國立交通大學

資訊科學與工程研究所

博 士 論 文

無線網狀網路之頻道分配與認證機制

Channel Allocation and Authentication Schemes for Wireless Mesh Networks

研究生：史永健

指導教授：曾建超 教授　　曹孝櫟 教授

中 華 民 國 一 ○ ○ 年 一 月

Channel Allocation and Authentication Schemes for Wireless Mesh Networks

Student: Yung-Chien Shih

Advisors: Dr. Chien-Chao Tseng and Dr. Shiao-Li Tsao

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

Hsinchu, Taiwan, Republic of China

January 2011

Abstract in Chinese

無線網狀網路之頻道分配與認證機制

學生：史永健　　　　　　　　　　　　　　　　　　指導教授：曾建超 博士

曹孝櫟 博士

國立交通大學 資訊科學與工程研究所 博士班

摘　　要

無線網狀網路(wireless mesh network)是一種無需實體電纜連接的網路架構，因此具有低成本與快速佈署等優勢。隨著應用需求的日趨成長，如何在此網路下提供安全通訊(secure communication)與支援快速換手(fast handoff)為迫切面對的問題。針對這個問題，我們提出將IEEE 802.1X認證者(authenticator)角色移往網狀網路入口(mesh portal)擔任的新安全架構，藉以建立行動節點(mobile station)與網狀網路入口間的端點對端點安全通道(end-to-end secure channel)。藉由我們所提出的架構，在不失去安全性的前提下，行動節點利用快速認證機制可以在漫遊時快速的與網路端互相認證，並且同時建立安全連線，而不需要一再的執行IEEE 802.1X認證與金鑰分配流程，因而可以有效降低行動節點換手時的認證延遲。

多躍步(multi-hop)無線網狀網路的另一個嚴峻挑戰就是必須克服頻道干擾問題(co-channel interference)。既使隨著無線技術的進步而使得無線介面(radio interface)的資料傳輸率(data bit rate)不斷提昇，但仍可能因干擾問題而使得整

體網路的流量(throughput)無法獲得相對應的提昇。這個問題同時也將造成無線網狀網路會有不可預期的傳輸延遲，成為快速換手機制的不安定因素。針對此問題，過去已有相當多研究提出利用分配頻帶不重疊的頻道(non-overlapping channels)來降低干擾與利用多無線介面的架構來提昇整體網路流量。然而考慮到無線網狀網路的應用特性，頻道分配(channel allocation)機制應該要同時考量到端點對端點的傳輸與同網域(intra-mesh)及跨網域(inter-mesh)通訊並存的情況，這也是過去研究所忽略的問題。因此，我們提出一套基於頻道與時間切割(radio-frequency-slot)的端點對端點的頻道分配機制，除了使得任意端點對端點傳輸路徑本身的多躍步之間都可以避免干擾外，不同傳輸路徑之間的干擾也可以被避免。雖然分離封包(packet)傳輸的接收與發送到不同的頻道上進行並非是一個新的發現，但是我們觀察到若是導入此概念至我們所提出的機制中，將可提升頻道的再利用率並且使我們的機制得以應用在多網路介面架構的無線網狀網路。接著，進一步配合適用於此機制的路徑選擇(route selection)方法，使得每一路徑的傳輸流量與延遲得以被維持，且有助於提昇整體網路的流量。

最後，為了讓行動節點能即時感知到網路環境的變化，因而能適時執行快速換手機制，我們提出一套跨網路協定階層(cross-layer)的互動機制與中介平台(middleware platform)。基於此平台，其上層的應用程式可透過程式開發介面(application programming interfaces)使用跨階層訊息交換機制(cross-layer signaling mechanism)來取得下層的網路狀態及通知下層改變網路連接點等。應用程式同樣可透過此介面使用事件通知機制(event notification mechanism)來即時感知其關注的網路變化。

透過端點對端點的安全通訊架構，在不失安全性前題下有效改進行動節點換手時的認證延遲。端點對端點的頻道分配機制可提昇網路流量且避免非預期傳輸延遲的發生，也連帶確保了行動節點換手時的訊息交換延遲。跨網路協定階層互動機制與中介平台則讓行動節點有能力即時感知網路變化並適時進行換手程序。結合前述三項研究，我們提供一套可在無線網狀網路下支援快速換手的解決方案。

關鍵詞： 無線網狀網路、端點對端點、快速換手、認證機制、頻道干擾問題、頻道分配機制、跨網路協定階層、中介平台。

Abstract

Channel Allocation and Authentication Schemes for Wireless Mesh Networks

Student: Yung-Chien Shih

Advisor: Dr. Chien-Chao Tseng
Dr. Shiao-Li Tsao

Department of Computer Science
National Chiao Tung University

ABSTRACT

While wireless mesh networks (WMNs) are gaining momentum in widespread application, we are concerned with fast handoff in a secure mesh environment. To this end, we present a means in the context of IEEE 802.11s by allowing a mesh portal to act as an IEEE 802.1X authenticator, to reduce costly IEEE 802.1X authentications during handoff. As the mesh portal (MPP) engages in IEEE 802.1X authentication and cryptographic key management, our scheme maintains an end-to-end secure channel between a mobile station and the MPP wherever the station roams in the network. Therefore, without compromising required robust security, IEEE 802.1X authentication can be bypassed during handoff to reduce overall delay in an approach suggested for moderately sized networks.

A WMN suffers from a co-channel interference problem when mesh nodes share the same wireless access channels. Therefore, the overall throughput of WMNs may not be able to increase substantially even with broadband physical layer technologies. The problem also causes unexpected transmission delays in the network, which could seriously influence the process of authentications. As a remedy, we propose an end-to-end channel allocation scheme, extending the radio-frequency-slot method and providing stable throughput for end-to-end packet delivery in WMNs. Although separating transmissions of data and acknowledgment packets on two different channels is not our new finding, we observe that the non-overlapping channels can be efficiently reused if the concept is introduced into our scheme. Moreover, by applying link and path metrics for route selection, the end-to-end throughput and delay can be maintained, and the overall throughput of WMNs can be improved.

For fast handoff, a mobile station should be able to detect immediately the changes of a network environment, such that the station can perform handoff process at correct time. To this end, we designed and implemented a middleware platform, providing application programming interfaces (APIs) for upper applications to use cross-layer signaling and event notification mechanisms. The applications can configure and acquire status of underlying protocol stack via the cross-layer signaling mechanism, and can immediately detect changes of a network environment via the event notification mechanism.

Keywords: Wireless mesh network, end-to-end, fast handoff, authentication, interference problem, channel allocation mechanism, cross-layer, middleware.

Acknowledgment

發自內心深深覺得，今日之所以能順利取得博士學位，實是有衆人們不斷的支持與鼓勵才得以達成；感謝這一路走來所經歷的種種人事物。

首先感謝我的父母對於我的包容與支持，即使像我這樣一個孩子三十好幾仍未能負擔家計，也無半點苛責；即使一直離家在外不能隨侍左右，也極盡包容。感謝我的兄姐們，代替我全力承擔家計，因而給予我機會追尋與挑戰自我，我一生的成就皆受家人所賜。

特別感謝我的指導教授：曾建超教授與曹孝櫟教授。曾教授除了提供我良好的學習環境並給予生活上的支持，並且不斷在研究上給予殷切的指導，尤其在我數度迷失自我時將我及時導正。與曹教授合作期間，在英文寫作上受益匪淺，從中學到學術研究應有的嚴謹與態度，體認自己所不足而需更加的努力。感謝教授們的指導，這段期間讓您們多所費心了。

同時要感謝許健平教授、鄭瑞光教授、曾煜棋教授與簡榮宏教授，教授們針對我的論文提出非常精闢的見解與建議，使得我的研究得順利進行並且有了好的結果。特別感謝我敬重的學長紀光輝教授，從學長那不只學習到學術研究應有的態度，我更深受學長那待人接物的謙和包容所折服。

楊人順學長、徐元瑛學姐、王瑞堂學長、何承遠學長，感謝有你/妳們一路相挺，我們都了解博士班的過程就是一場耐力的考驗，謝謝有你/妳們不時的加油打氣，在我需要幫助時適時提供協助，你/妳們是我的良師益友。

另外，我必須感謝這一路陪我走來的學弟妹們，特別是黃貴笠、范榮軒、何承運、何姿欣等(仍有很多同甘共苦的學弟妹不及備載，但我絕沒有忘了你/妳們)，感謝你/妳們在這段期間的情義相挺與陪伴。當然不能忘記的還有系辦助理吳畹鳳(吳姐)、李倖禎、楊耀萱、馮詠喬，感謝妳們一直以來在各種行政事務上的協

助與通融，因為有妳們，讓我在取得學位的過程中更為順利。

最後，感謝上天讓我有這個機會經歷並且完成這個博士學位，讓我知道世界的寬廣與自己的渺小，今後我會不斷的自我反省與自我警惕，以謙和的態度對待人們，以卑微的心理來面對浩瀚無涯的學海。

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

IEEE 802.11s specifies how IEEE 802.11 devices are interconnected for mesh networking [3, 8, 13]. A wireless mesh network (WMN) does not necessitate cabling, as opposed to a typical architecture where stations communicate via access points (APs) attached to a wired medium. This new type of network architecture facilitates rapid deployment and is evolving as a vital means of public access to Internet services. However, a WMN still suffers from three serious hindrances when a station moves in the network, as described below.

1. A handoff process occurs when a station moves its association from one access point to another, causing a blackout period of communication disruption. Handoff involves AP discovery, authentication, reassociation establishment, and inter-AP transfer of physical connectivity or credential information specific to the mobile station. Access point discovery by the station identifies APs within range. The authentication procedure refers to a legacy open-system and IEEE 802.1X authentications [9]. For user authentication and keying material distribution, the IEEE 802.1X framework has been adopted as a mandatory part of Robust Security Networks (IEEE 802.11i [11].) With IEEE 802.1X transactions at a remote site, internetwork operations account

for another potentially prohibitive delay. As far as secure communication is concerned, however, we should avail ourselves of IEEE 802.11i mechanisms to the greatest extent possible.

2. A WMN suffers from the co-channel interference problem [27] when mesh nodes share the same wireless access channels. Therefore, the overall throughput of WMNs may not be able to increase substantially even with broadband physical layer (PHY) technologies. The problem also causes unexpected delay when a station is performing a handoff process. To reduce co-channel interference, a number of solutions have been proposed to allocate multiple non-overlapping channels to mesh nodes. Unfortunately, most previous studies have considered co-channel interference between neighboring nodes, maximizing the total WMN throughput, but failing to address end-to-end issue of a WMN. End-to-end factors, such as delay and throughput, play a key role for real-time communication services, such as Voice over IP (VoIP) over WMNs, and should be carefully investigated. Previous studies, such as Hyacinth [45] and ROMA [51], have considered the end-to-end issue and have been aware of path loading. They have aimed to minimize intra-path interference (see Section 2.4 for categories of interference problem) in paths between mesh access points (MAPs) and a gateway. However, paths could be established between any mesh nodes within an intra-mesh domain, and these paths have also played a key role for packets forwarding and bi-casting as stations move around in a domain (also causing unexpected delays due to inter-path interference). To provide stable throughput and support fast handoff in a WMN, we require a generic solution to allocate channels for any paths without intra- and inter-path interference.

3. Furthermore, to achieve fast handoff in a WMN, another challenge is how a mobile station can immediately detect the changes of a network environment. An application running on a mobile station may visit different networks and encounter changes in bandwidth, delays, or IP addresses. To contend with

network fluctuations, network-aware applications, which can adapt themselves to changes in a network environment, may need to periodically issue system calls to retrieve lower-layer status, such as IP addresses, entries of routing tables, and connectivity of network adaptors. However, network status normally do not change frequently. Periodic system calls within short intervals may waste system resources for obtaining the same information. Conversely, with long intervals between system calls, the applications cannot react promptly to the changes of a network environment. Additionally, a mobile station with multiple network interfaces requires a mobility manager to monitor status of network adaptors and perform handoff decisions accordingly. To acquire the status and conduct a handoff, the mobility manager must use system calls to communicate with underlying network protocol stack, such as a link layer, network layer, or transport layer. However, the use of system calls is not only error-prone nor portable.

1.2 Main Contribution

In this dissertation, we propose an avenue to streamline secure communication and an approach to end-to-end channel allocation scheme for WMNs. To complete our fast handoff scheme, we also propose a cross-layer signaling and middleware platform for multi-interface mobile stations. With the integration of the three studies, WMNs can support fast handoff and provide stable throughput as mobile stations move in the network.

Our proposed secure communication scheme distinguishes itself from previous work in the following aspects:

- The security level in line with IEEE 802.11i is well maintained, without compromising required robust security.
- We avoid distributing sensitive keying material, such as pairwise master keys (see Section 2.2), over the wireless medium and thereby rule out undue access

from the air.

- IEEE 802.1X authentication can be bypassed during handoff to reduce overall delay in a way suggested for moderately sized networks.

- Current protocols at the station side remain operable without any changes, allowing backward and forward compatibility.

- Our approach is interoperable with the emerging IEEE 802.11s standard, indicating its usefulness in practical applications.

- Our design elegantly lends itself to generic multi-hop wireless networks, in particular clustered networks accommodating IEEE 802.1X.

- The proposed channel allocation scheme extends a radio-slot-frequency method and is aware of end-to-end path loading in WMNs. Based on a radio-slot-frequency method, our scheme can divide transmission time into several slots for multiple access and then allocate non-overlapping channels for these slots to prevent interference. Unfortunately, the original design is not feasible to be applied on multi-radio WMNs, because the limited non-overlapping channels could be quickly exhausted (see discussion in Section 4.1). Although separating transmissions of data and acknowledgment packets on two different channels is not a new observation, we find that the non-overlapping channels can be efficiently reused if the concept is introduced into our scheme. Consequently, we can model the channel allocation problem as a directed graph and derive a solution via set operations. We also extend previous link and path metrics (see discussion in Section 4.2.4) to measure path loads in our scheme. By the integration of our allocation scheme and extended metrics, the channel allocation for an end-to-end communication path can be easily determined and the throughput of the path can also be maintained. The results demonstrate that the proposed mechanism improves the channel utilization and achieves a more effective end-to-end throughput than other approaches.

- We also propose a cross-layer signaling and middleware platform, which is a software platform for the development of network-aware applications. The platform adopts a middleware [62] approach and provides application programming interfaces (APIs) for the application to use cross-layer signaling mechanism. Based on this mechanism, an application can interact with lower-layer network protocols to acquire network status and manage network interfaces. Moreover, the platform also provides an event notification mechanism for an application to register interested events of network changes so that the middleware can notify the application immediately when an event of interest occurs.

1.3 Synopsis

The remainder of this dissertation is organized as follows. Chapter 2 introduces related works. Chapter 3 presents our fast handoff approach in secure IEEE 802.11s mesh networks. Chapter 4 describes an end-to-end channel allocation scheme for wireless mesh networks. We then describe the proposed cross-layer signaling and middleware platform for multi-interface mobile devices in Chapter 5. Finally, we conclude this dissertation in Chapter 6.

Chapter 2

Background and Related Work

2.1 Mesh Networking Security Architecture

As shown in Figure 2.1, a mesh network comprises a mesh security domain and AP security domains. A security domain refers to a set of network entities on which a same security policy is exercised under a single administrative authority [23]. The mesh security domain covers mesh points (MPs) connecting to a mesh portal (MPP), while an AP security domain encompasses a mesh access point (MAP) and its local stations. (A MAP is an MP providing additional AP functionality.) Observe that current policies adopted in these security domains are different; links among MPs are protected by IEEE 802.11s, whereas connectivity between a station and its local MAP is protected by IEEE 802.11i mechanisms.

Whenever a mobile station switches its association to a new MAP, IEEE 802.1X requires the station and an Authentication Server situated its home network to authenticate each other. IEEE 802.1X authentication involves mostly multiple rounds of message exchanges through the Internet, at the expense of nontrivial delay. For this, a number of fast handoff schemes have been developed, e.g., [9, 17–20, 22, 24] (see [21] for an expository survey). However, these schemes did not take mesh infrastructure into account.

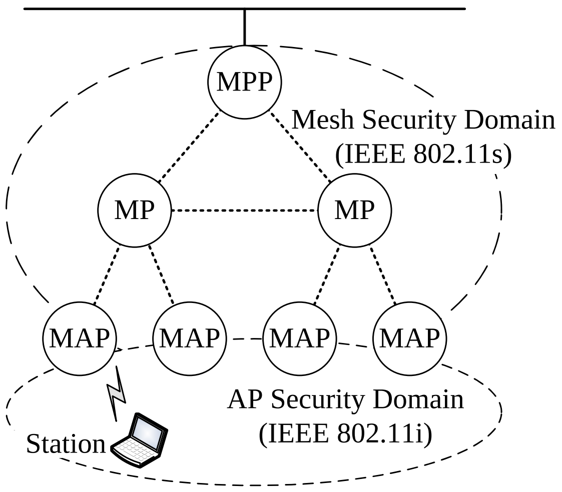


Figure 2.1: A mesh networking security architecture.

2.2 Security Association

A security association refers to the establishment of shared security information like credentials or cryptographic keys between two network entities to support secure communication. Within an AP security domain IEEE 802.11i mechanisms, namely IEEE 802.1X authentication, 4-Way Handshake and encryption protocols (TKIP or CCMP), are used. These mechanisms are collectively meant to establish a security association between a station and its MAP.

For access control, as shown in Figure 2.2, IEEE 802.1X authentication starts with an EAPOL-Start (Extensible Authentication Protocol over LAN) frame by the supplicant station or an EAP-Request/Identity packet by the associated MAP. In response, the station provides its identity using an EAP-Response/Identity packet, which is then encapsulated in a RADIUS-Access-Request message addressed to the backend Authentication Server. Following are challenge-response interactions between the station and the Authentication Server in several round trips, depending on what EAP method in use [1]. After such EAP interactions have terminated successfully, these two parties generate a common master session key (MSK) for authorization purpose. The Authentication Server then sends the MSK to the serving MAP in a RADIUS-Access-Accept packet for authorization. Upon reception, the

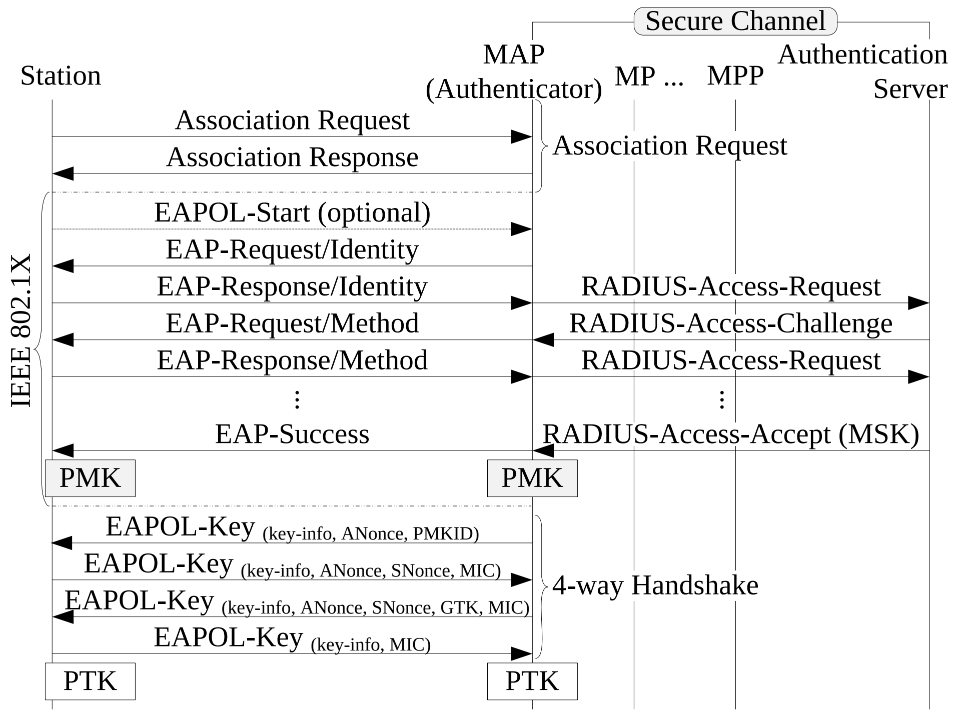


Figure 2.2: Message flow to and from an AP security domain.

IEEE 802.1X authentication process is completed when an EAP-Success packet is generated toward the supplicant station, whereby a pairwise master key (PMK) is derived as stipulated by IEEE 802.11i.

After successful IEEE 802.1X authentication, 4-Way Handshake takes place (Figure 2.2.) This procedure involving four EAPOL-Key frames is used by both the station and its local MAP to confirm the possession of a correct PMK and to derive a PTK (pairwise transient key.) 4-Way Handshake is initiated by the MAP sending the first EAPOL-Key frame to the station, containing a randomly chosen number ANonce and an identifier PMKID indicative of which PMK shall be used. Then the station derives a fresh PTK and sends back a frame containing SNonce (another random number selected by the station), certain information elements, and a message integrity code (MIC.) Subsequently the MAP derives a PTK identical to that on the station side and checks the integrity of the received frame. If valid, the MAP acknowledges the station using the third EAPOL-Key frame that may include a group temporal key (GTK.) Finally the station responds to the MAP using the fourth frame to terminate the handshake, whereupon a security association is established between these two sides.

Apart from IEEE 802.11i, IEEE 802.11s defines efficient mesh security association (EMSA) [13], involving EMSA authentication, 4-Way Handshake, key distribution, and encryption protocols. While the IEEE 802.1X framework is still adopted, an MP can act as both a supplicant and authenticator. Each MP authenticates its neighboring MP, establishes PTK, and sends GTK through key distribution and authentication processes. It is noted that key distribution includes the delivery of PMKs for mesh authenticators, termed PMK-MAs, from some mesh key distributors. Based on a PMK-MA, an authenticator MP and its supplicant MP mutually derives a PTK. EMSA allows multiple MPs to instantiate security association without invoking IEEE 802.1X authentication. In this study EMSA is honored. We aim to augment AP security domains so that our development can cooperate additionally with well-defined mechanisms such as EMSA in the mesh security domain.

2.3 Standard Fast Handoff Mechanisms

As stated, a handoff process involves AP discovery, commit phase (legacy open-system authentication and reassociation), IEEE 802.1X authentication, and 4-Way Handshake. Among others, the study in [5] indicates that IEEE 802.1X authentication accounts for most handoff delay (75% to 95%). In view of this major determinant, we shall leverage IEEE 802.11i preauthentication mechanism to let a station perform IEEE 802.1X authentication with a target AP beforehand. Essentially such preauthentication operates in the same way as IEEE 802.1X authentication prescribed in Section 2.2, except that all EAPOL frames are forwarded via the station's current MAP as an intermediary. After successful preauthentication, a newly derived PMK is cached at the station and the target MAP for future use. When reassociating with the target MAP later, the station provides PMKID in its Association Request frame to signify that a corresponding PMK has been cached. If the provided PMKID is valid, the target MAP can save itself IEEE 802.1X authentication and carry out 4-Way Handshake straightaway.

We remark that IEEE 802.11r is another standard specifying mechanisms to speed transitions of a station among APs within a *mobility domain*, a managed set of APs sharing security associations [12]. IEEE 802.11r optimizes message exchanges and separates IEEE 802.1X authentication from network access control during each handoff. However, a means of target AP prediction is required somehow. For backward compatibility with a vast set of end-user equipment, we propose an alternative approach to secure fast handoff in mesh networks[1]. Neither MSK nor PMK is transmitted over wireless media. In alignment with standard IEEE 802.11 machinery, our approach keeps current protocols at the station side operable without change, as shall be seen in Chapter 3.

2.4 Interference Problem

Figure 2.3 reveals two types of co-channel interference problems in an end-to-end communication paradigm. In this figure, there are 12 nodes, denoted as *node-0* to *node-11*, and two routing paths, named *path-1* and *path-2*. *Node-2* and *node-4* are in the communication range of *node-3*, and hence cannot transmit or receive packets when *node-3* is sending packets. In addition, *node-1* and *node-5* are also affected if they want to send packets to *node-2* and *node-4*. This situation is called *intra-path interference*. Moreover, *node-9* is in the communication range of *node-3* and cannot transmit or receive packets from *node-8* and *node-10*. Those nodes in other communication path suffer from *inter-path interference*.

[1]A number of studies have shown that target AP prediction techniques are of use to reduce handoff delay. Such techniques can be programmed in IEEE 802.11 devices via firmware updates or controlled by user space software. In comparison, we take another avenue to achieve similar objectives by restructuring the IEEE 802.1X framework, making our scheme transparent to end users and independent of AP prediction. However, our scheme can combine with target AP prediction techniques for secure fast handoff support. Our scheme is indeed complementary to these techniques.

