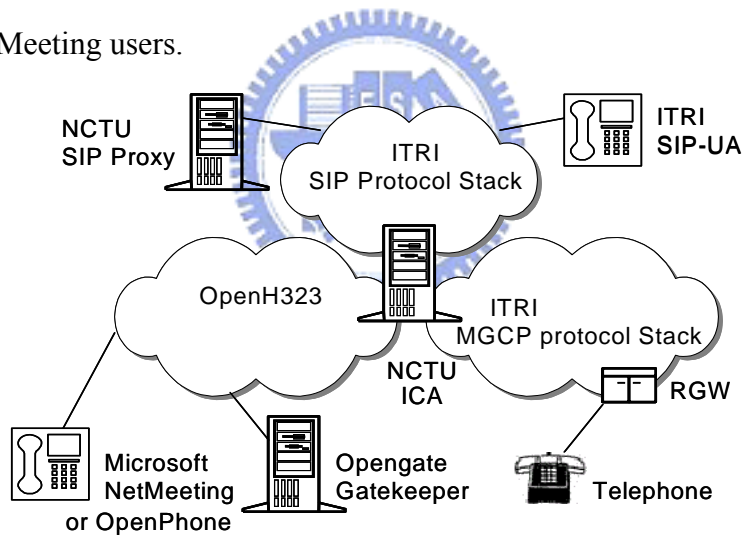


Figure 23. Components used in our platform.

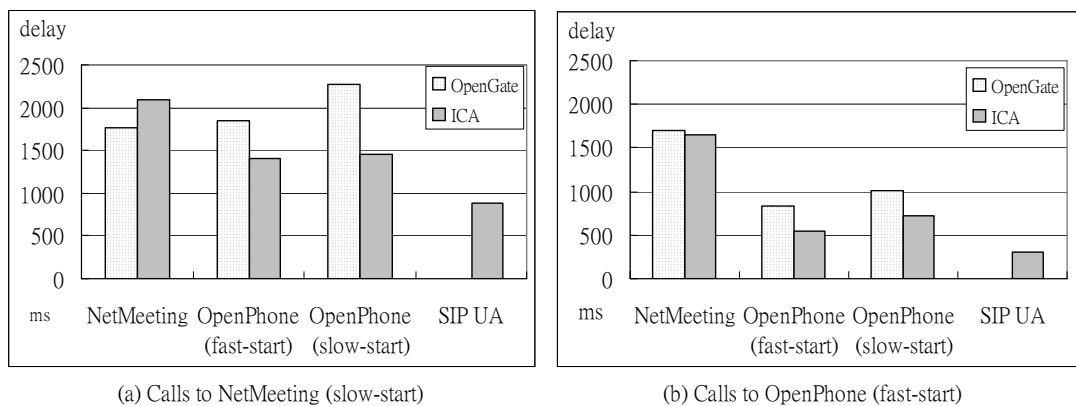


Figure 24. Call establishment delays.

## Chapter 4 VoIP's NAT traversal

Due to the un-routable signaling and digitized voice packets inward into a NAT, two major issues of the VoIP traversal over NAT are how to locate an internal user and how to deliver digitized voice packets into a NAT. We present a NAT traversal solution that works for all different NAT types without any modification to the NAT, user agent software.

### 4.1 NAT traversal architecture

In our architecture as depicted in Figure 25, there are three major differences to other solutions. First, we move the public SIP proxy (usually indispensable to most solutions) into the internal side of a NAT and confine modification to this internal SIP proxy (IPX) only; no other entities need to be modified. This IPX participates in locating an internal user, so that public SIP proxies are unchanged and can be optional. Second, the function of the ALG is extracted from a NAT and integrated into the IPX. An IPX is acted as an SIP proxy to internal users, hence every SIP message will pass through the IPX and examined by it. Third, the task of hole punching on a NAT is transferred to the IPX, leaving the user agent software intact.

For symmetric NATs, our design adapts a TURN-like solution that uses a relay agent (RA) outside a NAT to relay packets through a symmetric NAT. However, this RA is further responsible for NAT type determination and NAT hole punching.

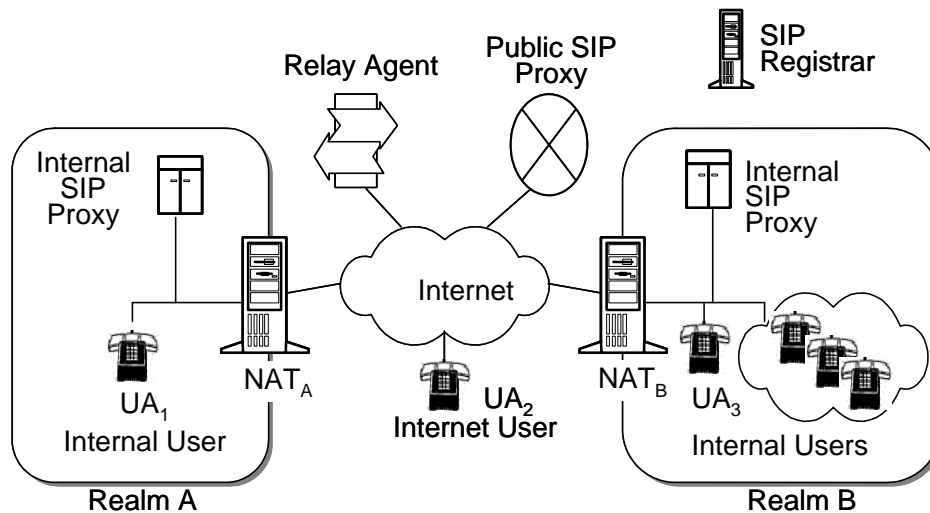


Figure 25. The NAT traversal for VoIP environment.

#### 4.2 NAT type decision

Different NAT types leads to different traversal solutions. According to the NAT specification [26] the full cone, restricted cone, and port restricted NAT possess different constraints on restricting inbound packets but use the same policy for selecting a port, i.e. they always binds an identical external port to all the connections that are initiated from the same port of an internal host. Since the hole punching operation makes translation on the port number, they can be treated as one category (non-symmetric NAT) in our design. Therefore, we can identify a non-symmetric NAT by issuing two connections from the same port of an internal host (IPX) to different ports of an external host (RA) and checking the port bindings. If the bindings are the same, it is a non-symmetric type; otherwise, a symmetric NAT.

#### 4.3 Locate an internal user

In a NAT environment, to receive an inbound call request, an internal user should have a deputy public address, so that it becomes addressable. Our solution requires internal users to treat the IPX in their NAT realm as their SIP proxy. This can be done by setting the user profile that is provided by almost every user agent programs.

### 4.3.1 Without a public SIP proxy

Figure 26 elucidates the user registration procedure without a public SIP proxy. We assume that SIP user agents and proxies are using well-known signaling port number (5060) and an IPX initially maintains a keep-alive connection with the RA. The RA allocates a relaying port (s) for an IPX that any packet destined to this port will be retransmitted through the keep-alive connection to the IPX. Therefore, inbound packets to the IPX can reach an RA first and then be relay to the IPX by the RA.

Since acting as a SIP proxy, an IPX can intercept the SIP register message from an internal user and know the user's signaling (contact) address (step 1). Then, the IPX keeps the user record in some database, replaces the public and contact addresses in the payload of the SIP *REGISTER* message with the RA's address (R:s) and sends it to a SIP Registrar (step 2). If anyone (Bob) wishes to make a call to an internal user (Alice), after asking the public address of Alice from the SIP registrar, Bob sends the SIP *INVITE* message directly to the RA at port s (step 4). The RA need not inspect the message and just relays it to the IPX using the keep-alive connection (step 5). The message is further dispatched to Alice by the IPX after looking up the real contact address of the Alice (step 6).

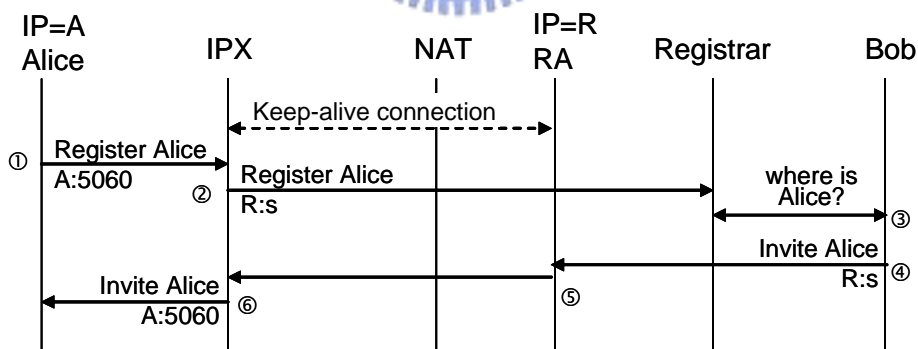


Figure 26. User registration without a public SIP proxy.

### 4.3.2 With a public SIP proxy

The above registration scenario is valid for all NAT types because all the inbound packets to NAT are relayed by an RA that has a connection created and kept alive by an IPX. However, an RA needn't relay signaling messages if there is an external (public) SIP proxy (EPX) and the users are behind a non-symmetric type NAT. In this configuration, all

signaling packets are proxied by the EPX so that the signaling port of the EPX acts as a fixed source address that feeds packets into NATs. Hence, the role of feeding packets by an RA can be superseded by the EPX.

Figure 27 depicts how internal users can be addressed without the help of an RA in such configuration. First, an IPX has to punch a hole (port) on a NAT and use this hole as the public address of the internal users. An empty packet sent from an IPX to an RA (step 1) is used to create a port on a NAT. The RA can learn the source address of this packet (i.e. N:y, the port binding on the NAT) and report it to the IPX (step 2). Hereafter, the IPX should keep this binding valid by sending an empty packet periodically from the same source address to any public destination<sup>3</sup> (step 3). If a user (Alice) is registered, the IPX will replace the public address field with the signaling address of the EPX and the contact address field with the hole (N:y) in the *REGIST* message and then send it to the EPX (step 4, 5). As mentioned above, the NAT still binds port y for the connection from IPX to the EPX and allows those packets back from the EPX. Thus, any incoming messages from the EPX to this hole will be relayed by the NAT directly to the IPX and be dispatched by the IPX to Alice without passing through the RA (step 9-11).

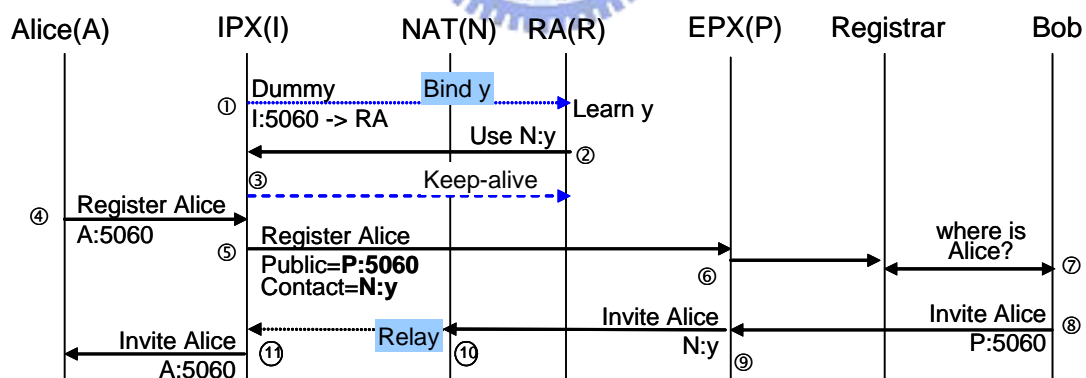


Figure 27. User registration using a public SIP proxy.

#### 4.4 Transport address translation

Similar to the case of locating an internal user, before RTP packets can penetrate into a NAT and reach an internal user, a port binding should be created first and RTP packets should

<sup>3</sup> Since the packet is sent from the same port number, the NAT will use the same port number no matter where

be destined to this port on the NAT. However, unlike signaling packets proxied by an IPX, the incoming RTP packets should be redirected to an internal user instead of an IPX to avoid one more store-and-forward. Figure 28 illustrates the case of a non-symmetric NAT environment. Having the RTP transport address (A:x) in the SDP of the SIP *INVITE* message coming from Alice, the IPX triggers first faked packet<sup>4</sup> from A:x to an RA, pretending this packet is issued from Alice. The bound address (N:z) on the NAT will be sensed and reported by an RA to the IPX and will be used as the external RTP transport address of Alice. When the SIP *OK* message is returned from Bob to the IPX, the IPX knows the exact destination of the outgoing RTP connection (B:p). Then, a second faked packet is issued by the IPX from A:x to B:p to exempt the NAT constraint. Since the port binding on the NAT is the same (z), the incoming RTP packets to the external transport address (N:z) on the NAT will be relay correctly to the internal user's RTP transport address (A:x) without passing through the RA and the IPX.

An IPX also takes the responsibility to keep all RTP connections alive if the silence depression feature is enabled by the user agent program. The IPX knows the birth and death of RTP connections as well as the port bindings they are using, so that the IPX can periodically issue faked empty messages masqueraded as if they are sent by the internal users.

---

goes the packet. If the packet destinate to an RA, the RA can just ignore it.

<sup>4</sup> A faked packet can be send by setting the network adaptor in a promiscuous mode and replacing the source IP address and source port to the pretended IP and port. The receiver thinks this packet is sent from the pretended address.

