Building Mobile L2TP/IPsec Tunnels

A thesis submitted in fulfilment of the requirements for the degree of Master of Engineering

Chen Xu
B.Eng

Electrical and Computer Engineering
Science, Engineering and Health
RMIT University
November 2009

Statement of Original Authorship

I certify that except where due acknowledgement has been made, the work is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program; and any editorial work, paid or unpaid, carried out by a third party is acknowledged; and, ethics procedures and guidelines have been followed.

Signature: _____

Chen Xu

Date: 09/11/2009

Acknowledgments

I would like to express my sincere gratitude to my supervisor Dr Peterjohn Radcliffe for his patient guidance, encouragement, suggestion, support in this project and correcting my technical writing.

I am thankful to Dr Joseph Yick Hon So for his patient guidance and generous assistance during the time. Thanks also to all the staff in School of Electrical and Computer Engineering at RMIT for their professional jobs.

I am also thankful to Mr Charlie Chen for helping me to solve problems and difficulties in the project.

Last, but not least, this thesis is dedicated to my parents, Mr Zhenying Xu and Mrs Li Chen, for their support and encouragement.

Keywords

Virtual Private Network (VPN), Internet Protocol Security (IPsec), Layer 2 Tunneling Protocol (L2TP), Point-to-Point Protocol (PPP), Mobile IP, VMware

Abstract

Wireless networks introduce a whole range of challenges to the traditional TCP/IP network, especially Virtual Private Network (VPN). Changing IP address is a difficult issue for VPNs in wireless networks because IP addresses are used as one of the identifiers of a VPN connection and the change of IP addresses will break the original connection. The current solution to this problem is to run VPN tunnels over Mobile IP (MIP). However, Mobile IP itself has significant problems in performance and security and that solution is inefficient due to double tunneling.

This thesis proposes and implements a new and novel solution on simulators and real devices to solve the mobility problem in a VPN. The new solution adds mobility support to existing L2TP/IPsec (Layer 2 Tunneling Protocol/IP Security) tunnels. The new solution tunnels Layer 2 packets between VPN clients and a VPN server without using Mobile IP, without incurring tunnel-re-establishment at handoff, without losing packets during handoff, achieves better security than current mobility solutions for VPN, and supports fast handoff in IPv4 networks.

Experimental results on a VMware simulation showed the handoff time for the VPN tunnel to be 0.08 seconds, much better than the current method which requires a new tunnel establishment at a cost of 1.56 seconds.

Experimental results with a real network of computers showed the handoff time for the VPN tunnel to be 4.8 seconds. This delay was mainly caused by getting an IP address from DHCP servers via wireless access points (4.6 seconds). The time for VPN negotiation was only 0.2 seconds. The experimental result proves that the proposed mobility solution greatly reduces the VPN negotiation time but getting an IP address from DHCP servers is a large delay which obstructs the real world application. This problem can be solved by introducing fast DHCP or supplying an IP address from a new wireless access point with a strong signal while the current

Internet connection is weak. Currently, there is little work on fast DHCP and this may open a range of new research opportunities.

The novel L2TP/IPsec VPN concepts have been the basis of one conference and one journal papers as listed below:

C. Xu and P. Radcliffe, "Building Secure Tunnel from PPP Wireless Network", *Wireless Personal Communications*, DOI 10.1007/s11277-009-9894-x, 2009.

C. Xu and P. Radcliffe, "A novel mobility solution based on L2TP/IPsec tunnel", *2009 IEEE Sarnoff Symposium*, 2009.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

3G	Third Generation (International Mobile Telecommunications-2000)
AES	Advanced Encryption Standard
AH	Authentication Header
AP	Access Point
API	Application programming interface
ATM	Asynchronous Transfer Mode
BS	Base Station
CA	Certificate Authority
CBC	Cipher Block Chaining
CE	Customer Edge
CHAP	PPP Challenge Handshake Authentication Protocol
CN	Correspondent Node
CoA	Care-of-Address
DH	Diffie-Hellman
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DOI	Domain of Interpretation
DoS	Denial of Service
ESP	Encapsulating Security Payload
FA	Foreign Agent
FR	Frame Relay
FTP	File Transfer Protocol
GRE	Generic Routing Encapsulation
GW	Gateway
HA	Home Agent
HMAC	keyed-Hash Message Authentication Code
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IKE	Internet Key Exchange
IP	Internet Protocol
IPCP	Internet Protocol Control Protocol
IPR	Intellectual Property Rights
IPsec	Internet Protocol Security
IPX	Internetwork Packet Exchange
ISAKMP	Internet Security Association and Key Management Protocol
ISP	Internet Service Provider

ITU	International Telecommunication Union
KINK	Kerberized Internet Negotiation of Keys
L2F	Layer 2 Forwarding
L2TP	Layer 2 Tunneling Protocol
LAC	L2TP Access Concentrator
LAN	Local Area Network
LCP	Link Control Protocol
LNS	L2TP Network Server
MAC	Medium Access Control
Mbps	Megabit per second
MMC	Microsoft Management Console
MPD	Multi-link PPP daemon for FreeBSD
MPLS	Multiprotocol Label Switching
MPPE	Microsoft Point-to-Point Encryption
MN	Mobile Node
MOBIKE	IKEv2 Mobility and Multihoming Protocol
MS-CHAP	Microsoft Challenge-Handshake Authentication Protocol
NAT	Network Address Translation
NCP	Network Control Protocol
NGN	Next generation wireless communication network
OSI	Open System Interconnection
OSPF	Open Shortest Path First
PAN	Personal Area Network
PAP	Previous anchor point
PDA	Personal Digital Assistance
PE	Provider Edge
PKI	Public Key Infrastructure
POSIX	Portable Operating System Interface for Unix
PPP	Point-to-Point Protocol
PPTP	Point-to-Point Tunneling Protocol
PVC	Permanent Virtual Circuit
QoS	Quality of Service
RA	Registration Authority
RFC	Request for comment
SA	Security Association
SADB	Security Association Database
SHA	Secure Hash Algorithm
SIP	Session Initial Protocol
SP	Security Policy
SPD	Security Policy Database
SPI	Security Parameter Index
TCP	Transmission Control Protocol

UDP	User Datagram Protocol
ULP	Upper and Lower Protocol
UMTS	Universal Mobile Telecommunications System
WAN	Wide Area Network
WAP	Wireless Access Point
WiMAX	Worldwide Interoperability for Microware Access
WLAN	Wireless LAN
xDSL	all types of Digital Subscriber Lines

Chapter 1: Introduction

1.1 BACKGROUND

Wireless networks have seen unprecedented growth during the past few decades. More and more people play or work on mobile devices via wireless networks.

Users can access wireless networks through many different ways, for example UMTS [78], WiMAX [77] and Wireless Access Points. UMTS (Universal Mobile Telecommunications System) is one of the third-generation (3G) mobile telecommunications technologies. Mobile phones can get an IP address directly through the base stations of UMTS. WiMAX (Worldwide Interoperability for Microwave Access), is a telecommunications technology that provides broadband wireless transmission of data via the public Internet. Wireless Access Point (WAP) is a device that also allows wireless communication devices to connect to a wireless network or to a wired network, and finally to the public Internet.

Virtual Private Networks (VPNs) are commonly used to provide secure connections between fixed nodes via the low-cost public network. Telecommuters, business travellers or remote office workers use VPN to connect to their company network when security is a serious concern. However, current VPNs are unusable when running on a wireless network and moving from one network to another. This means that the user must not change IP address otherwise the VPN connection will be lost and will need to be restarted. VPN under wireless networks will soon become a popular new research area, as the number of telecommuters and business travellers has continued to increase.

The VPN mobility problem is caused by the need for the VPN server to check the IP addresses of the VPN tunnel packets. The current solution to this problem is to run VPN tunnels over Mobile IP (MIP). In this case, the IP change is handled by Mobile IP and the IP addresses inside VPN tunnel do not change. However, Mobile IP itself has significant problems in performance and security and that solution is inefficient due to double tunneling.

A VPN usually consists of two components: a VPN concentrator and a VPN server. The VPN concentrator generally receives packets on the OSI Layer 2, 3 or 4, encapsulates and encrypts the packets into network frames, and then sends the network frames to the VPN server via public networks. The VPN server receives the network frames, decrypts and decapsulates the network frames into the original packets, and then sends the original packets to the destination computer. Similar things happen to the returning traffic. People on public networks may capture the packets, but they cannot decrypt the network frames. The network traffic is protected by this process.

Internet Protocol Security (IPsec) is one of the VPN technologies and is a suite of protocols. It receives packets on the OSI Layer 3 (IP Layer) and secures the packets inside the IP layer. IPsec uses Internet Key Exchange (IKE) protocol to generate security keys and to handle security key exchange between the VPN server and the VPN concentrator, and uses Authentication Header (AH) or Encapsulating Security Payload (ESP) to encrypt and to protect IP packets. IPsec has strong security and it has already been integrated into the next generation network (IPv6).

Layer 2 Tunneling Protocol (L2TP) is one of VPN protocols. It receives packets on the OSI Layer 2 (Data Link Layer) and secures the packets inside the OSI Layer 5 (Session Layer). It does not provide strong authentication method by itself and often the L2TP packets are sent inside IPsec for a better security. Compared with current VPN technologies, a VPN that transfers Layer 2 packets has a better range of applications as it can transfer almost all kinds of Internet packets: IP packets, non-IP packets (such as IPX packets) and Layer 2 packets (such as PPP packets [39]). This thesis focuses on VPN technologies that transfer Layer 2 packets.

1.2 THESIS GOALS & SIGNIFICANCE

The main purpose of this thesis is to propose and to implement a new and novel solution on simulators and real devices to solve the mobility problem in a VPN. The new solution adds mobility support to existing L2TP/IPsec (Layer 2 Tunneling Protocol/IP Security) tunnels. It tunnels Layer 2 packets between a VPN client and a VPN server without using Mobile IP,

without incurring tunnel-re-establishment at handoff, without losing packets during handoff, achieves better security than current mobility solutions for VPN, and supports fast handoff in IPv4 networks.

The new solution has particular application when several persons inside a moving vehicle are connected to a network at layer 2 (for example a PPP link [39]). An L2TP/IPsec concentrator inside the vehicle is used to encapsulate Layer 2 packets and then to tunnel to the company network. It is also possible to encapsulate IP packets inside the L2TP/IPsec tunnel. The real world topology of the solution is shown in Figure 1-1. The new solution explained in this thesis handles the situation perfectly and quickly when the vehicle is moving and the L2TP/IPsec concentrator inside the vehicle changes IP addresses from time to time.