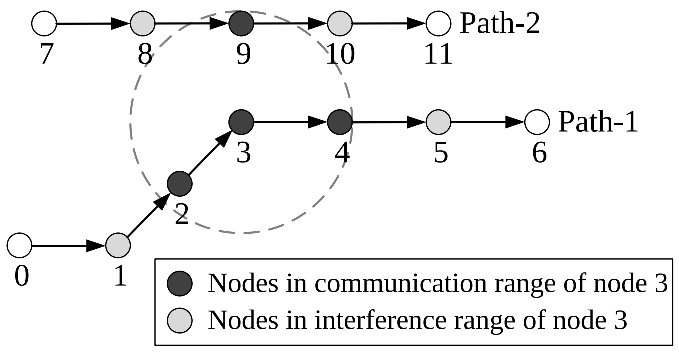

Path-2
7
8
9
10
11
Path-1
3
4
5
6
2
0
1
● Nodes in communication range of node 3
○ Nodes in interference range of node 3

Figure 2.3: The intra-path and inter-path interference problems in multi-hop WMNs

2.5 Channel Allocation Mechanisms

Most recent works in channel allocation for multi-radio mesh networks tend to jointly solve routing problem. Centralized solutions [44, 54–59] aim to find the best combination of routes, channel assignment and transmission schedules given the network topology on all channels and traffic pattern. However, these optimization algorithms lack practical solution for coordinating topology measurement and disseminating channel assignments.

In [54,55], both studies assume that the knowledge of the traffic demands is available and that the system operates synchronously in a time slotted mode. A Linear Programming (LP) is formulated in [54] to route the given traffic demands in order to maximize the system throughput subject to fairness constraints. Constraints on the number of radios and on the sum of the flow rates for the links in the interference range are also included. Because the resulting formulation includes integer variables which make the problem NP-hard, the authors solve the LP relaxation of the problem. The result is a network flow along with a possibly unfeasible channel assignment. The proposed channel assignment algorithm then serves the purpose to adjust the flow on the graph to ensure a feasible channel assignment. The flow on the graph is further re-adjusted by a post processing and a flow scaling steps. Finally, a scheduling algorithm produces an interference free link schedule.

In [55], the traffic demands are formulated as a multi-commodity flow problem, where one among several different objectives can be defined. Besides including the same constraints as in [54], the LP formulation in this work makes use of time-indexed variables, hence solving such LP gives a solution for the entire channel assignment, routing and scheduling. The presence of integer variables makes the problem NP-hard and thus, the authors solve the LP relaxation of the problem. Then a channel assignment along with scheduling based on greedy coloring is used to resolve the potential conflicts. The authors present both static and dynamic algorithms to assign distinct channels to a link for different time slots.

To address the practical coordinating protocol, a number of studies in distributed allocation algorithm for multi-radio WMNs use identical channels on all nodes [38–40], and few works dynamically allocate channels in distributed fashion, which is a hard challenge.

SSCH [33] is a very first solution with radio-frequency-slot and closest in spirit to our EECAS. SSCH implements channel hopping schedule and schedules packets within each channel by using the seed knowledge. Nodes with SSCH transmit the channel hopping schedule to neighboring nodes and update the schedule to adapt to changing traffic patterns. SSCH is a distributed protocol, like EECAS, for coordinating channel switching decisions. However, SSCH schedules channel hopping for a node to communicate with its all neighboring nodes that could cause redundant allocations due to missing the end-to-end issue in WMNs. Applying the radio-frequency-slot method on multi-radio WMNs is an issue due to the waste of limited non-overlapping channels. One difference between EECAS and SSCH is that EECAS separates data and acknowledgement packets of transmissions to increase the efficiency of channel reuse so that the radio-frequency-slot concept can be applied to multi-radio WMNs. Moreover, EECAS allocates channels for end-to-end paths on demand, which reduces channel waste on idle links.

In [42], the authors propose a distributed channel assignment protocol. One interface of each node is dedicated to a default channel common to all nodes, and the protocol relies on the common channel across the network to ensure connectivity. Each node runs the distributed assignment protocol to select channels for its other radios and prefers channels that are least used by its interfering neighbors.

This study uses the MR-LQSR protocol [38] with the WECTT [38] metric as its routing protocol. In comparison, EECAS also reserves a common channel for control messages exchanging but does not need a dedicated radio. Moreover, EECAS is aware of channel usage within an interfered range so that EECAS can allocate channels without interference and can use channels more efficiently to achieve higher performance in throughput.

Hyacinth [45] is a distributed load-aware algorithm for channel assignment. The solution explicitly builds a spanning tree rooted at each gateway node. Each node separates own radios into two groups and then independently chooses channels for radios in one group that is dedicated to communicate with its children. The assignment of channels changes on a per-flow basis as load conditions in the network changes. Note that this requirement is challenging to realize in a large-scale mesh because of the complexity involved in assigning channels to radios. Moreover, the solution is designed for WMNs specifically used for the wireless Internet access applications, and their algorithm works only for gateways whose connectivity graph is a tree.

ROMA [51], like Hyacinth, does not rely on a common channel to keep the network connected and optimizes channels assignments along routes between mesh nodes and a few gateways. One fundamental difference between ROMA and Hyacinth solutions is that ROMA has considered link losses and loss fluctuations. Moreover, ROMA uses ETT [38, 41] metric for link measurement and extends SIM [39] metric for routes selection such that ROMA is aware of channel-diverse for an end-to-end path. However, both the two studies do not consider the influence of intra-mesh communications. These communications build short-cut routes between nodes within an intra-mesh domain, and the short-cut route causes unexpected inter-path interference, which is one of most important factors affecting throughput.

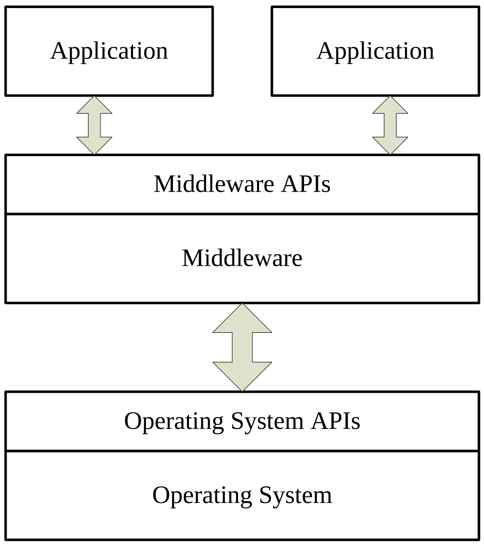


Figure 2.4: An operation model of middleware.

2.6 Middleware

Many researchers proposed middleware approaches to support fast handoff in heterogeneous network environment [65–70]. In the research, the middleware, provide interaction mechanisms among applications and lower-layer protocols. Accordingly, the middleware is situated between applications and the operating system (OS) as shown in Figure 2.4. It hides the complexity of error-prone system calls, OS APIs, from applications by encapsulating the complex system calls in simple middleware APIs that provide high level functions for applications. Therefore, applications need not handle the connectivity and heterogeneity of underlying networks. Further, they can manage the networks and react to the changes of networks accordingly through middleware APIs.

There are three benefits to develop applications via the middleware: easy porting, quick development and error avoidance. For application porting, because developing an application via middleware APIs can hide the OS heterogeneity, programmers need not modify any codes when transplanting applications onto another OS. For example, Java Virtual Machine (JVM) is one well-known embodiment implementation of the middleware concept. It allows the same Jave code to run on different

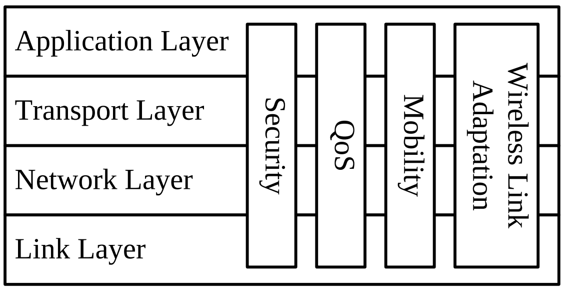


Figure 2.5: The architecture of a cross-layer notification mechanism.

OSs. As for application development and error avoidance, middleware APIs provide simple interfaces for programmers to acquire and configure status of lower-layer protocols, so programmers need not care for the details of system calls. Consequently, programmers can develop applications quickly and correctly.

2.7 Cross-Layer Design

According to [71], traditional network protocols used on mobile devices do not operate efficient enough because protocol stack are working independently, without knowledge of status of other stack. As a remedy, the cross layer feedback method proposed in [72, 73] provides an interaction mechanism for a protocol of a layer to share status with one of another layer. For example, a link layer protocol can share status with an application layer protocol and thus applications can adjust transmission rate according to the link status.

The cross-layer notification mechanism in [74] suggested that a network protocol need a means to inform other protocols about changes of network status. Therefore, cross-layer architecture, shown in Figure 2.5, was proposed to satisfy needs of various applications, such as security, QoS or mobility requirement. For example of wireless link adaptation, when link quality of currently attached network changes, the link layer will notify an application layer program of the link quality of each wireless adaptor or the transport layer of the maximum transmission rate. In response, applications can inform the link layer to switch wireless adaptor.

2.8 Driver-level Network Event Notification

The driver-level network event notification mechanism has been mentioned in [75]. This mechanism adopted a signaling mechanism to notify applications implemented in user space. Furthermore, it also modified the OS scheduling algorithm to eliminate the needs for an application to issue system calls periodically to retrieve low-layer status. Implementation of the mechanism includes three major parts: (1) event notification, (2) process management, and (3) scheduler. In our implementation, before a process invocation in the kernel space returns to the user space, the mechanism will check whether the registered events of this process occur or not. If so the registered event occurs, the mechanism will call the corresponding procedure firstly. In this dissertation, we adopted our proposed event notification mechanism and then modified some algorithms and data structures to satisfy requirements of our goal.

Chapter 3

Fast Handoff in Secure IEEE 802.11s Mesh Networks

This study deals with roaming in an IEEE 802.11s mesh network while maintaining secure communication as per IEEE 802.11i. Indeed, our approach is complementary to, but not competing with, IEEE 802.11s-based schemes nor is developed in place of the ritual of IEEE 802.11s. In our architecture an alternative end-to-end secure communication channel between a mobile station and its MPP is ensured such that repeated encryptions and decryptions in frame delivery can be saved substantially. We stress that authentications and secure messaging for mobile stations are of concern to this study, but otherwise we exploit IEEE 802.11s mechanisms like MP authentication or secure transfer of signaling messages in the mesh infrastructure where available.

3.1 The Proposed Approach

We extend AP security domains to a mesh network (Figure 3.1), so as to reduce repeated IEEE 802.1X authentications and encryption/decryption processing of data frames for mobile stations. In our architecture, a station performs IEEE 802.1X authentication only when (re)associated with a MAP in the network for the first time. The role of authenticator is now shifted from MAP to MPP. As an MPP

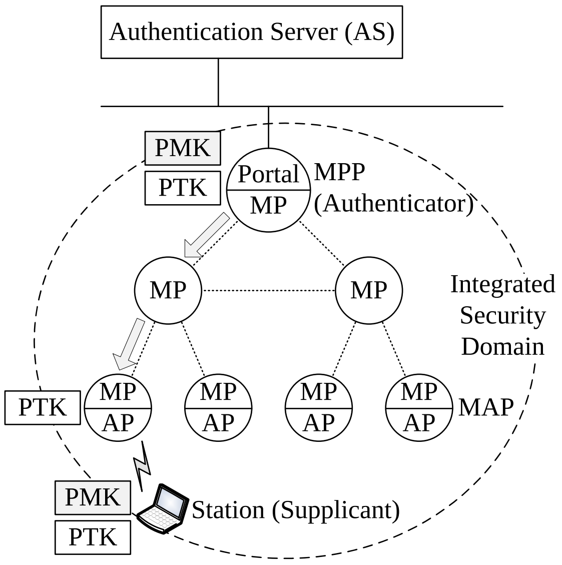


Figure 3.1: Extending AP security domains to a mesh network environment.

partakes in IEEE 802.1X authentication and cryptographic key management, we maintain an end-to-end secure channel between the station and the MPP wherever the station roams in the network. MAPs at the edge of a mesh network remain responsible for blocking unauthenticated stations from access to the system. In order to enable a MAP to verify frame integrity, PTK and GTK are distributed from the MPP to the serving MAP via secure mesh links immediately after 4-Way Handshake (see also Figure 3.2.)

3.1.1 Security Association Establishment

When a station is initially (re)associated to any MAP managed by an MPP, it performs IEEE 802.1X authentication and 4-Way Handshake to create the security association with the MPP as per IEEE 802.11i. In our architecture, however, since the MPP is *defacto* an authenticator, the serving MAP acts as a transit node that forwards IEEE 802.1X traffic between the station and the MPP, as depicted in Figure 3.2 in following lines:

1. The serving MAP examines whether a valid PMKID is included in the (Re)Association Request frame by the station. If not, we let the MAP generate a designated

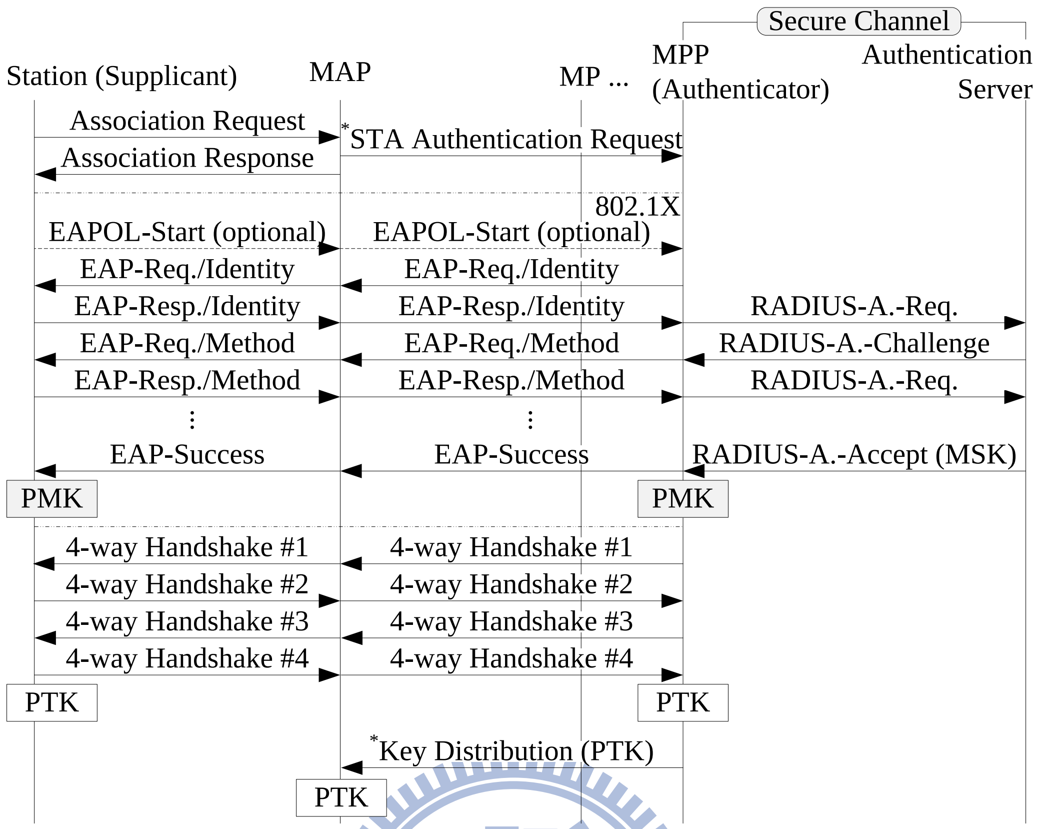


Figure 3.2: Message flow of establishing a security association in our architecture. Messages marked with '*' represent our addenda.

message named STA Authentication Request toward the MPP to initiate IEEE 802.1X authentication.

2. The station proceeds to IEEE 802.1X authentication and 4-Way Handshake with the MPP via the serving MAP.

3. Then the MPP uses a Key Distribution message to inform the serving MAP of the newly derived PTK. A GTK assigned by the MPP, if any, in 4-Way Handshake can also be included in the same message meant for the MAP.

4. Upon receipt of the PTK, the serving MAP allows the station to access the network hereinafter.

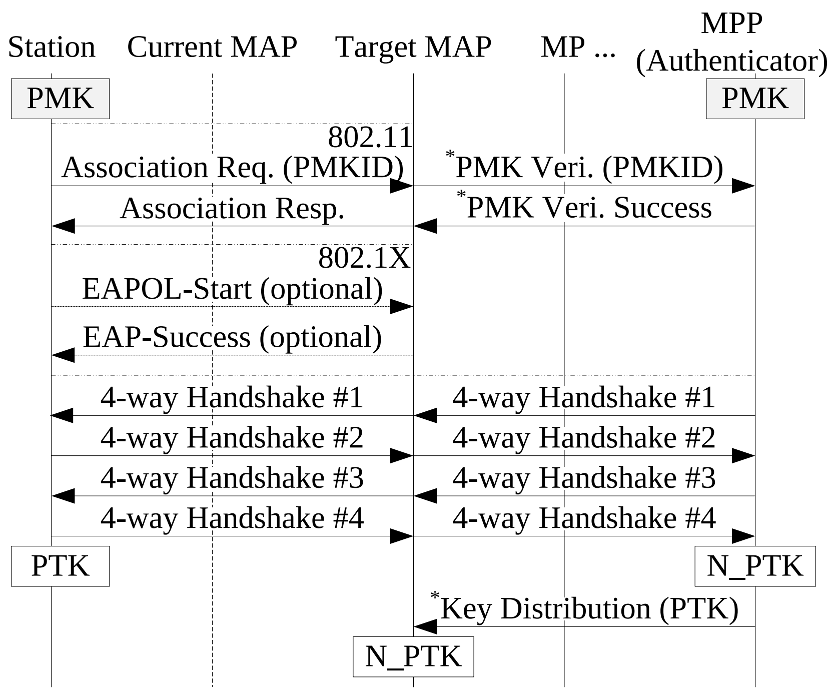


Figure 3.3: Intra-MPP handoff message flow without the involvement of any IEEE 802.1X Authentication Server.

3.1.2 Handoff Procedure

Handoff occurs when a mobile station reassociates from one MAP to another, in two possible cases: intra-MPP and inter-MPP handoff. The former means that the station reassociates with another MAP connected to the same MPP. Since the IEEE 802.1X authenticator (MPP) remains unchanged, PMKs cached on the station and the MPP sides serve as a basis for mutual authentication, leaving out IEEE 802.1X authentication. Figure 3.3 shows the message flow of such an intra-MPP handoff procedure.

1. The reassociating station sends PMKID to its target MAP, which then forwards PMKID to the MPP to validate the cached PMK on the station side.

2. The PMKID is used to identify the PMK cached in the MPP. If the PMKID is found valid, the MPP notifies the target MAP using a PMK Verification Success message.

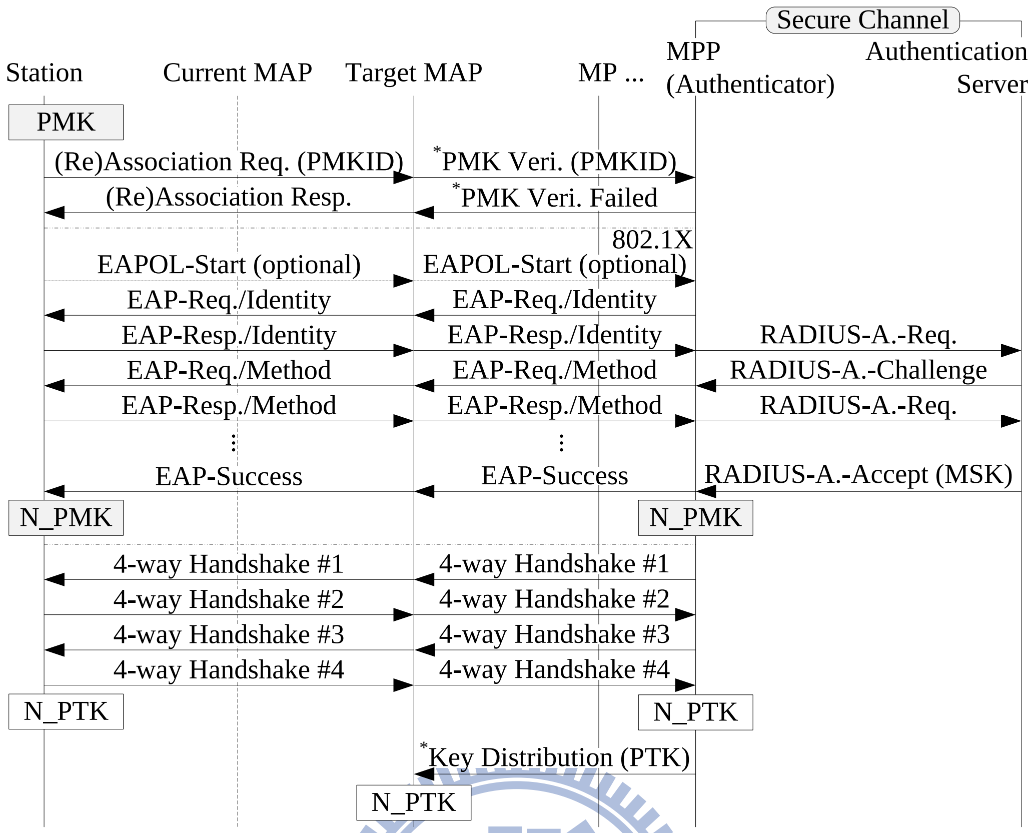


Figure 3.4: Inter-MPP handoff message flow.

3. Upon receiving an EAPOL-Start frame, the target MAP replies an EAP-Success message directly, with no need to contact the backend Authentication Server.

4. Then 4-Way Handshake and PTK distribution follow as in Section 3.1.1.

On the other hand, inter-MPP handoff takes place when a station moves across MAPs managed by different MPPs. In this case, the station performs IEEE 802.1X preauthentication with its new target MPP (Figure 3.4) unless the station has retained a correct PMK for the newly visited domain. Message flow is described below:

1. The station reassociates with the target MAP. The PMKID included in the (Re)Association Request frame is forwarded by the receiving MAP to the new MPP for validating the cached PMK, if any.

2. Provided that the new MPP cannot find the corresponding PMK, a message (PMK Verification Failure) is sent to the target MAP to trigger IEEE 802.1X authentication for the station.

3. Accordingly IEEE 802.1X authentication and 4-Way Handshake are activated, and then followed by PTK distribution. These procedures operate as prescribed in Section 3.1.1.

It is noted that IEEE 802.1X authentication delay remains present if preauthentication fails due to some reasons.

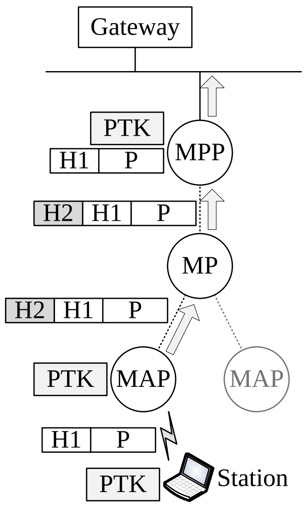
3.1.3 Frame Delivery

In our architecture, we establish an end-to-end secure channel between each station and its MPP. Therefore, in the event of communication with a correspondent host outside the mesh network, only a station's serving MAP and MPP are required to encrypt/decrypt frames. This can be accomplished by extending the use of IEEE 802.11i encryption protocols to the entire mesh network while preventing the MAC header from being altered. In practice, we introduce a bidirectional MAC tunnel between serving MAP and MPP. Figure 3.5a gives an example of our encapsulation process.

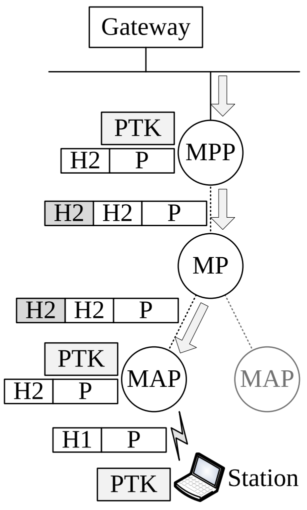
Supposing that a station communicates with a node outside the mesh network, frame delivery takes following steps.

1. The station prepares an outgoing frame of the form <H1, P>, where H1 and P denote the frame header and payload, respectively. The frame is encrypted using the station's PTK and then transmitted to the serving MAP.

2. The serving MAP verifies the MIC of the received frame using the PTK. A frame with correct MIC is further processed or discarded otherwise. If the destination site is found outside the mesh network, the MAP encapsulates the frame in a form <H2, H1, P>, where H2 is the MAC header meant for IEEE 802.11s routing. The encapsulated frame is then forward hop by hop toward



(a) Outbound frame



(b) Incoming frame

Figure 3.5: Frame delivery involves bi-directional MAC tunneling.

the MPP. Note that the outer header may be replaced anew at intermediate MPs for next-hop forwarding, whereas the inner header (H1) is kept unchanged throughout the routing process.