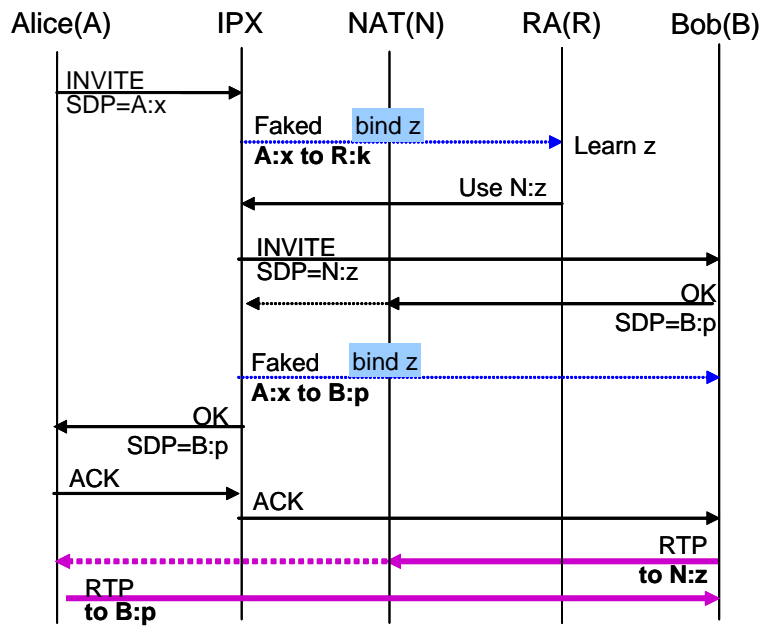


Figure 28. NAT hole punching.

As a symmetric NAT always selects different port bindings for different connections, the above scheme is not suitable because the second faked packet will cause another port binding. To solve this, the IPX follows a TURN-like solution using an RA to relay the RTP packets for internal users. The hole punching operation is performed by the IPX rather than by the user itself in the TURN solution. A faked packet is also sent by the IPX as if from an internal user to the RA to create a backward path into the internal user. The RA keeps a database that records the association between the allocated ports and the reverse paths, so that the incoming RTP path will go directly to the internal user rather than routing through the IPX. Though the symmetric type NAT traversal problem is solved, the RTP media is store-and-forward one more time and the performance will be deteriorated.

#### 4.5 Scenarios for call establishment

The following examples explain the message flows of connecting a call using our scheme in different environments. The IP addresses and port numbers are detailed to clarify the port translation and messages routing. We assume that the users are all registered to the same SIP proxy and the NAs have determined what type of the realm it belongs. We focus on three major issues: creating bindings, meeting NAT constraints, and routing, which are blocked in



those shadowed areas. The translation of the 'Contact', 'From', and 'To' fields are not detailed here since they have been explained in section 3.2.

Figure 29 is an example of call establishment between two internal users that locate within different non-symmetric NAT realms and use different RAs. One user, Alice at <Alice@P:5060>, wishes to make a call with another user, Bob at <Bob@P:5060>, issuing an *INVITE* with a receiving RTP transport address (U:p) to its thought SIP proxy, NA1. The NA1, knowing itself is in a non-symmetric NAT realm, performs the transport address translation by sending a faked *Probe* message from U:p to RA1's Learn1 port (R1:5001) and getting the port binding on the NAT1 (N1:q) in the *Report* message from the RA1. Then, the NA1 replaces the transport address with the N1:q (and other parameters such as the 'Contact' field) in the *INVITE* message. Since there is a SIP proxy involved, the signaling routing will go from the NA1 to the To-PROXY port on the RA1, and to the SIP proxy. Because Bob has a deputy address R2:5120 registered on the SIP proxy, the SIP proxy will forward the *INVITE* message to R2:5120, which is the RELAY-NA port on the RA2. Therefore, the RA2 redirects this *INVITE* message to the NA2 (A2:5060) through the NAT2 (N2:y). From the URI of the *INVITE* message, the NA2 searches its database for the user <Bob@P:5606> and knows he is located at V:5060. Consequently, Bob receives the modified *INVITE* from NA1 and knows Alice requests a call with an RTP receiving address at N1:q.

The following steps are similar to what NA1 have done, the NA2 translates the transport address and routes the modified *OK* message correctly to the NA1. In addition, the NA2 issue a faked *Probe* message from V:s to N1:q to relieve the constraint on the NAT2. After the NA1 receives the *OK* message with a modified RTP receiving transport address at N2:r, it also issues a faked *Probe* from U:p to N2:r to lift the ban on the NAT1 and sends the *OK* message to Alice. Eventually, the call is established. The RTP streams can successfully deliver to each other. In this case they do not go through the two relay servers.

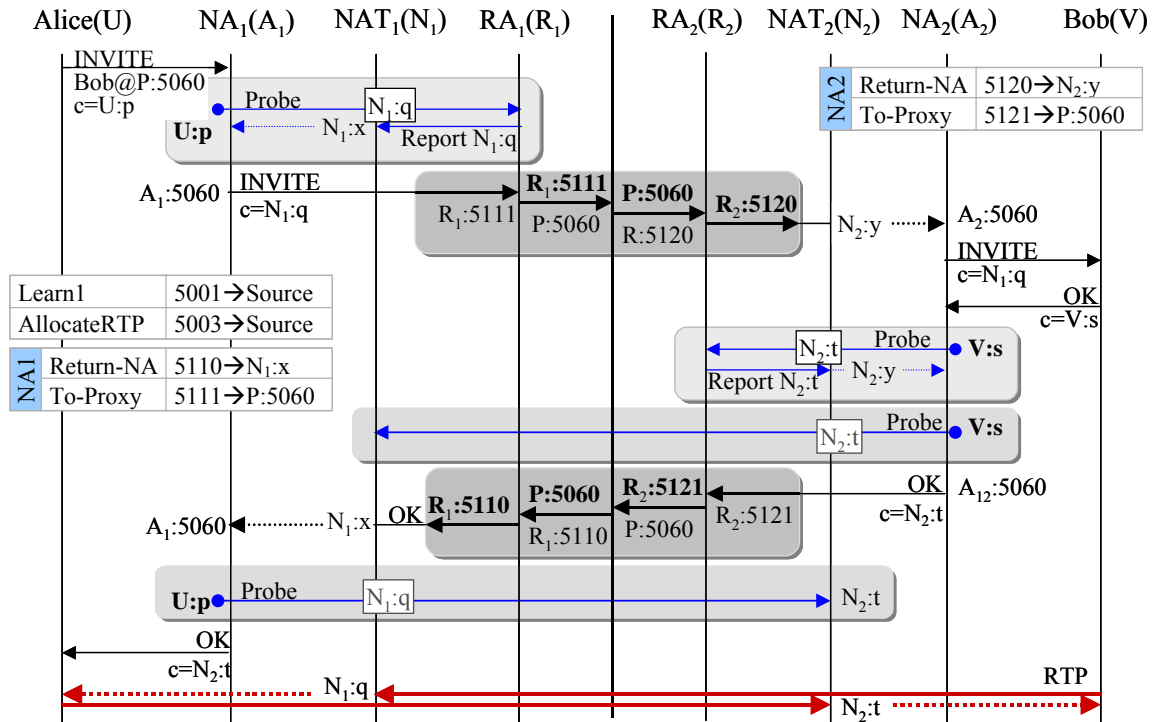


Figure 29. Call flow in a non-symmetric NAT environment.

Another example in Figure 30 describes the call flow in which both call parties resided in different symmetric NAT realms and shares the same RA. In this example, we only brief the transport address translation at the caller site and the message routing toward NA2. In the RTP transport address translation procedure, the NA1 first issues an *RtpRequest* to RA's *AllocateRTP* port and gets the allocated port number 6001 carried in the *RtpUse* message. Then, NA1 further sends a faked *Probe* from U:p to R:6001 in order to open a hole in the NAT2 for the inbound RTP packet (from Bob to Alice). The RA sets the source address of the faked *Probe* as port 6001's destination address (N1:q). The NA will modify Alice's transport address to R:6001. Because Alice subscribes to a SIP proxy, the NA1 transmits the modified *INVITE* message to the To-Proxy port (R:5111) on the RA, which forces the RA to relay this *INVITE* message to the SIP proxy (P:5060). Since Bob also registers itself by a contact address as <Bob@R:5120> at the SIP proxy, the SIP proxy will forward this *INVITE* to RA:5120, which forces the RA forwarding the *INVITE* message through the NAT2 to the NA2. Finally, the NA2 will cast the *INVITE* message to Bob. Once Bob accepts the call request, Bob issues an *OK* message with an RTP transport address V:s to the NA2. The NA2 also translates and replaces the RTP transport address as R:6002 and routes the *OK* messages

along the way to Alice. In this symmetric NAT case, the RTP packets are relayed by a RA, so one more hub count is introduced.

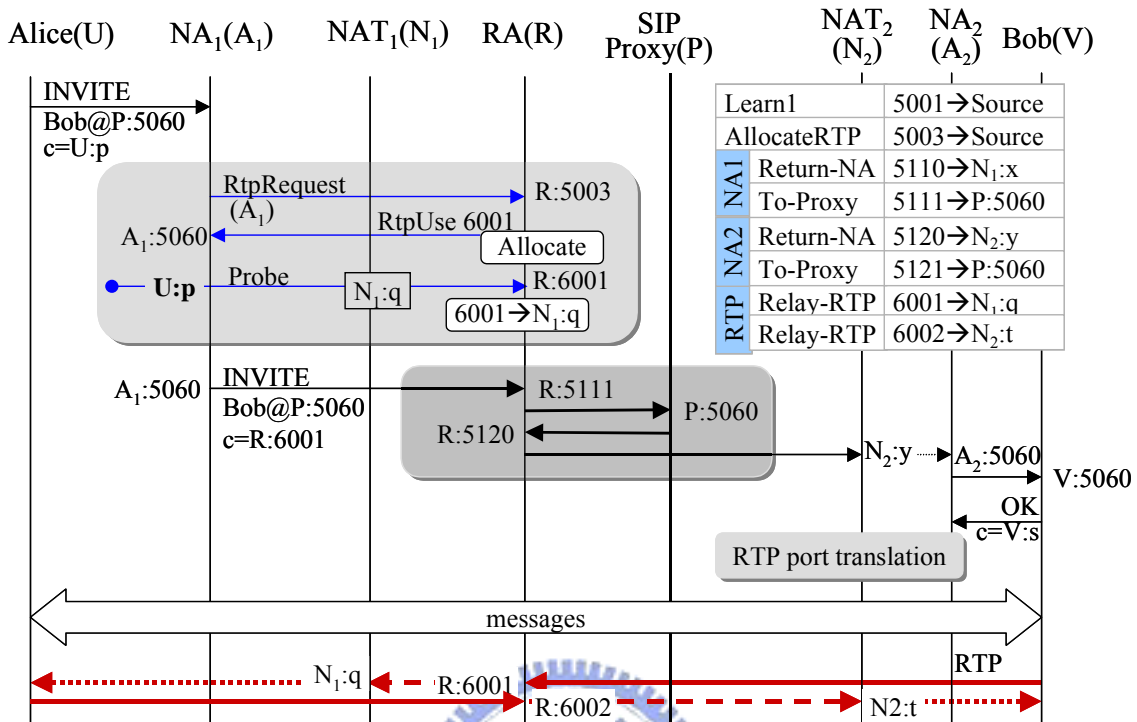


Figure 30. Call flow in a symmetric NAT environment.

The call parties in our example are all resident in NAT realms. However, when any one of the call parties is located directly on the Internet, there is no need to perform transport address translation and follow the SIP standard to establish a session.

## Chapter 5 Topology-aided fast handoff

Above solutions mainly address one specific layer but do not consider cross-layer effects and procedures. On the above discussion, the latency of a generic handover that might consist of the delays of link-layer, network-layer and application-layer handovers can be modeled as:  $t_{\text{Handover}} = t_{\text{LK}} + t_{\text{NW}} + t_{\text{App}}$ , where the delay of a link-layer handover consists of scan, authentication, and association delays ( $t_{\text{LK}} = t_{\text{Probe}} + t_{\text{Auth}} + t_{\text{Assoc}}$ ), the delay of a network-layer handover is composed of network change detection, IP acquisition and setup delays ( $t_{\text{NW}} = t_{\text{Detec}} + t_{\text{DHCP}} + t_{\text{Conf}}$ ), and finally the delay of an application-layer handover involves SIP registration, SIP update and packet delays ( $t_{\text{App}} = t_{\text{Reg}} + t_{\text{Update}} + \text{Max}\{t_{\text{Send}} + t_{\text{Recv}}\}$ ).

In this chapter, we aim to investigate each factor contributing handover delays, and to propose cross-layer solutions to reduce the handover latencies. To reduce the delay of a link-layer handover, we utilize a direct association design in the link layer to diminish the scan delay. Then, we develop a location-aware mobility protocol (LAMP) to reduce both network and application-layer delays. Our solution can cooperate with other handoff technology to further eliminate the SIP update delays and avoid the packet loss and further. The LAMP scheme can applies applications that based on mobile IP.

### 5.1 LAMP scheme

The concept behind the proposed LAMP scheme, is to pre-register to the neighbor service agents and to pre-allocate network resources such as IP addresses in the neighbor areas for an MN according to the AP's location information. This AP location information could be statically or dynamically configured and distributed to an MN while it moves to a new service area. The AP location information also contains pre-allocated IP addresses so that an MN can know the IP address in the new attached area. By knowing the allocated IP addresses in the AP areas in advanced where the MN might roam, the MN can directly configure the reserved IP address as soon as it detects the new AP. Hence, the IP address assignment delay that is

introduced by DHCP after a handover is vanished.

Figure 31 describes the network architecture to which the LAMP scheme and SIP are applied. An SIP domain is defined as a network that offers SIP services. It is further divided into several SIP sub-domains, each provided SIP service by an SIP proxy. For simplicity, we assume that a DHCP server and other resource management servers all co-locate with the SIP proxy. In other words, an SIP proxy in this paper also performs DHCP functions for IP assignment and is in charge of resource allocations such as bandwidth for MNs. For an SIP sub-domain that cover more than one sub-network, inter SIP sub-domain handovers will involve IP change and SIP updates. This inter SIP sub-domain handover is identical to the inter sub-network. Without loss of the generality, we assume the network configuration that one SIP proxy associates with one AP and controls one sub-network, so that an intra SIP sub-domain handover is an intra sub-network handover.