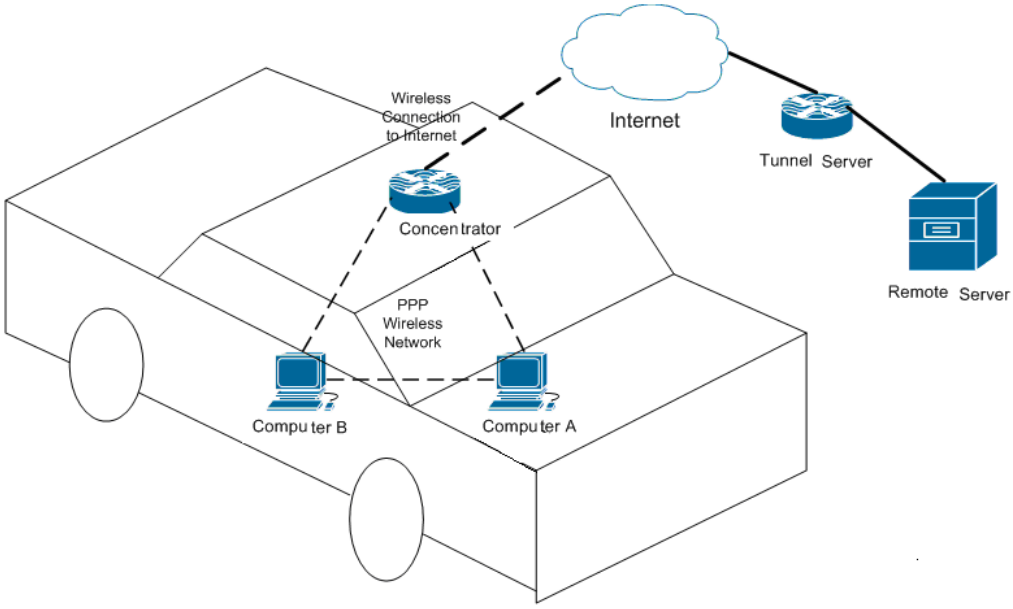


Figure 1-1 Real World Topology

This thesis focuses on reducing the VPN handoff time as much as possible. The time for VPN handoff is mainly caused by the time to get an IP address and the time for VPN negotiation. The time to get a new IP address varies from situation to situation as the new IP address can be got from UTMS, WiMAX [77] or even a wireless access point. Therefore, the main goal of this thesis is to reduce the VPN negotiation time. In the simulation, the time to get a new IP address was minimized by using the "ifconfig" command and a very small VPN handoff time was

achieved. In the real world experiment, a larger VPN handoff time was measured and analysed. This thesis also gives some suggestion of how to reduce the handoff time in the real world which may open a range of new research opportunities.

One journal paper and one conference paper related to this thesis have been published:

C. Xu and P. Radcliffe, "Building Secure Tunnel from PPP Wireless Network", *Wireless Personal Communications*, DOI 10.1007/s11277-009-9894-x, 2009.

C. Xu and P. Radcliffe, "A novel mobility solution based on L2TP/IPsec tunnel", *2009 IEEE Sarnoff Symposium*, 2009.

1.3 LITERATURE SEARCH

The most common solution to mobility problems in a VPN is to run tunnels, such as L2TP or IPsec, over Mobile IP (double tunneling) [21, 22]. However, that is inefficient due to the overhead of network traffic (see Chapter 8.4). The packet structures of these tunnels are shown below.

Figure 1-2 Packet Structures of Double Tunneling

Some solutions [19, 53] are to add mobility support only to IPsec. These solutions can only transfer IP packets (Layer 3 packet). If user wants to send Layer 2 packets (such as PPP packets [39]), an IPsec tunnel cannot provide a solution. Furthermore, these solutions lose packets as they do not provide a method for sending old packets to the new IP address, and have relatively lower performance when handling handoff. Synchronization problems may also occur when the IPsec server updates tunnel information and sends IPsec packets at the same time. Some solution [19] also has security problems. A detailed analysis of these solutions will be shown in Chapter 8.4.

Another solution may be to setup a tunnel to a Mobile IP system using only Mobile IP. However, security is a problem in this solution as will be shown in Chapter 3.

1.4 THESIS OUTLINE

The rest of this thesis is organized as follows:

- Chapter 2 introduces an overview of VPN together with its history and classifications. Then some of the concurrent VPN technologies are reviewed and compared, suggesting why L2TP/IPsec technology is a better choice to focus on.
- Chapter 3 introduces the key concepts of different versions of Mobile IP and analyses the drawbacks of MIPv4 and MIPv6. The possibility to use Mobile IP to transfer packets as a part of VPN without double tunneling is also discussed in this chapter.
- Chapter 4 introduces L2TP and identifies its benefits over similar protocols. The detailed analysis of L2TP packet structure and tunnel establishment as well as its implementation in UNIX (FreeBSD) is also presented.
- Chapter 5 overviews IPsec and presents tunnel establishment and packet protection procedures in detail. Then IPsec architecture and its implementation in FreeBSD are explained. Finally the differences between Microsoft Windows IPsec and UNIX IPsec are discussed.
- Chapter 6 presents a case for using L2TP/IPsec tunnels, and discusses details of L2TP/IPsec tunnels including packet structure, tunnel establishment and tunnel authentication. Finally, different solutions of the L2TP/IPsec tunnel are proposed and compared. The loopback interface solution is chosen to create the L2TP/IPsec tunnel in FreeBSD (UNIX) in this thesis.
- Chapter 7 proposes a detailed solution to handle the mobility problem in the L2TP/IPsec tunnel. This chapter also analyses and discusses the security and the performance of the new solution.

- Chapter 8 presents simulation experiments to validate the proposed solution discussed in the previous chapter. VMWARE software is introduced in this chapter and is used for the simulation. The experiment result is shown and analysed, and compared with four concurrent studies on VPN mobility support.
- In chapter 9, the same theory is proved on real devices (a wireless LAN). The VPN handoff time is fully discussed in this chapter and the experiment result is measured and analysed. Some suggestions are also proposed to further reduce the VPN handoff time.
- Chapter 10 concludes the thesis with future work listed.

Chapter 2: VPN Overview

A Virtual Private Network (VPN) is a connection which provides secure private communication over an insecure network such as the public network [63]. Typically, a VPN provides connections between fix network devices. The term "Private" means that all the traffic inside the VPN is encrypted and the resources are only shared among an authorized group of users, and are controlled by different levels of access control. The term "Virtual" indicates that VPN looks like a private network from the user's perspective and consists of an independently administered virtual topology, although the underlying network is shared by anyone using the network. Furthermore, VPN is cheap, as it normally uses the public network instead of costly leased line services.

This chapter will first introduce some VPN history and classifications. Next different VPN technologies will be reviewed and compared and finally a VPN will be chosen to add mobility support.

2.1 VPN HISTORY

Originally, the VPN was associated with Frame Relay networks [33]. Companies used dedicated lines and layer 2 services such as Frame Relay to interconnect their nodes with links that they owned. Frame relay networks are considered secure, as customer traffic will be sent through a predetermined path (Permanent Virtual Circuit). However, with the rapid development of IP network, VPN began to migrate from a conventional Layer 2 Frame Relay to a Layer 3 IP-based network.

The primary advantages of IP VPNs over Frame Relay VPNs are:

- Reduced network cost (Internet Service Providers charge more for a Frame Relay Permanent Virtual Circuit).
- Easy to provide network connectivity to geographically dispersed offices and remote users.

- Convergence of other services such as voice and video, which reduces cost.

Currently, VPNs provide connections at different OSI layers. VPN has become more and more popular for a variety of reasons; a VPN can be encrypted for security or to defeat firewalls and proxy servers. VPNs make it easier to manage geographically separated physical networks as if they were one network. Businessmen and other persons from remote offices often use VPNs to connect to company networks.

2.2 VPN CLASSIFICATION

VPN can be classified in a variety of ways.

- By topology:
  - Peer to Peer VPN

    Peer to Peer VPN sets up a secure tunnel between two computers via public networks. An IP address will be assigned to each end of the tunnel so that the two computers can communicate with each other as if they are connected by a physical Ethernet cable.

    The limitation of Peer to Peer VPN is that the VPN tunnel can be shared by only two computers. This solution is not widely used due to the limitation.

    The topology of Peer to Peer VPN is shown as follows

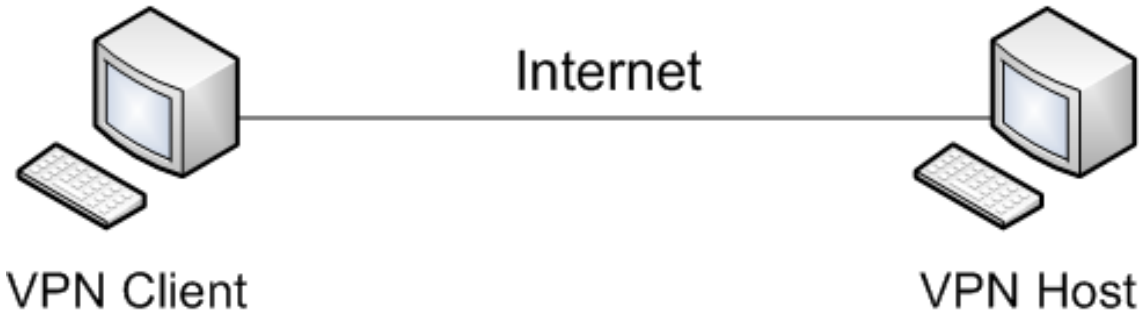


Figure 2-1 Peer to Peer VPN

- Client to Server VPN

Client to Server VPN sets up a secure tunnel between a VPN client and a specific network via public networks. The VPN client can connect to all the computers inside the specific network. However, unlike peer to peer VPN, Client to Server VPN only encrypts the traffic between VPN Client and VPN server, and the traffic between VPN server and other computers in the specific network is not protected.

Although it does not protect the full path between end users (no protection within the company network), Client to Server VPN is widely used in today's networks because businessmen outside usually want to connect to company network, not a single computer.

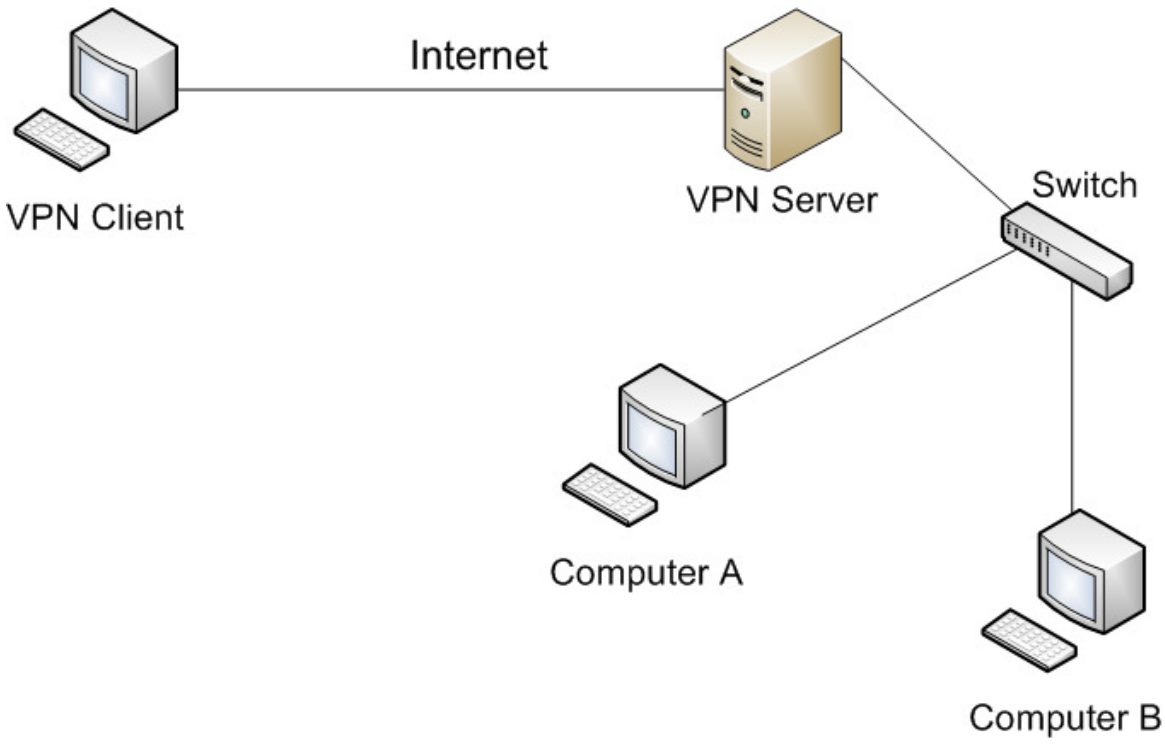


Figure 2-2 Client to Server VPN

- Site to Site VPN

Site to Site VPN sets up a secure tunnel between 2 networks via the public Internet where the tunnel endpoints are a VPN concentrator and a VPN server. These VPNs only encrypt the traffic between VPN concentrators and VPN servers, and any traffic outside the tunnel endpoints is not protected.