3. Upon receipt, the MPP removes the outer header (H2) and decrypts the inner frame <H1, P> using the corresponding PTK. Subsequently the MPP encapsulates the payload (P) into an Ethernet frame and forwards the frame to the destination.

In contrast, Figure 3.5b illustrates a scenario where a node outside the mesh network sends a data frame to a mobile station. The inbound delivery is as follows.

1. The MPP translates the received frame (from the gateway) into an IEEE 802.11 format of the form <H2, P>. The frame is encrypted using the corresponding PTK, encapsulated in a form <H2, H2, P>, and then forwarded hop by hop toward the destination station. Here two identical MAC headers are used to keep the inner header intact during routing within the mesh network.

2. When the frame arrives at the local MAP of the intended station, the outer header is removed and the resulting inner frame is decrypted using the PTK.

3. The MAP encapsulates the decrypted payload (P) into an IEEE 802.11 frame <H1, P> and encrypts the resulting frame using the PTK again. Then the MAP forwards the frame to the station.

In case that both source and destination nodes are located in the same mesh network, the IEEE 802.11s shortcut routing [13] technique applies.

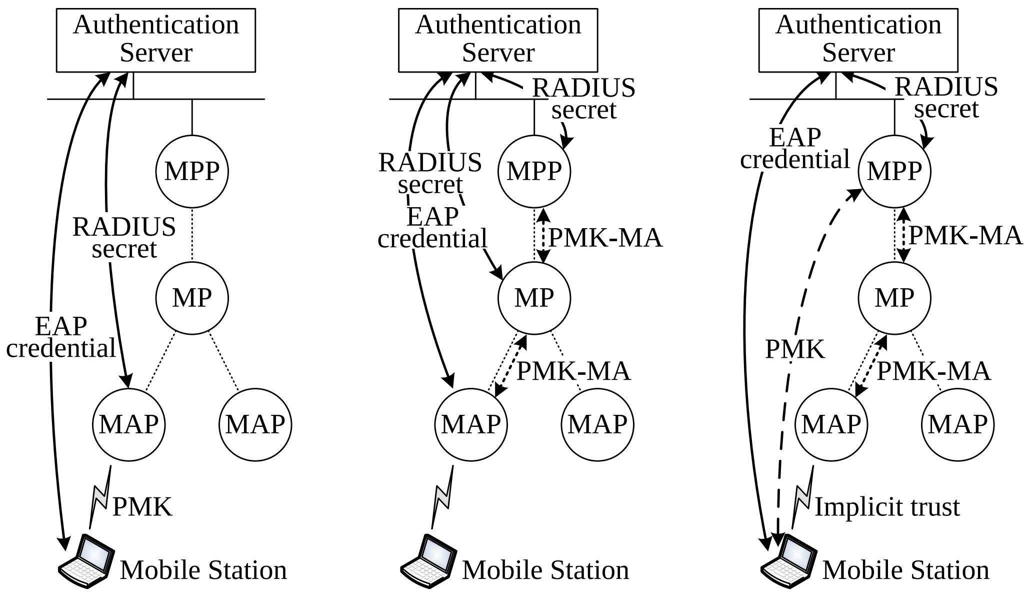
Note that our bidirectional MAC tunneling mechanism is not part of IEEE 802.11s but similar methodology appeared in some contexts like Virtual Private LAN Service [15] or IEEE 802.1ah bridge protocols. The frame size limit on our MAC tunneling is the maximum allowed frame size in IEEE 802.11s (1552 bytes) minus the length of the IEEE 802.11s frame header, as the extra header we use for encapsulation is of the IEEE 802.11s-specified format. To prevent fragmentation due to exceeding the maximum transmission unit a link-layer protocol can carry, the mesh network can be configured with setting the maximum transmission unit to 1552 bytes accordingly.

3.1.4 Security Considerations

We argue that in our architecture wireless communication between each station and its MAP remains secure, and that our achieved security level is no weaker than IEEE 802.11i. Recall that stations and MAPs under discussion are IEEE 802.11i-capable. In addition, mesh links among MPs have been protected by EMSA. In what follows we reason about trust relationships and then discuss threat models.

Trust Relationships

In IEEE 802.11i, a station and its serving MAP perform IEEE 802.1X authentication and 4-Way Handshake to establish a security association (based on PMK as shown in Figure 3.6a), with a station-MAP trust relationship. The relationship results from two security associations: i) between the mobile station and the Authentication Server (abbreviated to MS↔AS relationship) ensured by EAP credentials, and ii)



(a) IEEE 802.11i (b) IEEE 802.11s (c) Our architecture

Figure 3.6: Trust relationships in different network paradigms.

between the Authentication Server and the MAP (AS↔MAP relationship) by the RADIUS secret. Therefore, IEEE 802.11i implies an MS↔AS↔MAP trust chain.

To secure communication between the station and its MAP, our proposed solution should be proved to attain an equivalent station-MAP trust relationship. To see this, recollect that mesh links between two MPs are protected by EMSA (PMK-MA ensures a MAP↔MP trust relationship), as indicated in Figure 3.6b. Furthermore, a MAP, MP, and the MPP maintain security associations with the Authentication Server using their respective RADIUS secrets (these MAPs and MPs are also using EAP credentials.) Coalescing these relations gives a MAP↔AS↔MP↔AS↔ ⋯ ↔MPP trust chain in the mesh security domain.

In our approach, as shown in Figure 3.6c, the station performs IEEE 802.1X authentication and 4-Way Handshake with the MPP, forming an MS↔AS↔MPP trust chain. Besides, since there is MAP↔MPP trust relationship resulting from PMK-MAs within the mesh security domain, the MS↔MPP↔MAP trust chain can be deduced from the former two trust relationships. Therefore, we assure our architecture of station-MAP trust relationship equivalent to that in IEEE 802.11i.

Threat Models

To validate that our development maintains the due network security, we identify and discuss potential threats against our proposed mechanism as follows.

- PMKID leakage: Even if an attacker may learn the corresponding PMKID from previous eavesdropping and be able to skip IEEE 802.1X authentication, this does not cause security flaw impairing our architecture. This is because MSK or any form of pre-shared keys is never transmitted over the wireless media, and thus a valid PTK cannot be derived by the attacker. As a consequence, the attacker cannot compute the correct MIC of the second EAPOL-Key frame in 4-Way Handshake, so the attacker is blocked by the MAP right away.

- Unauthorized disclosure: Compromised mesh links may result in unauthorized disclosure of keying materials. For IEEE 802.11i, an MSK is transmitted from the Authentication Server to the serving MAP through mesh links. If mesh links are compromised, the MSK is possibly divulged to an attacker. As opposed to transmitting MSK, our proposal transfers only PTK via mesh links. Since PTK takes a lower level than MSK in key hierarchy, the exposed PTK accounts for less security degradation as compared with the compromised MSK.

- Compromise of authenticators: When an authenticator is compromised or stolen, an attacker may obtain all the PMKs cached in that authenticator. In this scenario, the attacker can access the mesh network via any MAP connected to that authenticator. However, such vulnerability is recognized reluctantly acceptable in IEEE 802.11 working groups. For example, IEEE 802.11r is also susceptible to the situation where a compromised authenticator may expose PMK-R0s to the attacker. Since IEEE 802.11 working groups permit of this situation, we believe the potential vulnerability is acceptable.

The qualitative analysis above shores up security aspects of our architecture. Next we proceed to quantitative analysis of the proposed protocol.

3.2 Performance Discussion

This section compares our fast-handoff approach with the best-known IEEE 802.11i scheme in terms of handoff delay and the amount of signaling traffic introduced by a handoff process. Our performance evaluation is conducted by means of analytical modeling and simulation modeling.

3.2.1 Analytical Model

Our performance analysis is inspired from a clustering technique by Akyildiz *et al* [4] that simplifies two-dimensional random walks over hexagonal and mesh planes to a reduced set of cells. As in that study, Figure 3.7 illustrates a hexagonal plane consisting of several regions of cells. Each cell is labeled with (x, y), where x denotes the hop count from the cell to the MPP and y its type. Stations in cells of the same type are assumed to leave the cells with the same routing pattern out of certain symmetry. For labeling, every region is partitioned into six congruent segments through diagonal chords such that cells within each segment are labeled in the following manner:

1. From center $(0, 0)$ outward, mark cells along the diagonal line as $(x, 0)$, with an increment of 1 in x. When done, reset x to 2.

2. Initializing y to 1, mark cells with x hops away from the MPP, in an arc-like form, clockwise as (x, y), with an increment of 1 in y.

3. Increase x by 1 and repeat the above step till no unmarked cells can be found.

Thanks to such marking in conjunction with Markov chain (see below) reasoning, the simplified random walk model was shown to behave exactly the same as the original two-dimensional random walks [4]. Consequently we restrict attention to

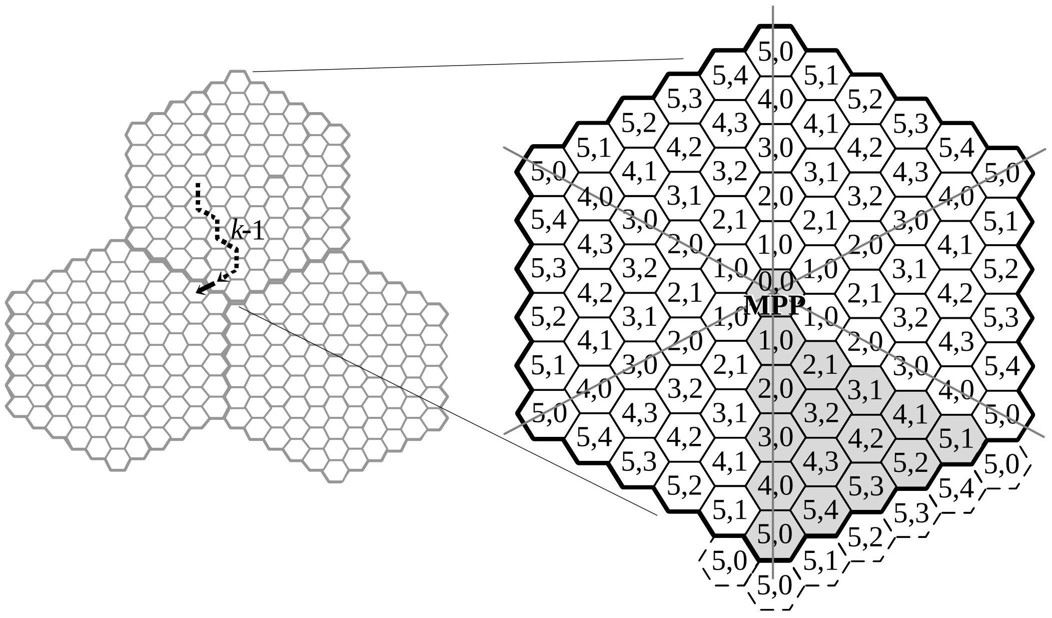


Figure 3.7: A mesh network is viewed to consist of regions of cells, each region being served by an MPP. Every cell represents the radio coverage of a MAP, an MP, or an MPP.

such a single segment, which is equivalent to considering the extent of a cluster as a whole.

We consider a mesh network of hexagonal cells in that such topology is arguably generic to represent how mesh networking entities are interconnected to cover a geographical area with maximized channel capacities. Concerning handoff in a mesh network, hop count between MAP and MPP forms an essential factor in performing a handoff procedure. While Figure 3.7 relates hop counts among cells covered by network entities, cell $(0,0)$ signifies MPP's radio coverage; other cells are coverages of MAPs connected to this MPP. (We presume each cell to be serviced by a MAP which also functions as an MP for routing purpose.) Here a station residing in a cell is assumed to migrate to any of its neighboring cells with equi-probability 1/6.

The reduced two-dimensional random walk model is applicable to capturing a station's movement behavior and obtaining the ratio of inter-MPP handoffs to intra-MPP handoffs occurred in the course. Concerning a random walk process for an n-subarea cluster (e.g. $n = 6$, n indicative of how many levels are present in a cluster),

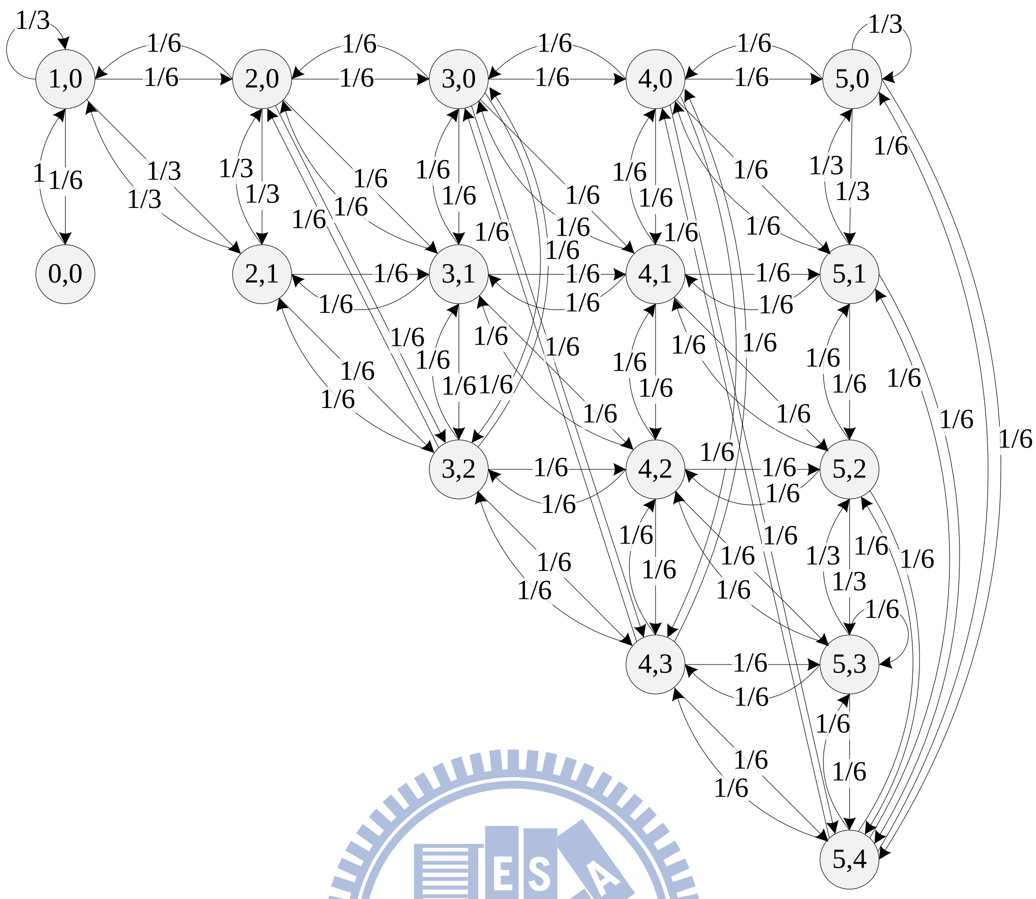


Figure 3.8: State transition diagram for a 6-subarea cluster.

Figure 3.8 shows a state transition diagram for a station; state (x, y) indicates a station residing in a cell (x, y). Totally there are $S(n) = \frac{n(n-1)}{2} + 1, n > 0$ states in an n-subarea cluster. We consider the MPP capable of offering a mobile station a radio link as well, so Figure 3.8 also depicts the possibility that the station migrates to cell $(0, 0)$.

Let $p_{(x,y),(x',y')}$ be the one-step transition probability from states (x, y) to (x', y') due to handoff. For a 6-subarea cluster, the transition probability matrix $\mathbf{P}$ =

$[p_{(x,y),(x',y')}]$ is

$$\mathbf{P} = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 1/6 & 1/3 & 1/6 & 1/3 & \cdots & 0 & 0 \\ 0 & 1/6 & 0 & 1/3 & \cdots & 0 & 0 \\ 0 & 1/3 & 1/3 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1/6 & 1/6 \\ 0 & 0 & 0 & 0 & \cdots & 1/6 & 0 \end{bmatrix}_{S(6)\times S(6)}$$

In $\mathbf{P}$, rows and columns are indexed in order of states $(0,0)$, $(1,0)$, $(2,0)$, $(2,1)$, $\cdots$, $(n-1,0)$, $(n-1,1)$, $(n-1,2)$, $\cdots$, and $(n-1,n-2)$. The general form of $\mathbf{P} = [p_{(x,y),(x',y')}]$ for an n-subarea cluster is given as following:

$$p_{(x,y),(x',y')} = \begin{cases} 6 \times w_0((0,0),(x',y')) & \text{if } x=y=0 \\ \sum\limits_{i=\{0,1,2,4,5,6\}} w_i((x,0),(x',y')) & \text{if } 0<x<n-1 \text{ and } y=0 \\ \sum\limits_{i=\{0,2,3,4,5,6\}} w_i((x,y),(x',y')) & \text{if } 0<x<n-1 \text{ and } 0<y<x \\ \sum\limits_{i=\{2,4,5\}} w_i((x,0),(x',y')) + 2\times v_0((x,0),(x',y')) + v_1((x,0),(x',y')) & \text{if } x=n-1 \text{ and } y=0 \\ \sum\limits_{i=\{2,3,4,5\}} w_i((x,y),(x',y')) + v_0((x,y),(x',y')) + v_1((x,y),(x',y')) & \text{if } x=n-1 \text{ and } 0<y<x \end{cases}$$

where

$$\begin{aligned}
w_0((x,y),(x',y')) &= \begin{cases} 1/6 & \text{if } x'=x+1 \text{ and } y'=y \\ 0 & \text{otherwise} \end{cases} \\
w_1((x,y),(x',y')) &= \begin{cases} 1/6 & \text{if } x'=x+1 \text{ and } y'=(y+x'-1) \bmod x' \\ 0 & \text{otherwise} \end{cases} \\
w_2((x,y),(x',y')) &= \begin{cases} 1/6 & \text{if } x'=x \text{ and } y'=(y+x'-1) \bmod x' \\ 0 & \text{otherwise} \end{cases} \\
w_3((x,y),(x',y')) &= \begin{cases} 1/6 & \text{if } x'=x-1 \text{ and } y'=(y+x'-1) \bmod x' \\ 0 & \text{otherwise} \end{cases} \\
w_4((x,y),(x',y')) &= \begin{cases} 1/6 & \text{if } x'=x-1 \text{ and } (y'=x'=0 \text{ or } y'=y \bmod x') \\ 0 & \text{otherwise} \end{cases} \\
w_5((x,y),(x',y')) &= \begin{cases} 1/6 & \text{if } x'=x \text{ and } y'=(y+1) \bmod x' \\ 0 & \text{otherwise} \end{cases} \\
w_6((x,y),(x',y')) &= \begin{cases} 1/6 & \text{if } x'=x+1 \text{ and } y'=(y+1) \bmod x' \\ 0 & \text{otherwise} \end{cases} \\
v_0((x,y),(x',y')) &= \begin{cases} 1/6 & \text{if } x'=x=n-1 \text{ and } (y+y') \bmod (n-1)=0 \\ 0 & \text{otherwise} \end{cases} \\
v_1((x,y),(x',y')) &= \begin{cases} 1/6 & \text{if } x'=x=n-1 \text{ and } (y+y') \bmod (n-1)=1 \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

Letting X_i be the random variable representing the mobile station in which state at step i, element $p_{(x,y),(x',y')}$ in $\mathbf{P}$ is given by $Pr[X_{i+1} = (x',y')|X_i = (x,y)]$, the probability of going to (x',y') on the next step, provided that the station currently resides in state (x,y).

Let $\Pi = [\pi_{(0,0)}, \pi_{(1,0)}, \cdots, \pi_{(n-1,n-2)}]$ be the stationary distribution, where $\pi_{(x,y)}$ denotes the equilibrium probability of finding the station in state (x, y). We obtain Π by solving the following two equations.

$$\Pi = \Pi\mathbf{P} \tag{3.1}$$

$$\pi_{(0,0)} + \sum_{x=1}^{n-1}\sum_{y=0}^{x-1} \pi_{(x,y)} = 1 \tag{3.2}$$

Since one of the rows in Equation (3.1) is linearly dependent on the others, it is necessary to introduce the additional conservation relationship as given in Equation (3.2) to obtain Π. From Π we can derive the ratio of inter-MPP handoffs to intra-MPP handoffs accordingly (see Section 3.2.2). This ratio is of avail to estimate the expected handoff costs of different schemes because in our architecture, a station only performs 4-Way Handshake during intra-MPP handoff and IEEE 802.1X authentication during inter-MPP handoff. In contrast, IEEE 802.11i requires IEEE 802.1X authentication in each handoff unless the PMK is cached at the target MAP. (PMK can be found in the MAP's cache when preauthentication ahead of handoff has been completed or when the station visited the MAP before, whose corresponding security context remains in credit.) Such operational differences will lead to distinct performance results[1].

Note that our Markov chain is constructed in states' points of view. A state (x, y) may refer to corresponding cells in different clusters; state (x, y) is interpreted to consolidate all the cells labeled with (x, y) of different clusters in the system. Such unification can be done because of high-symmetry cluster layouts. As such, $\pi_{(x,y)}$ is the aggregate probability of finding the mobile station in cell (x, y) of any cluster. Thus, $p_{(x,y),(x',y')}$ may implicitly subsume inter-cluster transition probabilities. Taking Figure 3.7 as an example, two transitions from border cells $(5, 0)$ to $(5, 1)$ are possible, causing one inter-cluster and one intra-cluster handoffs. So

[1]Another way is to model the station appearing at a random cell, making a number of handoffs (represented with a random variable), and eventually disappearing. The expectation can then be computed over two random variables: the starting cell and the number of handoffs made in the course.

$p_{(5,0),(5,1)} = 1/6 + 1/6 = 1/3$. Provided $p_{(x,y),(x',y')}$, the fraction of inter-cluster transitions is pertinent to the station's whereabouts, as shall be substantiated in Section 3.2.2 below.

3.2.2 Handoff Latency

We are now in a position to discuss intra-MPP handoff and inter-MPP handoff latencies. As depicted in Figure 3.3, the intra-MPP handoff procedure in our architecture involves two messages (for PMK Verification) exchanged between the target MAP and MPP, 4-Way Handshake messages between the station and MPP, and a message containing PTK distributed by the MPP to the target MAP. In general, intra-MPP handoff delay L_{INTRA} is composed of authentication latency (L_{INTRA_AUTH}) and 4-Way Handshake latency (L_{INTRA_4W}). These measures can be formulated using several notations:

- x is the hop count between a MAP and the MPP.
- S counts MAPs in a concerned segment with x hops to the MPP (see the grayed area of Figure 3.7.) Note that $S = 1$ if $x = 0$; otherwise $S = x$ for $x \geq 1$.
- A segment of an n-subarea cluster contains $1 + n(n-1)/2$ MAPs (including the MPP), as exemplified in Figure 3.7.

Therefore, on average we have

$$L_{INTRA\_AUTH} = 2 \cdot T \cdot H \tag{3.3}$$

where T denotes the expected single-hop transmission delay and H is the mean hop count between a MAP and the MPP, namely,

$$H = \frac{\sum_{x=0}^{n-1} (x \cdot S)}{1 + n(n-1)/2} \tag{3.4}$$

H results from averaging all possible hop counts over the total number of MAPs within a segment.

In the handshake phase, 4-way messages are delivered between the station and the MPP (message exchanges between the station and the intermediary MAP causing a delay of L_{4W}), followed by an additional message for sending PTK to the target MAP. Hence we have

$$L_{INTRA\_4W} = L_{4W} + 5 \cdot T \cdot H \tag{3.5}$$

Concerning IEEE 802.11i intra-MPP handoff, IEEE 802.1X authentication is performed if the PMK is not cached at the target MAP. Thus, the average authentication delay (L'_{INTRA_AUTH}) results in

$$L'_{INTRA\_AUTH} = L_{1X} + M_{RADIUS} \cdot T \cdot H \tag{3.6}$$

where L_{1X} is the time required for transporting IEEE 802.1X traffic between the station and its Authentication Server, excluding that spent over the mesh backbone. IEEE 802.1X message delay over the mesh backbone is singled out as $M_{RADIUS} \cdot T \cdot H$, where M_{RADIUS} is the number of RADIUS messages delivered between the target MAP and the MPP. Note that 4-Way Handshake delay in IEEE 802.11i remains to be L_{4W}.

Using Equation (3.6), the mean intra-MPP handoff delay (L'_{INTRA}) in IEEE 802.11i is:

$$L'_{INTRA} = (1 - P_{PMK\_MISS}) \cdot L_{4W} + P_{PMK\_MISS} \cdot (L'_{INTRA\_AUTH} + L_{4W}) \tag{3.7}$$

P_{PMK_MISS} above denotes the probability of PMK being absent in the target MAP. Given the probability P_v that the station moves to a previously visited authenticator and P_{PF} the probability of IEEE 802.11i preauthentication failure, we find

$$P_{PMK\_MISS} = (1 - P_v) \cdot P_{PF} \tag{3.8}$$

P_{PF} arises when the station moves to a target MAP before its initiated preauthentication is not yet completed. This corresponds to a sudden, premature handoff[2].