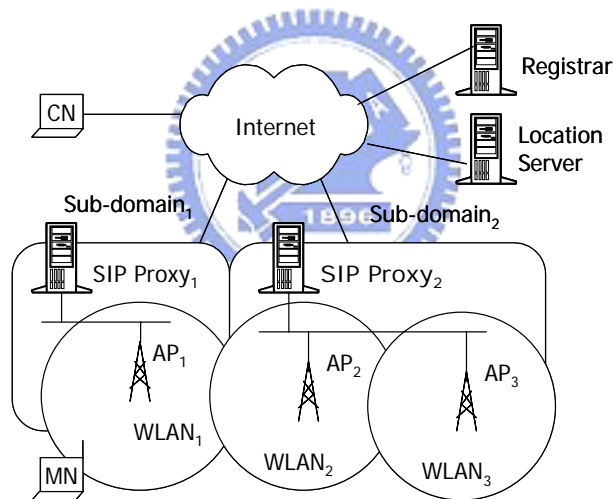


Figure 31. LAMP architecture.

Figure 31 also shows a location server (LS), which maintains the network deployment table. A network deployment table contains the AP location information, and provides an MN the network configurations in the neighbor sub-domains. This table maintains the neighbor relationships between APs and SIP proxies and the profile of each SIP proxy. An example of the network deployment information for the architecture in Figure 7 is shown in Table 1. This table can be stored on a dedicated location server (LS) or can be distributed to all SIP proxies. An MN thus uses the identifier of its current AP (such as BSSID, Basic Service Set Identifier)

as the key to query the LS or the current associated SIP proxy in order to know its neighbor SIP proxies and APs.

Table 1. An example of a network deployment table.

AP Identifier	Channel	Associated Proxy(IP)	Neighbor(s)
AP <sub>1</sub>	1	140.113.1.23	AP <sub>2</sub>
AP <sub>2</sub>	6	140.113.2.46	AP <sub>1</sub> , AP <sub>3</sub>
AP <sub>3</sub>	11	140.113.2.46	AP <sub>2</sub>

### 5.1.1 Registration and pre-allocation

The MN needs to complete the SIP registration procedure as the Step 1 to Step 4 in Figure 32 to access the SIP service. In addition, in order to enable the LAMP support the MN has to pre-allocate resources in the neighbor sub-domains by sending a pre-allocation request to the current SIP proxy. This request can be embedded into particular SIP messages such as the SIP *INFO* message in our implementation. After an MN finishes the registration process, the MN sends an SIP *INFO* message that carries pre-allocation request to the current SIP proxy to pre-allocate the resources (Step 5). The SIP proxy relays the *INFO* message to its neighbor sub-domains according the network deployment table (Step 6). After the current SIP proxy obtains the reserved IP addresses and resources in the neighbor sub-domains for the MN (Step 7), the SIP proxy sends an SIP *OK* message piggybacked with the resource tables to the MN (Step 8). Thus, the MN has adequate information to determine which sub-domain the AP resides and knows what resources it should use in that particular sub-domain according to the relationship between SIP proxies and APs.

Alternative implementation is to embed the pre-allocation request in the *REGISTER* message in Step 1, then Step 1 and Step 5 can be combined together to reduce the number of message exchanges. To register an SIP proxy that was attached before, the pre-allocation procedure (Step 6 and Step 7) for the same MN could be omitted if the pre-allocated resources for that MN are not expired.

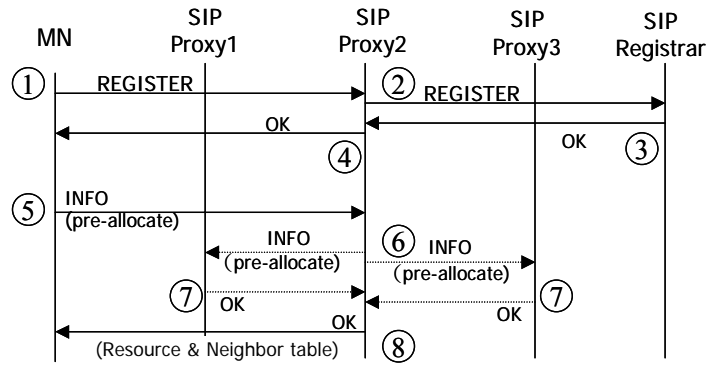


Figure 32. The registration and pre-allocation procedures.

While a centralized location database in an LS is used, the pre-registration and pre-allocation messages are issued by an MN to an LS directly, as the Step 5 shown in Figure 33. This request conveys the current associated AP's ID to the LS, so that LS knows the MN's neighbor SIP sub-domains and can thus relay the pre-registration and pre-allocation messages to those SIP proxies to allocate resources and construct a network deployment table for the MN (Step 6 to Step 8).

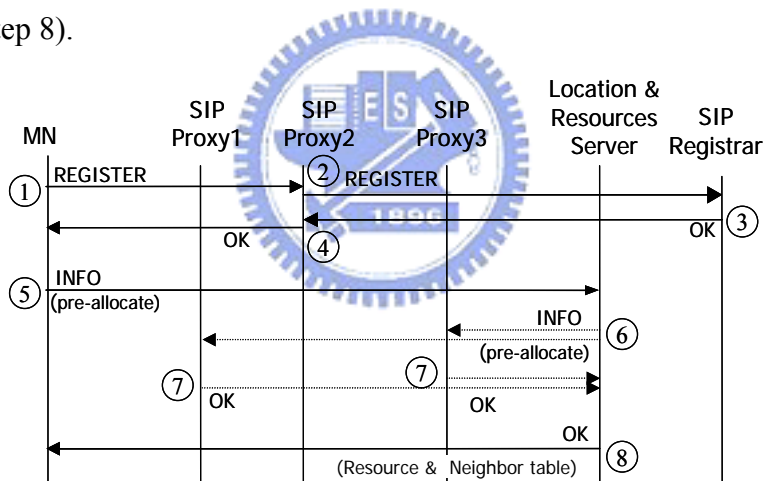


Figure 33. Registration and pre-allocation procedures using a dedicated LS.

### 5.1.2 Handoff

After an MN has registered to an SIP proxy, it can initiate a communication session with a CN. When the MN detects that a handover is about to happen during the session, the MN can learn the new AP's ID (BSSID) from the background scan<sup>1</sup>. If the SIP sub domain that the new AP attached is not changed, the imminent handover is an intra sub-network handover. The MN re-associates with the new AP and the handover procedure is complete. If an MN moves to a pre-registered SIP sub-domain that having pre-allocate resources for the MN, an



inter sub-network handover is triggered. The MN seeks its local resource table to find its reserved IP in the newly attached sub-network. Then, the MN configures this IP without requesting resources. As soon as the MN's IP is setup, the MN updates the CN its location by sending an SIP *RE-INVITE* message. The CN replies *OK* to the MN, and resumes communications with the MN. Hence, the handover latency is largely reduced by avoiding the IP acquisition after a network-layer handover. In order to comply with the conventional SIP protocol, after the MN moves to a new SIP sub-domain, the MN needs to register to the SIP proxy in the sub-domain again. During the registration process to the new SIP proxy, the pre-allocation procedures can be performed in advanced so that the delay for next handovers can be reduced.

If the AP that the MN is going to handover is not presented in MN's resource reservation table, this situation implies that the MN hasn't pre-registered and pre-allocated resources in the new SIP sub-domain before. The MN must perform a complete handover based on the standard SIP mobility procedures.



## **5.2 The integrated cross-layer fast handover**

### **5.2.1 Predicted bi-casting**

In order to reduce the location update delay as well as the packet lost, a bi-casting mechanism is applied. Unlike multicast, a CN only duplicates the RTP packets to a limited number of IP addresses that the MN may likely move to. In other words, when an MN detects a candidate AP that the MN might move to by the background scan<sup>5</sup>, the MN can know the AP's BSSID and the correspondent IP address allocated in this AP area from its resource reservation table. Then, the MN can issue an SIP *INFO* message with its new IP address to the CN. The CN starts to bi-cast the RTP packets to both IP addresses. As soon as the MN

---

<sup>5</sup> The background scan here means that during the packet exchanging between an AP and an MN, the MN can inform the AP that it wants to switch to power saving mode (PSM). During the sleep period of an MN in PSM, the AP has to queue the incoming packets from networks to that MN. Then, the MN changes the current associated channel, scans the signal strength of other channels, and then switches back to the current associated channel. The MN notifies the AP again to return from PSM and retrieves its packets queued on the AP. In order words, by using the background scan, the MN can obtain the signal strength of other channels without breaking



handovers to a new AP, the MN can receive RTP packets before the *RE-INVITE* message is sent. The packet loss is reduced significantly and there is no location update delay. The bi-casting is stopped when the CN receives a *RE-INVITE* message from the MN.

### 5.2.2 AP direct association

The AP probe delay contributes the major part of a link-layer handover delay. Fortunately, the AP probe delay can be eliminated by the proposed LAMP scheme if the wireless network adaptor supports the AP direct association. The AP direct association facilitates an MN to associate to a specific AP with a given BSSID and channel number directly without an AP probe procedure.

To fully utilize the benefits of the AP direct association, the channel information of the neighbor APs is included in the location information table. Then, during communications, the MN can use background scan to probe the channels of the neighbor APs. Once it confirms the AP to roam, the MN directly associates with the AP without interrupting the communications. Therefore, the AP probe delay is diminished. Having the candidate AP's information before handovers, the MN can also determine the IP address to use from the pre-allocated IP list in advanced. The proposed AP direct association tries to rearrange the conventional procedures of a handover that a link-layer handover must be performed first and then informs the upper network layer to proceed the network-layer handover. Instead of the conventional procedures, the proposed AP direct association determines the AP to handover, the IP to use, and then forces the link and network layer changes at the same time. Therefore, the AP scan time and the detection time of the network changes are both reduced.

The LAMP scheme, predicted bi-casting, and AP direct association mechanisms are integrated together to achieve fast handovers. The handover procedures by applying the proposed cross-layer solutions are shown in Figure 34. The period of the disconnected session begins from the step of AP direct association request, i.e. Step 5, to the step of receiving the bi-casted RTP from the CN, i.e. Step 6. The resource reservation and allocation are performed

---

the connection with the current AP, but packet queuing delay on AP might be introduced.

in advanced before a handover, and the most of the SIP *RE-INVITE* and *REGISTER* message exchanges are postponed after a handover. The handover delay is thus reduced.

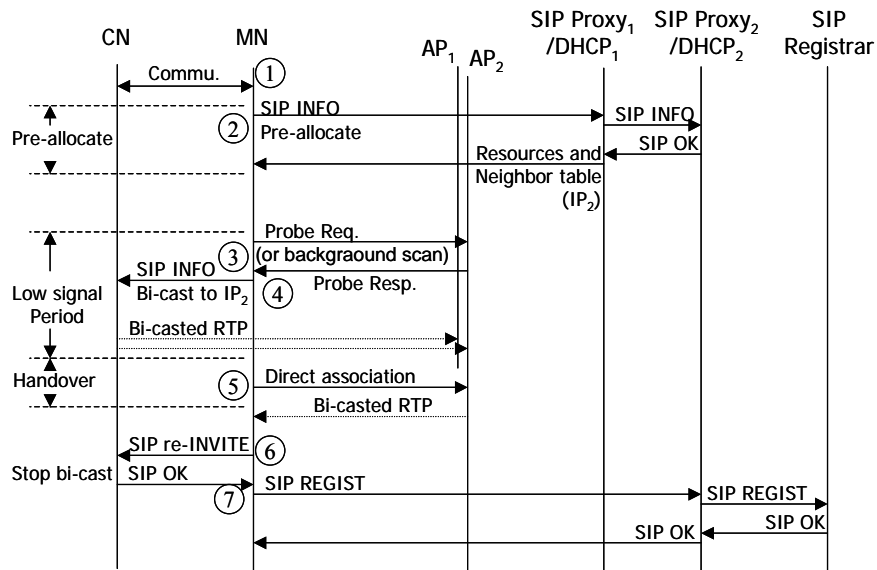


Figure 34. Combined handover procedures by applying the proposed cross-layer solutions.

### 5.3 Mobile IP handoff

The layer-2 triggers and cross-layer topology information are also useful to speedup MIP handoffs. In the following paragraphs, we present a protocol design that incorporates both techniques. The protocol utilizes pre-handoff triggers for agent discovery or address configuration prior to layer-3 handoffs, while post-handoff triggers are used to eliminate the move detection delay. We assume that there is an independent location server (LS) that maintains location information, handoff-to relationships, and AP/MA or AP/DHCP association for a set of APs.

LS can be implemented either as a stand-alone server or as an add-on software module at MAs, DHCP Proxies, or RADIUS servers. The information maintained by LS is assumed to be manually configured, as such information rarely changes in most cases. However, it is also possible to include a function in LS that periodically collects relevant information from associated entities. The contents of LS can be duplicated or hierarchically organized, as DNS (Domain Name Service) servers are usually treated, to distribute processing loads.

### 5.3.1 Using Foreign-Agent CoA

Our protocol coarsely consists of three phases: neighbor request, agent pre-discovery, and pre-registration (Figure 35).



Figure 35. Message flow for the FA-CoA case.

In our design, neighbor MAs periodically exchange their advertisements. After an MN has registered successfully in the visited network, the MN sends a *NeighborRequest* to LS to request topology-related information. LS responds a *NeighborReply* to the MN, which contains a list of all neighbor APs with their locations and associations with MAs. With this information and MN's current location plus moving direction, the MN can determine which neighbor AP is most likely the next serving AP and which FA is the corresponding nFA on the occurrence of a pre-handoff trigger. The MN then starts agent pre-discovery by sending a *ProxyAgentSolicitation* to cFA, requesting nFA's advertisement. cFA returns the requested information via a *ProxyAgentAdvertisement*.