Site to Site VPN is widely used between company's main office and remote office.

This thesis has chosen to concentrate on this topology.

The topology of Site to Site VPN is shown in Figure 2-3.

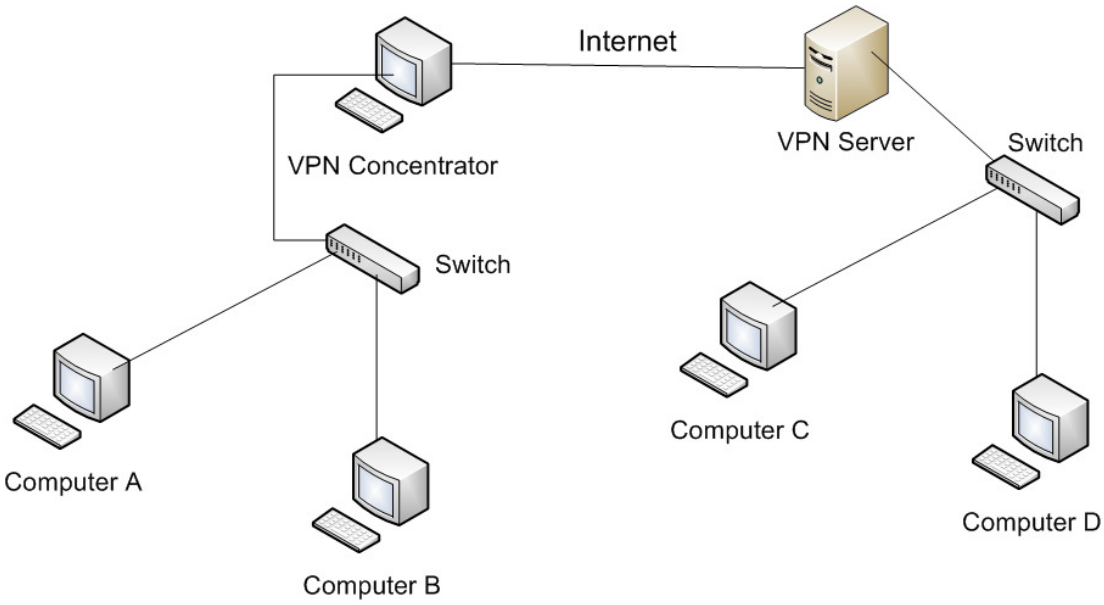


Figure 2-3 Site to Site VPN

- By protocols:

The choice of a VPN protocol depends on the type of traffic to be sent via the tunnel. VPN protocols can be classified according to OSI layers of received packets used for encryption. There are currently 3 kinds of VPN:

- Layer 2 VPN

A Layer 2 VPN encapsulates packets on the OSI Layer 2: Data Link Layer. Main Layer 2 VPN protocols are: Layer 2 MPLS VPN, OpenVPN, PPTP and L2TP. Chapter 2.3 discusses the details of Layer 2 VPN protocols.

- Layer 3 VPN

Layer 3 VPN encapsulates packets on the OSI Layer 3: Network Layer. Main Layer 3 VPN protocols are: Layer 3 MPLS VPN, IPsec and OpenVPN. Chapter 2.4 discusses the details of Layer 3 VPN protocols.

- Layer 4 VPN

Transport Layer Security (TLS) and its predecessor Secure Sockets Layer (SSL) are Layer 4 VPN protocols that encrypt segments of network connections at the OSI Layer 4 (transport layer). A prominent use of TLS is for securing web traffic carried by HTTP to form HTTPS. Although TLS is widely used, it can only encrypt Layer 4 packets, not lower layers. This greatly limits its applications. This thesis uses VPN protocols at lower layers.

2.3 VPN PROTOCOLS IMPLEMENTED ON OSI LAYER 2

This section will analyse and compare main VPN protocols implemented on OSI layer 2 (data link layer).

2.3.1 Layer 2 MPLS VPN

Multiprotocol Label Switching (MPLS) [48] is a mechanism used in high-performance networks and it carries data from one network node to the other. In an MPLS network, labels are added to each data packet and packets are switched according to these labels. MPLS is a scalable protocol as MPLS labels can be added to various network protocols.

Layer 2 MPLS VPN is a type of Virtual Private Network (VPN) that uses MPLS labels to transport OSI Layer 2 packets. It is commonly used when customers want to communicate between remote offices through the Internet Service Provider (ISP) network [35], but they have no access to the public Internet. The edge routers on the Service provider side are called Provider

Edge (PE) routers and the edge routers on the customer side are called Customer Edge (CE) routers. The topology of a Layer 2 MPLS VPN network is shown in Figure 2-4.

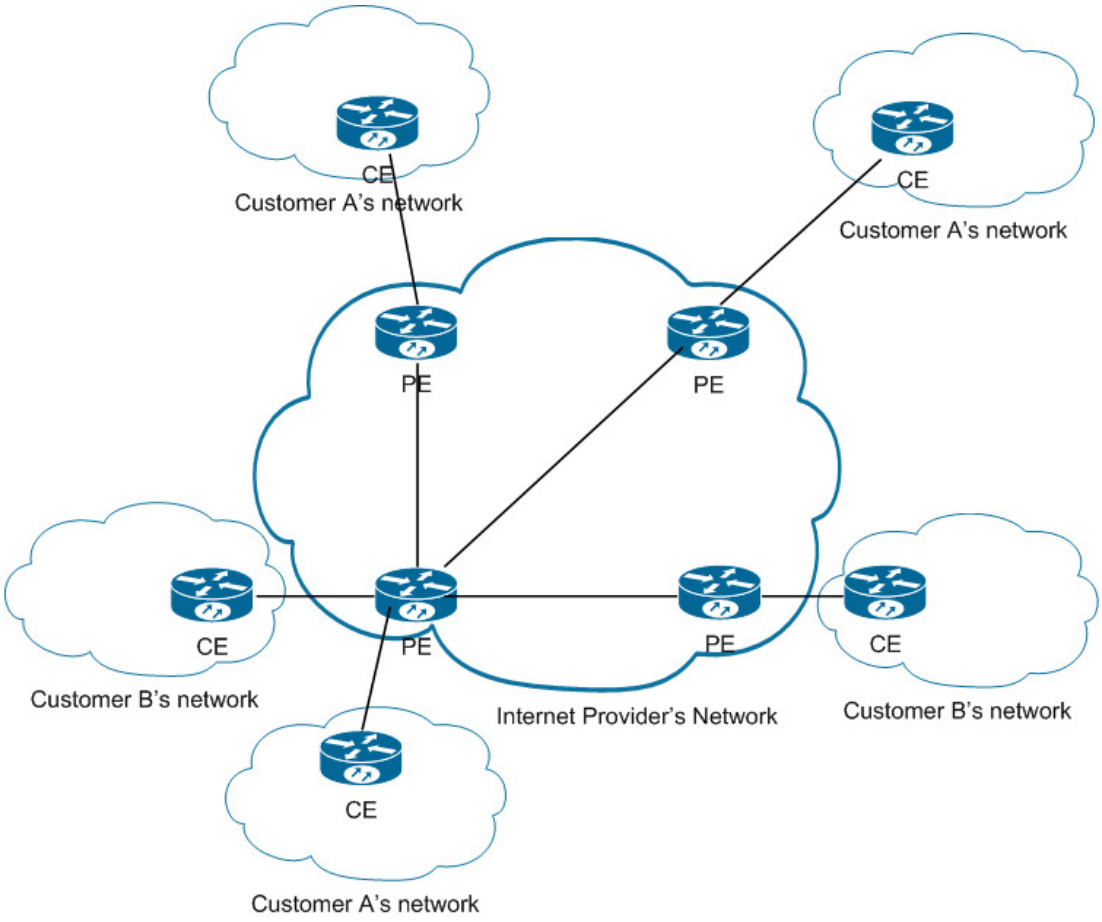


Figure 2-4 Layer 2 MPLS Network Topology

Layer 2 MPLS VPN networks are quite fast. All kinds of traffic, i.e. Frame Relay (FR), Asynchronous Transfer Mode (ATM) and Ethernet traffic, can be sent through the network. The Provider Edge (PE) routers are not responsible for routing and they only forward packets according to Layer 2 information and MPLS labels.

All traffic going through Internet Provider's network is protected by Layer 2 MPLS VPN because other customers cannot access these packets.

Security is a big issue for Layer 2 MPLS VPN. If several customers share a Layer 2 medium on ISP network, there is often no control over the packets transferred to that device so that the packets from other customers can be easily captured. The chance for using exclusive network devices on ISP network is very limited because of the high cost. One solution is to use a port-based Ethernet connection between two physical data ports provided across an MPLS

network. This means that the Layer 2 packets are encapsulated in 802.1Q Ethernet frames and sent to the destination. Another big security issue is that Layer 2 MPLS VPN packets are not encrypted in ISP network. [34]

Layer 2 MPLS VPN has not been chosen to add mobility support because of its security issues.

2.3.2 **OpenVPN**

OpenVPN is an open source Layer 2 or Layer 3 tunneling protocol. It works by encapsulating Layer 2 and Layer 3 packets inside UDP or TCP packets and sending them to the destination. It uses OpenSSL for encryption and implements SSL and TLS (the advanced and standardized version of SSL) [4]. It uses pre-shared, certificate-based, and username/password-based key for authentication. It is capable of establishing direct links between computers across network address translators (NATs) and firewalls. It is easy to configure but it has not been widely used [9].

The packet structure of OpenVPN is shown in Figure 2-5.

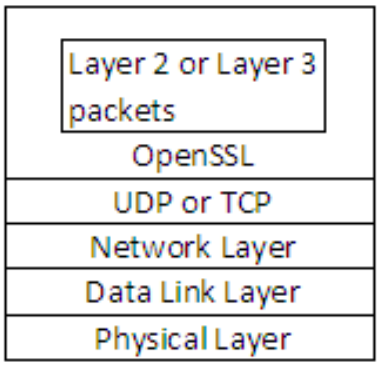


Figure 2-5 Packet Structure of OpenVPN

The main problem in OpenVPN is security. The key exchange in TLS is weak, for example completely anonymous sessions are vulnerable to man-in-the-middle attacks and public key and private keys are exposed in RSA key exchange. OpenVPN is not recommended when security is a concern [5].

OpenVPN by itself is not useful for mobile business scenarios as it has no native ability to cope with mobile clients.

2.3.3 **PPTP**

PPTP (Point-to-Point Tunneling Protocol) [3] is a layer 2 tunneling protocol which works by sending a regular PPP session [39] to a peer with the Generic Routing Encapsulation (GRE) protocol. A second session is used to initiate and manage the GRE session. This session is a simple TCP connection from the PPTP client to port 1723 on the PPTP server. PPTP also works in sending IPX packets [27].