[2]The value of P_{PF} varies from station to station, depending on 1) the elapsed time from the point when such advance operations are initiated to the point when the station disassociates

Note that intra-MPP handoff delay in our architecture is L_{INTRA}, which is generally by far lower than that (L'_{INTRA}) incurred in IEEE 802.11i. If a station moves to a new MAP before preauthentication is completed, L_{INTRA_AUTH} should still be included in our intra-MPP handoff delay.

Next consider inter-MPP handoff. L_{INTER} represents inter-MPP handoff delay for a station in our architecture, equating IEEE 802.1X authentication latency (L_{INTER_AUTH}) and 4-Way Handshake latency (L_{INTER_4W}). With reference to Figure 3.4, when the station moves out of a cluster but its valid PMK is not cached at the new target MPP, IEEE 802.1X authentication with a delay of L_{INTER_AUTH} remains necessary. To quantify,

$$L_{INTER\_AUTH} = L_{1X} + M_{1X} \cdot (n-1) \cdot T \tag{3.9}$$

where M_{1X} is the number of IEEE 802.1X messages exchanged between the target MAP and the MPP, and $(n-1)$ is the hop count between the target MAP and the new MPP. Since a station doing inter-MPP handoff will reassociate with some boundary MAP in another cluster, the hop count between the target MAP and the new MPP is thus $(n-1)$. With L_{INTER_4W} denoting the average latency for 4-Way Handshake and PTK distribution during inter-MPP handoff, we have

$$L_{INTER\_4W} = L_{4W} + 5 \cdot (n-1) \cdot T \tag{3.10}$$

Inter-MPP handoff in the context of IEEE 802.11i resembles intra-MPP handoff in the form of Equation (3.6), except that messages are forwarded via the boundary MAP. The incurred authentication latency yields

$$L'_{INTER\_AUTH} = L_{1X} + M_{RADIUS} \cdot (n-1) \cdot T \tag{3.11}$$

from the current MAP (preauthentication to a target MAP cannot be done if the station loses contact with its current MAP), and 2) the time required for completing preauthentication over the network (network dynamics such as congestion or variable transmission rates govern how long preauthentication message exchanges can take.) As opposed to formulating P_{PF} in terms of the two determinants through probabilistic modeling as in [6], this study does not resolve P_{PF} but considers all its possible values ranging between 0 and 1 in Section 3.2. We then discuss performance results from different settings of P_{PF} to demonstrate general system behavior.

Overall, provided Equations (3.8) and (3.11), the inter-MPP handoff delay L'_{INTER} in IEEE 802.11i results in

$$L'_{INTER} = (1 - P_{PMK\_MISS}) \cdot L_{4W} + P_{PMK\_MISS} \cdot (L'_{INTER\_AUTH} + L_{4W}) \quad (3.12)$$

From Equations (3.3), (3.4), and (3.12) it can be seen that the handoff delay for a station amounts to

$$L_S = L_{INTER} \cdot (\Pi \mathbf{Q}) + L_{INTRA} \cdot (1 - \Pi \mathbf{Q}) \quad (3.13)$$

where $\mathbf{Q} = [q_{(0,0)}, q_{(1,0)}, q_{(2,0)}, q_{(2,1)}, \cdots, q_{(n-1,n-2)}]^T$ is a column vector of probabilities. Each entry of $\mathbf{Q}$ takes the form $q_{(x,y)}$, the probability for the station performing an inter-MPP handoff from cell (x, y). The equation above expresses the mean handoff delay experienced by a station moving in the network. The measure L_S takes the weighted sum of inter-MPP handoff delays and intra-MPP handoff delays, starting mobility from any cell (x, y). For the former, the weight is the product of $\pi_{(x,y)}$ and the probability $q_{(x,y)}$ of inter-MPP handoff occurrences. Likewise, the weight of intra-MPP handoff delay involves the probability complementary to that for inter-MPP handoffs. Now that a station migrates to any of its neighbor cells with equi-probability, we have $q_{(n-1,0)} = 1/2$ and $q_{(n-1,y)} = 1/2$ for $y = 1, 2, \cdots, n-2$, but otherwise $q_{(x,y)} = 0$.

Notice that in our scheme, inter-MPP handoff brings about IEEE 802.1X authentication delay only if a station moves to a new domain before its intended preauthentication can be completed. Thus, the demand for IEEE 802.1X authentication is substantially reduced. In essence we trade multi-hop transmissions of 4-Way Handshake messages between MAP and MPP within a same administrative domain for expensive cross-realm IEEE 802.1X authentication. Such a design is of value as long as IEEE 802.1X authentication takes a longer delay than that for conveying handshake messages within a regional cluster. As shall be seen shortly in Section 3.2.4, it is imperative to select the parameter n moderately for scalability considerations.

3.2.3 Signaling Traffic

Similar to the foregoing treatment, we now formulate the amount of signaling traffic (number of message transmissions) incurred in intra- and inter-MPP handoff procedures, respectively, over the mesh backbone. Observe that intra-MPP handoff involves authentication traffic (M_{INTRA_AUTH}) and 4-Way Handshake traffic (M_{INTRA_4W}). As illustrated in Figure 3.3, our approach requires 7 message transmissions over the wireless infrastructure: 2 messages for PMK Verification, 4-Way Handshake messages, and 1 message for PTK distribution. Hence,

$$M_{INTRA\_AUTH} = 2 \cdot H \tag{3.14}$$

$$M_{INTRA\_4W} = 5 \cdot H \cdot R \tag{3.15}$$

where R is the ratio of 4-Way Handshake to IEEE 802.1X authentication in average message size. Equation (3.15) reflects a normalized expression for two different sizes of messages in a unified scale. With Equations (3.14) and (3.15) as well as corresponding measures, intra-MPP handoff signal-ing traffic takes the form

$$\begin{aligned} M_{INTRA} = {} & (1 - P_{PMK\_MISS}) \cdot M_{INTRA\_4W} + \\ & P_{PMK\_MISS} \cdot (M_{INTRA\_AUTH} + M_{INTRA\_4W}) \end{aligned} \tag{3.16}$$

Regarding intra-MPP handoff within IEEE 802.11i, Figure 2.2 shows that only RADIUS messages are transmitted. Therefore, $M'_{INTRA\_AUTH} = M_{RADIUS} \cdot H$ but M'_{INTRA_4W} is zero. The resulting handoff signaling traffic can still be expressed by Equation (3.16), yet with M'_{INTRA_AUTH} and M'_{INTRA_4W} in place of M_{INTRA_AUTH} and M_{INTRA_4W}, respectively.

For the inter-MPP handoff case, let M_{INTER} represent the amount of signaling traffic generated due to a station performing inter-MPP handoff, containing authentication traffic (M_{INTER_AUTH}) and 4-Way Handshake traffic (M_{INTER_4W}). In our architecture, as shown in Figure 3.4, all EAPOL and 4-Way Handshake messages are transmitted over the mesh infrastructure. So $M_{INTER\_AUTH} = M_{1X} \cdot (n-1)$ and $M_{INTER\_4W} = 5 \cdot (n-1) \cdot R$.

As for IEEE 802.11i, inter-MPP handoff causes the same number of message exchanges as in intra-MPP handoff. It follows that $M'_{INTER\_AUTH} = M_{RADIUS} \cdot (n-$

1) and $M'_{INTER\_4W} = 0$. Thus, inter-MPP handoff signaling traffic can be written as

$$\begin{aligned} M_{INTER} = (1 - P_{PMK\_MISS}) \cdot M_{INTER\_4W} + \\ P_{PMK\_MISS} \cdot (M_{INTER\_AUTH} + M_{INTER\_4W}) \end{aligned} \tag{3.17}$$

The above equation also holds for the IEEE 802.11i scheme with M'_{INTER_AUTH} and M'_{INTER_4W} in lieu of M_{INTER_AUTH} and M_{INTER_4W}, respectively.

In the presence of Equations (3.16) and (3.17), the handoff signaling traffic M_S in terms of a cluster gives

$$M_S = M_{INTER} \cdot (\Pi \mathbf{Q}) + M_{INTRA} \cdot (1 - \Pi \mathbf{Q}) \tag{3.18}$$

The measure M_S quantifies the amount of signaling traffic caused by a station doing handoffs in the mesh network. M_S counts all possible transmissions of signaling messages out of inter-MPP and intra-MPP handoffs. The weights of the two types of handoff are identical to those in Equation (3.13).

3.2.4 Performance Results

In order to validate our analytical model and demonstrate performance results, we developed a simulator in C# that mimicked mobile stations moving about within a clustered environment as shown in Figure 3.7. Under consideration were $100,000$ stations, each being simulated to start off from a randomly chosen cell, make 800 cell-crossing movements, and bring in an accumulation of intra-MPP or inter-MPP handoff costs. A station performing an inter-MPP handoff, say to cell (x', y') of another cluster, was taken to emerge next at the corresponding cell (x', y') of the current cluster in that all clusters were considered congruent in appearance. For data representativeness, our simulation results assume the average behavior of all the stations.

Additionally, experiments were conducted to determine parameter values of concern. As shown in Figure 3.9, the experimental environment consists of an Authentication Server, two au-thenticators, and a supplicant station residing in a wireless Local Area Network without any other entities attached. The supplicant station is a

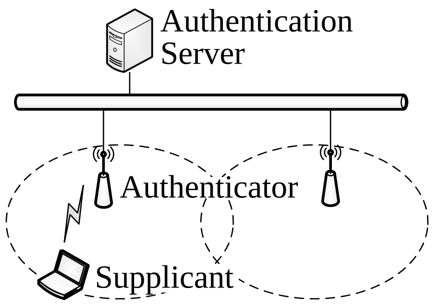


Figure 3.9: Experimental environment.

laptop PC, equipped with an IEEE 802.11g network interface, running on Windows XP SP2 with built-in Windows Zero Configuration Service. Authenticators are another IEEE 802.11g-capable PCs running the open source daemon hostapd-0.5.7. The Authentication Server employs FreeRADIUS-1.1.4. All these four machines are kept synchronized to the same NTP (Network Time Protocol) server throughout. The encryption protocol in use is WPA2/AES and the EAP method is PEAP/EAP-MSCHAPv2. A widely-used packet analyzer–Wireshark–is run on the supplicant station and the Authentication Server sides to capture all the packets of interest.

Table 3.1 summarizes average values of system parameters resulting from 20 full authentications, where T is in particular out of 80 measurements. Besides, we built n-subarea clusters and carried out $1,200,000$ independent, identical simulations of random walks to resolve the ratio P_v that a cell-crossing station migrates to a previously visited authenticator. The collected statistics of P_v in different n-subarea settings are listed in Table 3.2. Note that a network environment where authenticators are co-located at MAPs corresponds to the setting $n = 1$, a typical IEEE 802.11s scenario.

According to IEEE 802.11s [13], the suggested level of a mesh network is 3 (i.e. $n = 3$). Hence we elaborate primarily on the performance in terms of a 3-subarea clustered environment. Figure 3.10a plots handoff delay (L_S) versus P_{PF} in an n-subarea network environment. Overall, our numerical results conform closely with simulation results. Also, this figure reveals a trend that our proposal reduces handoff delays significantly. In particular, when P_{PF} is 1.0, i.e., the station does not perform

Table 3.1: System parameters.

T	2.44 ms
L_{1X}	401.63 ms
L_{4W}	20.76 ms
M_{1X}	22 messages
M_{RADIUS}	18 messages
R	1.0492

Table 3.2: Mean P_v.

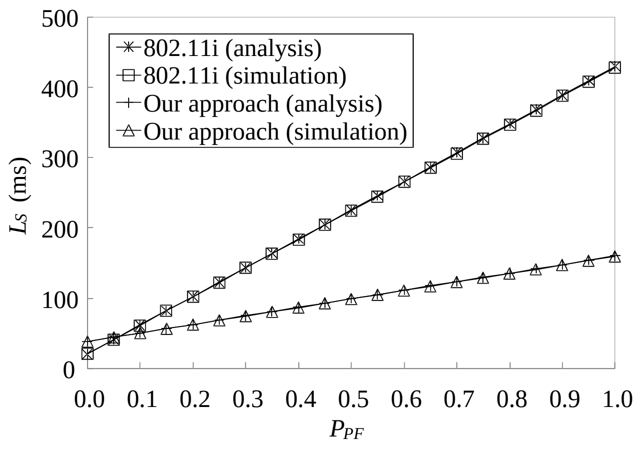
$n = 1$	0.000000
$n = 2$	0.064579
$n = 3$	0.120625
$n = 4$	0.164704
$n = 5$	0.199851
$n = 6$	0.229387
$n = 7$	0.254347
$n = 8$	0.275391

preauthentication[3], our approach can achieve a saving of handoff delay by up to 62.7%, a marked improvement. Therefore, our design is of utility, even though most IEEE 802.11 devices do not support preauthentication. As a note, our approach maintains its advantage over the counterpart scheme unless $P_{PF} < 0.05$, which signifies a rare case though. This case arises because 4-Way Handshake messages forwarded between MAP and MPP bring about comparatively longer delay in our architecture.

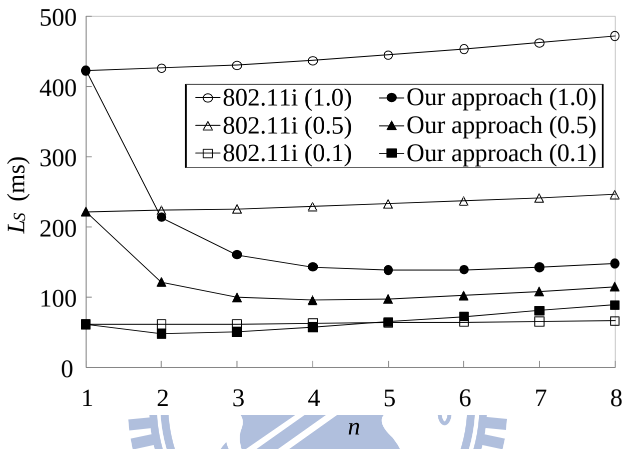
Further experiments in other settings reflect a consistent trend that numerical results accord with simulation results substantially, which justifies the correctness of our analytical model. For conciseness and clarity reasons, hereinafter we provide analytical results as main parts of figure layouts.

Figure 3.10b shows handoff delays under different cluster sizes and preauthentication failure probabilities (P_{PF} is represented in parenthetical numbers of the

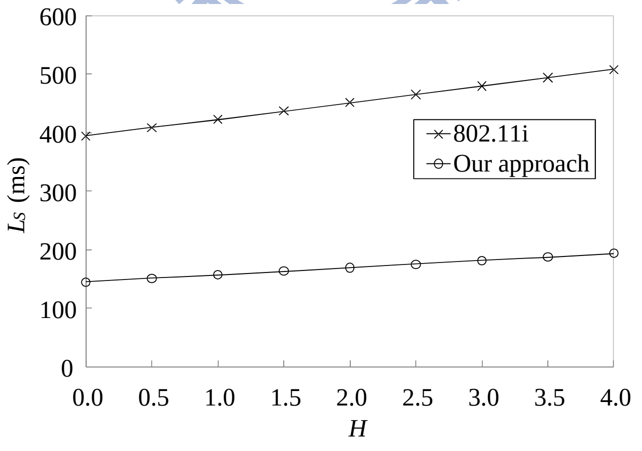
[3]The preauthentication function in Windows XP with WPA2 is disabled by default.



(a) L_S versus P_{PF} ($n = 3$)



(b) L_S versus n



(c) L_S versus H ($n = 3$; $P_{PF} = 1.0$)

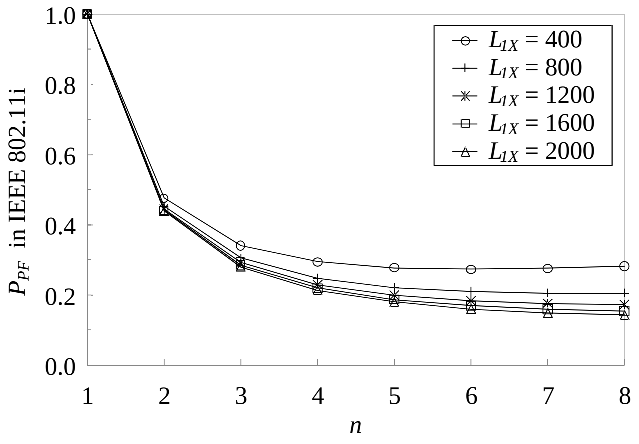
Figure 3.10: Handoff delays of two subject schemes.

legends.) This figure suggests that, as long as the cluster is large enough, say, with $n \geq 3$, handoff delays appear immaterially different for whichever P_{PF}, meaning that P_{PF} becomes a less dominant factor for a sufficiently large network. Hence, for concise presentation, we next confine ourselves to discussing performance results from P_{PF} equal to 1.0. Performance statistics in other P_{PF} settings can be estimated proportionately with reference to Figure 3.10a.

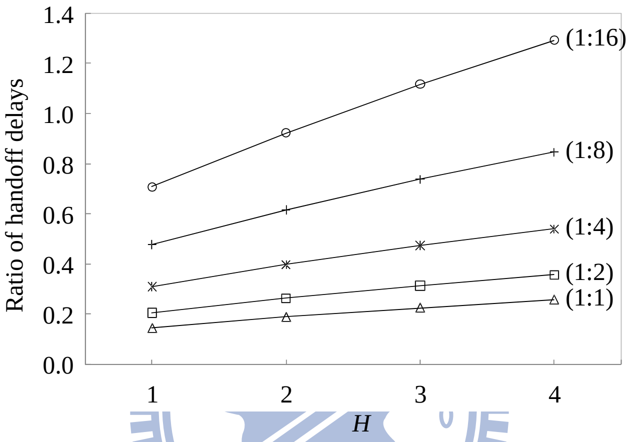
From Figure 3.10b it can be seen that, when n is 5, our approach results in the least handoff delay. Indeed, handoff delay in our architecture appears indistinct if $n \geq 3$. This finding is justifiable, since as n grows larger, a mobile station is more likely to experience intra-MPP handoff upon a move (saving costly IEEE 802.1X authentication) but meanwhile suffer more hop-by-hop transmissions of 4-Way Handshake messages between MAP and MPP. In other words, the increasing overhead of multi-hop transmissions for 4-Way Handshake messages may counteract the benefit brought by a larger cluster when n becomes sufficiently large. For IEEE 802.11i, a similar argument applies. That is, the larger n grows, the more multi-hop transmissions (longer delay) of EAP traffic requires. So L_S increases with n.

As regards the effect of different network topologies, Figure 3.10c shows handoff delays relating to the changes of mean hop count H between a MAP and the MPP. Figure 3.10c results from experiments by varying the parameter S in Equation (3.4). Varying S implies a floating number of MAPs with certain hops away from the MPP, thereby leading to different interconnectivity of MAPs. Performance results show that our approach can reduce handoff delay of the counterpart scheme by an appreciable amount, resulting in a reduction of around 63%.

Let us investigate under what conditions the two subject schemes exhibit comparable handoff delay. To gain fairer bases for comparison, we vary L_{1X} in a large extent representing possible scenarios. Here suppose that our approach operates with P_{PF} preset to 1.0. Supposing that both schemes are made produce nearly identical handoff delay, we are now concerned with the relationship between P_{PF} in the IEEE 802.11i scheme and the cluster parameter n. Figure 3.11a indicates that our proposed approach achieves almost equivalent effectiveness resulting from the



(a) P_{PF} versus n



(b) Handoff delay comparison versus H

Figure 3.11: Handoff delay of our approach relative to the counterpart scheme.

IEEE 802.11i scheme with P_{PF} ranging between 0.15 and 0.3. This means that effectively our approach functions as if the IEEE 802.11i scheme were operating with high likelihood of 70% to 85% successful preauthentication, without resorting to additional facility like handoff prediction techniques, network topology information, or profiles of handoff behavior.

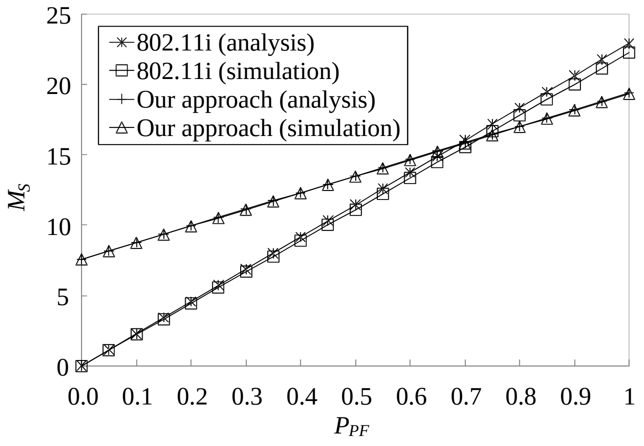
Observe that system-wide performance is also subject to which entities enact the role of IEEE 802.1X authenticator and how frequent reauthentications or PTK updates take place. When a MAP acts as an authenticator, the system involves heavier outlays for PMK distribution and reauthentication, but less overhead for

4-Way Handshake due to PTK updates. On the other hand, an MPP behaving as an authenticator contributes to fewer messages for PMK distribution and reauthentication, yet at the expense of more signaling overhead for renewing PTKs in that 4-Way Handshake always goes to MPP. Similar arguments for handoff delay apply here as well. Since IEEE 802.1X (re)authentication (PMK updates) and PTK updates occur at times, their resulting effects are of interest. Figure 3.11 compares their incurred delays in different cases of PMK or PTK updates carried out at varying frequencies with different hop count H. It can be seen that our approach remains advantageous unless PTK updates appear comparatively too often. However, as evidenced in [14], a PTK is arguably allocated with lifetime in order of several hours. Therefore it is deduced that PTK updates are mostly occasional and less likely to cause overwhelming counteraction.

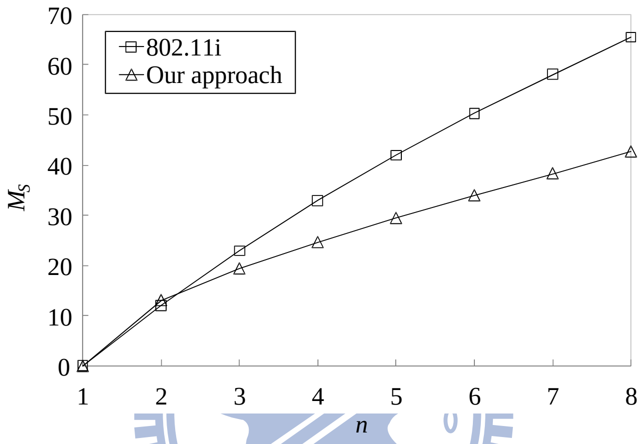
Handoff signaling traffic is another performance aspect. In this regard, Figure 3.12a shows the relationship between M_S and P_{PF} when $n = 3$. Again, this figure exhibits osculating results from analytical and simulation models without apparent discrepancies. Yet this figure shows that our approach may lose its advantage over the IEEE 802.11i scheme in certain circumstances. To be more specific, when $P_{PF} < 0.7$, our approach generates more signaling traffic than IEEE 802.11i scheme does. This is mainly because our proposed approach assumes a centralized architecture, where any generated 4-Way Handshake messages are conveyed to some MPP over multi-hop mesh links, causing more transmissions than those induced by IEEE 802.1X authentication. Nevertheless, Figure 3.12a depicts that signaling traffic is kept low in few tens, placing insignificant burden on the network.

Figure 3.12b shows M_S under different cluster sizes for $P_{PF} = 1.0$. Results indicate that overall handoff signaling traffic generated by our approach is less than that by IEEE 802.11i scheme except for $n = 2$. (For $n = 2$, our signaling traffic is slightly higher.) This reflects the cost effectiveness of our approach in that the benefit due to restricting most signaling traffic within a larger cluster outvalues the potential overhead of increased multi-hop transmissions.

Further extensive evaluation reveals that signaling traffic in both schemes grows



(a) M_S versus P_{PF} ($n = 3$)



(b) M_S versus n ($P_{PF} = 1.0$)

Figure 3.12: Signaling traffic of two subject schemes.

linearly with average hop count H between MAP and MPP. Given $n = 3$, $P_{PF} = 1.0$, and H ranging over the interval $[0, 4]$, for instance, we find that IEEE 802.11i scheme generates signaling traffic in quantity of 8.33 to 54.98, whereas our approach in 12.94 to 33.59. Our approach is found to introduce less handoff signaling traffic than the counterpart scheme does throughout our experiments. This implies that our design caters better for a variety of network topologies.

Cross referencing Figures 3.10 to 3.12 gives a more thorough view of how and when the proposed approach outperforms the conventional scheme. These figures delimit the usefulness of our approach in practical sense, assuring that our approach is suited where a mesh network is neither too small nor unduly large in scale. (An excessively large network is liable to suffer overwhelming overhead of message exchanges.) The foregoing discussions corroborate that a network of MAPs or MAPs with maximum 3 hops to the MPP adopting our approach operates fairly well, even in the absence of any preauthentication support.