*ProxyAgentSolicitation/Advertisement* can be skipped, if *NeighborReply* also contains nFA's advertisement. To this end, the contents of LS should include advertisements of associated MAs.

An MN acquires its CoA after the agent pre-discovery phase. If the layer-2 handoff is not

yet completed, the MN may then wait for a post-handoff trigger and attempt registration after then. This option eliminates the latency of agent discovery but not the registration delay.

Alternatively, the MN may directly initiate the pre-registration phase after the agent pre-discovery phase. The MN does it by encapsulating a *RegistrationRequest* that ought to be sent to HA in a *Pre-RegistrationRequest* and sending the *Pre-RegistrationRequest* to cFA. cFA forwards the *Pre-RegistrationRequest* to nFA, where the *RegistrationRequest* is decapsulated and sent to the HA. When the *RegistrationReply* sent by the HA is received by nFA, nFA encapsulates it in a *Pre-RegistrationReply* and sends the *Pre-RegistrationReply* to cFA. cFA then forwards the *Pre-RegistrationReply* to the MN.

When the MN receives a post-handoff trigger indicating the completion of a layer-2 handoff, the MN sends a *HandoffNotify* to inform nFA of its arrival. After then, the MN may start a new neighbor request phase to acquire new topology-related information.

Similar to the pre-registration proposal [40], this design has the merit that a layer-3 handoff can occur in parallel with a layer-2 handoff and therefore the overall handoff latency is greatly shortened. Moreover, a layer-3 handoff in our design can be completed even *before* the completion of a layer-2 handoff. Layer-3 handoff latency is completely eliminated in this case.

In fact, the HA can start tunneling packets to nFA, in addition to cFA, even before the completion of the layer-2 handoff. In MIP, an MN can register multiple CoAs by setting “S” bit (the Simultaneous Binding flag) in a *RegistrationRequest*. The advantage of simultaneous binding is the ability of *bi-casting*, i.e., the HA can encapsulate and send packets simultaneously to all registered CoAs. If this option is enabled in our scheme, the MN can start collecting packets by sending a *HandoffNotify* to nFA as soon as the link to the new AP has been established.

### 5.3.2 Using Co-located CoA

We assume that there is a DHCP Proxy in every subnet. In addition to conventional

DHCP functionality, the DHCP Proxy can allocate a CoA and perform Duplicate Address Detection (DAD) and Proxy ARP (Address Resolution Protocol) on behalf of a remote MN (an MN not in the current subnet). The DHCP Proxy also has the ability of encapsulating and decapsulating *Pre-RegistrationRequest* and *Pre-RegistrationReply* messages, respectively. LAS in this case keeps AP/DHCP Proxy association rather than AP/MA association.

Figure 36 shows a message flow where both address pre-configuration and pre-registration are conducted. Similar to the FA-CoA case, after an MN has registered successfully, it sends a *NeighborRequest* to LAS to request topology-related information. On the occurrence of a pre-handoff trigger, the MN determines which AP is most likely MN's next serving AP and which DHCP Proxy corresponds to the next serving AP (i.e., nDHCP Proxy) based on location information. The MN then starts address pre-configuration by exchanging Proxy DHCP messages with nDHCP Proxy. Proxy DHCP messages are similar to regular DHCP messages (Discovery, Offer, Request, and Ack). However, Proxy DHCP messages aim to request a valid CoA in the next network domain rather than in the client's (i.e. the MN's) current domain.

After an MN acquires its CoA, the MN may request registration after a post-handoff trigger or initiate pre-registration immediately. The pre-registration process is similar to that occurs in the FA-CoA case. The difference is that *Pre-RegistrationRequest/Reply* messages are now encapsulated and decapsulated, respectively, at nDHCP Proxy rather than at nFA. When nDHCP Proxy receives a *HandoffNotify* sent by the MN, it stops performing Proxy ARP and DAD defense for the MN.

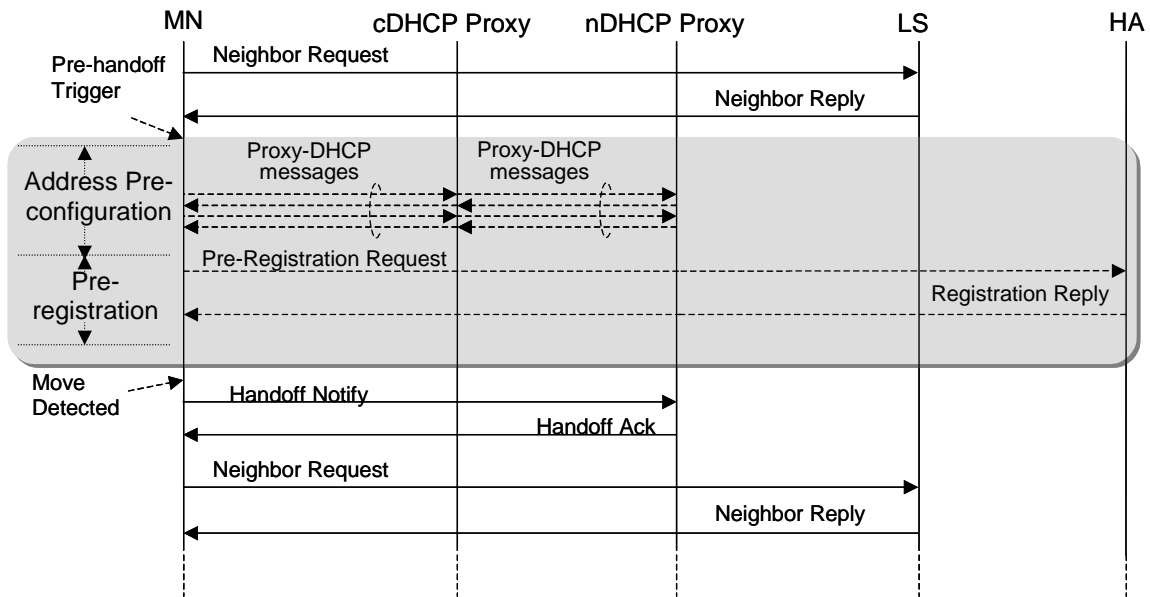


Figure 36. Message flow for the CCoA case.

## 5.4 Implementation and result

### 5.4.1 SIP handoff

We have integrated the aforementioned cross-layer handover mechanisms and implemented the proposed methods to the SIP protocol suite developed by Industrial Technology Research Institute [72].

Our improvement does not only address the protocol delays but also considers the delays caused by the SIP software. To simplify the implementation effort, we pre-construct the location information and distribute the information to each SIP proxy, and we only modify the SIP user agent and the SIP proxy.

Most of the current SIP UA is designed for a fixed network environment and do not consider the mobile environment, we observed that the current SIP UA design for a fixed network further generates latencies in a mobile environment. The internal procedures of the SIP UA program during initiation, call establishment, and handover are first analyzed in Table 2. We measure the detail handover latencies that are caused by both protocol stack and software design and list them in Table 3. Those items in Table 3 marked by asterisk are the

delays caused by the software design.

Table 2. Internal procedures of an SIP UA for initiation, call establishment and handover.

	Procedure	Actions
Initiation	Profile	initialize server address/SIP ports
	Media Management	initialize soundcard I/O, and codecs for encode and decode
	Call Management	setup internal callback function and event dispatch start server state machine
Call establishment	New Client	prepare local SDP start a new client transaction state machine
	Invite	establish an SIP connection to CN play ring-back tone wait OK, send ACK messages
	Media Management	receive CN's SDP listen the incoming RTP packets open RTP port for CN
Handover	AP Handover	probe authentication association
	Network Detection	network detection IP acquisition set IP/gateway IP change notification
	Application Handover	stop RTP transmission stop event dispatch update profile start new event dispatch
	New Client	prepare local SDP start a new client transaction state machine
	Invite	establish an SIP connection to CN play ring-back tone wait OK, send ACK messages
	Media Management	receive CN's SDP listen the incoming RTP packets open RTP ports for CN

Many software design delays can be eliminated if we properly re-arrange the internal program procedures for a mobile environment. For example, the UA stops old RTP transmission and event dispatcher, i.e. Item 8 in Table 3, they are not necessary if the UA uses the same RTP ports before and after a handover.

Table 3. Measurement results of handover delays before the proposed improvements.

Stage	Item	Action	Delay
Link-layer handover	1	Probe ( $t_{\text{Prob}}$ )	320ms
	2	Authentication ( $t_{\text{Auth}}$ )	20ms
	3	Association ( $t_{\text{Assoc}}$ )	20ms
Network-layer handover	4	Network detection ( $t_{\text{Detec}}$ )	100ms
	5	DHCP IP acquisition ( $t_{\text{DHCP}}$ )	5s
	6	Configure IP & Gateway ( $t_{\text{Conf}}$ )	60ms
	7	IP change notification.	40ms
Application-layer handover	*8	Stop old media RTP & dispatch thread.	155ms
	*9	Update location profile.	9ms
	*10	UA reinitiating.	50ms
	*11	Construct re-INVITE message.	16ms
	12	INVITE negotiation ( $t_{\text{Update}}$ )	80ms
	*13	Create new listening port & dispatch thread.	45ms
	14	Wait RTP from CN ( $t_{\text{Recv}}$ ) & RTP Connection to CN ( $t_{\text{Send}}$ )	40ms
Total Time			5.95s

We summarize the delays that are reduced by the proposed mechanisms in Table 4, and also list the experimental measurement results of handover delays by applying the cross-layer fast handover mechanisms in Table 5.

Table 4. Summary of the reduced delays by using different mechanisms.

Mechanisms	Delays eliminated during handovers	Delays postponed after handovers	Delays reduced during handovers
Same RTP port/ Software re-arrangement	Items 8, 10, 13		
Pre-allocation		Item 5	Item 6 (60ms->40ms)
LAMP scheme	Item 7	Items 9, 11, 12	
Bi-casting			Item 14 (MN side)
AP Direct Association	Items 1, 4		

Table 5. Handover delays by applying the proposed improvements.

Stage	Items	Action	Delay
Link-layer handover	2, 3	AP Direct Association	40ms
Network-layer handover	6	Gateway Configuration	40ms
Application-layer handover	14	RTP Connections to CN	40ms
Total delay			120ms



The handover latency of the conventional SIP-based real-time communication application such as VoIP is up to 6 seconds before applying our mechanisms. By employing our designs, the latency can be reduced to about 50ms for link-layer handovers and 120ms for network and application-layer handovers respectively. The implementation results demonstrate that the presented mechanisms consider both software designs and cross-layer optimizations of handovers, and achieve low latency handovers for wireless real-time communications.

5.4.2 Mobile IP handoff

We conducted experiments to measure handoff delays and the number of lost packets in case of using CCoAs. In each experiment, either an MN or a corresponding node (CN) generates packets at constant rate (one per 20 ms). The destination of the packets is the MN (CN) if the CN (MN) is the message source. During the handoff period, a sequence of packets will be lost. We recorded the time at which the last packet was received before a handoff and the time at which the first packet was received after the handoff. The handoff delay is measured as the time difference between these two instants. We also measured the number of lost packets during the handoff period.

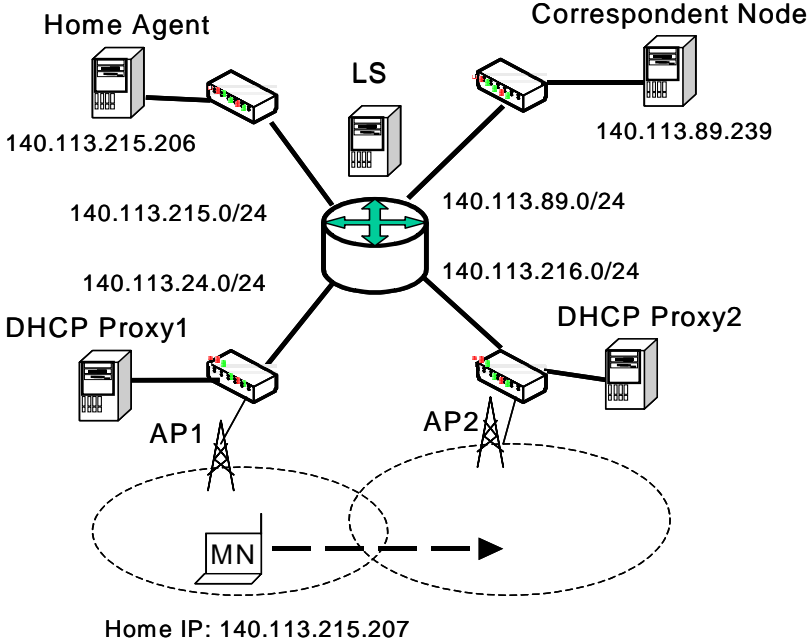


Figure 37. The Mobile IP handoff experimental setup.

The experimental setup is shown in Figure 37. We use Dynamics MIP, which was originally developed at Helsinki University of Technology, as an implementation of MIP. We assume the use of CCoA in the MN. The MN is equipped with two identical Intersil prism2-based IEEE 802.11b wireless interfaces and located in a place where it can associate with either AP1 or AP2. The experimental procedure is as follows.

1. Before handoff, we associate one of the MN's interfaces (Interface 1) with AP1 and the other (Interface 2) with AP2. Moreover, we configure Interface 1 with a CoA through DHCP Proxy 1.
2. Start generating and transmitting packets.
3. Detach the CoA of Interface 1.
4. Configure a new CoA for Interface 2 through DHCP Proxy 2.
5. Perform MIP registration.