The main disadvantage in PPTP is the security. PPTP itself does not specify any authentication or encryption algorithms, and the only algorithms used are inside the PPP sessions [39]. Microsoft Challenge-handshake authentication protocol (MS-CHAP) [37] and Microsoft Point-to-Point Encryption (MPPE) [38] are used for PPP authentication and encryption. MS-CHAP is known to be a weak algorithm, easily cracked by software such as L0phtcrack. MPPE is also weak in security because an attacker can spoof resynchronize keys packets easily [36]. Also, there are many unauthenticated control packets that are readily spoofed [3].

PPTP is widely used in Microsoft Windows and some parts of it are patent encumbered. It has no native ability to cope with mobile clients.

2.3.4 **L2TP**

L2TP (Layer 2 Tunneling Protocol) [28] is an open source layer 2 tunneling protocol. It is originally used to encapsulate PPP frames into UDP packets and send UDP packets over existing networks. The two endpoints of an L2TP tunnel are the LAC (L2TP Access Concentrator) and the LNS (L2TP Network Server). The LAC receives PPP packets from users, encapsulates the PPP packets into UDP packets and then sends these to the LNS. The LNS decapsulates the UDP packets and sends the PPP packets to the destination computers. IP packets can also be tunnelled through L2TP and the process of tunneling IP packets is similar to that of tunneling PPP packets. L2TP does not provide strong authentication by itself and often uses IPsec to secure the tunnel [28]. The topology of an L2TP tunnel is shown in Figure 2-6.

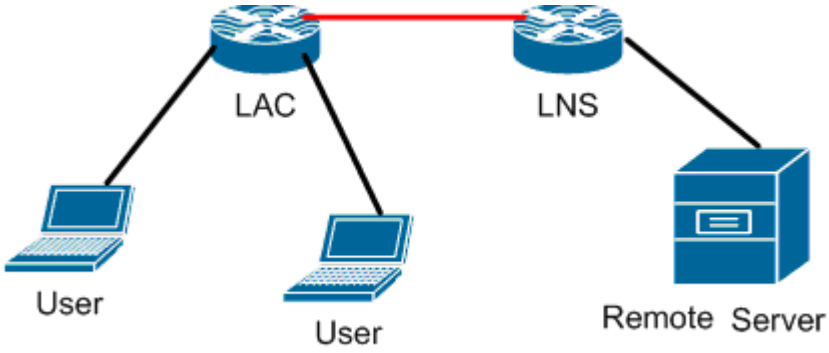


Figure 2-6 L2TP Topology

The details of L2TP will be discussed in Chapter 4.

A problem with L2TP/IPsec tunneling is that it does not support NAT. However, IPv6 (next generation network) has an almost infinite number of addresses that makes NAT unnecessary [18].

L2TP by itself is not useful for mobile business scenarios as there is no native ability to cope with mobile clients.

2.4 VPN PROTOCOLS IMPLEMENTED ON OSI LAYER 3

This section will analyse and compare main VPN protocols implemented on OSI layer 3 (Network Layer).

2.4.1 Layer 3 MPLS VPN

Similar to Layer 2 MPLS VPN, Layer 3 MPLS VPN, also known as L3VPN, is a type of VPN that uses MPLS labels to transport OSI Layer 3 packets. It is commonly used when customers want to communicate between remote offices through the Internet Service Provider (ISP) network [35]. Customers can still access the public Internet through L3VPN via an Internet Customer Edge router though strict security policies should be applied to the Internet Customer Edge router. The topology of a Layer 3 MPLS VPN network is shown in Figure 2-7.

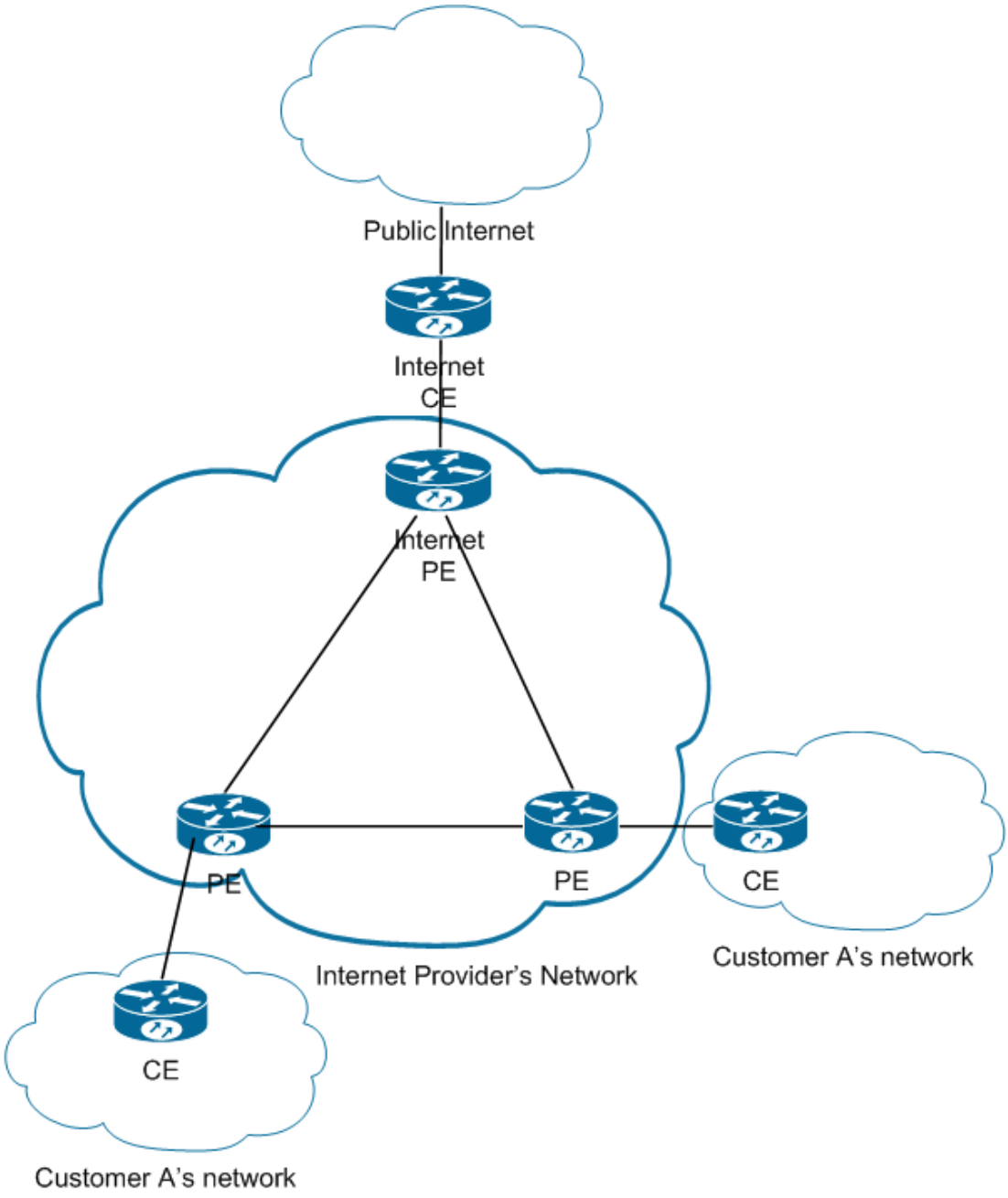


Figure 2-7 Layer 3 MPLS Network Topology

Layer 3 packets are protected by Layer 3 MPLS because other customers cannot access these packets. Unlike Layer 2 MPLS VPN, the Provider Edge (PE) routers in Layer 3 MPLS VPN are responsible for routing and forwarding packets according to IP addresses and MPLS labels.

Security is also a big drawback of Layer 3 MPLS VPN. The VPN does not provide any confidentiality or integrity services. This means that a service provider can easily sniff VPN data

and there is no guarantee that the packets are not corrupted or changed during transfer. Customers can only trust the service provider, or give up this VPN solution [34].

Layer 3 MPLS VPN has not been chosen to add mobility support because of its security issues.

2.4.2 IPsec

Internet Protocol Security (IPsec) [8, 30] is a suite of protocols for securing IP communications at the OSI Network Layer. It encrypts IP frames into IPsec packets and sends the packets to the other end of the networks. It supports peer authentication, data integrity and data confidentiality (encryption). IPsec can be used to protect IP packets (OSI Layer 3 packets) between a pair of hosts (Peer to Peer VPN), between a security gateway and a host (Client to Server VPN), or between a pair of security gateways (Site to Site VPN).

Compared to other VPN protocols, IPsec is a suite of VPN protocols with very strong security. It is very popular and has already integrated into the next generation network (IPv6).

IPsec is a complex system which includes encapsulation, encryption, authentication, and key exchange and management. The details of IPsec will be discussed in Chapter 5.

IPsec by itself is not useful for mobile business scenarios as there is no native ability to cope with mobile clients. An IPsec extension adds mobility support to IPsec, which is discussed in RFC 4555 [53]. However, that solution has some limitations which will be discussed in Section 8.4.

2.4.3 OpenVPN

OpenVPN can tunnel packets on the OSI Layer 3 as well as Layer 2. At Layer 3 it has the same advantages and disadvantages as at Layer 2. See Section 2.3.2 for more details.

2.5 CHOOSING A VPN TO ADD MOBILITY SUPPORT

The VPN protocols examined do not have a native ability to cope with mobile clients. Adding mobility support to existing VPN protocols is one way to solve the problem. The final solution should have a wide range of applications, good security, small handoff time and simplicity of usage.

A VPN that transfers Layer 2 packets will be chosen as it has a better range of applications and can transfer almost all kinds of Internet packets: IP packets, non-IP packets (such as IPX packets) and Layer 2 packets (such as PPP packets [39]). A brief comparison among different Layer 2 VPN is shown below.

- Layer 2 MPLS VPN has big security issues. It assumes that ISP network can be trusted and all the packets within ISP network are not encrypted (see Section 2.3.1).
- OpenVPN is not widely used and is relatively weak in security (see Section 2.3.2).
- PPTP is weak in security and is patent encumbered (see Section 2.3.3). It is difficult to modify PPTP.
- L2TP provides Layer 2 tunneling functions (see Section 2.3.4) and together with IPsec provides good security (see Section 2.4.2). Although L2TP/IPsec tunnels do not support NAT, IPv6 (next generation network) has an almost infinite number of addresses that makes NAT unnecessary.

The L2TP/IPsec tunnel has been chosen to add mobility support because it has a good range of applications (transferring Layer 2 packets) and is strong in security (using IPsec).

2.6 SUMMARY

In this chapter, firstly, the definition of VPN is introduced together with its history and classification. Secondly, some of the dominant VPN technologies have been reviewed and compared. Finally, a comparison is made among different VPN technologies and a decision is made to choose a particular VPN technology to add mobility support.

A Virtual Private Network (VPN) is a connection which provides secure private communication over an insecure network. VPNs can be classified by topology or by protocol and the examined VPNs do not have native mobility support. L2TP is an open source layer 2 tunneling protocol which does not provide strong authentication by itself and often uses IPsec to

secure the tunnel. L2TP/IPsec is most suitable for adding mobility support as other VPN protocols have problems with security or other issues.

Chapter 3: Mobile IP Overview

Mobile IP (MIP) is one of the most popular solutions for handling IP mobility problems at OSI Network Layer. It is a standard communications protocol that is designed to allow mobile device users to move from one network to another while maintaining a permanent IP address [23]. There are two versions of Mobile IP: Mobile IPv4 [23] and Mobile IPv6 [15], which work for IPv4 [40] and IPv6 [41] networks respectively.