To summarize, performance results indicate that our approach is promising if MPs or MAPs are organized into MPP-centric clusters, each with approximately 19 to 37 MPs interconnected to an MPP. Such cluster size of several tens in order is agreeable to the suggested scale of a mesh network by IEEE 802.11s [13]. As another remark, our approach allows MPs to be clustered in moderate size such as to minimize IEEE 802.1X authentication while keeping inter-MP communication within the cluster manageable. This facilitates a battery-powered mobile station to balance its power consumption and handoff performance. Note that our scheme comes to some limitation in scalability (overloading the MPP) when cluster size grows unduly large. Nevertheless, Figures 3.10 to 3.12 lead us to argue that the proposed scheme suffices for real mesh networks.

3.3 Summary

We presented an approach to secure fast handoff in an IEEE 802.11s mesh network, a field that warrants closer study. Our approach reduces repeated IEEE 802.1X authentications and encryption/decryption processing of data frames for mobile stations by an appreciable amount. Our development exhibits several features. First, the security level in line with IEEE 802.11i is maintained. Section 3.1.4 has reasoned that our approach does not lead to security vulnerability. Nor did we trade performance for security and robustness to the extent that security requirements for IEEE 802.11i or 802.11s are unduly weakened. Further, our scheme prevents pairwise master keys from being distributed over the wireless medium, protecting against any unauthorized access from the air.

Second, following our design tenet, the entities holding pairwise mater keys are shifted to MPP in a way that fast handoff can be accomplished by bypassing prohibitive IEEE 802.1X authentication during handoff. Such a design enables the setup of an end-to-end secure communication channel–a bidirectional MAC tunnel–between a station and its MPP to keep data frame exchanges intact in the mesh environment.

Meanwhile our approach is characterized by backward and forward compatibility; we leverage the use of standard protocols at the station side without tailoring current protocol fabric. Hence interoperability with IEEE 802.11i- or 802.11s-conformant devices is maintained. Besides, our approach may serve as an optional but not a costly add-on to any IEEE 802.11-based mesh network. Note that original security and routing mechanisms predefined in IEEE 802.11s can fully co-operate with our development.

Moreover, Section 3.2 provided an analytical model to formulate handoff delay and incurred signaling overhead. With our analysis in place, an optimal number of MAPs managed by an MPP (or the best location of an IEEE 802.1X authenticator somewhere between a MAP and the MPP) can be determined accordingly. Performance results have demonstrated the strengths of our proposed approach and augur well for our design in practical application.

To conclude this study, we outline several future directions to work on. First, we continue implementing our approach using hostapd, a widespread open source platform. Second, it is apropos to relate our analytical model to evaluating handoff methodology in other contexts, e.g., IEEE 802.11r or 802.16e (WiMAX). Derived analytical results can give an indication of how these handoff processes perform in a quantitative fashion, which may become part of feasible studies on any subsequent development. Third, we note that reauthentication mechanisms of EAP methods shall undergo some optimization for seamless handoff in mobile wireless environments. There has been active work in progress in a new IETF (Internet Engineering Task Force) work group called *Handover Keying*. We are keeping closely aligned with the development of the new work group. Lastly, our treatment can be extended to other types of multi-hop wireless networks by allowing network entities within the same do-main (analogous to MPs in the mesh architecture) to share out the workload of 4-Way Handshakes and encryption and decryption in frame delivery. While such an extension entails context transfers among network entities, access points accommodating mobile stations need to keep with adjustments of new authenticators. This suggests a topic that warrants more thorough investigation in the future.

Chapter 4

End-to-End Channel Allocation Scheme for Wireless Mesh Networks

In this study, we propose an End-to-End Channel Allocation Scheme, henceforth referred to as EECAS, which can provide load and interference awareness paths for end-to-end packet deliveries in WMNs. We model the channel allocation problem as a directed graph and derive a solution via set operations. By applying the proposed approach and path metrics, channel allocation for an end-to-end communication path can be easily determined and the throughput can also be maintained. As shall be seen in Section 4.3, performance results demonstrate that the proposed mechanism improves channel utilization and better end-to-end throughputs than other approaches.

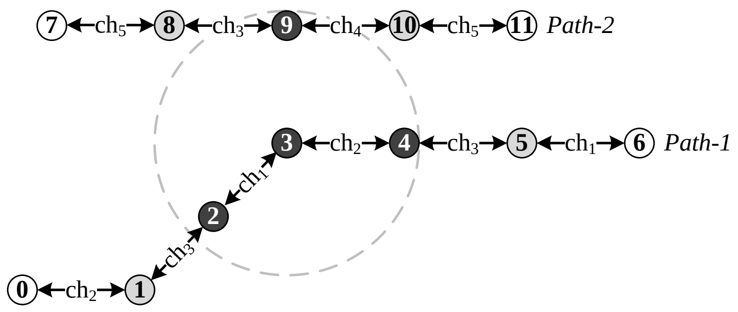
4.1 Overview

EECAS extends radio-frequency-slot method as in SSCH and takes end-to-end issues into account. Although separating two-way communications on two different channels is not a new finding, we observe that channel utilization can be improved while the concept is introduced into our scheme.

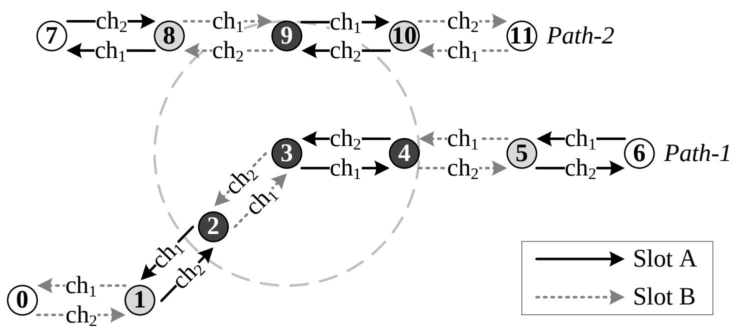
With the benefits of radio-frequency-slot method, we can allocate channels for end-to-end paths and ensure that all paths do not suffer intra-path and inter-path interferences, if we have a mechanism to determine which channels are interfered on a specified hop of path. For example, as shown in Figure 4.1, there are 12 nodes, denoted as 0 to 11, and two routing paths, named *path-1* and *path-2*. Each node is equipped with two radio interfaces. In other words, each node can be allocated up to two channels on each slot. Figure 4.1a shows conventional channel allocation with radio-frequency-slot. In this figure, we try to allocate channels for each hop of the two paths on the same slot. To avoid intra-path interference, we cannot allocate same channel within three continuous hops of a path, so both *path-1* and *path-2* have to take three different channels. Moreover, to avoid inter-path interference, two neighboring nodes that is belonging in the two paths respectively cannot be allocated same channels. In Figure 4.1a, in-use channels of *node-3*, which are ch_1 and ch_2, have to differ from channels of *node-9*, which are ch_3 and ch_4. In summary, the conventional radio-frequency-slot is workable if there are at least five non-overlapping channels. The requirement quickly rises as network density or number of radios increases.

On the other hand, as shown in Figure 4.1b, both *path-1* and *path-2* just take two different channels for avoiding intra-path interference, if we can separate Tx and Rx of transmissions on two different channels and a node's communications with different neighbors on different slots. For example, at slot A, *node-3* uses ch_1 for sending and ch_2 for receiving packets from *node-4*. And, *node-3* also uses ch_2 for sending and ch_1 for receiving packets from *node-2* at slot B. Besides, inter-path interference is also easily avoided if we can ensure that receiving channel(s) of a node is not interfered by the node's neighbors.

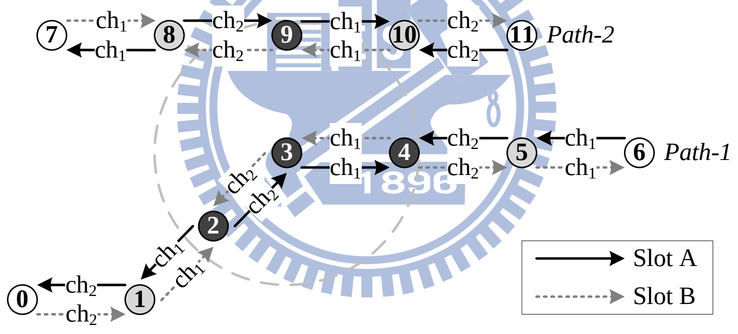
Furthermore, Figure 4.1c shows a alternative allocation result compared to Figure 4.1b. If we can separate transmissions not only on two different channels but also on different slots, the alternative result could achieve higher performance in end-to-end delay due to less slot switching. For example, in Figure 4.1c, the trans-



(a) Conventional radio-frequency-slot



(b) Extended radio-frequency-slot



(c) Extended radio-frequency-slot with less slot switching

Figure 4.1: Comparison between conventional and extended radio-frequency-slot.

missions between *node-2* and *node-3* are separated on different slots, and therefore we can allocate different channels for the Tx from *node-2* to *node-3* and the Tx from *node-3* to *node-4* but place them on a same slot. Then, packets can be fast delivered from *node-2* to *node-4* without slot switching.

A node can communicate with each of its all neighbors by applying the radio-frequency-slot method even though the node could be equipped with any number

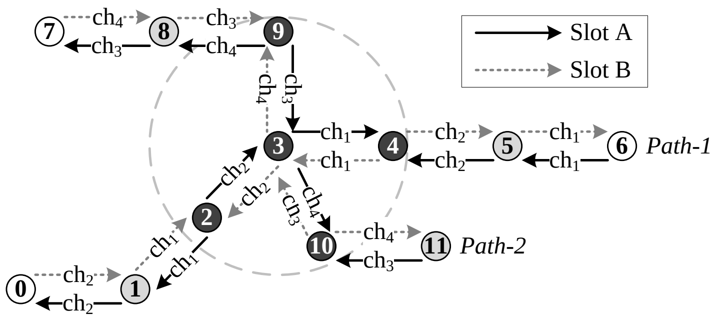


Figure 4.2: Easing the hot-spot problem by adding radios.

of radios that is less than the number of its neighbors. Moreover, the numbers of radios on nodes do not be equal to one another. Thus, our EECAS is also easily applied to this asymmetric radio architecture, which is helpful to address the hot-spot problem in a WMN. For example, in Figure 4.2, we equip *node-3* with four radios and other nodes with two radios. There are two routing paths and *node-3* is the crossover node (it also could be a mesh portal in a WMN) of the two paths. In this scenario, *node-3* can be allocated four different channels on each slot to simultaneously communicate with its neighbors, and thus end-to-end throughput of the two paths can be maintained due to without sharing bandwidth at the crossover node. The number of total in-use channels is four. As a result, the hot-spot problem can be easily addressed by adding radio while we apply two-way separation in asymmetric radio WMNs to reduce the waste of channels.

In summary, we find that separating transmissions on different channels can improve channel utilization as well as interference can be easily determined in distributed fashion. Therefore, we propose a general dynamic, distributed mechanism, which extends the radio-frequency-slot method and takes end-to-end issues into account, to allocate channels to paths free from both intra-path and inter-path interferences in asymmetric multi-radio WMNs. Our mechanism is also helpful to address hot-spot problem by adding radio interfaces.

4.2 The Proposed Scheme

4.2.1 Definitions

We model a WMN as a directed graph $G(V, E)$ and derive the solution of the channel allocation based on set operations. Table 4.1 lists the symbols used in proposed EECAS. We assume that V is a set of vertices that represent all mesh nodes in a WMN, and u and v are mesh nodes in the set V. Then we define the relationships between u and v as:

$$\forall u,\ v \in V,\ D_{(u,v)} < R_{com} \Leftrightarrow (u, v) \in E\ \wedge\ (v, u) \in E \tag{4.1}$$

Equation (4.1) reveals that if the distance between u and v, denoted as $D_{(u,v)}$, is less than the communication range of mesh nodes, denoted as R_{com}, there should have two directed edges, denoted as (u, v) and (v, u), in the graph $G(V, E)$. Each edge indicates a logical channel that offers directional communication between nodes. In addition, a sequence of edges indicates a directional path in a WMN.

Moreover, the set of neighbors of a mesh node u, denoted as N_u, is defined as:

$$\forall u \in V,\ N_u \equiv \{x \mid \forall x \in V,\ D_{(u,x)} < R_{com}\} \tag{4.2}$$

Equation (4.2) shows that for any mesh node in V, denoted as x, if the distance between u and x is less than R_{com}, the node x is included in N_u and x is one of neighbors of u. In other words, all nodes in the set N_u are within the signal coverage of u so that these nodes can directly communicate with u.

Then, we define the available logical channels in a multi-radio multi-channel WMN. A mesh node may have multiple radio interfaces. In addition, the transmission time could be divided into several time slots, like *Time Division Multiple Access* (TDMA). In each time slot, the radio interface can switch to a logical channel and communicate with peer nodes which are also in the same channel. To consider timing inaccuracies and different propagation delays of individual nodes, a guard period is usually introduced between successive time slots and therefore perfect time synchronization is not necessary. In recent years, several decentralized synchronization

Table 4.1: Symbols used in EECAS.

Symbol	**Description**
V	A set of vertices, which represent nodes in a WMN
E	A set of edges, which represent logical channel between transmitter and receiver in a WMN
$G(V, E)$	A directed graph, which represents a WMN including nodes (V) and logical channels (E)
R_{com}	Radius of communication range of the node in a WMN
C^*	A set of all non-overlapping channels or *Cartesian product* of non-overlapping channels and time-slots
$D_{(u,v)}$	Distance between nodes u and v
$S_u(k)$	State of channel slot k of node u, $\forall k \in C^*$
N_u	A set of neighbor nodes of node u
$ACS_{(u,v)}$	A set of channel slots that have been allocated on the edge (u, v)
PCS	A set of parity channel slots
$FCS_{(u,v)}$	A set of free channel slots on the edge (u, v)
$FCSC_{(u,v)}$	A set of free channel slot combinations on the edge (u, v)
$CCSC_{(u,v)}$	A set of candidate channel slot combinations on the edge (u, v), which is a subset of the $FCSC_{(u,v)}$

algorithms, such as [52,53], have been proposed and can get the accuracy of no more than 10 microseconds. If EECAS uses a 10 ms time slot, the overhead due to the guard period is 10 μs / 10 ms = 0.1%. Therefore, the logical channel set can be expressed by a *Cartesian product* of non-overlapping channels and time slots. For example, assume let C denote the set of non-overlapping logical channels 1, 2, and 3, i.e., $C = \{1, 2, 3\}$. Further, suppose that S is the set of time slots containing slots a, b, c, and d. In other words, $S = \{a, b, c, d\}$. Thus, the new set of available logical channels can be denoted as $C^* = C \times S = \{\langle 1, a\rangle,\ \langle 2, a\rangle,\ \cdots,\ \langle 3, d\rangle\}$, which includes 12 combinations. The element $\langle 1, a\rangle$ represents channel 1 at time slot a. For simplicity, we refer to element $\langle 1, a\rangle$ as channel slot $\langle 1, a\rangle$.

Additionally, we introduce two terms: *directional logical channel* and *non-directional logical channel*. As mentioned earlier, we suggest separating packets in different directions and scheduling them on different logical channels. For example, data and acknowledgment packets in a path between neighboring nodes are separated into two directional logical channels. On the other hand, a logical channel with data and acknowledgment packets together without separation is called a non-directional logical channel.

4.2.2 Channel Allocation Mechanism

The proposed EECAS has two phases of operations. The first is called *Information Maintenance* and composed of two operations, *Channel State Table (CST) Maintenance and Exchange*, and *Free Channel Slot (FCS) Discovery*. All mesh nodes perform the procedures of the first phase periodically and can update information for the subsequent channel allocation. The second phase is called *Channel Allocation* and the phase has two operations, *Free Channel Slot Combination (FCSC) Computing* and *Channel Decision*. The second phase is performed in an on-demand basis and it is triggered when a communication is newly established or modified. We illustrate these procedures by using the examples below.

CST Maintenance and Exchange

To find the channel slot which is in-use, all mesh nodes need to maintain a CST and the state of each channel slot. Also, they have to broadcast their own CST to their neighbors, defined in Equation (4.2). As shown in Figure 4.3, there are six mesh nodes, named A, B, $\cdots$, F, and D and E equipped with two radio interfaces. The channel slot set $C^* = \{\langle 1, a\rangle,\ \langle 2, a\rangle,\ \cdots,\ \langle 3, d\rangle\}$ includes 12 available channel slots, and each node maintains its own CST. The channel slot state has five conditions: transmitting (T), receiving (R), interfered (I), free (F) and parity (P). Moreover,

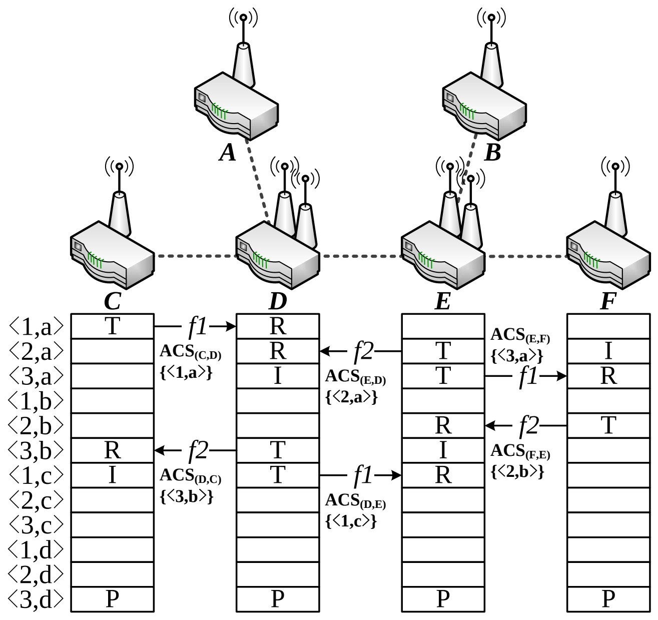


Figure 4.3: In-use channel awareness by applying the Channel State Tables (CSTs).

each node also keeps the life time of channel slots in its own CST.

As shown in Figure 4.3, there is a data flow, named $f1$, from C to F, through D and E. Furthermore C uses $\langle 1, a\rangle$ to send packets to D so that C marks the state of $\langle 1, a\rangle$ as 'T' in its CST and associates the life time $t_{\langle 1,a\rangle}$ of the channel slot. The life time is updated whenever packet is sent or received through the channel slot, and the channel slot is released if the life time expires. The channel slot $\langle 1, a\rangle$ is also put into the set of *Allocated Channel Slots* (ACS) for the corresponding edge, denoted as $ACS_{(C,D)}$, indicating an in-use channel slot on the edge. Moreover, D marks the channel slot as 'R' in its CST because it receives packets from C through the channel slot.

Furthermore, we can define an interfered channel slot by Equation (4.3). A node u deems that a channel slot k has interference, denoted as $S_u(k) = I$ and maintains the state in its CST, if at least one neighbor node x of the node u is using the channel slot to send packets, denoted as $S_x(k) = T$, and the node u is not the receiver. The condition is updated by sensing wireless carrier and/or after every CST exchanges. For example, A marks $\langle 1, a\rangle$ as 'I' in its CST because D uses the channel slot to send packets and A is not the receiver. Finally, if a channel slot is not in the above states, the channel slot is marked as blank to represent its availability.

$$\forall u \in V,\ k \in C^*,\ S_u(k) = I \Leftrightarrow \exists x \in N_u,\ S_x(k) = T \tag{4.3}$$

To avoid channel conflict, a mesh node needs to know every channel slot's usage in the interference range. To achieve this goal, every node periodically broadcasts its own CST to its neighbors and keeps all CSTs of its neighbors. Therefore, every node is aware of interfered channels of neighbors. This information implies channel slot usages of three hops away so that every node can know what channel slot usage in the interference range is. For example, C knows that D deems $\langle 3, a\rangle$ as interfered, as shown in Figure 4.3, so C surmises that the channel slot had been allocated within three hops away.

Finally, all nodes have to reserve a parity channel slot for control message exchange. For example, in Figure 4.3, all nodes reserve channel slot $\langle 3, d\rangle$ and mark the channel slot as 'P' in their own CSTs to indicate the parity channel slot. The detail of control mechanism is described in Section 4.2.3.

FCS Discovery

To allocate channel slots without co-channel interference, every node needs to discover free channel slots which have not been allocated. A mesh node performs the discovery procedure whenever the node receives CST of its neighbor. To solve channel allocation with the directional logical channels, the problem can be seen as allocating channel slots to directed edges. Therefore, a node needs to discover free channel slots from its outgoing edges. Equation (4.4) presents the conditions of free channel slots with the directional logical channels. We assume that there are two neighboring nodes, named u and v, so that two directed edges, denoted as (u, v) and (v, u), exist. For the outgoing edge (u, v) of u, node u deems channel slot k as free on the edge and puts the channel slot into the $FCS_{(u,v)}$. This denotes the set of free channel slots on the edge (u, v), if the following three conditions are satisfied: the state of k in the CST of u is free or interfered, denoted as $S_u(k) = F/I$; the state in the CST of v is free, denoted as $S_v(k) = F$; and the state in CSTs of all neighbors

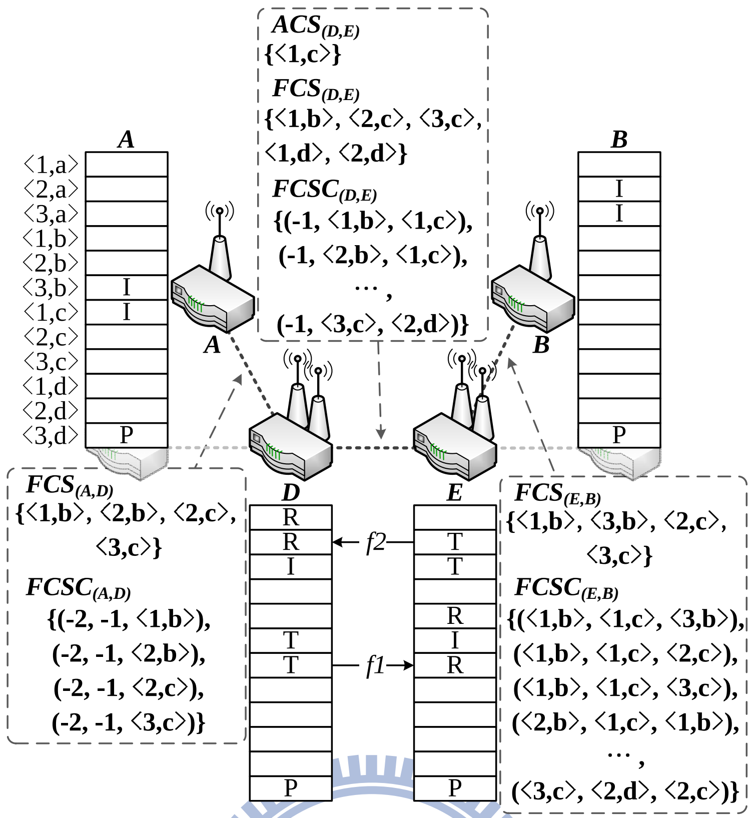

ACS(D,E)
{⟨1,c⟩}
FCS(D,E)
{⟨1,b⟩, ⟨2,c⟩, ⟨3,c⟩, ⟨1,d⟩, ⟨2,d⟩}
FCSC(D,E)
{(-1, ⟨1,b⟩, ⟨1,c⟩), (-1, ⟨2,b⟩, ⟨1,c⟩), … , (-1, ⟨3,c⟩, ⟨2,d⟩)}
A
⟨1,a⟩ ⟨2,a⟩ ⟨3,a⟩ ⟨1,b⟩ ⟨2,b⟩ ⟨3,b⟩ ⟨1,c⟩ ⟨2,c⟩ ⟨3,c⟩ ⟨1,d⟩ ⟨2,d⟩ ⟨3,d⟩
I I P
B
I I P
A
B
D
E
R R I T T P
T T R I R P
f2
f1
FCS(A,D)
{⟨1,b⟩, ⟨2,b⟩, ⟨2,c⟩, ⟨3,c⟩}
FCSC(A,D)
{(-2, -1, ⟨1,b⟩), (-2, -1, ⟨2,b⟩), (-2, -1, ⟨2,c⟩), (-2, -1, ⟨3,c⟩)}
FCS(E,B)
{⟨1,b⟩, ⟨3,b⟩, ⟨2,c⟩, ⟨3,c⟩}
FCSC(E,B)
{(⟨1,b⟩, ⟨1,c⟩, ⟨3,b⟩), (⟨1,b⟩, ⟨1,c⟩, ⟨2,c⟩), (⟨1,b⟩, ⟨1,c⟩, ⟨3,c⟩), (⟨2,b⟩, ⟨1,c⟩, ⟨1,b⟩), … , (⟨3,c⟩, ⟨2,d⟩, ⟨2,c⟩)}

Figure 4.4: An example of Free Channel Slot (FCS) discovery and Free Channel Slot Combination (FCSC) computing.

of u is not receiving packets. As mentioned above, u owns its CST and CSTs of its neighbors, so that u can obtain $FCS_{(u,v)}$ and maintain the information.

$$\begin{aligned} &\forall u,\ v \in V,\ k \in C^*,\ k \in FCS_{(u,v)} \\ \Leftrightarrow\ &S_u(k) = F/I\ \wedge\ S_v(k) = F\ \wedge\ \forall x \in N_u,\ S_x(k) \neq R \end{aligned} \tag{4.4}$$

A simple example of the FCS discovery with directional logical channels is given in Figure 4.4, where A and D are neighboring nodes such that A can receive D's CST periodically. When A receives D's CST, A performs the FCS discovery algorithm, shown in Figure 4.5, to update FCS of the outgoing edge (A, D). The result of $FCS_{(A,D)}$ is $\{\langle 1, b\rangle,\ \langle 2, b\rangle,\ \langle 2, c\rangle,\ \langle 3, c\rangle\}$, indicating that allocating any one of these channel slots for the edge (A, D) does not affect existing flows. Similarly, D can also update its outgoing edges $FCS_{(D,A)}$, $FCS_{(D,B)}$, $FCS_{(D,E)}$, and so on.