This procedure measures only layer-3 handoff delay without address pre-configuration and pre-registration. Step 3 emulates the event that the link to AP1 breaks. Because Step 4 is carried out immediately after Step 3, there is no move detection delay. Also, erratic layer-2 handoff delay is not taken into account.

We slightly changed the procedure to measure layer-3 handoff delay with address pre-configuration. In Step 1, we additionally configure a CoA for Interface 2 through DHCP Proxy 2. We then skip Step 4. To measure layer-3 handoff delay with both address pre-configuration and pre-registration, the MN additionally enables bi-casting by performing a simultaneous registration for the CoA of Interface 2 with the HA in Step 1. Steps 4 and 5 are totally skipped as neither address configuration nor registration is needed. Table 6 lists the obtained results, where each value was measured based on 10 experiment results.

Table 6. Means and standard deviations of handoff delay and the number of lost packets in different settings.

Metrics \ Setting		Address configuration + Registration	Address pre-configuration + Registration	Address pre-configuration + Pre-registration
		Handoff delay	MN sending to CN	Avg. 3416 ms Std. 1188.1
	CN sending to MN	Avg. 2463 ms Std. 914.2	Avg. 88 ms Std. 39.1	Avg. 43 ms Std. 15.7
Num. of lost packets	MN sending to CN	Avg. 166 Std. 58.6	Avg. 3 Std. 2.1	Avg. 1 Std. 1.2
	CN sending to MN	Avg. 121 Std. 46.8	Avg. 3 Std. 1.7	Avg. 0 Std. 0.0
Needed L2 trigger		Post-handoff	Pre-handoff, Post-handoff	Pre-handoff, Post-handoff
Needed location/topology information			AP location, MN location, AP/DHCP Proxy association	AP location, MN location, AP/DHCP Proxy association
Other technique needed				simultaneous binding

As can be seen from the table, original layer-3 handoff delay is intolerable for VoIP applications. If address pre-configuration is used, the handoff delay sharply drops to around 85 ms, which may still be unacceptable for time-critical applications for the rather-high variation. If address pre-configuration plus pre-registration is used, the handoff delay further drops to only around 45 ms with low variation. Such result should meet the delay requirement of VoIP applications.

The number of lost packets in each setting generally agrees with the handoff delay. Original MIP incurs more than one hundred lost packets. The value drops to only 3 when address pre-configuration is used and to 0 or 1 when pre-registration is further performed.

## Chapter 6 Mobility database restoration

To simplify our discussion, all the events that lead to location update of a mobile user, such as registration, call origination, and crossing of location areas, will be referred to as registration. Since accessing a radio channel is more expensive than accessing a local storage, we assume that no autonomous registration is performed. Note that for a per-user database checkpointing algorithm, the checkpointing timer for each user may be different.

### 6.1 Three checkpointing algorithms

Three per-user database checkpointing algorithms are depicted in Figure 38. In the figures,  $t_r$  denotes the interval between two consecutive registrations and  $T_C$  denotes the checkpointing timer. In general, when  $T_C$  expires, the user record is checkpointed if it has been updated.

#### 6.1.1 Periodically checkpointing the modified record

The first scheme is essentially the same as the UMTS checkpointing method except that it is performed on a per-user basis and that only the modified location record is checkpointed. It works as in the following.

1. When a user record is checkpointed, a timer,  $T_C$ , of fixed expiration time is set on (see  $t_0$  in Figure 38.a).
2. When  $T_C$  expires, if the user record has been modified, the user record is checkpointed (see  $t_3$  and  $t_6$  in Figure 38.a). Otherwise, if the user record has not been modified when  $T_C$  expires,  $T_C$  is restarted (see  $t_4$  in Figure 38.a) and the process repeats.

This scheme will be referred to as FIXED, because a timer of fixed expiration time is used.

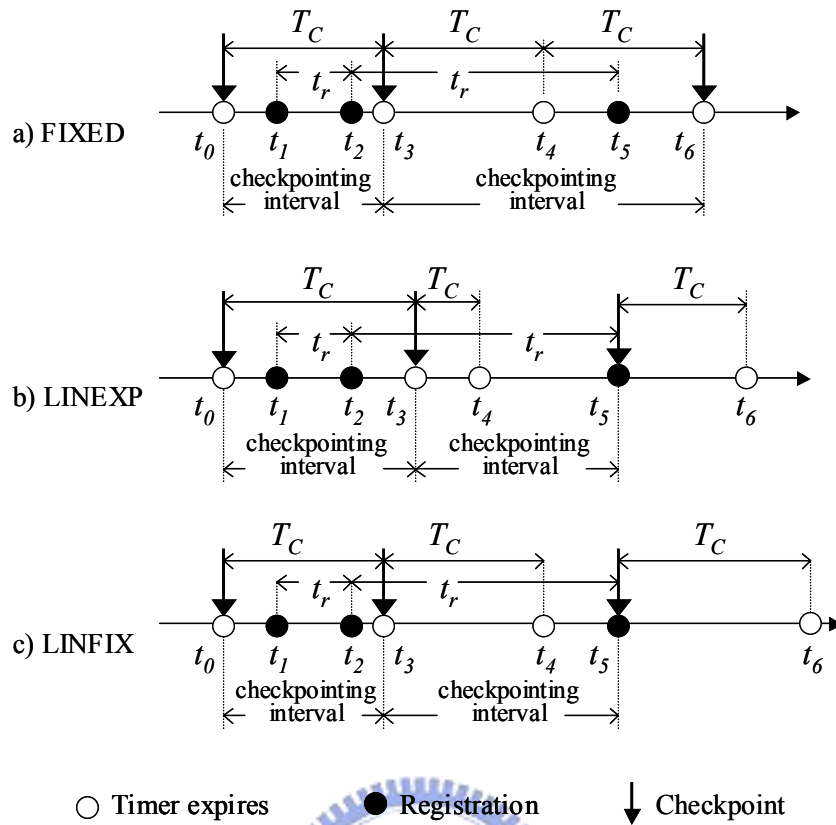


Figure 38. Three per-user checkpointing algorithms.

### 6.1.2 Lin's per-user checkpointing algorithm with an exponential timer

Lin presented a per-user checkpointing algorithm, which is illustrated in Figure 38.b. His algorithm assumes that timer  $T_C$  is exponentially distributed with mean  $1/\lambda$ . The algorithm is described as in the following.

1.  $T_C$  is started when a user record is checkpointed ( $t_0$  in Figure 38.b).
2. After  $T_C$  expires, if the user record has been updated (see  $t_3$  in Figure 38.b), it will be checkpointed. Otherwise, if the user record is not updated (see  $t_4$  in Figure 38.b), the user record will be checkpointed at the next user registration (see  $t_5$  in Figure 38.b).

Lin's algorithm differs from the FIXED scheme in that when the timer expires and the user record is not modified, the user record is checkpointed at the next user registration, but the scheme FIXED waits until the timer expires after the next user registration. This algorithm will be referred to as LINEXP.

### 6.1.3 Lin's per-user checkpointing algorithm with a fixed checking interval

To study the effects of exponential timers and fixed timers, we apply fixed timers to Lin's per-user checkpointing algorithm. The algorithm is identical to LINEXP except that timer  $T_C$  is of fixed expiration interval. An example user registration and checkpointing scenario can be found in Figure 38.c. This algorithm will be referred to as LINFIX.

## 6.2 Cost function analysis

The cost function we consider in the paper includes the cost of paging a user with an invalid location record and the cost of checkpointing a user's location record. Let  $P_{ib}$  denote the probability that a user record in the backup database is invalid when the main database fails. When an invalid user record is encountered by an incoming call, the network pages the user. Let  $t_f$  denote the average database failure interval. It can be shown that the paging cost is proportional to  $P_{ib}/t_f$ . Let  $I$  denote the expected length of the checkpointing interval. The checkpointing cost is proportional to  $1/I$ . Let  $c_b$  denote the cost of checkpointing a location record, and  $c_p$  denote the expected cost to page a user with an invalid location record due to mobility database failure. For the cost function we consider, the checkpointing cost equals to  $c_b(1/I)$ , and the paging cost equals to  $c_p(P_{ib}/t_f)$ . The cost function is given as follows

$$C = c_b \cdot 1/I + c_p \left( P_{ib}/t_f \right) \quad (1)$$

We will study the effects of changing the expiration interval of  $T_C$  on the total cost, and try to find the optimal timeout interval to minimize the total cost. For our analytic models,  $t_r$  is assumed to be exponentially distributed with mean  $1/u$ . However, later in the computer simulation,  $t_r$  can have a gamma distribution with mean  $1/u$  and variance  $\sigma$ .

### 6.2.1 FIXED

Let  $T$  denote the expiration interval of timer  $T_C$ . Consider two consecutive checkpoints, checkpoint A and checkpoint B, as shown in Figure 39. At checkpoint A, the user record is checkpointed and timer  $T_C$  is activated. Since the user registers after the expiration of  $T_C$  for the  $(i-1)$ th time and before the  $i$ th time, the user record is checkpointed when  $T_C$  expires for the  $i$ th time, at checkpoint B.

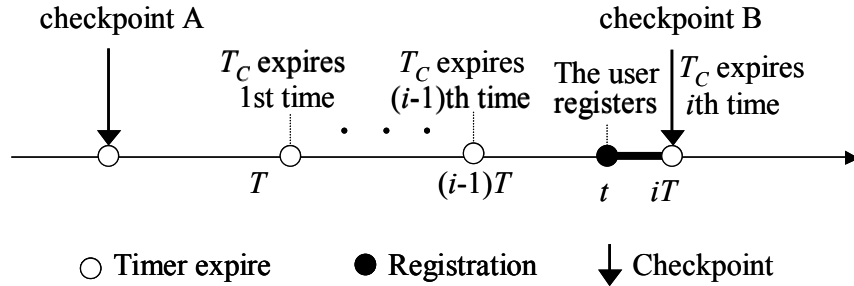


Figure 39. Two consecutive checkpoints (FIXED).

Let  $Q_i$  denote the probability that the interval between two consecutive checkpoints is of length  $iT$ , i.e., the user registers between time  $(i-1)T$  and  $iT$ . We have

$$Q_i = \int_{(i-1)T}^{iT} ue^{-ut} dt = e^{-u(i-1)T} (1 - e^{-uT})$$

The expected checkpointing interval can be obtained as follows.

$$I_{FIXED} = \sum_{i=1}^{\infty} iTQ_i = \frac{T}{1 - e^{-uT}} \quad (2)$$

Since the inter-registration interval has an IID (independent identically distribution). The user registrations can be modeled as a renewal process. The behavior of checkpointings is also a renewal process; because at each checkpoint, timer  $T_C$  is restarted and the registration interval is exponentially distributed. For a reliable mobility database, we expect that interval between two consecutive database failures is significantly larger than the user registration interval and the checkpointing interval. In this situation, the time when the database fails can be seen as a random observer to the renewal process of user registration and that of checkpointing. The backup user record is invalid only after the user registers and before the record is checkpointed. If the main database fails during this period, the system restores an invalid backup record. Thus, we have

$$P_{ib\_FIXED} = \left( \sum_{i=1}^{\infty} \int_{(i-1)T}^{iT} u e^{-ut} (iT-t) dt \right) / I_{FIXED} = 1 - \frac{1-e^{-uT}}{uT} \quad (3)$$

From (1), (2), and (3) the cost function can be obtained as follows,

$$C_{FIXED} = c_b \cdot \frac{1-e^{-uT}}{T} + \frac{c_p}{t_f} \left( 1 - \frac{1-e^{-uT}}{uT} \right) = \frac{c_p}{t_f} + \frac{1}{u} \left( uc_b - \frac{c_p}{t_f} \right) \left( \frac{1-e^{-uT}}{T} \right) \quad (4)$$

Our goal is to minimize the cost by choosing an appropriate  $T$ .

$$\frac{d}{dT} \left( \frac{1-e^{-uT}}{T} \right) = \left( -1 + \frac{1+uT}{e^{uT}} \right) / T^2$$

Since  $e^{uT} = 1 + uT + \frac{(uT)^2}{2!} + \frac{(uT)^3}{3!} + \dots$  and  $uT > 0$ , we have  $\frac{1+uT}{e^{uT}} \leq 1$  and

$\left( -1 + \frac{1+uT}{e^{uT}} \right) / T^2 \leq 0$  for  $T \geq 0, u \geq 0$ . This leads to that  $\frac{1-e^{-uT}}{T}$  is a monotonic

decreasing function of  $T$ . From (4), we can draw the following conclusions:

1. If  $uc_b < \frac{c_p}{t_f}$ ,  $C_{FIXED}$  is a monotonic increasing function of  $T$ .  $C_{FIXED}$  can be minimized when  $T = 0$ , i.e., the expiration interval of the timer is of length 0. At each user registration, since the timer must have expired, the user record should be checkpointed. In this case,  $C_{FIXED} = uc_b$ .

2. If  $uc_b > \frac{c_p}{t_f}$ ,  $C_{FIXED}$  is a monotonic decreasing function of  $T$ .  $C_{FIXED}$  can be minimized

when  $T = \infty$ , i.e., the expiration interval of the timer is of infinite length. Since the timer never expires, the user record should never be checkpointed. In this case

$$C_{FIXED} = \frac{c_p}{t_f}.$$

3. If  $uc_b = \frac{c_p}{t_f}$ ,  $C_{FIXED} = uc_b = \frac{c_p}{t_f}$ ;  $C_{FIXED}$  is a constant independent of  $T$ .  $T$  can be any

value, i.e., at a user registration, the user record can be either checkpointed or not checkpointed. The cost of checkpointing the record and the cost of not checkpointing (the expected paging cost) are the same.