This chapter will first discuss the key concepts of different versions of Mobile IP and then analyse the problems faced by Mobile IP when transferring packets as a part of VPN without double tunneling.

3.1 OVERVIEW OF MIPV4

Mobile IPv4 is a protocol enhancement that allows routing of IP packets to a moving node under IPv4 networks. Only the OSI Network Layer is enhanced to handle the problems so that the upper layer softwares can be used without any modification.

The basic components of Mobile IPv4 are:

(1) Mobile Node (MN): A host that changes its point of attachment from one network to another. Its IP address will change in this situation.

(2) Correspondent Node (CN): A host communicating with a Mobile Node.

(3) Home Agent (HA): A router on a Mobile Node's home network. It keeps a permanent IP address (Home Address) for each Mobile Node and maintains current location (Care-of Address) for the Mobile Node. It is responsible for tunneling packets to Foreign Agent when the Mobile Node is away from home, and advertising itself.

(4) Foreign Agent (FA): A router on a Mobile Node's visited network which provides routing services to the Mobile Node when registered. It is responsible

for advertising itself, de-tunneling packets from the Home Agent, and sending the packets to the Mobile Node.

(5) Home Address: The IP address assigned to a Mobile Node in the Home Network. The IP address will not change when the Mobile Node is roaming.

(6) Care-of Address (CoA): The IP address which is assigned to the Foreign Agent. A Home Agent tunnels packets to Mobile Node's Care-of Address.

Home Agents and Foreign Agents advertise their presence by broadcasting Agent Advertisement messages. The Mobile Node examines the Agent Advertisement messages to determine its location. If the MN is connected to its home network, it operates without mobility services.

When the Mobile Node is connected to a foreign network, it registers the Care-of Address with its Home Agent. The registration process sets up necessary services to re-route packets.

When the Correspondent Node sends packets to the Mobile Node, the packets are first sent to the Home Agent. The HA encapsulates and tunnels the packets to the Foreign Agent and the FA de-tunnels the packets and sends the packets to the Mobile Node.

When the Mobile Node sends packets to the Correspondent Node, the packets are first sent to the Foreign Agent. The FA directly forwards the packets to the Mobile Node.

Figure 3-1 shows the operation of a Mobile IP (IPv4) connection.

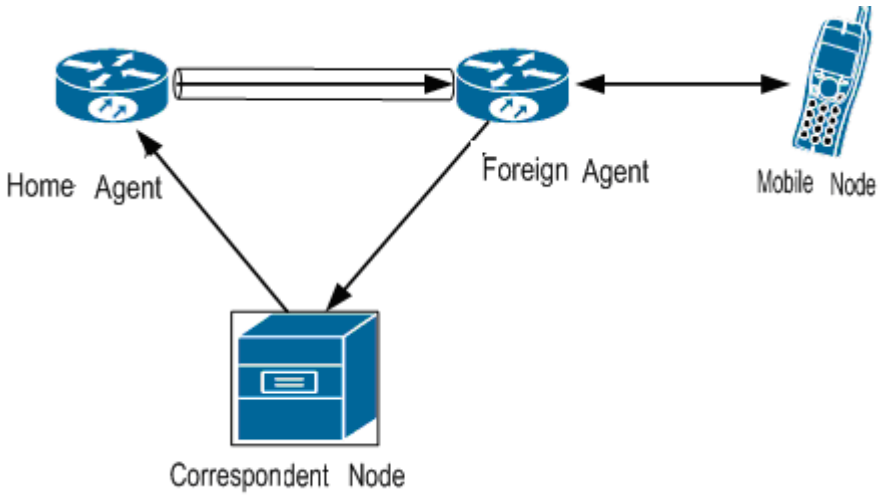


Figure 3-1 Mobile IPv4

3.2 PROBLEMS IN MIPV4

When the Mobile Node roams in foreign networks, no matter how close the correspondent node and the mobile node are, MIPv4 forces all packets from the Correspondent Node to be routed to the Home Agent and then to be routed to the Mobile Node. However, packets from the Mobile Node can be routed to the Correspondent Node through the Foreign Agent directly. This mechanism has poor packet routing performance and is labelled "Triangular Routing" [82] which is depicted in Figure 3-1. The word "Triangular Routing" is routing which causes sending packets to a proxy system before sending to the intended destination. Both Mobile IP and Skype have this problem.

The security in Mobile IPv4 is weak. One of the security problems is that the Address Resolution Protocol (ARP) is not authenticated, and can potentially be used to steal another host's traffic [20]. Another security problem is that Mobile IPv4 security uses Authentication, Authorization, and Accounting (AAA) systems for key distribution. However, AAA systems are not supported by all parties [20]. One AAA example is RADIUS (Remote Authentication Dial In User Service), a networking protocol that provides centralized authentication, authorization and accounting management for computers to use a network service [26].

Another problem with Mobile IPv4 is the large external equipment requirement as Mobile IPv4 systems need to setup many Home Agents and Foreign Agents.

Due to the security and other problems stated above, it is not recommended to use Mobile IPv4 to transfer packets to Mobile Node as a part of VPN without double tunneling.

3.3 OVERVIEW OF MIPV6

Mobile IPv6 is the Mobile IP protocol working under IPv6 (next generation network). Mobile IPv6 is similar to Mobile IPv4 and it has many improvements. Mobile IPv6 discards Foreign Agents and supports Route Optimization as a fundamental component. Route Optimization provides a method for Correspondent nodes to cache the binding of a mobile node

and then to send their own packets directly to the Care-of Address (the Mobile Node itself) indicated in that binding, and so bypassing the mobile node's home agent.

When the Mobile Node is away from its Home Network, it uses its current IP address as the Care-of Address. It registers the Care-of Address with the Home Agent and binds the new address with the Correspondent Node through the Home Agent. The registration process is protected by IPsec and the binding is protected by cookies (random numbers used to prevent spoofing) [15] and tokens (numbers used to compute keys) [15].

After a successful binding update, packets are sent between the Correspondent Node and the Mobile Node directly, without using Home Agents. The Correspondent Node examines its binding cache before sending any packets. If the binding cache has an entry for the address, a routing header with Home Address is added to the packet and the Correspondent Node sends the packet to the Care-of Address of the Mobile Node.

The operation of Mobile IPv6 is shown in Figure 3-2 and Figure 3-3.

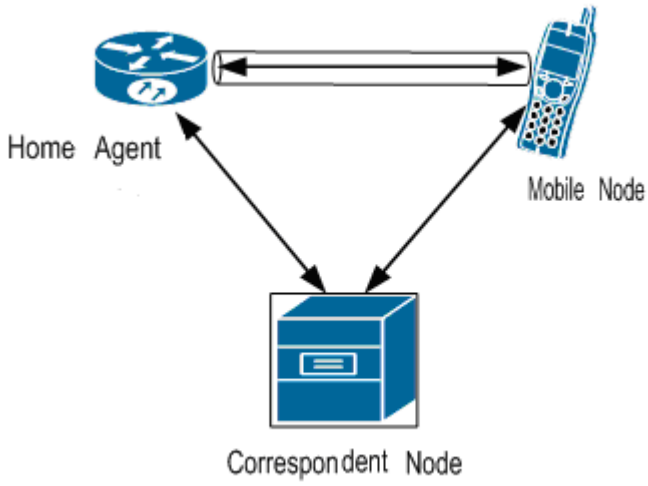


Figure 3-2 Binding and Registration at Mobile IPv6 Handoff

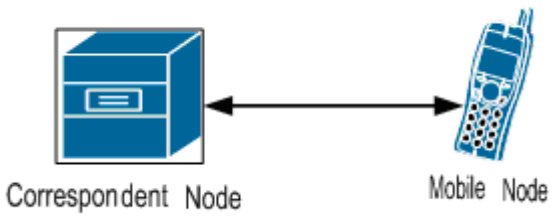


Figure 3-3 Normal Traffic of Mobile IPv6

3.4 PROBLEMS IN MIPV6

Although Binding Updates and Mobile Prefix Discovery is protected by the use of IPsec extension headers [15], the introduction of route optimization introduces new security threats from many kinds of denial of service (DoS) and redirection attacks [14].

Also, data packets exchanged with mobile nodes are exposed to similar threats as that of regular IPv6 traffic [15]. For example, packet confidentiality is not provided by Mobile IPv6. In other words, on-path attackers can easily capture the packets and analyze the content of the packets.

Due to the security problems stated above, it is not recommended to use Mobile IPv6 to transfer packets to Mobile Node as a part of VPN without double tunneling. Double tunneling is necessary to transfer packets to Mobile Nodes in a secure VPN using Mobile IP.

3.5 SUMMARY

In this chapter, an overview of different versions of Mobile IP has been presented together with a discussion of the possibility to use Mobile IP to transfer packets as a part of VPN without double tunneling. Mobile IP is one of the most popular solutions for handling IP mobility problems at OSI Network Layer. However, Mobile IPv4 has the triangle routing problem, needs significant equipment and has a lot of security problems. Mobile IPv6 improves Mobile IPv4 by introducing Route Optimization and IPsec, but it still has security problems. The most severe problem of Mobile IPv6 is that it does not provide packet confidentiality which is essential to a secure VPN. Due to the problems stated above, it is not recommended to use Mobile IP to transfer packets to Mobile Node as a part of a secure VPN without double tunneling.

Chapter 4: L2TP

This chapter will introduce the Layer 2 Tunneling Protocol (L2TP). It will explore why L2TP is developed, its benefits, how does it work and its implementation under the FreeBSD Operating System (a UNIX Operating System).

This discussion is based on a proposed standard RFC 2661 [28]. A new version of this protocol, L2TPv3, was published on RFC 3931 [42]. L2TPv3 provides better security and adds a mechanism to tunnel Layer 2 packets other than PPP packets [42]. The implementation of L2TPv3 on FreeBSD is not stable till now (18/09/2009) and the algorithm used to tunnel other Layer 2 packets in L2TPv3 is similar to the algorithm used to tunnel PPP packets in L2TP. Also, security is not important for L2TP because L2TP is protected by IPsec, a robust security protocol, in our scenario. For the above reasons this thesis has chosen to use L2TP instead of L2TPv3.

4.1 OVERVIEW OF L2TP

The Point-to-Point Protocol (PPP) [43] is a popular data link (Layer 2) protocol commonly used to transfer Layer 2 packets between adjacent nodes, such as a serial cable, a fiber optic link etc. It can provide authentication, encryption and compression. Most Internet Service Providers (ISPs) use PPP for customer's dial-up and xDSL access to the Internet.

PPP is comprised of three main components:

(1) A method for encapsulating network packets.

(2) Link Control Protocol (LCP) [43, 44] for configuring and testing data-link connections.

(3) Network Control Protocols (NCPs) [43] for establishing and configuring network layer protocols. Among NCPs, the most famous one is Internet Protocol Control Protocol (IPCP) [45] for the Internet Protocol (IP).

L2TP is an open source layer 2 tunneling protocol. It was originally introduced to extend the PPP model by allowing the Layer 2 (PPP) endpoints to reside on different devices interconnected by a packet-switched network [28] instead of adjacent nodes.

L2TP [49] origins from two older tunneling protocols for PPP: Layer 2 Forwarding (L2F) [47] and Point-to-Point Tunneling Protocol (PPTP) [27]. L2F is a Layer 2 tunneling protocol developed by Cisco. It has been merged to L2TP because it supports simultaneous tunnels. PPTP is also a Layer 2 tunneling protocol which has been discussed in Section 2.1.3. The main disadvantage of PPTP's early versions is that it supports only one tunnel at a time for each user. L2TP successfully combines the advantages of L2F and PPTP without suffering from their disadvantages.