In addition, channel slot $\langle 1, a\rangle$ is not included in $FCS_{(E,B)}$ since that E has two radio interfaces and the two interfaces had been assigned to other channel slots at

Algorithm 1 FCS discovery

```
1: $\mathrm{FCS}_{(src,dst)} \Leftarrow \{\emptyset\}$
2: for each $c \in$ channel slot set C* do
3:   if $src$ and $dst$ has at least one interface to be allocated the channel slot $c$ then
4:     if ($\mathrm{S}_{src}(c)$ = 'F' or $\mathrm{S}_{src}(c)$ = 'I') and $\mathrm{S}_{dst}(c)$ = 'F' then
5:       $\mathrm{NonRecv}_{(src,dst)} \Leftarrow$ 'true'
6:       for each $n \in$ neighbor set $\mathrm{N}_{src}$ do
7:         if $\mathrm{S}_n(c)$ = 'R' then
8:           $\mathrm{NonRecv}_{(src,dst)} \Leftarrow$ 'false'
9:         end if
10:      end for
11:      if $\mathrm{NonRecv}_{(src,dst)}$ = 'true' then
12:        $\mathrm{FCS}_{(src,dst)} \Leftarrow \mathrm{FCS}_{(src,dst)} \cup \{c\}$
13:      end if
14:    end if
15:  end if
16: end for
```

Figure 4.5: FCS discovery algorithm.

time slot a. The node has no interface to be assigned at the same time slot. Thus, channel slot $\langle 1, a\rangle$ is removed from the $FCS_{(E,B)}$. The proposed FCS discovery algorithm shown in Figure 4.5 can also handle this case.

We also demonstrate how to obtain the free channel slots when the conventional non-directional logical channel is applied. In this case, the channel allocation problem is to map channel slots to the edges of both directions between neighboring nodes. This is because the conventional non-directional logical channel assumes no direction on the link, and the communication packets may be exchanged in both directions. Therefore, the edges in both directions between two nodes are occupied if the link is assigned to the nodes. For example, u deems channel slot k as free on both edges (u, v) and (v, u), if the following conditions are satisfied: the k is free in the CSTs of both u and v, denoted as $S_u(k) = F$ and $S_v(k) = F$; and none of k in the CSTs in all neighbors of u and v are receiving packets. Thus, channel slot k is included in both $FCS_{(u,v)}$ and $FCS_{(v,u)}$. In addition, u and v need to exchange the information for FCS discovery because none of them have sufficient information to know the channel states in the CSTs of another's neighbors.

$$\begin{aligned}\forall u,\ v \in V,\ k \in C^*,\ k \in FCS_{(u,v)}\ \wedge\ k \in FCS_{(v,u)} \\ \Leftrightarrow S_u(k) = F\ \wedge\ \forall x \in N_u,\ S_x(k) \neq R \\ \wedge\ S_v(k) = F\ \wedge\ \forall y \in N_v,\ S_y(k) \neq R\end{aligned} \tag{4.5}$$

FCSC Computing

By applying FCS discovery, we individually allocate channel slots for different edges and can avoid the inter-path interference. However, the end-to-end issue has not yet been considered. The end-to-end issue is to ensure that all edges along a routing path from source to destination can be successfully allocated with sufficient bandwidth for a specific flow. To achieve this goal, we have to find out channel slot mappings along the path and the channel slots for the flow have sufficient resources. The FCSC computing derives all channel slot mapping combinations for a flow's path and it is triggered when a node initiates a new data flow or modifies the existing flow. The procedure is performed node by node from the flow's source to the flow's destination. In other words, the nodes along the routing paths have to perform the FCSC computing algorithm individually in order to derive channel allocations. The algorithm for this is shown in Figure 4.6.

We illustrate the FCSC computing using an example, shown in Figure 4.4. We assume that A initiates a flow, called $f3$, from itself to B so that it triggers the FCSC computing procedure and then uni-casts a *Flow Request* (FREQ) message that carries the current outgoing edge's FCSC to the destination B. Then A needs to determine which combinations are available on the edge (A, D), and puts these combinations into the corresponding set $FCSC_{(A,D)}$. Note that a channel slot mapping combination for an edge of a flow's path is defined as a three-tuple that states available channel slot mappings for the neighboring three edges along the path. This definition ensures that any consecutive three edges of a flow's path are not allocated the same channel slot to avoid the intra-path interference. Equation (4.6) shows the conditions of the combination on the first edge. Node A may pick in-use channel slots from the $ACS_{(A,D)}$ if the remaining bandwidth of the channel slots can sat-

Algorithm 2 FCSC computing

```
1: FCSC$_{(nbr,src)}$ $\Leftarrow$ the FCSC that is carried by FREQ
   and is sent from neighbor node $nbr$
2: FCSC$_{(src,dst)}$ $\Leftarrow \{\emptyset\}$
3: if ACS$_{(src,dst)} \neq \{\emptyset\}$ and $\exists a \in$ACS$_{(src,dst)}$ satisfy the
   flow's bandwidth requirement then
4:   if $src$ is the flow's source node then
5:     FCSC$_{(src,dst)}$ $\Leftarrow$ FCSC$_{(src,dst)} \cup \{(-2,-1,a)\}$
6:   else
7:     for each $(c_1,c_2,c_3) \in$ FCSC$_{(nbr,src)}$ do
8:       if $c_2 \neq a$ and $c_3 \neq a$ then
9:         FCSC$_{(src,dst)}$ $\Leftarrow$ FCSC$_{(src,dst)} \cup \{(c_2,c_3,a)\}$
10:      end if
11:    end for
12:  end if
13: else
14:  for each $c \in$ FCS$_{(src,dst)}$ do
15:    if $src$ is the flow's source node then
16:      FCSC$_{(src,dst)}$ $\Leftarrow$ FCSC$_{(src,dst)} \cup \{(-2,-1,c)\}$
17:    else
18:      for each $(c_1,c_2,c_3) \in$ FCSC$_{(nbr,src)}$ do
19:        if $c_2 \neq c$ and $c_3 \neq c$ then
20:          FCSC$_{(src,dst)}$ $\Leftarrow$ FCSC$_{(src,dst)} \cup \{(c_2,c_3,c)\}$
21:        end if
22:      end for
23:    end if
24:  end for
25: end if
```

Figure 4.6: FCSC computing algorithm.

isfy the bandwidth requirement of the flow, or node A may collect all free channel slots from the $FCS_{(A,D)}$. Then these channel slots, denoted as c, are placed on the third channel slot mapping of combinations to indicate available channel slots on the edge (A, D). Because node A is the source of the flow $f3$, the first and the second channel slot mappings of combinations are not necessary. In our design, A places two constants "-2" and "-1" on the first and the second channel slot mapping to indicate that no channel slot is needed. In this example, $ACS_{(A,D)}$ is empty, so that A can collect channel slots only from $FCS_{(A,D)}$. The resulting $FCSC_{(A,D)}$, shown in Figure 4.4, is carried by the FREQ message which will be sent to the next node.

$$FCSC_{(A,D)} \equiv \{(-2,-1,c) \mid \forall c \in FCS_{(A,D)} \ \vee \ c \in ACS_{(A,D)}\} \tag{4.6}$$

Upon receiving the FREQ message, D also performs the FCSC computing algorithm to derive the second edge's combinations $FCSC_{(D,E)}$ and then puts it into the FREQ message. In Equation (4.7), D may also pick channels from $ACS_{(D,E)}$ or $FCS_{(D,E)}$ and then performs the function $Comb$ to ensure that all combinations of the selected channel slots and the received FCSC satisfy the definition of channel slot mapping combination. In this example, we assume that in-use channel slot $\langle 1, c \rangle$ in $ACS_{(D,E)}$ satisfies the bandwidth requirement of the flow and D picks the channel slot. Then the channel slot $\langle 1, c \rangle$ and the combinations in $FCSC_{(A,D)}$ are fed into the function $Comb$ to derive $FCSC_{(D,E)}$, which is

$$FCSC_{(D,E)} \equiv \{Comb(p,\ c) \mid \forall p \in FCSC_{(A,D)} \wedge (\forall c \in FCS_{(D,E)} \vee c \in ACS_{(D,E)})\} \tag{4.7}$$

Recall that to avoid the intra-path interference, we cannot allocate the same channel slot for consecutive three edges of a flow's path. The function $Comb$, shown in Equation (4.8), ensures that the second and third mappings, denoted as c_2 and c_3, of the input combination are not equal to the input channel slot, denoted as c_4.

$$Comb((c_1, c_2, c_3),\ c_4) = \begin{cases} (c_2, c_3, c_4), & \text{if } c_2 \neq c_4 \text{ and } c_3 \neq c_4 \\ \emptyset\ , & \text{otherwise} \end{cases} \tag{4.8}$$

Note that the first mapping, denoted as c_1, of the input combination is not significant because the corresponding edge of c_1 is out of the interference range of current edge. In this example, only the channel slot $\langle 1, c \rangle$ and combinations in $FCSC_{(A,D)}$ are fed into the function and the result $FCSC_{(D,E)}$ is shown in Figure 4.4. In addition, the third mapping of all combinations in $FCSC_{(D,E)}$ is $\langle 1, c \rangle$ and the channel slot was allocated to the edge (D, E). This indicates that multiple flows sharing the same edges may be aggregated into the same channel slot on the common edges. Furthermore, a node may also collect all free channel slots from the FCS to derive the FCCS. For example, because $ACS_{(E,B)}$ is empty, E can only collect channel slots from $FCS_{(E,B)}$ and derive $FCSC_{(E,B)}$.

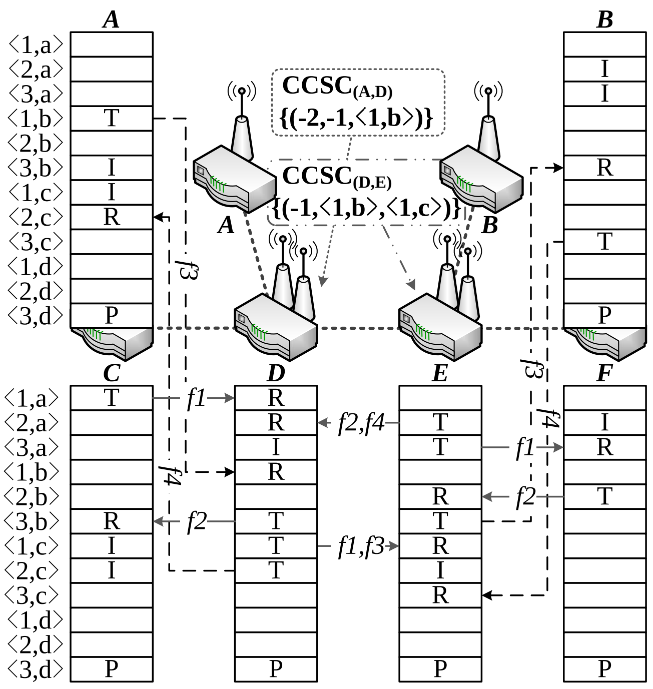


Figure 4.7: An example of channel allocation by applying the EECAS.

Channel Decision

When the destination receives the FREQ message carrying the last edge's FCSC, the FCSC computing procedure is completed. Then, the destination triggers the channel decision procedure and uni-casts a *Flow Reply* (FREP) message back to the source to confirm the resource allocation. The channel allocation of the edges in a path is confirmed in a reversed order. There may have several channel allocation choices which satisfy the requirements. A simple decision policy for the channel allocation is to randomly pick a combination from the set of *Candidate Channel Slot Combination* (CCSC) which is a subset of the FCSC. Initially, the last edge's CCSC is equal to the edge's FCSC. For example, the last edge's $CCSC_{(E,B)}$ is equal to the edge's $FCSC_{(E,B)}$ and B randomly picks a combination such as $(\langle 1,b\rangle, \langle 1,c\rangle, \langle 3,b\rangle)$ from $CCSC_{(E,B)}$, denoted as $fcs_{(E,B)}$. Then B configures channel slot $\langle 3,b\rangle$ as the receiving state in its CST and uni-casts the FREP message that carries the decision back to the source.

Upon receiving the FREP message, each node along the path configures the channel slot as the transmitting state and records the bandwidth requirement in its CST. For example, E configures channel slot $\langle 3, b\rangle$ as the transmitting state in its CST. As shown in Figure 4.7, the flow $f3$ has been established on the edge (E, B). One edge's channel decision determines subsequent edge's CCSC because a combination not only designates the current edge's mapping but also states mappings of the next two edges along the reversed path. For example, E needs to derive $CCSC_{(D,E)}$ before it makes the decision for edge (D, E). The $fcs_{(E,B)}$ carried by the FREP message determines which channel slot combinations are included in $CCSC_{(D,E)}$, shown in Figure 4.7. A combination in the $FCSC_{(D,E)}$ is also included in the $CCSC_{(D,E)}$, if the second and third mappings of the combination are equal to the first and second mappings of $fcs_{(E,B)}$. In this example, only one combination $(-1, \langle 1, b\rangle, \langle 1, c\rangle)$ satisfies this condition. The restriction indicates that E needs to decide only the first mapping of combinations in $CCSC_{(D,E)}$ because other mappings have been decided.

Finally, when the source receives the FREP message, the channel decision procedure and the second phase are completed and the source can send the flow's packets. As shown in Figure 4.7, flow $f3$ has been established from A to B and the flow $f4$ may be established to transmit $f3$'s acknowledgement packets. In addition, the flows $f1$ and $f3$ share the channel slot $\langle 1, c\rangle$ in their common edge (D, E). This indicates that the EECAS allows multiple flows to share or to contend capacity of a channel slot.

4.2.3 Control Mechanism

Dedicated Control Radio

Each node in the networks setting up a dedicated control radio is the simplest implementation for control mechanism. All nodes configure the dedicated radio to permanently switch on a common channel and then run the CSMA/CA on the radio. Therefore, control messages and synchronous information can be delivered with the radio. However, this dedicated radio approach is not efficient because the radio could be idle at most time.

Parity Channel Slot

A alternative approach to exchange control messages and synchronous information in our scheme is that all nodes reserve a common parity channel slot on each cycle, like SSCH, and configure the slot to switch on a common channel. Therefore, the nodes run the CSMA/CA on the parity channel slot so that collision can be avoided. In addition, we can reserve multiple parity channel slots within a cycle to improve performance in delivery of control messages. The overhead of parity channel slot reservation of a node x is $\frac{|PCS|}{|C^*| \times I_x}$, where $|PCS|$ is the number of parity channel slots per cycle; $|C^*|$ is the total number of channel slots per cycle; and I_x is the number of radio interfaces equipped on the node x.

Recovery

At the channel decision, some node could select unsuitable channel slot due to inconsistent CSTs. In our implementation, a simple recovery is to roll control packet back to the node that made the unsuitable decision if some node along the allocating path is aware of this. Therefore, the decision node is triggered to re-select channel slot. The worst case is FREQ timeout because the source node does not receive FREP message. In this case, the source node resends FREQ message and triggers FCSC computing procedure again, and the overhead of EECAS is increased.

4.2.4 Link Metric

The most popular link metric are ETX and its extension ETT. The ETX metric measures the expected number of transmissions, including retransmissions, needed to send a uni-cast packet across a link. The derivation of ETX starts with measurements of the underlying packet loss probability in both the forward and reverse directions, denoted by p_f and p_r, and then calculates the expected number of transmissions. Let p denote the probability that the packet transmission is not successful: $p = 1 - (1 - p_f) * (1 - p_r)$. Besides, the IEEE 802.11 MAC retransmits a packet whose transmission was not successful. Let $s(k)$ denote the probability that the packet is successfully delivered after k attempts: $s(k) = p^{k-1} * (1 - p)$. Finally, the expected number of transmission is:

$$ETX = \sum_{k=1}^{\infty} k * s(k) = \frac{1}{1-p} \tag{4.9}$$

Although the ETX metric performs better than shortest-path routing, it does not necessarily select good routes in environments with different data rates or multiple radios. The ETT metric addresses this issue by taking link bandwidth into account, and starts with the ETX and then multiplies by the link bandwidth to obtain the time spent in transmitting the packet. Let P denote the size of the packet (for example, 1024 bytes) and B the bandwidth (raw data rate) of the link. Then:

$$ETT = ETX * \frac{P}{B} \tag{4.10}$$

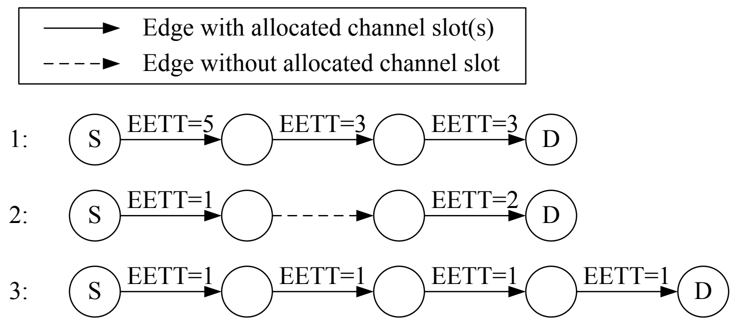
To calculate the transmitting time of an edge in our proposed scheme, we modify the conventional ETT metric, and define the Edge ETT (EETT) metric. For an edge (u, v), we denote this value by $EETT_{(u,v)}$ and define it as follows:

$$EETT_{(u,v)} = \frac{\sum_{i \in ACS_{(u,v)}} ETT_i}{|ACS_{(u,v)}|} \tag{4.11}$$

Note that the ETT is used to calculate the transmitting time of an allocated channel slot in our scheme. To calculate the value of channel slot i on edge (u, v), denoted by ETT_i, we need to know the joint loss rate of channel slot i and its reverse (allocated channel slot(s) on edge (v, u)). The rate can be approximated by sending the uni-cast packets on the specified channel slot and gathering the acknowledgement statistics. Therefore, according to Equation 4.11, the EETT is the average transmission time of allocated channel slots on a specified edge.

4.2.5 Path Metric

WECTT and SIM are two path metrics that resolve tension in choosing between shorter paths with lower total transmission overhead (e.g. sum of ETTs) and longer paths consisting of better performing links on different channels. We do not consider the channel-diverse on a path for easing intra-path interference because EECAS can


Edge with allocated channel slot(s)
Edge without allocated channel slot
1:
S
EETT=5
EETT=3
EETT=3
D
2:
S
EETT=1
EETT=2
D
3:
S
EETT=1
EETT=1
EETT=1
EETT=1
D

Path	Sum	H	M (β=0.1)	M (β=0.5)	M (β=0.9)
1	11	0	9.9	5.5	1.1
2	3	1	2.8	2.0	1.2
3	4	0	3.6	2.0	0.4

Figure 4.8: An illustration of the tradeoff by M values.

allocate channels for paths without interference problem. So that EECAS's path metric modifies the two path metrics to take overhead of channel allocating into account and is calculated as follows:

$$M = (1 - \beta) \cdot \sum_{j \in path} EETT_j + \beta \cdot H \tag{4.12}$$

The EECAS's path metric (M) is a linear combination of path overhead and allocating overhead with parameter β balancing the tradeoffs between them. The path overhead is approximated by the sum of EETTs along a routing path and the allocating overhead (H) is the sum of edges without allocated channel slot. The tradeoff is illustrated in the paths shown in Figure 4.8. We consider three possible paths from the source S to the destination D. The EETTs on the edges in these paths are shown in the figure. The M values for $\beta = 0.1$, $\beta = 0.5$ and $\beta = 0.9$ are also shown.

Consider the first path. This path is one of the shortest paths and all edges along this path have allocated channel slot(s). Unfortunately, its sum of EETTs, which is 11, is higher than sums of other paths. This fact implies that this path is occupied by heavy transmission loads or shared among multiple flows.

Now consider paths 2 and 3. The path 2 is another shortest path and its sum of EETTs is the smallest, but it contains an edge without allocated channel slot. If we assign $\beta = 0.1$, the overhead of allocating channel slot for those edges without allocated channel slot is undervalued and therefore this path is chosen. On the other hand, even through the length of path 3 is longer than path 2, this path does not contain edge without allocated channel slot and its sum of EETTs is close to the sum of path 2. In this case, if we assign β more than 0.5 to imply that the overhead of allocating channel slot is taken into account, path 3 is clearly better than path 2. We further explore the tradeoffs offered by β in Section 4.3.

4.3 Simulation Results

The simulations are conducted to evaluate the performance improvements by applying the proposed EECAS. The NS-2 simulator is used and various network configurations are considered. We assume IEEE 802.11a as the physical layer technology, and the raw data rate is set to 54 Mbps. A previous work [49] indicates that the channel switching delay of an IEEE 802.11a interface is between 40 μs to 80 μs, so we assume that a mesh node requires 40 μs to switch the channel. The performance of the EECAS mechanism is evaluated and compared with the previous channel allocation mechanisms.

4.3.1 Comparison between EECAS and IEEE 802.11a

In the first simulation, we evaluate the throughput improvement by applying the proposed EECAS against the conventional IEEE 802.11a. We configure ten mesh nodes, each with only one radio interface as a 2 row by 5 column mesh network, and every node can hear neighboring nodes. We then establish two four-hop flows which do not share any node or edge. The EECAS uses a 10 ms channel slot and therefore the switching overhead is 40 μs / 10 ms = 0.4%. We sequentially activate the two flows, called flow-1 and flow-2, at 0 and 15 seconds and the two flows generate constant bit rate (CBR) UDP packets.

Figure 4.9 shows the variation of the maximal throughput by applying the

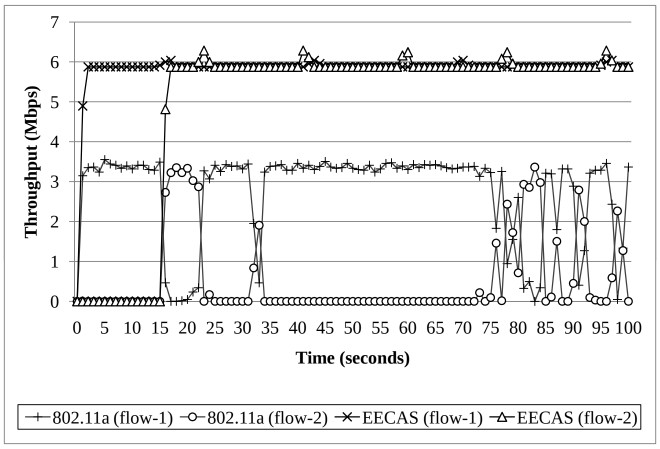


Figure 4.9: Throughput comparison between the EECAS and the conventional IEEE 802.11a.

EECAS to IEEE 802.11a and the conventional IEEE 802.11a. The horizontal axis indicates the simulation time in seconds and the vertical axis indicates the throughput in Mbps. As can be seen from the figure, the maximal throughput with the conventional IEEE 802.11a infrastructure mode, i.e. one-hop, could achieve approximately 13 Mbps, but an IEEE 802.11a WMN encounters serious interferences. For an IEEE 802.11a WMN, the throughput of flow-1 drops to 3.4 Mbps due to the intra-path interference. Moreover, flow-1 and flow-2 also suffer from the inter-path interference such that the two flows conflict each other and cannot maintain stable throughput.

One the other hand, the EECAS allows mesh nodes to use various channels. As shown in Figure 4.9, the maximal throughput of both flow-1 and flow-2 by applying the EECAS achieves approximately 5.9 Mbps. A single radio mesh node requires approximately half the time for receiving and half the time for forwarding packets, i.e. the optimal throughput is about 6.5 Mbps. The results show that the EECAS has about 9.23% overheads spent on parity channel slot, channel switching delay, and guard period. Moreover, the throughput of the two flows by applying the EECAS becomes more stable than that by employing the conventional IEEE 802.11a.