Note that the minimum cost that scheme FIXED can achieve equals to  $\text{Min} \left( uc_b, \frac{c_p}{t_f} \right)$



## 6.2.2 LINEXP

The analysis of the LINEXP is similar to that of the FIXED. Considering two consecutive checkpoints, there are two possible conditions as shown in Figures 3.a and 3.b. For Case I, shown in Figure 40.a, the user registers before timer  $T_C$  expires, so that the checkpointing interval is equal to the expiration interval of timer  $T_C$  ( $s$  in Figure 40.a). For Case II, shown in Figure 40.b, the user registers after timer  $T_C$  expires, so that the checkpointing interval is equal to the user registration interval ( $t$  in Figure 40.b).

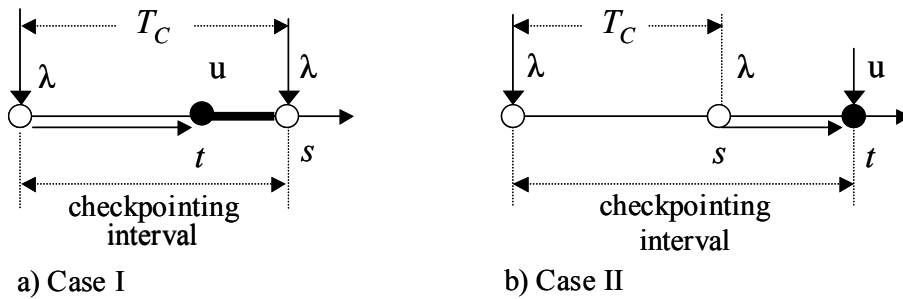


Figure 40. Two possible cases of checkpointing (LINEXP).

Since the registration interval and the checkpointing timer are both exponentially distributed, the expected length of checkpointing interval can be obtained by adding the intervals of both conditions.

$$\begin{aligned}
 I_{LINEXP} &= \int_0^{\infty} \int_0^s s \cdot u e^{-ut} \cdot \lambda e^{-\lambda s} dt ds + \int_0^{\infty} \int_t^{\infty} t \cdot u e^{-ut} \cdot \lambda e^{-\lambda s} dt ds \\
 &= \frac{1}{\lambda} + \frac{\lambda}{u(\lambda + u)}
 \end{aligned} \tag{5}$$

For Case I, the backup user record is invalid only after the user registration at time  $t$ . For Case II, the backup user record is always up-to-date because when the user registers, the record is also checkpointed. From the random observer property,  $P_{ib}$  can be obtained as

$$\begin{aligned}
 P_{ib\_LINEXP} &= \left( \int_0^{\infty} \int_0^s (s-t) \cdot u e^{-ut} \cdot \lambda e^{-\lambda s} dt ds \right) / I_{LINEXP} \\
 &= \frac{u^2}{u^2 + \lambda u + \lambda^2}
 \end{aligned} \tag{6}$$

From (1), (5), and (6), the cost function can be obtained as follows,

$$C_{LINEXP} = c_b \cdot \left( \frac{1}{\lambda} + \frac{\lambda}{u(\lambda + u)} \right) + \frac{c_p}{t_f} \cdot \frac{u^2}{u^2 + \lambda u + \lambda^2} \quad (7)$$

Since  $\frac{d}{d\lambda} \left( \frac{1}{u^2 + \lambda u + \lambda^2} \right) = \frac{-(u + 2\lambda)}{(u^2 + \lambda u + \lambda^2)^2} \leq 0$  for  $\lambda \geq 0, u \geq 0$ ,  $\frac{1}{u^2 + \lambda u + \lambda^2}$  is a monotonically decreasing function of  $\lambda$ . We also obtain the following results, which are essentially the same as those obtained from FIXED. Note that the expected timeout interval of the exponential timer is  $1/\lambda$ .

1. If  $uc_b < \frac{c_p}{t_f}$ ,  $C_{LINEXP}$  is a monotonically decreasing function of  $\lambda$ . The optimum

$C_{LINEXP} = uc_b$ , when  $\lambda = \infty$ , i.e., the timeout interval of the checkpointing timer is of length 0.

2. If  $uc_b > \frac{c_p}{t_f}$ ,  $C_{LINEXP}$  is a monotonically increasing function of  $\lambda$ . The optimum

$C_{LINEXP} = \frac{c_p}{t_f}$ , when  $\lambda = 0$ , i.e., the timeout interval of the checkpointing timer is of infinite length.

3. If  $uc_b = \frac{c_p}{t_f}$ ,  $C_{LINEXP} = uc_b = \frac{c_p}{t_f}$ , a constant independent of  $\lambda$ .  $\lambda$  can be any value.

### 6.2.3 LINFIX

Since this algorithm is identical to LINEXP algorithm except that it utilizes a checkpointing timer with fixed expiration interval, the two checkpointing cases of LINEXP shown in Figure 40 can also be used to analyze LINFIX. For Case I, the checkpointing interval is equal to the expiration interval of the timer, which is  $T$ . For Case II, the checkpointing interval is equal to the user registration interval ( $t$ ). The expected length of checkpointing interval can be obtained as

$$I_{LINFIX} = \int_0^T T \cdot ue^{-ut} dt + \int_T^\infty t \cdot ue^{-ut} dt = T + \frac{e^{-uT}}{u} \quad (8)$$

$P_{ib}$  equals to the probability that the main database fails in Case I after the user registration.

$$P_{ib\_LINFIX} = \left( \int_0^T (T-t) \cdot u e^{-ut} dt \right) / I_{LINFIX} = 1 - \frac{1}{uT + e^{-uT}} \quad (9)$$

From (1), (8) and (9), the cost function can be obtained as

$$C_{LINFIX} = \frac{c_b}{T + \frac{e^{-uT}}{u}} + \frac{c_p}{t_f} \cdot \left( 1 - \frac{1}{uT + e^{-uT}} \right) = \frac{c_p}{t_f} + \left( uc_b - \frac{c_p}{t_f} \right) \cdot \frac{1}{uT + e^{-uT}} \quad (10)$$

Since  $\frac{d}{dT} \left( \frac{1}{uT + e^{-uT}} \right) = \frac{-u(1 - e^{-uT})}{(uT + e^{-uT})^2}$  and  $e^{-uT} \leq 1$  for  $T \geq 0, u \geq 0$ , we get

$\frac{-u(1 - e^{-uT})}{(uT + e^{-uT})^2} \leq 0$ . Thus,  $\frac{1}{uT + e^{-uT}}$  is a monotonically decreasing function of  $T$ . We can

obtain the same results in FIXED and LINEXP.

1. If  $uc_b < \frac{c_p}{t_f}$ ,  $C_{LINFIX}$  is a monotonically increasing function of  $T$ . The optimum

$$C_{LINFIX} = uc_b, \text{ when } T = 0.$$

2. If  $uc_b > \frac{c_p}{t_f}$ ,  $C_{LINFIX}$  is a monotonically increasing function of  $T$ . The optimum

$$C_{LINFIX} = \frac{c_p}{t_f}, \text{ when } T = \infty.$$

3. If  $uc_b = \frac{c_p}{t_f}$ ,  $C_{LINFIX} = uc_b = \frac{c_p}{t_f}$ ,  $T$  can be any value.

It is important to note that the analyses of three algorithms all lead to the same conclusions. If the checkpointing cost out-weights the paging cost ( $uc_b > \frac{c_p}{t_f}$ ), we should

never checkpoint a user record. On the other hand, if  $\frac{c_p}{t_f} > uc_b$ , we should use a duplicated database.

### 6.3 Results

Without loss of generality, we let the expected user registration rate,  $u$ , to be 1 per

unit-of-time. This can be interpreted as one registration per  $x$  minutes. A small  $x$  means the user registers frequently. First we consider exponential registration interval and examine the effects of the timeout interval ( $T$ ) on the expected checkpointing interval ( $I$ ) and on the probability of invalid backup record at database failure ( $P_{ib}$ ). The expiration interval of timer  $T_C$  used in FIXED and LINFIX varies in the range of 0.2-8 unit-of-time. In addition, the expected expiration interval of the exponential timer in LINEXP also varies in the range of 0.2-8.

The curves in Figure 41.a are obtained from Equations (2), (5), and (8), and those in Figure 41.b are from Equations (3), (6), and (9). The results indicate that all the three algorithms obtain similar results; both the expected checkpointing interval and the probability of invalid backup record increase as the timeout interval increases. The differences between the three algorithms are small, but for a given timeout interval, LINFIX has the smallest  $P_{ib}$  at the cost of the shortest checkpointing interval,  $I$ . When the timeout interval is larger than 4 (i.e., four times the registration interval), all checkpointing algorithms act much the same. This is because when a long checkpointing timer expires, the user record is most likely modified and needs to be checkpointed for all algorithms.

Figure 42 shows the cost functions at various paging costs; the user registration interval is exponentially distributed. The curves are obtained from Equations (4), (7), and (10). Without loss of generality, let  $u=1$ ,  $c_b=1$  and  $\frac{c_p}{t_f}$  vary in the range of 0.5-1.5. The results indicate that when  $uc_b = \frac{c_p}{t_f}$ , the cost of all algorithms becomes 1 ( $= \frac{c_p}{t_f} = uc_b$ ), and the total cost is independent of the timer expiration interval. Furthermore, when  $\frac{c_p}{t_f} > uc_b$ , the cost increases as the timeout interval,  $T$ , increases, and when  $uc_b > \frac{c_p}{t_f}$ , the cost decreases as  $T$  increases.

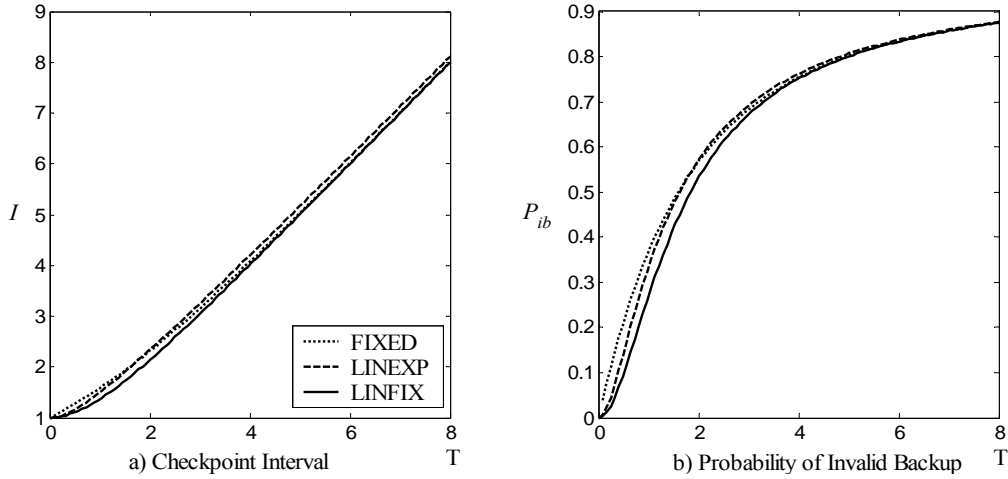


Figure 41. Comparison of checkpointing algorithms for exponential registration interval.

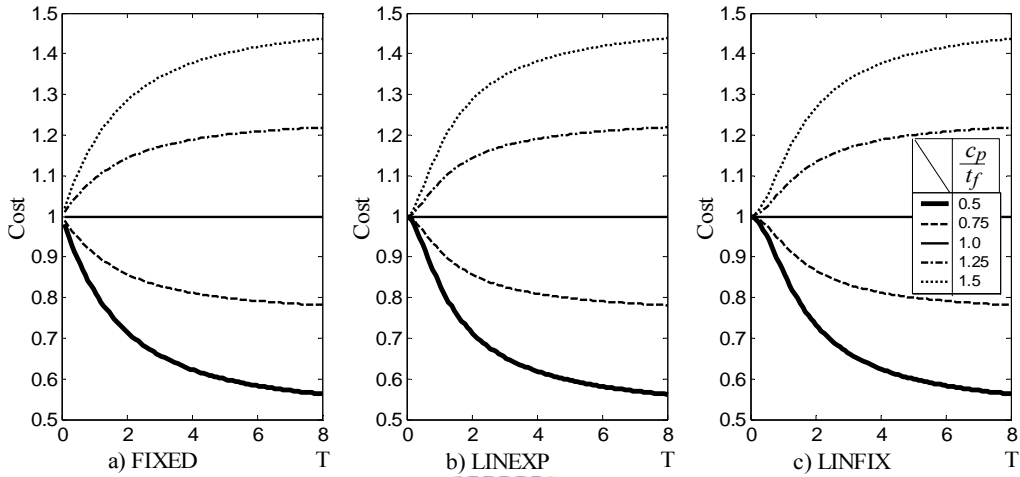


Figure 42. Cost functions for various weighting factors (exponential registration interval).

### 6.3.1 Simulation result

Computer simulations have been used to study the effects of changing the registration interval variance on  $P_{ib}$  and  $I$ . The registration interval is assumed to have a gamma distribution with mean 1 unit-of-time and variance  $\sigma$ . In order to speedup the simulation, the database failure rate is chosen to be  $\frac{1}{500}$ , which may be unrealistically large but is small enough (compared to the user registration rate) to obtain correct simulation results, i.e., the random observer property still holds. In each computer simulation, the database fails for at least 10,000 times until stable results are obtained. The simulation is verified to be corrected as shown in Figure 43. The results in Figure 44 indicate that for all algorithms,  $I$  increases as

$\sigma$  increases. This is because as  $\sigma$  increases, there are more short registration intervals that are shorter than the checkpointing timer, and no checkpointing is needed at registration. In addition,  $P_{ib}$  drops as  $\sigma$  increases. This is because as  $\sigma$  increases, there are also longer registration intervals during which the backup record is always valid. As a random observer, the database failure is more likely to occur at long registration intervals. As a result,  $P_{ib}$  drops.