Like PPTP, L2TP tunnels Layer 2 packets but it transfers these packets on OSI Layer 5. It encapsulates PPP frames into a UDP packet and tunnels network traffic over existing networks. The two end points of an L2TP tunnel are L2TP Access Concentrator (LAC) and L2TP Network Server (LNS). LAC is used to collect customer packets and send in an L2TP tunnel. LNS is used to receive L2TP packets from different customers (LACs). A typical L2TP topology is shown in Figure 4-1.

The LAC receives PPP packets from different users, encapsulates the PPP packets into UDP packets, and then sends the UDP packets to the LNS. The LNS receives the UDP packets from different customers (different LACs), decapsulates the UDP packets and sends the original PPP packets to the destination computers (Remote Server in Figure 4-1). IP packets can also be tunnelled through L2TP and the process of tunneling IP packets is similar to that of tunneling PPP packets.

L2TP allows different sets of PPP peer terminals (computers behind LAC) to utilize one tunnel via different sessions. In other words, an L2TP tunnel will be created between LAC and LNS, and different customers will use separate sessions inside the L2TP tunnel. Figure 4-2 shows multi-sessions inside a tunnel [49].

L2TP tunnel is bidirectional while L2TP session is directional. Typically one L2TP tunnel and two L2TP sessions are used between a pair of end users.

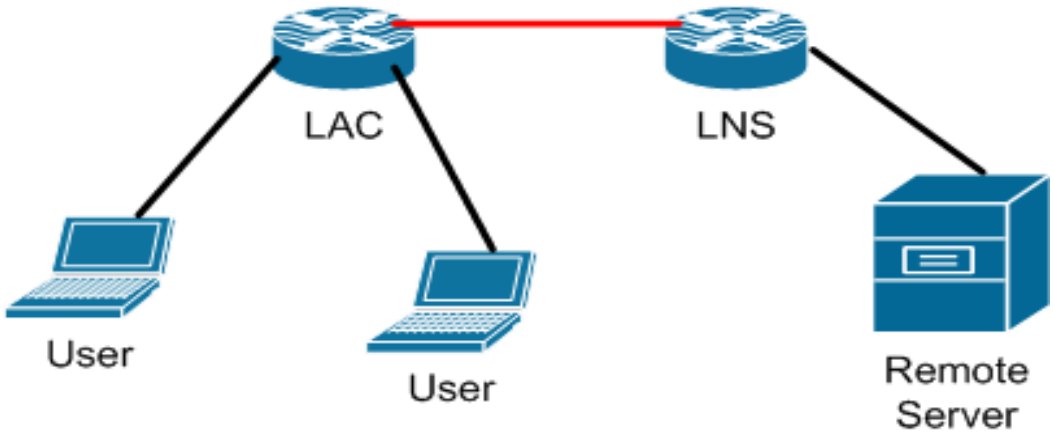


Figure 4-1 L2TP Topology

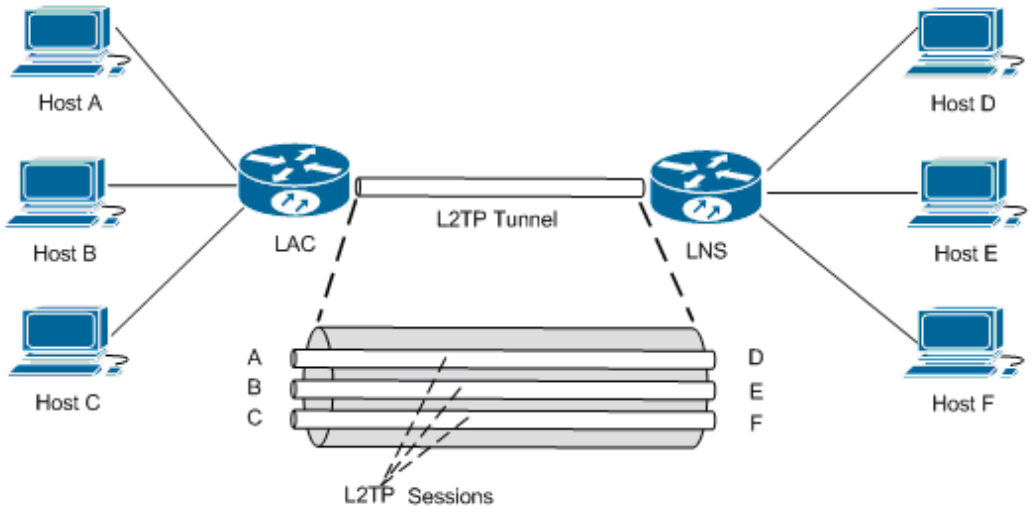


Figure 4-2 L2TP Tunnel and Sessions

L2TP does not provide strong authentication by itself and often uses IPsec to secure the tunnel [28].

A problem with L2TP/IPsec tunneling is that it does not support NAT [51, 52]. However, IPv6 (next generation network) [41] has an almost infinite number of addresses that makes NAT unnecessary [18].

L2TP by itself is not useful for mobile business scenarios as there is no native ability to cope with mobile clients.

4.2 L2TP MESSAGES

The packets exchanged within an L2TP tunnel are categorised as either control messages or data messages. Control messages are used to establish, maintain and clear L2TP tunnels, and data messages are used to encapsulate PPP packets into L2TP tunnel packets and to transfer the data frames over the L2TP tunnel.

A common L2TP header is used for both L2TP control messages and L2TP data messages. The control messages are transported reliably by using sequence number in L2TP header and using retransmission algorithm, while the data messages are transported unreliably. Figure 4-3 and Figure 4-4 show the packet structure of L2TP data messages and L2TP control messages.

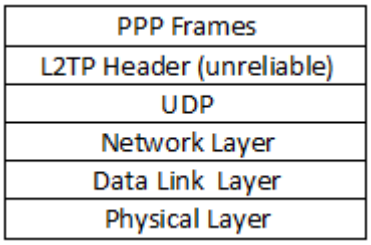


Figure 4-3 Structure of L2TP Data Messages

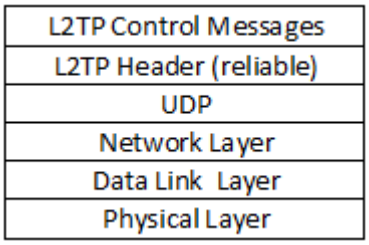


Figure 4-4 Structure of L2TP Control Messages

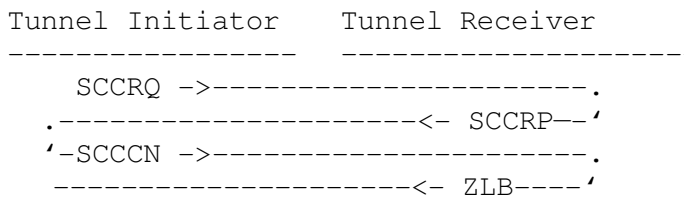
The payload (L2TP Control Messages in Figure 4-4) of an L2TP control message consists of several Attribute Value Pairs (AVPs). Each AVP represent an attribute of an L2TP control message, for example “control message type” is a mandatory attribute for all control messages and “random vector” (is used to hide sensitive control message data such as user passwords) is also a mandatory attribute when the message is hidden. Typically, multiple AVPs are used in an L2TP control message which will be exchanged between the LAC and the LNS.

Different L2TP tunnels use different UDP port numbers for communication. Port 1701 is only used in L2TP initial tunnel creation stage. In order to setup an L2TP tunnel, a tunnel receiver (LNS or LAC) listens on port 1701 for incoming L2TP connection. A tunnel initiator (LNS or LAC) selects an idle port (which is unnecessarily 1701) to send a packet to port 1701 of the LNS. The tunnel receiver receives the packet, selects an idle port (which is unnecessarily 1701 either) and notifies the LAC about the new communication port. From then on, the LAC-LNS pair uses the negotiated ports to communicate until the tunnel is disconnected.

4.3 L2TP TUNNEL ESTABLISHMENT

There are three steps to setup a PPP connection through L2TP tunnel [28]:

(1) Establish a control connection of an L2TP tunnel. A three message exchange is employed to setup control connection. A tunnel initiator sends a Start-Control-Connection-Request (SCCRQ) message to a tunnel receiver. The receiver sends a Start-Control-Connection-Reply (SCCRP) message back. The initiator responses with a Start-Control-Connection-Connected (SCCCN) message. The receiver acknowledges the packet by sending a Zero-Length Body (ZLB) message. L2TP tunnel authentication maybe used within these messages (typically not). The typical message exchange is shown below.



(2) Establish an L2TP session triggered by PPP request. Similar three message exchange is used in this step.

(3) PPP initialization. Each end of the PPP link must first send LDP [43] (Link Control Protocol) packets to configure and test the data link (OSI Layer 2). After the link has been established, the peer usually authenticates using PPP Challenge Handshake Authentication Protocol (CHAP). Then, PPP must send NCP [43] (Network Control Protocol) packets to choose and configure network layer protocols. After that, data

packets can be sent though the PPP [43] link. For more details about CHAP, please refer to RFC 1994 [46].

After the steps above, PPP packets can be exchanged through a L2TP tunnel.

4.4 L2TP IMPLEMENTATION

There are many L2TP implementations on Cisco equipment, Linux, UNIX and Windows. This thesis will only analyse the implementations under UNIX.

The implementation of L2TP under UNIX can be divided into 2 parts: kernel and user space. The experimental work done in this paper uses FreeBSD 7.0 (a free UNIX-like operating system) kernel and the user space daemon MPD 5.2 (Multi-link PPP daemon for FreeBSD) [2]. The kernel part is responsible for receiving and sending PPP packets to end users, encapsulating and decapsulating L2TP packets, and sending and receiving UDP packets between LAC and LNS. The user space part is responsible for creating and managing the L2TP tunnel, and saving and managing L2TP policies and information.

Different implementations use different software architectures. MPD uses the netgraph module [50] for implementation. The netgraph system provides a way to implement kernel networking functions under user space. In order to achieve this goal, programs under user space first register a node type (such as a protocol like PPP) into the kernel, and then register and implement other functions of that node (such as encapsulating). Finally, kernel functions are implemented under user space. In this way, MPD can handle different kinds of packets, for example PPP packets or IP packets, by implementing different protocols. In this way, only a small amount of code needs to be added when introducing a new packet type.

4.5 SUMMARY

In this chapter, firstly, the history of L2TP is introduced together with a discussion of its benefits over other protocols. Secondly, an overview of L2TP has been presented, followed by the detailed analysis of L2TP packet structure and tunnel establishment. Finally, the implementation of L2TP under FreeBSD is discussed.

L2TP is an open source Layer 2 tunneling protocol. It encapsulates PPP frames into UDP packets and tunnels network traffic over existing networks. It extends the PPP model by allowing the Layer 2 (PPP) endpoints to reside on different devices interconnected by a packet-switched network [28]. The latest version of L2TP (L2TPv3) can tunnel all kinds of Layer 2 packets, which extends the range of applications. L2TP does not provide strong authentication by itself and often uses IPsec to secure the tunnel [28].

The implementation of L2TP under UNIX can be divided into 2 parts: kernel and user space. The kernel part is responsible for sending, receiving, encapsulating and decapsulating packets. The user space part is responsible for creating and managing the L2TP tunnel, and saving and managing L2TP policies and information.