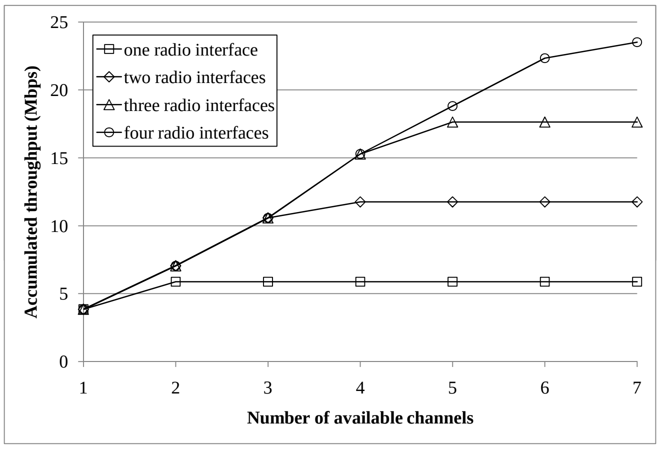


Figure 4.10: Aggregate throughput versus number of available non-overlapping channels.

4.3.2 The Impact of Available Resource

Figure 4.10 shows the effects of varying the number of available radio channels on the network aggregate throughput. The X-axis indicates the number of available non-overlapping channels, and the Y-axis indicates the aggregate throughput, which is the maximum throughput that a node can send or receive. These simulations were conducted on a 24-node grid network, and the maximum hop-count between any two nodes is limited to four hops. We also vary the number of radio interfaces per node to evaluate the impact of number of radio interfaces on the utilization efficiency of the channels. The aggregate throughput increases monotonically with the number of available non-overlapping channels when the number of radio interfaces per node is kept constant. When each node has one radio interface, the aggregate throughput saturates at about 2 channels. When the number of radio interfaces on each node is increased to four, the node can use up to 7 channels before its performance starts to saturate.

4.3.3 Throughput of Single Flow

In Figure 4.11, we examine how throughput differs for paths with different hop-counts, and compare EECAS against ROMA [51] and Hyacinth [45]. These simulations were conducted on a 24-node dual-radio grid network. In each run of the

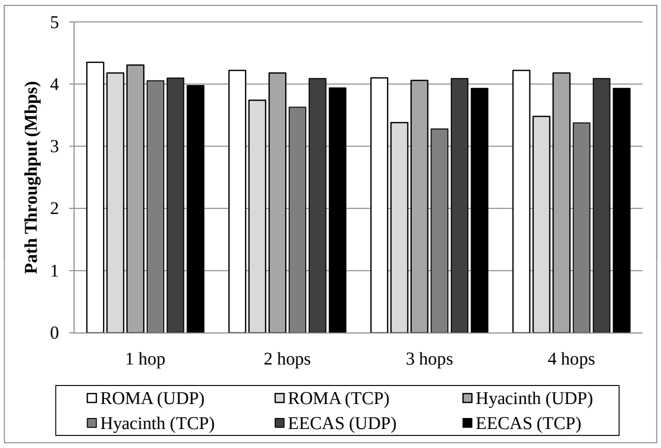


Figure 4.11: Throughput differs for paths with different hop-counts.

simulation, we randomly pick one of the 24 nodes to act as the gateway, and therefore initiate a UDP flow from the gateway to each of 23 non-gateway nodes, one at a time. The RTS/CTS is enabled. There were ten runs in total. In this figure, ROMA and Hyacinth are almost equal. They both build a spanning tree rooted at the gateway and then assign different channels for links with different hop counts from the gateway based on the spanning tree. In the single flow examination, only one flow is active at a time, so both ROMA and Hyacinth do not suffer intra-path and inter-path interference and perform well. In this case, there is about 6% degradation in EECAS, compared with ROMA, due to protocol overheads, such as channel switching delay and guard period.

We repeat the same single-flow simulation to evaluate the throughput of TCP flows. From Figure 4.11, we observe that TCP flows in EECAS achieve only marginally lower throughput when compared to the UDP flows, even for longer paths. For example, in 4-hop paths, the median TCP throughput is 3.93 Mbps, which is 4% lower when compared to the median UDP throughput of 4.09 Mbps.

4.3.4 Throughputs of Multiple Flows

We measure the throughputs of multiple randomly chosen simultaneous flows. These simulations were also conducted on 24-node dual-radio grid network. In each run, we divide 24 nodes into 12 source-destination pairs, and start a UDP flow on each

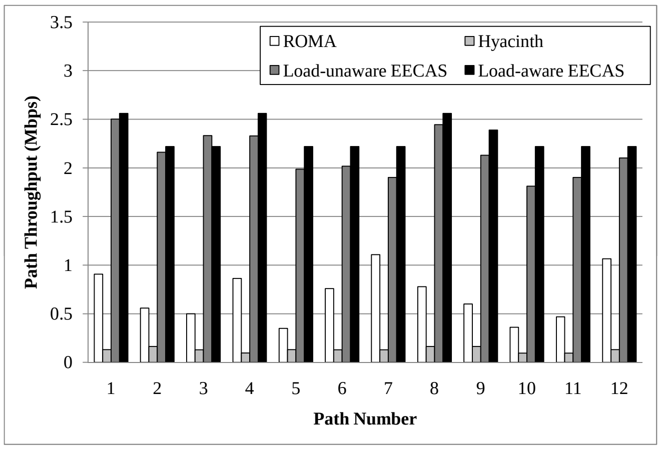


Figure 4.12: Throughputs of multiple intra-mesh paths.

pair. For path decision, two configurations are applied: (a) EECAS's metric (load-aware), $\beta = 0.5$; (b) shortest-path only (load-unaware). There were 12 runs in total. Figure 4.12 shows path throughput of each path with the two configurations, and compares the performances of EECAS, ROMA and Hyacinth in this scenario.

We first observe that almost paths with load-aware EECAS achieve higher performance than load-unaware. Applying EECAS's metric could cause a longer path or a path sharing bandwidth from other paths, which causes lower performance in particular paths, such as path 3 in Figure 4.12. However, the network throughput with load-aware can achieve 2.32 Mbps and have 8% improvement compared to load-unaware. In addition, none of paths is starved either with load-aware or load-unaware.

Moreover, to compare with ROMA and Hyacinth, we randomly choose a node to act as the gateway, and let both ROMA and Hyacinth build a spanning tree rooted the gateway. As can be seen from Figure 4.12, the variations of path throughput with EECAS, ROMA, and Hyacinth are totally different even though they have same source-destination pairs, because channel allocation mechanism dominates routes for these pairs. With Hyacinth, almost routes travel through the gateway, which causes serious interference among these routes, and thus path throughput falls quickly. ROMA allocates channels for all nodes residing on the same routing level based on the spanning tree as well as preserves cross links between paths, so most source-destination pairs can find out shortcut routes, which is helpful to improve path throughput. However, both ROMA and Hyacinth are not feasible in this intra-mesh communication scenario due to serious inter-path interference, even though they perform well in the Internet access scenario.

4.3.5 Setup Latency

Latency of route selection and channel allocation in EECAS is not only affected by number of hop-count but also length of channel slot (Ls) and length of parity slot cycle (Lc), because control messages are exchanged only at the period of parity slot(s) except edges with allocated channel slot(s). In other words, we can arrange multiple parity slots in a channel slot cycle to reduce the latency. Typically, EECAS spends two times of round-trip time on performing route selection and channel allocation for a new request. Figure 4.13 shows the total latency of route selection and channel allocation versus different number of hop-counts. We observe that the latency little rises as hop-count increases. When Ls = 5 ms and Lc = 5, the latency for 1-hop path is about 17.34 ms, which is about five times of ROMA's latency. And the latency rises to 23 ms as hop count increases to 4. If we double the length of channel slot (Ls = 10 ms), the latency rises to 26.55 ms (and 36.67 ms) in 1-hop (and 4-hops) paths. Besides, if we double the cycle length of parity slot (Lc = 10), the latency rises to 35.97 ms (and 43.89 ms) in 1-hop (and 4-hops) paths. Unlike ad-hoc networks, because nodes in mesh networks are stationary and act as the backbone of network, EECAS spending more efforts on route selection and channel allocation for long-term usage is sensible.

4.3.6 Route Stretch

EECAS uses the parameter β to balance the tradeoff between path performance and overhead of channel allocation. To study the effects by β, we set up a 5×5 mesh network that consists of one node as the gateway and 24 dual-radio non-gateway nodes. We know that the gateway is bottleneck if multiple non-gateway nodes simultaneously generate traffics and throw packets to the gateway. Fortunately

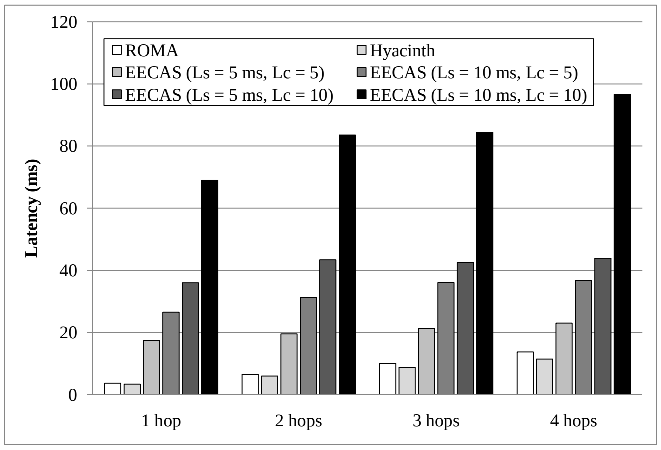


Figure 4.13: Setup Latency.

EECAS allows nodes to be equipped with different number of radio interfaces to each other due to different roles in network, so we set up the gateway as four-radio gateway node to solve the potential hot spot problem. In each run of simulations, 12 source nodes selected from the 24 non-gateway nodes generate data flows to the gateway. We try three different topologies: (a) the gateway is placed at a corner of the network and the source nodes are placed near the gateway; (b) the gateway is also placed at a corner, but the source nodes are placed opposite to the gateway; and (3) the gateway is placed at the center of network and the source nodes are randomly chosen from other non-gateway nodes. We also vary β to study its effect on the average path length as well as the average path throughput from the source nodes to the gateway. We use three different values of β (0.5, 0.7, and 0.9) and run the simulation three times for each of them.

In the topology (a), across the three runs, the average path length is 2.75, 2.5 and 2.5, and the average path throughput, as shown in Figure 4.14, is 0.85, 1.02 and 1.02 Mbps for $\beta = 0.5$, 0.7 and 0.9 respectively. Moreover, in the topology (b), the average path length is 3.67, 3.34 and 3.34, and the average path throughput is 0.94, 1.02 and 0.34 Mbps. We observe that as β is decreased, many of sub-optimal paths are replaced by the high-performance longer paths. For example, some 3 hop paths are replaced by 4 or 5 hop paths as β is decreased. But, too many longer paths could waste network resource and could cause that path throughputs of sub-

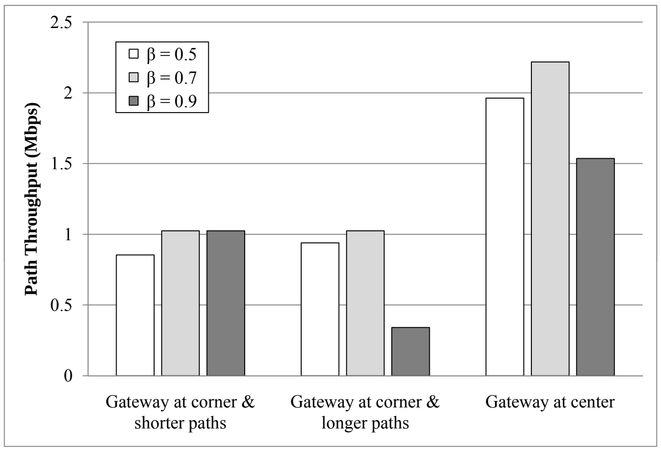


Figure 4.14: Path throughput versus different β values.

sequent flows fall quickly. On the other hand, as β is increased, many of flows, especially in the topology (b), have common paths and share bandwidth of channel slots on these common paths. But, too many flows sharing limited bandwidth of common paths also causes path throughputs of subsequent flows fall quickly. In the last topology (c), the average path length is 2, 1.75 and 1.67, and the average path throughput is 1.96, 2.22 and 1.54 Mbps, for $\beta = 0.5$, 0.7 and 0.9 respectively. Almost paths in topology (c) are shorter than paths in other two topologies and they can achieve higher throughput due to less common paths than other topologies. However, varying β also influences the performance at path length and path throughput. In summary, higher values of β improve performance at the cost of increased channel allocating overhead, while smaller β values tend to arrange longer paths with higher throughput. Assigning $\beta = 0.7$ can achieve better performance in all three simulation topologies.

4.4 Summary

In this chapter, we have presented the proposed EECAS, a simple channel allocation mechanism for end-to-end communications in WMNs. We apply the concept of the directional logical channel to the channel allocation and further consider the channel allocation from an end-to-end point of view so that the channel utilization can be improved and the end-to-end throughput of flows can be maintained. Our EECAS can achieve 335% improvement over the ROMA under a scenario of intra-mesh communications.

Chapter 5

A Cross-layer Signaling and Middleware Platform

5.1 Introduction

With the advance in wireless techniques, mobile devices, such as personal digital assistant (PDA), smart phone, and tablet PC, has become a popular electronic product recently. A mobile device may be equipped with multiple wireless network interfaces, such as WLAN, GPRS, or 3G adaptors, to attach to different networks as it moves. Therefore, an application running on the mobile device may visit different networks and encounters the changes in bandwidth, delays, or IP addresses. In order to tackle network fluctuations, network-aware applications that can adapt themselves to the changes in network environment have now become a major research topic in recent years.

A network-aware application may need to issue system calls periodically to retrieve lower-layer status, such as IP addresses, entries of routing table, and connectivity of network adaptors. However, the network status normally do not change frequently. Periodical system calls with short intervals may waste system resources for getting the same information. On the other hand, with long intervals between system calls, the applications can not react to the changes of network environment promptly.

Furthermore, a mobile device with multiple network interfaces needs a mobility manager to monitor status of network adaptor and perform handover decision accordingly. In order to acquire the status and conduct a handover, a mobility manager needs to use system calls to communicate with underlying network protocol stacks, such as link layer, network layer, or transport layer. However, the use of system calls is not only error-prone but also not portable.

In order to solve above problems, we proposed a software platform for the development of network-aware applications. The platform adopts a middleware [62] approach and provides application programming interfaces (APIs) for the application to use Cross-layer Signaling Mechanism. Based on the mechanism, an application can interact with the lower-layer network protocols to acquire network status and manage network interfaces. Besides, the platform also provides an Event Notification Mechanism for an application to register interested events of network changes so that the middleware can notify the application immediately when an event of interest occurs.

We also use network-aware applications, namely a Mobility Manager and a Modified Kphone [63], as two examples to demonstrate the effectiveness of out proposed platform. The Mobility Manager can monitor status of multiple network interfaces simultaneously and perform handover decision accordingly. The Modified Kphone is a voice over IP (VoIP) application with a new handover decision module that can interact with our proposed platform.

5.2 System Architecture

In this section, we will detail system architecture of the proposed middleware platform including Cross-layer Signaling and Event Notification mechanisms.

5.2.1 Middleware Platform and Cross-layer Signaling

System architecture of proposed middleware platform can be divided into two major parts that include user space and kernel space. The middleware was implemented in user space and between application layer and underlying network layers, and it provided APIs for the network-aware applications, such as mobility manager and modified Kphone, to interact with underlying stacks of network protocol as shown in Figure 5.1. For interaction between different network protocol stacks, the proposed middleware platform must provide Cross-layer Signaling Mechanism including status of underlying stacks acquisition, control messages dispatch and events notification for the applications. Therefore, the APIs are divided into three kinds including control, query and event interfaces. The control interface is used to notify underlying protocol stacks of changing status, such as adding or deleting entries of routing table, changing default gateway or attachment access point (AP). The query interface is used to acquire specific status of underlying protocol stacks, such as wireless signal strength, IP address configurations or data transmission rate. The event interface allows the network-aware application to register interesting events and get event notification immediately when an interesting event occurs. Our implementation of event interface will be detailed in next subsection.

On the other hand, because the middleware must help the network-aware applications to interact with underlying protocol stacks, control and query messages received from applications need to translate into corresponding system calls. Therefore, we implemented the system call functions in proposed middleware, called Middleware Core, to interact with specific protocol layer located in kernel space. This part adopts cross-layer signaling design to order specific layer to do something and acquire status of specific layer. For example, an application can acquire wireless signal strength of WLAN adaptor or configure IP address directly. Furthermore, the Middleware Core will maintain some data structures, such as registered event tables and event queues, for applications to query or use.

To adopt proposed middleware platform, complex system calls can be reduced to simple APIs and thus an application can easy to use. Furthermore, to acquire and configure low-layer status via the Middleware APIs, a network-aware application can be developed quickly and error-less, and thus application developer can pay attention more to design handover policies.

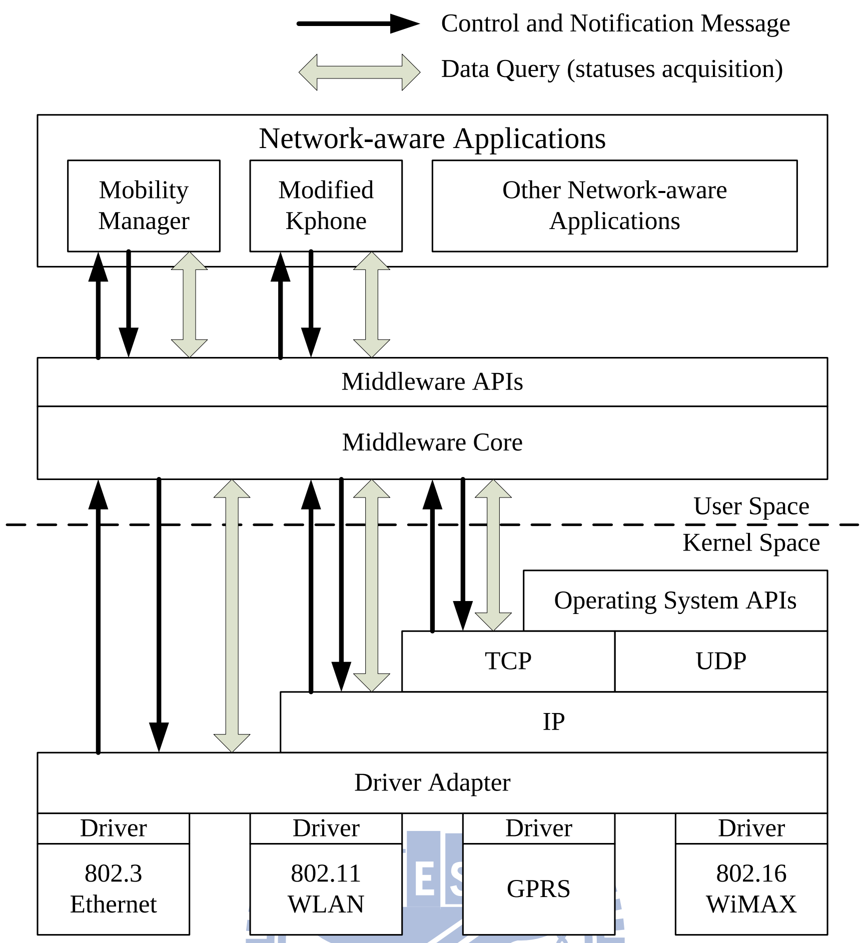


Figure 5.1: The middleware platform.

5.2.2 Network Event Notification Mechanism

The event interface includes two kinds of method to notify the network-aware applications: synchronous and asynchronous process. In the synchronous process, a network-aware application must use Middleware API to initiate one or more than one event queues, and then register those queues and interesting events, called registered event, in Middleware Core as shown in Figure 5.2. To illustrate this process, in the example program of synchronous process, an application must use the *win_event_init* function call and configure the parameter as “`NULL`” to initiate an event queue. Therefore, the application can use the *win_event_register* function call to register interesting events in Middleware Core, and then the Middleware Core will maintain a “Registered Event Table” including all interesting events of the ap-

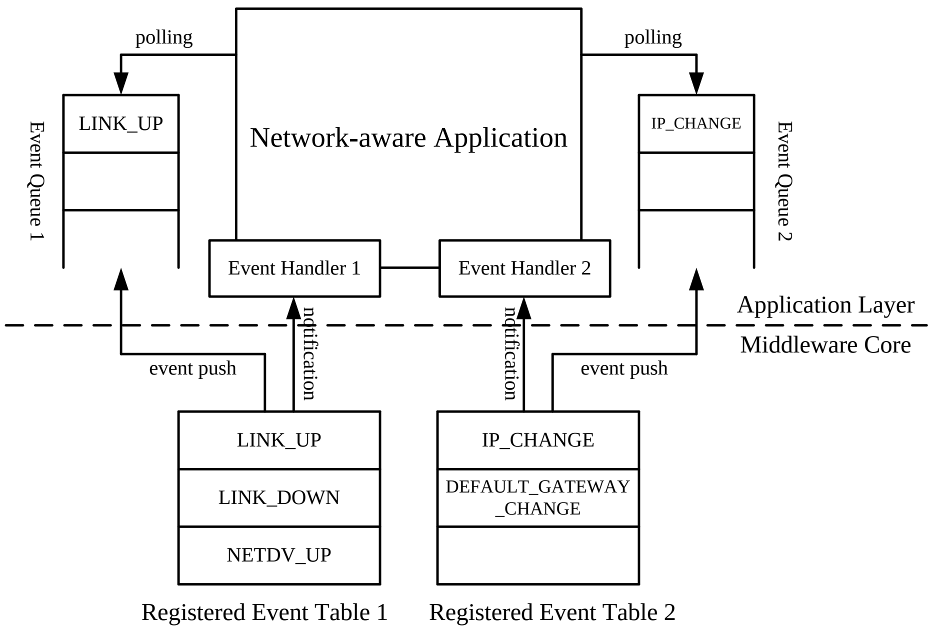


Figure 5.2: Two kinds of Event Notification Mechanism process.

plication, such as LINK_UP, LINK_DOWN and NETDV_UP in Registered Event Table 1. When an event occurs, the Middleware Core will check all of Registered Event Tables and then push the event to corresponding event queues. Finally, the application can use *win_check_event* function call to poll event queues periodically.

In the asynchronous process, an application must create an event handler to prepare for event notifications. Using this process, the application can get event notification immediately when an event occurs. For example, an event handler that is named for "`my_event_handler`" must be created firstly, and then the application uses *win_init_event* function call and configures the parameter as the handler name to initiate asynchronous process as shown in example program of asynchronous process. Therefore, the application can use *win_event_register* function call to register interesting events similarly. The Middleware Core will dispatch event notification to corresponding event handlers when an event occurs so the application knows event occurrence and the corresponding handler procedure will run immediately.

In the kernel level, we adopted the Driver-level Network Event Notification Mechanism that is developed by our research partner and mentioned in Section 2.8 to support our Event Notification Mechanism. Furthermore, we had added novel events, such as ROUTING_TABLE_CHANGE, to extend the notification mechanism so the middleware platform can be used to help network-aware application that it gets interesting event notifications correctly and immediately.

Listing 5.1: An Example Program of Synchronous Process

```
int main() {
        event_descriptor = win_event_init(NULL);
        win_event_register(event_descriptor, NETDEV_UP);
        win_event_register(event_descriptor, IP_CHANGE);
        ...
        if (win_check_event(event_descriptor, wevent) ==
            R_HAVE_EVENT) {
                ...
        }
        ...
}
```

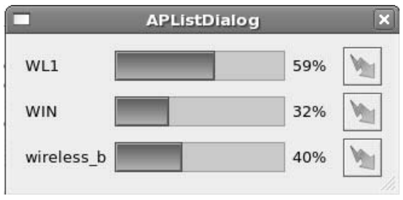
Listing 5.2: An Example Program of Asynchronous Process

```
void my_event_handler(struct win_event* wevent) {
    printf("Event Happen!!\n");
}

int main() {
        ...
        event_descriptor = win_event_init(my_event_handler);
        win_event_register(event_descriptor, NETDEV_UP);
        win_event_register(event_descriptor, IP_CHANGE);
        ...
}
```

5.3 Network-aware Application

Based on our middleware platform, a program developer can develop a network-aware application quickly and easily. In this section, we will demonstrate two network-aware applications including Mobility Manager and Modified Kphone.

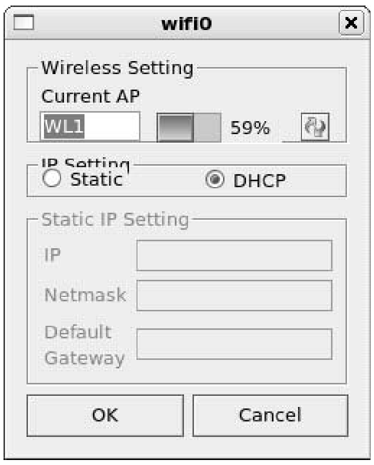

APListDialog

WL1 59%

WIN 32%

wireless_b 40%

Figure 5.3: A dialog of AP list.


wifi0

Wireless Setting

Current AP: WL1 59%

IP Setting: Static [✓ DHCP]

Static IP Setting

IP

Netmask

Default Gateway

Figure 5.4: A dialog for user to configure current attachment AP.