Figure 45 depicts the cost functions for various  $\frac{c_p}{t_f}$  values and different variance of registration intervals. The results in Figure 45.a, c, d, and f indicate that the optimal choice of the checkpointing timer is determined by the weighting ratio of the paging cost and the checkpointing cost; it is independent of the variance of registration intervals. The optimal expiration interval of the checkpointing timer is either of length 0 or infinity. Only when the checkpointing cost and the paging cost almost balance, i.e., when  $uc_b \approx \frac{c_p}{t_f}$ , the variance of registration intervals affects the choice of checkpointing algorithms. The results in Figure 45.b indicate that when  $\sigma (= 0.5)$  is small, a duplicated database should be used ( $T=0$ ). The results in Figure 45.e indicate that when  $\sigma (= 2)$  is large, all algorithms can outperform a duplicated database scheme or a non-checkpointing database scheme. This is because as  $\sigma$  increases, there are more short registration intervals; by setting appropriate timer length, all three checkpointing schemes can skip the short registration intervals (shorter than the checkpointing timer) without checkpointing, and thus reduce the overall cost. Our results indicate FIXED is the best at skipping short registration intervals, and the optimum expiration interval of the checkpointing timer can be obtained from computer simulation.

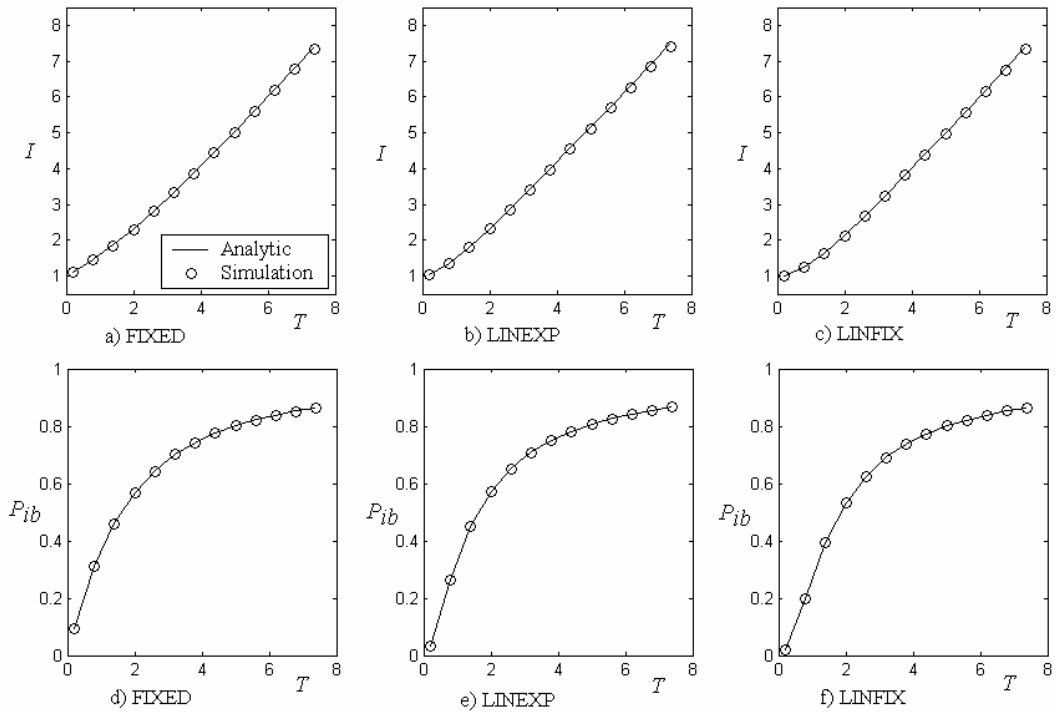


Figure 43. The effects of timer expiration interval.

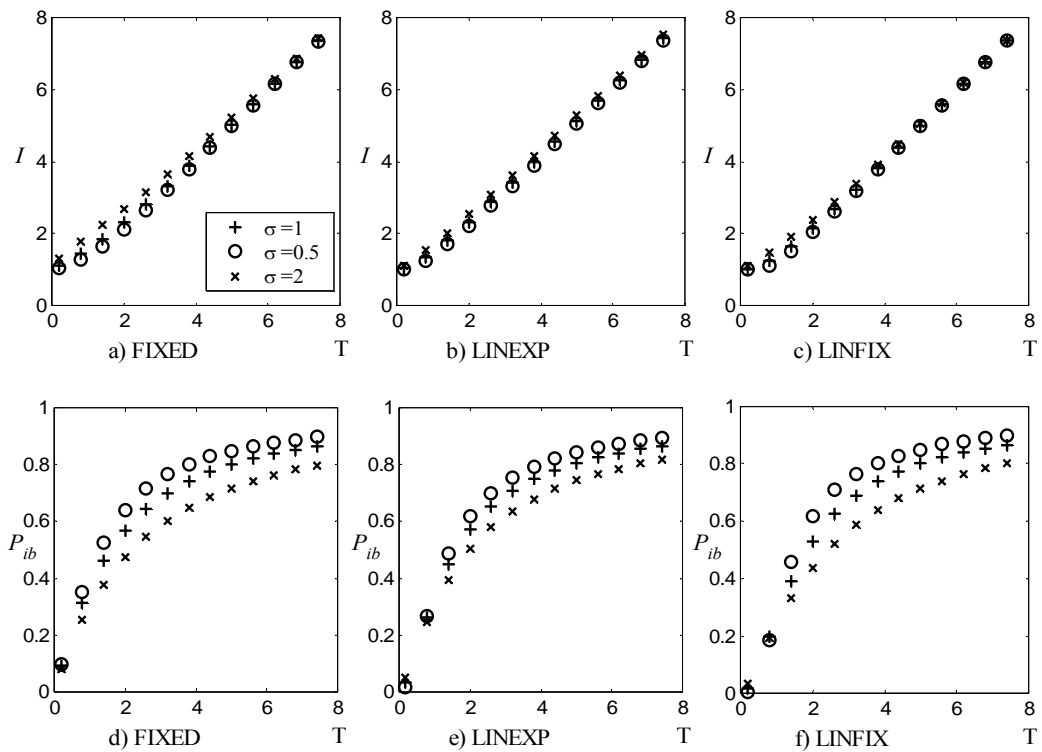


Figure 44. The effects of registration interval variance.

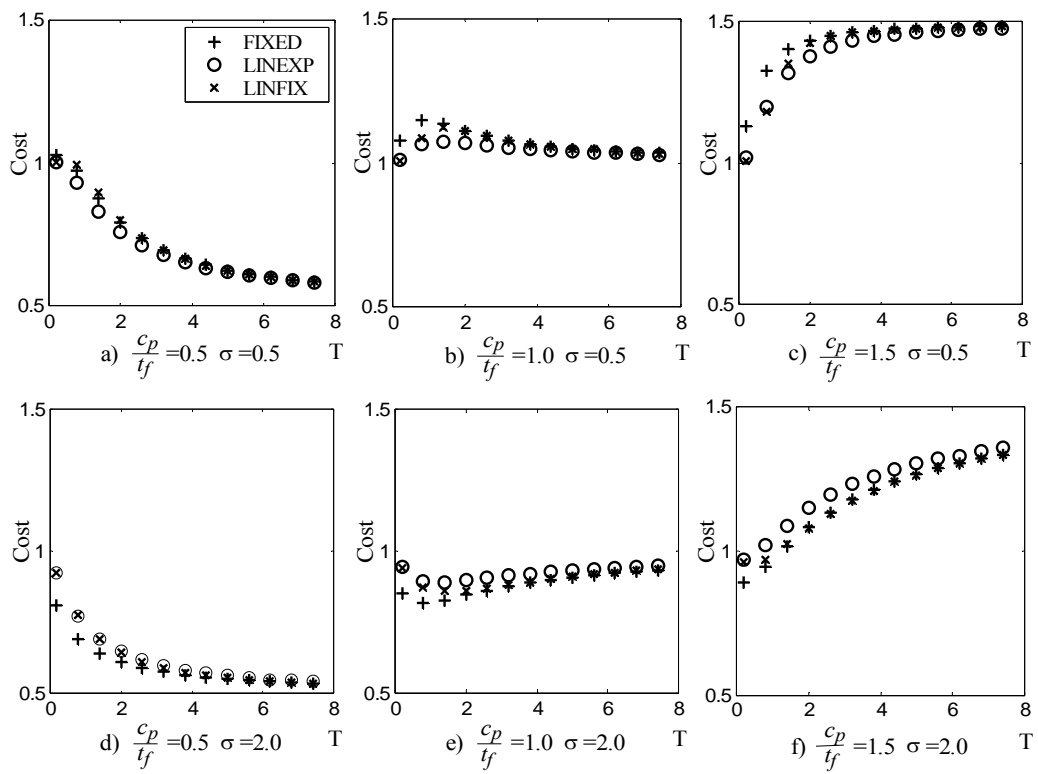


Figure 45. Cost function for various weighting factors and different variance of registration interval.





## Chapter 7 Conclusions

A VoIP converged environment can be achieved through the supports that are proposed in this thesis. We have presented a simple and flexible framework for the interworking functions of VoIP protocols based on IN half-call BCSM. In addition, we have implemented the basic gateway components, O\_BCSMs and T\_BCSMs, for SIP, H.323, and MGCP. Using these components, gateways for SIP/H.323, SIP/MGCP, and MGCP/H.323 can be constructed. We have demonstrated a novel solution to the NAT traversal problem using SIP as an example. Our method works no matter whether a public SIP proxy is present or not. Most important of all, it requires no changes to the existing VoIP software or devices, such as SIP user agents, SIP proxies, and NATs.

Also, we have demonstrated how those fast handoff techniques in different layers can be integrated to reduce the overall handoff delay. Our result shows that the handoff delay meets the delay requirements of VoIP applications if the MN can perform both address configuration and registration beforehand. We have analyzed three per-user location database checkpointing algorithms using numeric analysis and computer simulation. It is convinced that there may not need a sophisticated checkpointing algorithm if the checkpoint cost and paging cost are not fairly equal.

### 7.1 Integrated call agent

Our approach using half-call model simplifies the effort in interworking with a call signaling protocol, such as ISUP and Q.931, in the network. By using the same interaction events of the half-call model, the BCSMs of a call signaling protocol can interwork with the existing BCSMs. In addition, an ICA containing all the BCSMs is able to translate messages between call signaling protocols. Under this half-call control framework, a converged VoIP network can be managed by a group of coordinating ICAs such that two user devices managed by different ICAs can communicate with each other. The call routing function that

determines the location and protocol of the called party has not been fully investigated in this paper. As a mobile user may change his IP address and VoIP devices constantly, this problem becomes even more complicated. We need registration and/or paging schemes to track mobile users in the converged telecommunication network. Recently, P2P (peer-to-peer) VoIP communications, such as Skype, have become very popular. The interworking function for a P2P VoIP system and a client-server one (such as SIP) is an important issue that needs to be investigated.

## **7.2 VoIP's NAT traversal**

Although our NAT traversal solution introduces an internal SIP proxy and a relay agent, their main job is to relay packets and they are easy to implement. Though the internal SIP proxy is application dependent, it inspects only a small subset of the protocol messages, such as *REGISTER*, *INVITE*, *OK*, and *ACK* in SIP. This is a practicable solution that can adapt to most VoIP applications and protocols. We also simplify the NAT type test and reduce the unnecessary hop count of the RTP stream since our IPX participates only in call setup signaling but not in RTP exchanging. The keep-alive traffic are also minimized and confined between an IPX and an RA. Although the signaling messages was relayed by an IPX and an RA, they are lightweight and do not cause vast burden.

## **7.3 Handoff**

The handover latency of the conventional SIP-based real-time communication application such as VoIP is about 6 seconds before applying our mechanisms. By employing our designs, the latency can be reduced to about 40ms for link-layer handovers and 120ms for network and application-layer handovers, respectively. The implementation results demonstrate that the presented mechanisms consider both software designs and cross-layer optimizations of handovers, and achieve low latency handovers for wireless real-time communications.

We have reviewed several cross-layer techniques that aim to cut down handoff delays in IEEE 802.11/Mobile IP environments. Among them, a post-handoff layer-2 trigger can

successfully eliminate the link-switch detection delay. A pre-handoff layer-2 trigger, on the other hand, can be used as a signal to execute layer-3 handoff related activities prior to the associated layer-2 handoff. An MN may utilize cross-layer topology together with its location and moving direction information to determine the next serving AP and the associated FA or DHCP server to speedup both layer-2 and layer-3 handoffs.

#### **7.4 HLR**

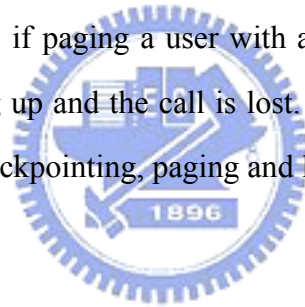
Checkpointing can be used to enhance the reliability of the location database of PCS networks. Since each user exhibits a unique calling and moving behavior, per-user checkpointing schemes can better serve both the users and operators. In this paper, we have analyzed three per-user location database checkpointing algorithms using numeric analysis and computer simulation. The costs that we considered include the checkpointing cost and the paging cost. Our results indicate that when inter-registration times are exponentially distributed, a user location record should either be always checkpointed at registration, or be never checkpointed at all, depending on the weighting ratio between the checkpointing cost and the paging cost. If the checkpointing cost is of more concern, the user record should never be checkpointed; otherwise, the user record should always be checkpointed (duplicated) at registration. We have also studied the effects of the variance of registration interval using computer simulation. When the checkpointing cost and the paging cost almost balance, and the variance of registration interval is large, a simple checkpointing algorithm using a fixed checkpointing timer is preferred and the optimal choice of the checkpointing timer can be determined by computer simulation. In this paper, we did not investigate the effects of incoming call arrivals on the optimal choice of the checkpointing frequency directly; we assumed that the expected paging cost is known.