L2TP by itself is not useful for mobile business scenarios as there is no native ability to cope with mobile clients.

Chapter 5: IPsec

Internet Protocol Security (IPsec) is a suite of protocols for securing IP communications by authenticating and encrypting each IP packet of a data stream. It defines a standard and robust mechanism to provide security to IP and upper layer protocols. Because of its strong security, IPsec has already become an integral part of the IPv6 (next generation network) protocol.

This chapter will first introduce the Internet Protocol Security (IPsec), how it works and its implementation under FreeBSD Operating System. Finally, this chapter will discuss the differences between Microsoft Windows IPsec and UNIX IPsec.

Advanced readers should also read this chapter carefully to get a better understanding of IPsec and its implementation, so that they can understand the rationality and advantages of the new and novel L2TP/IPsec solution explained in later chapters.

5.1 OVERVIEW OF IPSEC

IP packets have no inherent security, so it is easy to modify IP packets, replay old packets, or view the contents of IP packets in transmission. IPsec is a suite of protocols which provides authentication, integrity and confidentiality services to IP packets at Network Layer (OSI Layer 3, same layer as IP). IPsec can be used as a Client to Server VPN or a Site to Site VPN. The typical topology is shown in Figure 5-1. With a Site to Site VPN, the concentrator receives IP packets from computers, encapsulates the packets into IPsec packets and sends them to the tunnel server. The tunnel server decapsulates IPsec packets into original IP packets and sends the IP packets to their destination (Remote Server in Figure 5-1). The same algorithm is used in Client to Server VPN except that the concentrator and the customer computers (Computer A and Concentrator in Figure 5-1) are actually on the same computer.

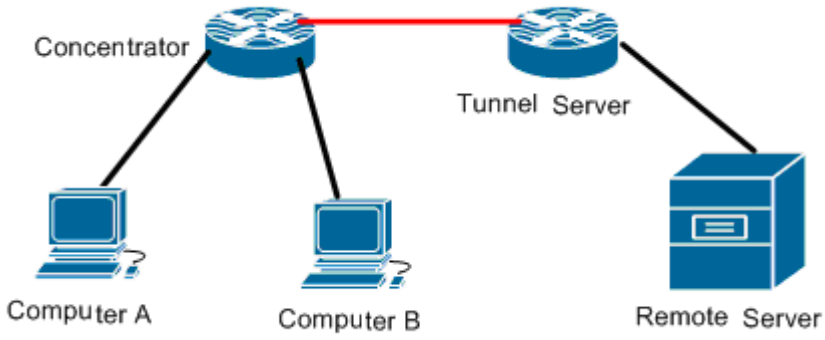


Figure 5-1 IPsec Tunnel Topology

IPsec is a suite of open standards, and it can be divided into four components. The fundamental components of IPsec are shown as follows:

(1) Encapsulation Protocols -- Authentication Header (AH) [13] and Encapsulating Security Payload (ESP) [12].

(2) Security Policy Management -- Security Policies (SPs) and Security Associations (SAs) are used and managed by IPsec [11].

(3) Key Management -- the generation, distribution and storage of cryptographic keys. The Internet Key Exchange (IKE) [11] is used to establish a shared secret key to create IPsec security associations.

(4) Algorithms for cryptographic algorithm.

AH and ESP are the packet structures used to protect IP packets. The AH and ESP headers are inserted after an IP header and before the data to be protected, so AH and ESP are Layer 3 protocols. There are “two modes” to protect IP packets: transport mode and tunnel mode. Transport mode is used to protect IP payload or upper-layer protocols, while tunnel mode is used to protect the entire IP packet (including IP header). Both transport mode and tunnel mode can be used under AH or ESP.

ESP [12] is used to provide confidentiality, data source authentication, integrity, an anti-replay service. It does so by inserting an ESP header and an ESP trailer. ESP authentication data may also be added to the end of an ESP packet. Protocol Number 50 is assigned to ESP packets.

In an ESP packet, that number will be put in the protocol field of an IPv4 header or in the "Next Header" field of an IPv6 header. In transport and tunnel mode, the packet structure of ESP is shown in Figure 5-2 and Figure 5-3. ESP is an integral part of IPv6 protocol and ESP header is an extension header in IPv6. ESP header should appear after hop-by-hop, routing, and fragmentation extension headers and the protected data should put after all extension headers.

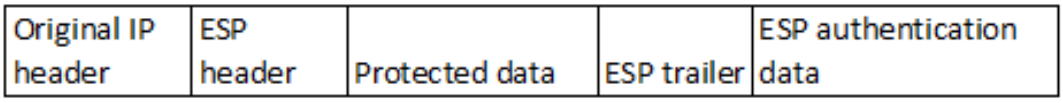


Figure 5-2 Packet Structure of ESP Transport Mode

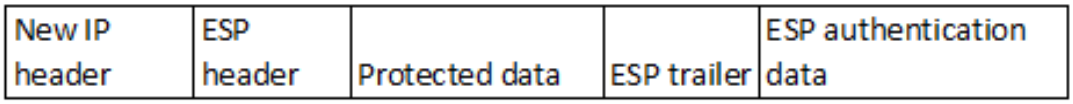


Figure 5-3 Packet Structure of ESP Tunnel Mode

The ESP header is not encrypted because of the specified order of processing of ESP packets: first verify the sequence number, then check the integrity of the data, and then decrypt the data according to Security Parameters Index (SPI, a part of ESP header). ESP packets cannot be decrypted if ESP header is encrypted as sequence number and SPI are necessary parts of ESP header.

Like ESP, AH [13] provides data source authentication, integrity, and optionally an anti-replay service. It only uses a header to protect data and it does not provide confidentiality. Without confidentiality service, hackers can easily view the contents of tunnelled packets. Fortunately, AH can be applied alone or in combination with ESP. Using AH with ESP is out of the scope of this thesis due to its complexity. AH provides authentication for much of the IP header in transport mode, as well as for upper level protocol data. Some IP header fields are not protected as they may change in transit and may not be predictable by the sender.

AH has a protocol Number 51. In an AH packet, that number will be put in the protocol field of an IPv4 header or in the "Next Header" field of an IPv6 header. AH is an integral part of IPv6 protocol and AH header is an extension header in IPv6. The AH header should appear after

hop-by-hop, routing, and fragmentation extension headers and the protected data should put after all extension headers. In transport and tunnel mode, the packet structure of AH is shown in Figure 5-4 and Figure 5-5.

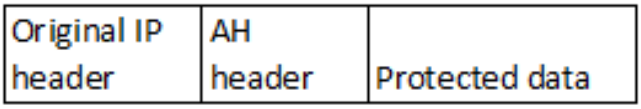


Figure 5-4 Packet Structure of AH Transport Mode

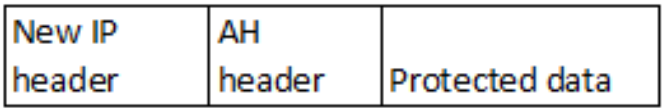


Figure 5-5 Packet Structure of AH Tunnel Mode

Security Association (SA) [11] is the shared security information between two network entities. There are 2 kinds of SAs: IKE (Internet Key Exchange) SA and Child SA. IKE SA is used to establish a secure authenticated communication channel by using Diffie-Hellman (D-H) [56] key exchange algorithm to generate a shared secret key to encrypt further IKE communications. Child SA includes ESP SA and AH SA and it is used to negotiate Security Associations on behalf of other services like ESP and AH. IKE SA should be established first and Child SA is based on it. Child SA defines the way to protect network traffic (encryption), the network or IP addresses to be protected and the lifetime of the protection. A Child SA also contains the state of an IPsec tunnel. A Child SA is unidirectional, which means that a Child SA only defines security services for one direction not the returning traffic. Typically, Child SAs exist in pairs, one in each direction, and they are negotiated when setting up IPsec tunnel. All SAs reside in the Security Association Database (SADB) which will be loaded in memory when starting up IPsec services. Each SA has a lifetime which will be negotiated between IPsec peers by the key management protocol [17].

Security Policy (SP) [11] defines the traffic to be protected by IPsec of a specific user. This allows for certain traffic to be protected by one kind of encryption while others can use different kinds of protection. An SP specifies the source and destination of the traffic to be

protected, the type of the traffic, the way to protect the traffic and with whom the protection is shared. Typically, SPs are set on the network security gateway and are maintained in the Security Policy Database (SPDB) which will be loaded in memory when starting up IPsec services. All IP traffic going through the network security gateway will be checked through SPDB. If the traffic matches an SP in SPDB, the network security gateway sends the traffic according to the SA specified by that SP. If the SP does not point to any existing SAs in the SADB, a new SA pair will be created before any traffic may pass [17].

Internet Key Exchange (IKE) [11] is the protocol used to authenticate and establish Security Association (SAs) using cryptographic keys. IKE was originally defined in 3 documents: Internet Security Association and Key Management Protocol (ISAKMP, RFC 2408) [29], The Internet IP Security Domain of Interpretation (DOI, RFC 2407) [61], and IKE (RFC 2409) [62]. IKE was updated to version two (IKEv2) [11] in December 2005. The new version obsoletes ISAKMP and DOI, and simplifies message exchange by merging two phases in IKE version 1.

IKEv2 communication only consists of pairs of messages: a request and a response. The pair is called an "exchange". Commonly, only two exchanges are used to setup SAs: IKE_SA_INIT and IKE_AUTH exchanges. The IKE_SA_INIT negotiates cryptographic algorithms (security parameters for the IKE SA), exchanges nonces (a random number), and does a Diffie-Hellman exchange [56]. The IKE_AUTH transmits identities, proves knowledge of the secrets (certificates) corresponding to the two identities, and establishes the first CHILD_SA (an ESP SA or an AH SA). Other types of exchanges are CREATE_CHILD_SA (an IKE SA can have many Child SAs) and INFORMATIONAL (witch deletes an SA, reports errors, or other maintenance work).

The certificates mentioned above can be distributed by Public Key Infrastructure (PKI) [55] or can be pre-shared keys. PKI is the system to distribute certificates. Typically, administrators are responsible for verifying the identities of users and granting users certificates. PKI is a complex system and it consists of Registration Authority (RA), Certificate Authority (CA) and repository of certificates. For more information about PKI, please refer to [55].

Detailed discussion about PKI is beyond the scope of this thesis. Pre-shared keys can also be used as certificates. Pre-shared keys can be easily deployed by copying the same pre-shared key to IPsec peers before establishing an IPsec tunnel, but they are a relatively weak authentication method. Using different kinds of certificates will not influence the performance of handling mobility problems in IPsec tunnel. In order to simplify the implementation of L2TP/IPSec in this thesis, it was decided to use pre-shared keys for certificates.

Cryptographic algorithms can be classified into two categories: Symmetric and asymmetric [54]. Symmetric cryptographic algorithms are a class of algorithms for cryptography that use the same cryptographic keys for both decryption and encryption, while asymmetric cryptographic algorithms use a different key for encryption than for decryption. In IPsec, key exchange (IKE) uses an asymmetric cryptographic algorithm (Diffie-Hellman Key Exchange) and encapsulation protocols (ESP and AH) use symmetric cryptographic algorithms [54].