5.3.1 Mobility Manager

A mobility manager for mobile device should include three major functions: display of network status, network configuration and handover policy. Therefore, we implemented the Mobility Manager application that used Middleware APIs to acquire and configure status of underlying protocol stacks. For example, in the Figure 5.3, wireless signal strength of APs can be displayed on a single graphic interface. Furthermore, a user of mobile device can use a graphic configuration interface to set attachment AP and choose acquisition method of IP address as shown in Figure 5.4.

For mobile device handover, we need to know the changes of network environment and then mobile device chooses another network to attach according user preference if current attachment network is unreachable or low quality. In our implementation, the Mobility Manager allows user to choose manual or automatic handover and to assign preferred interface as shown in Figure 5.5. User can configure a profile of handover policy that records when handover can be performed and what something is

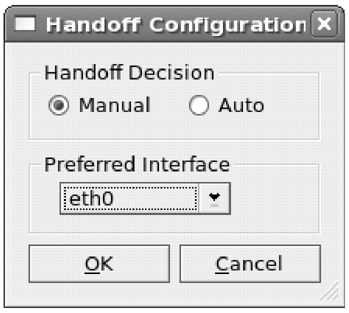


Figure 5.5: The handoff configuration interface.

needed to do in handover, if the Mobility Manager be configured automatic handover mode.

In summary, a mobile device can acquire network status, such as current IP address and signal strength of APs, and configure those network status easily if the Mobility Manager application runs on this mobile device. Furthermore, the mobile device can make handover decision automatically according network environment changes, such as link quality or reachable network, and user preference.

5.3.2 Modified Kphone

The Modified Kphone is another example of network-aware application that is sourced from a VoIP application called Kphone and is modified to perform handover via the Middleware APIs. We need to program some new procedures to make handover decision because the original Kphone application cannot support handover. In the Kphone communication, a mobile node (MN) and a correspondent node (CN) are transmitting voice data via Real-time Transport Protocol (RTP) packets to another. We need to ensure that Synchronization Source Identifier (SSRC) in RTP header is identical when the MN changes its IP address because same SSRC in RTP header implies same connection between MN and CN.

In our implementation of Modified Kphone, we divided handover procedure into four steps as shown in Figure 5.6. First, a MN receives a network event notification, such as wireless signal strength below a threshold, and then it triggers network layer handover that includes changing attachment network, acquiring a new IP address

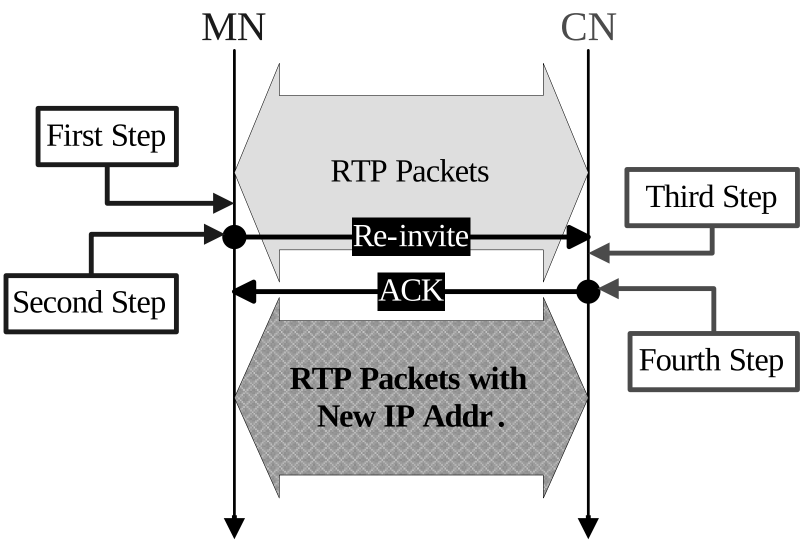


Figure 5.6: The handover procedure of the Modified Kphone.

and configuring default gateway. Second, the MN uses the new IP address and original SSRC to send Re-invite message to CN. Third, the CN will change destination IP address of sending packets according source IP address of received packet and the SSRC when it receives the Re-invite message. Finally, the CN will send ACK message to the MN and thus an application layer handover is completed.

In summary, our Modified Kphone application can perform handover quickly and correctly because the middleware platform can provides layer-2 trigger and some detail status for an application. Furthermore, a program developer can pay attention more to design of handover policies as a result of the middleware platform provides several mechanisms for the developer to implement applications.

5.4 Summary

In this chapter, we designed and implemented a middleware platform that includes Cross-layer Signaling and Network Event Notification mechanisms. The middleware was implemented in user space and then it provided Middleware APIs for applications to configure and acquire status of underlying protocol stack. Based on Cross-layer Signaling mechanism, control and query messages can order each protocol stack to do something directly. Beside, the Network Event Notification can help application that it is aware of changes of network environment immediately.

We also demonstrated two examples of network-aware application that include Mobility Manger and Modified Kphone. Those examples illustrated two kinds of Event Notification method and how to use in our middleware platform. According our illustration, if a network-aware application is developed via our middleware platform, this application can be developed quickly and easily.

In future works, we will continue to extend novel network events when a new type of network adaptor is created. Furthermore, we consider developing a handover analysis tool via the middleware platform and then the tool can help programmer to determine cause of handover delay.

Chapter 6

Conclusions

In this thesis, we first presented an approach to secure fast handoff in a WMN. Our approach reduces repeated IEEE 802.1X authentications and encryption/decryption processing of data frames for mobile stations by an appreciable amount. Our development exhibits several features. First, the security level in line with IEEE 802.11i is maintained. Second, following our design tenet, the entities holding pairwise mater keys are shifted to MPP in a way that fast handoff can be accomplished by bypassing prohibitive IEEE 802.1X authentication during handoff. Moreover, performance results show that our approach reduces handoff delay by up to 62.7%, or achieves comparable performance resulting from the counterpart IEEE 802.11i scheme with high likelihood of 70% to 85% successful preauthentication. Our performance analysis also suggests an optimal number of MAPs managed by one mesh portal in a network.

We then proposed an end-to-end channel allocation scheme and its path metric to solve the co-channel interference problem in a WMN. Our EECAS is a general solution due to following features. First, both intra-path and inter-path interferences can be avoided. Second, not only inter-mesh paths (established between MAPs and MPP) but also intra-mesh paths (established between any two nodes within same domain) have been considered. Third, EECAS is workable with any number of radio interfaces and asymmetric radio architecture. Moreover, simulation results show that EECAS achieves 335% improvement over ROMA in throughputs of multiple intra-mesh paths.

We have designed and implemented a middleware platform, which provides Cross-layer Signaling and Event Notification mechanisms, on mobile station for upper-layer network-aware applications. Based on the platform, network-aware applications can configure and acquire statuses of underlying protocol stacks via Cross-layer Signaling mechanism. Beside, the applications can also detect changes of network environment immediately via Event Notification mechanism. we also demonstrated two examples of network-aware application, which illustrated the usages of Event Notification mechanism.

At last but not least, by the integration of above three studies, our solution not only reduces handoff delay but also improves path throughput in a secure WMN. Besides, a WMN can provide stable backbone at any time.

Bibliography

[1] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz, "Extensible Au-thentication Protocol (EAP)," *RFC 3748*, IETF Network Working Group, June 2004.

[2] B. Aboba, D. Simon, and P. Eronen, "Extensible Authentication Protocol (EAP) Key Management Framework," Internet Draft <draft-ietf-eap-keying-22.txt>, IETF Network Working Group, November 2007.

[3] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless Mesh Networks: A Survey," *Computer Networks Journal*, 47(4): 445–487, March 2005.

[4] I. F. Akyildiz, Y.-B. Lin, W.-R. Lai, and R.-J. Chen, "A New Random Walk Model for PCS Networks," *IEEE Journal on Selected Areas in Commun.*, 18(7): 1254–1260, July 2000.

[5] A. Alimian and B. Aboba, "Analysis of Roaming Techniques," IEEE 802.11 Contribution 802.11-04/0377r1, March 2004.

[6] K.-H. Chi, C.-C. Tseng, and Y.-H. Tsai, "Fast Handoff among IEEE 802.11r Mobility Domains," *Journal of Information Science and Engineering*, 26(4): 1345–1362, July 2010.

[7] K.-H. Chi, J.-H. Jiang, and L.-H. Yen, "Cost-Effective Caching for Mobility Support in IEEE 802.1X Frameworks," *IEEE Trans. Mobile Computing*, 5(11): 1547–1560, November 2006.

[8] W. S. Conner, J. Kruys, and J. C. Zuniga, "IEEE 802.11s Tutorial: Overview of the Amendment for Wireless Local Area Mesh Networking," IEEE 802 Plenary, Dallas, USA, November 2006.

[9] H. Duong, A. Dadej, and S. Gordon, "Proactive Context Transfer and Forced Handover in IEEE 802.11 Wireless LAN-Based Access Networks," *ACM Mobile Computing and Comm. Rev.*, 9(3): 32–44, July 2005.

[10] IEEE 802.1 Working Group, "Port-Based Network Access Control," IEEE Standard 802.1X-2004, December 2004.

[11] IEEE 802.11 Working Group, "Amendment 6: Medium Access Control (MAC) Security Enhancements," IEEE Standard 802.11i-2004, July 2004.

[12] IEEE 802.11 Working Group, "Amendment 2: Fast BSS Transition," IEEE Standard P802.11r, July 2008.

[13] IEEE 802.11 Working Group, "Amendment: ESS Mesh Networking," IEEE Standard Draft P802.11s/D2.0, April 2008.

[14] P. Kiratiwintakorn and P. Krishnamurthy, "An Energy Efficient Security Protocol for IEEE 802.11 WLANs," *Pervasive and Mobile Computing*, 2(2): 204–231, April 2006.

[15] M. Lasserre and V. Kompella, "Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling," *RFC 4762*, IETF Network Working Group, January 2007.

[16] A. Mishra, M. H. Shin, and W. Arbaugh, "An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process," *ACM SIGCOMM Computer Commun. Rev.*, Vol. 33, pp. 93–102, April 2003.

[17] A. Mishra, M. H. Shin, and W. Arbaugh, "Context Caching Using Neighbor Graphs for Fast Handoffs in a Wireless Network," In *Proc. 23rd IEEE Conf. Computer Commun.* (INFOCOM), 2004.

[18] A. Mishra, M. H. Shin, N. L. Petroni, Jr., T. C. Clancy, and W. A. Arbaugh, "Proactive Key Distribution Using Neighbor Graphs," *IEEE Wireless Commun.*, 11(1): 26–36, February 2004.

[19] S. Pack and Y. Choi, "Fast Inter-AP Handoff Using Predictive Authentication Scheme in a Public Wireless LAN," In *Proc. IEEE Networks Conf.*, August 2002.

[20] S. Pack and Y. Choi, "Pre-Authenticated Fast Handoff in a Public Wireless LAN Based on IEEE 802.1x Model," *Proc. IFIP Personal Wireless Commun. Conf. '02*, pp. 175–182, October 2002.

[21] S. Pack, J. Choi, T. Kwon, and Y. Choi, "Fast Handoff Support in IEEE 802.11 Wireless Networks," *IEEE Commun. Surveys & Tutorials*, 9(1): 2–12, March 2007.

[22] S. Pack, H. Jung, T. Kwon, and Y. Choi, "SNC: A Selective Neighbor Caching Scheme for Fast Handoff in IEEE 802.11 Wireless Networks," *ACM Mobile Computing and Commun. Rev.*, 9(4): 39–49, October 2005.

[23] M. G. Rahman and H. Imai, "Security in Wireless Communication," *Wireless Personal Commun.*, Vol. 22, pp. 213–228, August 2002.

[24] M. H. Shin, A. Mishra, and W. Arbaugh, "Improving the Latency of 802.11 Handoffs Using Neighbor Graphs," In *Proc. Second Int'l Conf. Mobile Systems, Applications, and Services*, pp. 70–83, 2004.

[25] G. Xue, "An Improved Random Walk Model for PCS Networks," *IEEE Trans. Commun.*, 50(8): 1224–1226, August 2002.

[26] R. Bruno, M. Conti, and E. Gregori, "Mesh Networks: Commodity Multihop Ad Hoc Networks," *IEEE Communications Magazine*, vol. 43, pp. 123–131, March 2005.

[27] K. Jain, J. Padhye, V. N. Padmanabhan, and L. L. Qiu, "Impact of Interference on Multi-hop Wireless Network Performance," *Wireless Networks*, vol. 11, pp. 471–487, July 2005.

[28] H. Skalli, S. Ghosh, S. K. Das, L. Lenzini, and M. Conti, "Channel Assignment Strategies for Multiradio Wireless Mesh Networks: Issues and Solutions," *IEEE Communications Magazine*, vol. 45, pp. 86–93, November 2007.

[29] J. Crichigno, W. Min-You, and S. Wei, "Protocols and Architectures for Channel Assignment in Wireless Mesh Networks," *Ad Hoc Networks*, pp. 1051–1077, 2008.

[30] N. Jain, S. R. Das, and A. Nasipuri, "A Multichannel CSMA MAC Protocol with Receiver-based Channel Selection for Multihop Wireless Networks," In *the Tenth International Conference on Computer Communications and Networks (ICCN 2002)*, pp. 432–439, 2002.

[31] A. Tzamaloukas, and J. J. Garcia-Luna-Aceves, "Channel-hopping Multiple Access," In *Proc. IEEE International Conference on Communications (ICC 2000)*, pp. 415–419, 2000.

[32] A. Tzamaloukas, and J. J. Garcia-Luna-Aceves, "Channel-hopping Multiple Access with Packet Trains for Ad Hoc Networks," In *Proc. IEEE Device Multimedia Communications (MoMuC'00)*, Tokyo, Japan, 2000.

[33] P. Bahl, R. Chandra, and J. Dunagan, "SSCH: Slotted Seeded Channel Hopping for Capacity Improvement in IEEE 802.11 Ad-hoc Wireless Networks," In *Proceedings of the 10th annual international conference on Mobile computing and networking (MobiCom 2004)*, Philadelphia, PA, USA, 2004.

[34] R. Maheshwari, H. Gupta, and S. R. Das, "Multichannel MAC Protocols for Wireless Networks," In *the 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks (SECON '06)*, pp. 393–401. 2006.

[35] J. So, and N. H. Vaidya, "Multi-channel Mac for Ad Hoc Networks: Handling Multi-channel Hidden Terminals using A Single Transceiver," In *Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, Roppongi Hills, Tokyo, Japan, 2004.

[36] J. Chen, S.-T. Sheu, and C.-A. Yang, "A New Multichannel Access Protocol for IEEE 802.11 Ad Hoc Wireless LANs," In *the 14th IEEE Proceedings on Personal, Indoor and Mobile Radio Communications (PIMRC 2003)*, pp. 2291–2296, 2003.

[37] J. Zhang, G. Zhou, C. Huang, S. H. Son, and J. A. Stankovic, "TMMAC: An Energy Efficient Multi-Channel MAC Protocol for Ad Hoc Networks," In *IEEE International Conference on Communications (ICC '07)*, pp. 3554–3561, 2007.

[38] R. Draves, J. Padhye, and B. Zill, "Routing in Multi-radio, Multi-hop Wireless Mesh Networks," In *Proceedings of the 10th annual international conference on Mobile computing and networking*, Philadelphia, PA, USA, 2004.

[39] S. M. Das, Y. Wu, R. Chandra, and Y. C. Hu, "Context Based Routing: Technique, Applications and Experience," In *USENIX NSDI*, 2008.

[40] S. Biswas, "Meraki Networks' Next Generation Multi-radio Mesh Platform," private communication, 2008.

[41] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and Evaluation of An Unplanned 802.11b Mesh Network," In *ACM Mobicom*, 2005.

[42] B.-J. Ko, V. Misra, J. Padhye, and D. Rubenstein, "Distributed Channel Assignment in Multi-Radio 802.11 Mesh Networks," In *Wireless Communications and Networking Conference (IEEE WCNC 2007)*, pp. 3978–3983, 2007.

[43] K. N. Ramachandran, E. M. Belding, K. C. Almeroth, and M. M. Buddhikot, "Interference-Aware Channel Assignment in Multi-Radio Wireless Mesh Networks," In *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM 2006)*, pp. 1–12, 2006.

[44] A. Raniwala, K. Gopalan, and T.-c. Chiueh, "Centralized Channel Assignment and Routing Algorithms for Multi-channel Wireless Mesh Networks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 8, pp. 50–65, 2004.

[45] A. Raniwala and T.-c. Chiueh, "Architecture and Algorithms for An IEEE 802.11-based Multi-channel Wireless Mesh Network," In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005)*, vol. 3, pp. 2223–2234, 2005.

[46] A. H. Mohsenian Rad, and V. W. S. Wong, "Joint Optimal Channel Assignment and Congestion Control for Multi-channel Wireless Mesh Networks," In *IEEE International Conference on Communications (ICC '06)*, pp. 1984–1989, 2006.

[47] A. K. Das, H. M. K. Alazemi, R. Vijayakumar, and S. Roy, "Optimization Models for Fixed Channel Assignment in Wireless Mesh Networks with Multiple Radios," In *the Second Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON 2005)*, pp. 463–474, 2005.

[48] M. X. Gong and S. F. Midkiff, "Distributed Channel Assignment Protocols: A Cross-Layer Approach," In *Wireless Communications and Networking Conference (IEEE WCNC 2005)*, 2005.

[49] F. Herzel, G. Fischer, and H. Gustat, "An Integrated CMOS RF Synthesizer for 802.11a Wireless LAN," *IEEE journal of Solid-state Circuits*, 18(10), October 2003.

[50] V. Bahl, A. Adya, J. Padhye, and A. Wolman, "Reconsidering the Wireless LAN Platform with Multiple Radios," In *Workshop on Future Directions in Network Architecture*, 2003.

[51] A. Dhananjay, H. Zhang, J. Li, and L. Subramanian, "Practical, Distributed Channel Assignment and Routing in Dual-radio Mesh Networks" *ACM SIG-*

COMM Computer Communication Review, vol. 39 Issue 4, pp. 99–110, October 2009.

[52] Q. Yang, J. Shi, M. Xiao, and H. Chen, "A Decentralized Slot Synchronization Algorithm for TDMA-Based Ad Hoc Networks," In *International Conference on Wireless Communications, Networking and Mobile Computing (WiCom 2007)*, 2007.

[53] F. Wang, P. Zeng, and H. Yu, "Slot Time Synchronization for TDMA-Based Ad Hoc Networks," In *International Symposium on Computer Science and Computational Technology (ISCSCT '08)*, 2008.

[54] M. Alicherry, R. Bhatia, and L. Li, "Joint Channel Assignment and Routing for Throughput Optimization in Multi-radio Wireless Mesh Networks," In *Proc. ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2005.

[55] M. Kodialam, and T. Nandagopal, "Characterizing the Capacity Region in Multi-Radio Multi-Channel Wireless Mesh Networks," In *MobiCom '05*, August 28-September 2, 2005, Cologne, Germany.

[56] A. Mohsenian-Rad, and V. Wong, "Joint Logical Topology Design, Interface Assignment, Channel Allocation and Routing for Multi-channel Wireless Mesh Networks," In *Infocom*, 2008.

[57] S. Merlin, N. H. Vaidya, and M. Zorzi, "Resource Allocation in Multi-radio Multi-channel Multi-hop Wireless Networks," In *UIUC Technical Report*, 2007.

[58] M. Marina, and S. Das, "A topology Control Approach for Utilizing Multiple Channels in Multi-radio Wireless Mesh Networks," In *Proceedings of Broadnets*, 2005.

[59] P. Dutta, S. Jaiswal, and R. Rastogi, "Routing and Channel Allocation in Rural Wireless Mesh Networks," In *Infocom*, 2007.

[60] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A High-throughput Path Metric for Multi-hop Wireless Routing," In *Proceedings of the 9th ACM International Conference on Mobile Computing and Networking (MobiCom '03)*, San Diego, California, September 2003.

[61] R. Draves, J. Padhye, and B. Zill, "Comparison of Routing Metrics for Static Multi-hop Wireless Networks," In *Proc. ACM SIGCOMM Conference (SIGCOMM 2004)*, September 2004.

[62] D. Bakken, "Middleware," `http://www.eecs.wsu.edu/~bakken/middleware.pdf`

[63] "Kphone Project," `http://sourceforge.net/projects/kphone`

[64] C. Perkins, "IP Mobility Support for IPv4," *IETF RFC3344*, Nokia Research Center, August 2002.

[65] J. Sun, J. Riekki, M. Jurmu, and J. Sauvola, "Adaptive Connectivity Management Middleware for Heterogeneous Wireless Networks," In *IEEE Wireless Communications*, December 2005.

[66] J. Sun, J. Tenhunen, and J. Sauvola, "CME: A Middleware Architecture for Network-aware Adaptive Applications," In *Proceedings 14th IEEE PIMRC2003*, Beijing, China (2003).

[67] K. A. Hawick, H. A. James, "Wireless Issues for a Mobility Management Middleware," In *CCN2002*, August 2002.

[68] Y. Tian, S. Frank, V. Tsaoussidis, and H. Badr, "Middleware Design Issues for Application Management in Heterogeneous Networks," In *Networks 2000 (ICON 2000)*.

[69] B. Li, and K. Nahrstedt, "A Control-Based Middleware Framework for Quality of Service Adaptations," In *IEEE Journal of Selected Areas in Communication*, 17(9): 1632–1650, 1999.

[70] B. Kreller, A. S. B. Park, J. Meggers, G. Forsgren, E. Kovacs, M. Rosinus, and A. G. Siemens, "UMTS: A Middleware Architecture and Mobile API Approach," In *IEEE Personal Communications*, April 1998.

[71] X. George, and C. P. George, "Internet Protocol Performance over Networks with Wireless Links," In *IEEE Network*, 13(4): 55–63, 1999.

[72] D. D. Clark, "The Structuring of Systems using Upcalls," In *ACM Symposium on Operating Systems*, pp. 171–180, 1985.

[73] G. H. Cooper, "The Argument for Soft Layer of Protocols," In *Tech. Rep. Tr-300*, Massachussets Institute of Technology, Cambridge, May 1983.

[74] G. Carneiro, J. Ruela, and M. Ricardo, "Cross-Layer Design in 4G Wireless Terminals, " In *IEEE Wireless Communications*, 11(2): 7–13, April 2004.

[75] T. J. Chou, "Design and Implementation of Driver-Level Network Event Notification Mechanism in Linux," In *Thesis in Wireless Internet Lab.*, National Chiao Tung University, Hsinchu, Taiwan 2005.

Publication List

論文有關著作清單

(a) Journal papers:

1. Kuang-Hui Chi, Yung-Chien Shih, Ho-Han Liu, Jui-Tang Wang, Shiao-Li Tsao, and Chien-Chao Tseng, “Fast Handoff in Secure IEEE 802.11s Mesh Networks,” (to appear in) Journal of IEEE Transactions on Vehicular Technology (TVT).

2. Yung-Chien Shih, Shiao-Li Tsao, and Chien-Chao Tseng, “End-to-End Channel Allocation Scheme for a Wireless Mesh Network,” (in revision) Journal of IEEE Transactions on Mobile Computing (TMC).

(b) Conference papers:

1. Yung-Chien Shih, Kai-Cheng Hsu, and Chien-Chao Tseng, “A Cross-layer Signaling and Middleware Platform for Multi-interface Mobile Devices,” in Proceedings of the 2007 conference on emerging direction in embedded and ubiquitous computing.

(c) Patents:

1. 史永健, 楊人順, 曾建超, “分散式頻道配置方法及使用該方法的無線網狀網路系統,” with CCL/ITRI, 中華民國發明專利, 證書號第I327016號.

2. Yung-Chien Shih, Jen-Shun Yang, and Chien-Chao Tseng, “Distributed Channel Allocation Method and Wireless Mesh Network therewith,” with CCL/ITRI, USA Patent Pending, China Patent Pending.

其它著作清單

(a) Journal papers:

1. Yung-Chien Shih, Yuan-Ying Hsu, Chien-Hung Chen, Edwin Sha, and Chien-Chao Tseng, "Adaptive Attenuation Factor Model for Localization in Wireless Sensor Networks," International Journal of Pervasive Computing and Communications, Vol. 4, No. 3, pp. 257–267 (2008).

(b) Conference papers:

1. 蔡奇峰、史永健、曾建超、何文楨，行動式多媒體推播與跨終端機換手系統設計實作，全國計算機會議(NCS 2003)。

(c) Others:

1. 徐元瑛、史永健、曾建超(民91)，無線網路學習環境的面面觀，資訊與教育雜誌，第90期，頁3–13。