## Chapter 8 Future work

For the protocol interoperability, a protocol syntax and semantic analyzer would be necessary for the fast addition of a new signaling protocol into the ICA. Meanwhile, how to converge those parameters in the payload of the messages is also a challenge.

For the cross-layer fast handover scheme, it needs the layer-2 triggers and the direct association support build into the network adapter. However, most vendors did not add all these features to their products. We would find a vendor and form a partnership to develop the fast handover supports into the network adapter.

For the analysis of checkpoint schemes, further study is needed to obtain the paging cost in the PCS networks. In addition, if paging a user with an invalid location record cannot be done in time, the caller may hang up and the call is lost. It may be meritorious to consider a cost function consisting of the checkpointing, paging and lost-call costs.



## Reference

- [1] ITU, "Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear prediction (CS-CELP)," Recommendation G.729, ITU-T.1, Geneva, 1996.
- [2] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," IETF RFC 1889, Jan 1996.
- [3] H. Schulzrinne and L. C. Stephen, "RTP Profile for Audio and Video Conferences with Minimal Control," IETF Request for comments 1890, Jan 1996
- [4] ITU, "Packet-Based Multimedia Communication Systems," ITU-T Recommendation H.323, ITU, 1998.
- [5] ITU, "Call signaling over UDP," ITU-T Recommendation H.323 Annex E, ITU, 1998.
- [6] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," IETF Request for Comments 3261, June 2002.
- [7] M. Arango, A. Dugan, I. Elliott, C. Huitema, and S. Pickett, "Media Gateway Control Protocol (MGCP) Version 1.0," IETF Request for Comments 2705, IETF, 1999.
- [8] M. Handley and V. Jacobson, "SDP: Session Description Protocol," IETF Request for Comments 2327, IETF, 1998.
- [9] C. Groves and M. Pantaleo, "The Megaco/H.248v2 Gateway Control Protocol, version 2," IETF Internet draft draft-ietf-megaco-h248v2-03.txt, IETF, 2002.
- [10] U. Black, "The intelligent network: Customizing telecommunication networks and services," Prentice Hall, New Jersey, 1998.
- [11] EURESCOM: "Providing IN functionality for H.323 telephony calls," Project report P916, Research and Strategic Studies in Telecommunications, European Institute, 1999.
- [12] K.V. Vemuri, "SPHINX: A Study in Convergent Telephony," IP Telecom Services Workshop (IPT2000), Georgia, USA, 2000, pp. 9-18.
- [13] H. Agrawal, V. Palawat, and R.R. Roy, "SIP-H.323 Interworking," IETF Internet draft draft-agrawal-sip-h323-interworking-reqs-02.txt, IETF, 2001.
- [14] K. Singh, and H. Schulzrinne, "Interworking Between SIP/SDP and H.323," 1st IP-Telephony Workshop ( IPT2000), Berlin, 2000, pp. 75-92.
- [15] V. K. Gurbani, and V. Rastogi, "Accessing IN Services from SIP networks," IETF Internet draft draft-gurbani-iptel-sip-to-in-05.txt, IETF, 2001.

- [16] F. Haernes, "SIP-IN Interworking Protocol Architecture and Procedures," IETF Internet draft draft-haerens-sip-in-00.txt, IETF, 2001.
- [17] R. Ackermann, V. Darlagiannis, M. Goertz, M. Karsten, and R. Steinmetz, "An Open Source H.323 / SIP Gateway as Basis for Supplementary Service Interworking," 2nd IP Telephony Workshop ( IPTEL2001), New York, USA, 2001, pp. 1-7.
- [18] W. Jiang, J. Lennox, S. Narayanan, H. Schulzrinne and K. Singh, "CINEMA: Columbia InterNet Extensible Multimedia Architecture," Technical Report CUCS-011-02, Department of Computer Science, University of Columbia, 2002.
- [19] P. Srisuresh and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations," RFC 2663, August 1999.
- [20] Y. Rekhter, B. Moskowitz, D. Karrenberg, and G. de Groot, "Address Allocation for Private Internets," RFC 1597, March 1994.
- [21] D. Senie, "Network Address Translator (NAT)-Friendly Application Design Guidelines", RFC 3235, IETF, January 2002.
- [22] P. Srisuresh, M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, IETF, August 1999.
- [23] Jiri Kuthan and J. Rosenberg, "Middlebox Communications: Framework and Requirements", IETF Internet draft draft-kuthan-fcp-02.txt, IETF, November 2000.
- [24] R. P. Swale, P. A. Mart, P. Sijben, S. Brim, and M. Shore, "Middlebox Communications (midcom) Protocol Requirements", RFC 3304, IETF, August 2002.
- [25] B. Carpenter and S. Brim, "Middleboxes: Taxonomy and Issues", RFC 3234, IETF, February 2002.
- [26] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy, "STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)", RFC 3489, IETF, March 2003.
- [27] J. Rosenberg, J. Weinberger, R. Mahy, and C. Huitema, "Traversal Using Relay NAT (TURN)". IETF Internet draft draft-rosenberg-midcom-turn-01.txt, IETF, March 2003
- [28] J. Rosenberg, Mahy, Sen, "NAT and Firewall Scenarios and Solutions for SIP", IETF Internet draft draft-rosenberg-sipping-nat-scenarios-00.txt, dynamicsoft, Cisco, Nortel, November 14, 2001.
- [29] S. Davies, S. Read, and P. Cordell, "Traversal of non-Protocol Aware Firewalls & NATs", IETF draft draft-davies-fw-nat-traversal-00.txt, IETF, March 2001
- [30] Sanjoy Sen, Patrick Sollee, and Sean March, "Midcom-unaware NAT/Firewall Traversal", IETF Internet draft draft-sen-midcom-fw-nat-01.txt, IETF, April 2002.

- [31] A. Mishra, M. Shin, and W. A. Arbaugh, "An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process," *ACM Computer Communications Review*, vol. 33, no. 2, pp. 93-102, Apr. 2003.
- [32] H. Schulzrinne and E. Wedlund, "Application-Layer Mobility Using SIP," *ACM Mobile Computing and Communications Review*, vol. 4, no. 3, pp. 47-57, July 2000.
- [33] One-way Transmission Time, ITU-T Recommendation G.114, Feb. 2003.
- [34] R. Droms, "Dynamic Host Configuration Protocol," RFC 2131, IETF, March 1997.
- [35] IEEE 802.1X, "IEEE Standards for Local and Metropolitan Area Networks: Port-Based Network Access Control,"
- [36] IEEE 802.11i, June 2004."
- [37] M. Shin, A. Mishra, and W. Arbaugh, "Improving the Latency of 802.11 Hand-offs Using Neighbor Graphs," *ACM Mobisys*, 2004.
- [38] A. Mishra, M. Shih, and W. Arbaugh, "Context Caching Using Neighbor Graphs for Fast Handoffs in a Wireless Network," *IEEE INFOCOM*, 2004.
- [39] A. Mishra, M. H. Shin, N. L. Petroni Jr., T. C. Clancy and William A. Arbaugh, "Proactive Key Distribution Using Neighbor Graphs," *IEEE Wireless Comm.*, vol. 11, no. 1, pp. 26-36, Feb. 2004.
- [40] K. El Malki et al., "Low latency handoffs in Mobile IPv4," Internet Draft, Aug. 2005.
- [41] H. Fathi and R. Prasad, "Mobility Management for VoIP in 3G Systems: Evaluation of Low-Latency Handoff Schemes," *IEEE Wireless Comm.*, vol. 12, no. 2, pp. 96-104, Apr. 2005.
- [42] C.-C. Tseng, K.-H. Chi, M.-D. Hsieh, and H.-H. Chang, "Location-based Fast Handoff for 802.11 Networks," *IEEE Comm. Letters*, vol. 9, no. 4, pp. 304-306, 2005.
- [43] K. Pahlavan, X. Li, and J.-P. Makela, "Indoor Geolocation Science and Technology," *IEEE Comm.*, vol. 40, no. 2, pp. 112-118, Feb. 2002.
- [44] M. Shin, A. Mishra, and W. A. Arbaugh, "Improving the Latency of 802.11 hand-offs using Neighbor Graphs," in *2004 ACM Mobisys*, Boston, USA, pp. 70-83, 2004.
- [45] J. C.-S. Wu, C.-W. Cheng, N.-F. Huang, and G.-K. Ma, "Intelligent Handoff for Mobile Wireless Internet," *Mobile Networks and Applications*, vol. 6, pp. 67-79, 2001.
- [46] C. Perkins, "IP Mobility Support," IETF RFC 2002, Oct. 1996.
- [47] A. Dutta, F. Vakil, J.-C. Chen, M. Tauil, S. Baba, and N. Nakajima, "Application Layer Mobility Management Scheme for Wireless Internet," in *2001 IEEE 3G Wireless'01*, San Francisco, pp. 7, 2001.
- [48] T. T. Kwon, M. Gerla, Sajal Das, and Subir Das, "Mobility Management for VoIP Service:



- Mobile IP vs. SIP," IEEE Wireless Communications Magazine, vol.9, no.5, pp. 66–75, Oct. 2002.
- [49] Z. R. Turányi, C. Szábó, E. Kail, and A. G. Valkó, "Global Internet Roaming with ROAMIP," ACM Mobile Computing and Communications Review, vol. 4, no. 3, pp. 58-68, July 2000.
- [50] H. Yokota, "Link Layer Assisted Mobile IP Fast Handoff Method over Wireless LAN Networks," in 2002 Mobicom, Atlanta, USA, pp. 131-139, 2002.
- [51] E. Shim, H.-Y. Wei, Y. Chang, and R. D. Gitlin, "Low Latency Handoff for Wireless IP QOS with NeighborCasting," in 2002 IEEE Conf. on Communications, ICC02, pp. 3245-3249, 2002.
- [52] L. Ong and J. Yoakum, "An Introduction to the Stream Control Transmission Protocol (SCTP)," RFC 3286, IETF, May 2002.
- [53] M. Riegel and M. Tuexen, "Mobile SCTP," Internet draft draft-riegel-tuexen-mobile-sctp-02.txt, IETF, Feb. 2003.
- [54] J. Chan, B. Landfeldt, A. Seneviratne and P. Sookavatana, "Integrating Mobility Prediction and Resource Pre-allocation into a Home-Proxy Based Wireless Internet Framework," in 2000 IEEE conf. on Networks, ICON2000, Singapore, pp. 18-23, 2000.
- [55] V. Madisetti and A. Argyriou, "Voice & Video over mobile IP Networks," Internet draft draft-madisetti-argyriou-voice-video-mip-00.txt, IETF, May 2002.
- [56] EIA/TIA, "Cellular radio-telecommunication intersystem operation: Automatic roaming," BEIA/TIA Technical Report IS-41.3, 1991.
- [57] ETSI/TC, "Restoration procedures, version 4.2.0," ETSI Technical Report Recommendation GSM 03.07, 1993.
- [58] Z. J. Haas and Y.-B. Lin, "On the optimizing the location update costs in the presence of database failures," ACM-Baltzer Wireless Networks, no. 4, pp 419-426, August 1998.
- [59] Y. Fang, I. Chlamtac and H. B. Fei, "An active mobility database failure recovery scheme and performance analysis for PCS networks," in: Conference of the IEEE Communications Society, pp. 757-764, March 2000.
- [60] Y. Fang, I. Chlamtac and H. B. Fei, "Analytical results for optimal choice of location update interval for mobility database failure restoration in PCS networks," IEEE Transactions on Parallel and Distributed Systems, no 11, pp. 615-624, June 2000.
- [61] Z. J. Haas and Y.-B. Lin, "Demand re-registration for PCS database restoration," in: Proceedings Military Communications Conference, no 2, pp. 887-892, October-November 1999.



- [62] Y.-B. Lin and P. Lin, "Performance modeling of location tracking systems," *ACM Mobile Computing and Communications Review* 2, pp. 24-27, July-August 1998.
- [63] J. W. Young, "A first order approximation to the optimum checkpoint interval," *ACM communications*, no 17, pp. 530-531, September 1974.
- [64] E. Gelenbe, "On the optimum checkpoint interval," *Journal of the Association for Computing Machinery*, no 26, pp. 259-270, April 1979.
- [65] J. S. Plank and W.R. Elwasif, "Experimental assessment of workstation failures and their impact on checkpointing Systems," in: *28th International Symposium on Fault-Tolerant Computing*, pp. 48-57, June 1998.
- [66] J. S. Plank, K. Li and M. A. Puening, "Diskless Checkpointing," *IEEE Transactions on Parallel and Distributed Systems*, no. 9, pp. 972-986, October 1998.
- [68] Y.-B. Lin, Failure restoration of mobility database for personal communication networks, *ACM-Baltzer Wireless Networks*, no. 1, pp. 365-372, March 1995.
- [69] T.-P. Wang, C.-C. Tseng and W.-K. Chou, An aggressive approach to failure restoration of PCS mobility databases, *ACM Mobile Computing and Communications Review* 1, pp. 21-28, September 1997.
- [70] Y.-B. Lin, Per-user checkpointing for mobility database failure restoration, Accepted and to appear in *IEEE Transactions on Mobile Computing*.
- [71] G. Varghese and A. Lauck, Hashed and hierarchical timing wheels: Efficient data structures for implementing a timer facility, *IEEE/ACM Transactions on Networking*, no. 5, pp. 824-834, December 1997.
- [72] Industrial Technology Research Institute, <http://www.ccl.itri.org.tw>.

2.5 cm

畢業  
民國  
年度

1 cm

2.5 cm

94

博士論文

VoIP

聚合環境之行動性與互用性的支援

論文  
題目



校院  
所名

3 cm

1 cm

國立交通大學資訊學系 張弘鑫

94

博士論文

VoIP

聚合環境之行動性與互用性的支援

國立交通大學資訊學系 張弘鑫