Diffie-Hellman key exchange (D-H) is a cryptographic protocol that allows two parties to establish a shared secret key over an insecure communication channel. This key can then be used to encrypt subsequent communications using a symmetric algorithm. The D-H algorithm is shown in the example below: Alice and Tom agree to use a large prime number ‘p’ and base ‘g’; Alice chooses a secret integer ‘a’ and sends Tom the result of ‘g^a mode p’; Tom chooses a secret integer ‘b’ and sends Alice the result of ‘g^b mode p’; finally, Alice calculates the final shared secret key by ‘$(g^b \text{ mode } p)^a$ mode p’ and Tom calculates the same value by ‘$(g^a \text{ mode } p)^b$ mode p’. The algorithm is shown in the table below.

Alice				Tom	
Secret	Public	Value transferred to the peer	Value transferred to the peer	Public	Secret
	p, g			p, g	
a					b
		g^a mode p	→		
			←	g^b mode p	
	$(g^b \text{ mode } p)^a$ mode p			$(g^a \text{ mode } p)^b$ mode p	

Table 5-1 Diffie-Hellman algorithm

Different symmetric cryptographic algorithms can be used in encapsulation protocols (ESP and AH). The algorithms that must be implemented are: AES-CBC with 128-bit keys, Triple DES-CBC, NULL, and HMAC-SHA-96. For definitions of these algorithms, please refer to RFC 4835 [57].

The relationship between different IPsec modules is shown in Figure 5-6 [8, 30].

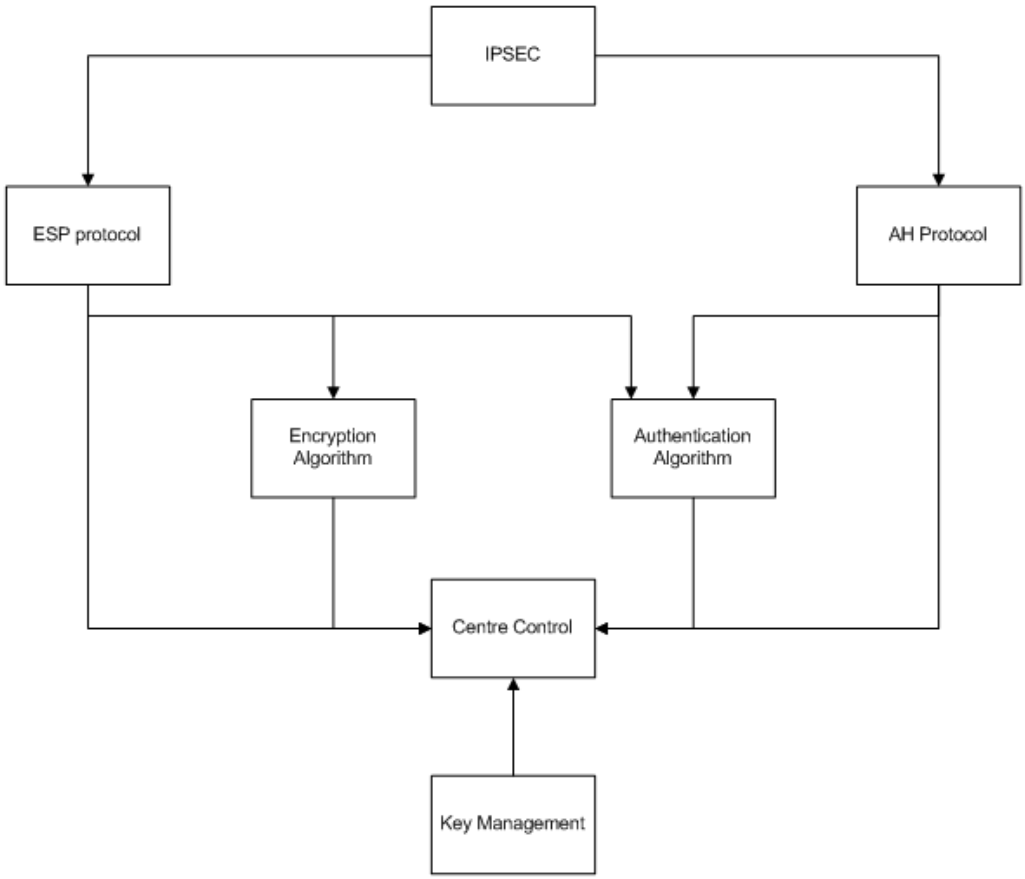


Figure 5-6 IPsec Module Architecture

5.2 IPSEC TUNNEL ESTABLISHMENT

Setting up an IPsec tunnel requires a relatively short period. The key steps are:

(1) Start up SPDB, SADB and IKE services.

(2) When the IPsec concentrator receives an IP packet, it checks the service type of the packet (port number), the source and the destination addresses of the IP packet. If these things match an entry in SPD, a new IKE SA and then a Child SA (ESP SA or

AH SA) will be negotiated via IKE according to the security policy. The new SA will be saved in SADB and a pointer to the new SA (Child SA) will be saved in SPD.

(3) Finally, all similar data packets can be sent inside IPsec tunnel.

5.3 SENDING DATA PACKETS

Similar to IPsec tunnel establishment, the IPsec concentrator checks every IP packet it received. If the packet matches an entry (SP) in SPD, the concentrator searches for the Child SA which is pointed to by the entry (SP). If a Child SA is found, the concentrator will encapsulate and encrypt the packet into an ESP or AH packet according to the Child SA, and sends the ESP or AH packet to the other end of the IPsec tunnel (Tunnel Server in Figure 5-1). The tunnel server fetches the SA from the SADB using destination, protocol (AH/ESP), and SPI of the packet. If an SA is found, the tunnel server decapsulates and decrypts the packet according to Security Parameter Index (SPI) in the ESP or AH header of the packet, checks the security policy according to SPD and sends the packet to the destination of the IP packets (Remote Server in Figure 5-1). A similar algorithm is used for the returning traffic.

In summary, in Figure 5-1, only the red line is the IPsec tunnel. The packets outside the IPsec tunnel are normal IP traffic, and the packets inside are ESP or AH packets. The concentrator and the tunnel server are the endpoints of the IPsec tunnel. They are responsible for encapsulating IP packets into ESP or AH packets, and decapsulating ESP or AH packets back to IP packets.

5.4 IPSEC IMPLEMENTATION

IPsec is a framework of open standards. There are IPsec implementations under Windows, Cisco IOS Software, Solaris, BSD operating system and Linux. This thesis will only analyse the implementations under FreeBSD.

FreeBSD is written in the C programming language. The implementation of IPsec under FreeBSD can mainly be divided into 2 parts: kernel (a part of the FreeBSD kernel) and user space (racoon2 project) [1, 7]. User space part is responsible for handling key exchange and

managing security policies. The kernel part is responsible for sending and receiving IP packets, encapsulating and decapsulating ESP or AH packets, and checking security policies. The architecture of the IPsec implementation is shown in Figure 5-7.

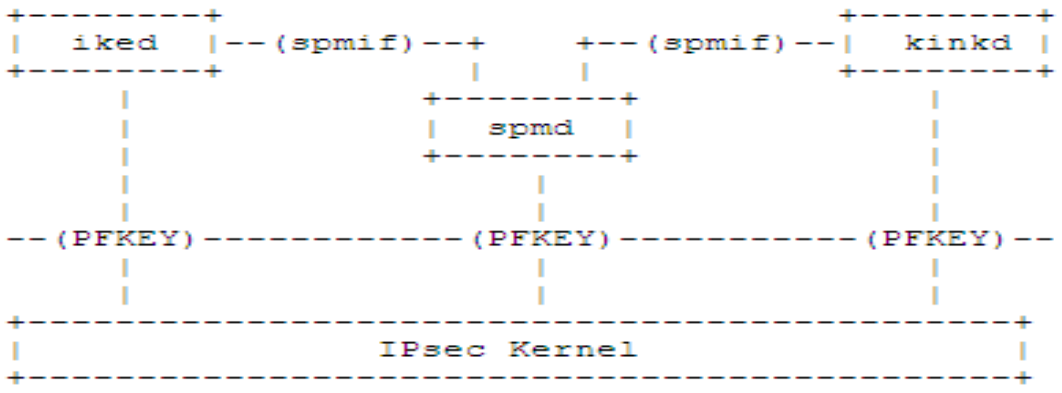


Figure 5-7 IPsec Implementation Architecture

User space part (racoon2) can be mainly divided into 3 components (iked, spmd and kinkd) and 2 main interfaces (spmif and PFKEY). Each component has to start in a separate process so that interfaces are used to communicate between different components (a process is an instance of a computer program, consisting of one or more threads, that is being executed by a computer that has the ability to run several computer programs concurrently).

Iked is responsible for handling key exchange using IKE protocol. Both IKE version 1 and version 2 are implemented by iked. In other words, 3 protocols are implemented to support IKE version 1: ISAKMP, DOI and IKE; IKEv2 are also implemented to support IKE version 2. Iked is only responsible for creating IKE packets and negotiation. The actual sending packets and cryptographic functions are handled by the FreeBSD kernel through system calls. In order to get or manage SA information, iked calls PFKEY interface to communicate with IPsec kernel. In order to view or change SP information, iked calls spmif interface to communicate with spmd.

Spmd manages the SPD and receives various requests from other racoon2 key management daemons. It reads security policies (SPs) from the user definition file, creates a temporary security policy database under user space and manages the actual SPD (inside IPsec kernel) via PFKEY interface. Other racoon2 key management daemons may make security

policy requests to spmd through spmif interface under user space. The actual SPD will be empty if spmd is not started so that spmd must be started before iked or any other IPsec components. The IPsec kernel will only check the actual SPD, not through spmd.

Kinkd [1] is similar to iked except that it processes the KINK (Kerberized Internet Negotiation of Keys) protocol [58]. KINK is not a popular protocol for handling key exchange. This thesis will not discuss the KINK protocol and the kinkd daemon.

Interface spmif defines a group of functions for security policy requests. The interface is created like an internal socket (socket is usually used to deliver data packets to other hosts on network. An internal socket is a local socket which does not go through Internet), but actually it is a file which is defined in racoon2 settings. Key management daemons like IKE can read or write to that interface, and spmd scans the interface regularly and handles these requests. If the security policy is changed according a request, spmd has to change the temporary security policy database first and then send a request to IPsec kernel through PFKEY interface.

Interface PFKEY defines a group of functions sending and receiving security policy requests and security association requests between user space and IPsec kernel. The PFKEY interface is also an internal socket. IPsec kernel listens to the PFKEY interface, updates the SPD or SADB accordingly, and sends response to that interface if necessary.

IPsec kernel is a part of the FreeBSD kernel. When starting up the FreeBSD operating system, the FreeBSD kernel will first be loaded and then libraries such as libc (the library for C programming language). Therefore only basic functions can be used under the FreeBSD kernel. All the source code of the kernel is under /usr/src/sys/ directory and the basic functions are declared in /usr/src/sys/sys/systm.h. The FreeBSD kernel is a complex system and the KAME project provides implementation for IPsec and IPv6 protocol stack under the FreeBSD kernel. This thesis will not explain the whole KAME project which is fully described in [7] and [59]. It will explain how IPsec works under the FreeBSD kernel.

The FreeBSD kernel is responsible for sending and receiving IP packets. Figure 5-8 illustrates an overview of the process flow of sending and receiving IP (IPv4) packets under the FreeBSD kernel. The main functions of handling IP packets under the FreeBSD kernel are

ip_input() and ip_output(). All IP packets are received by each network interface driver and passed to the ip_input() function. If the packet reaches its destination address, it will be passed to transport layer (or other upper layer) input functions according to packet type. If the packet should be sent to other interfaces or other network devices, it will be passed to the ip_forward() function and then to the ip_output() function. Transport layer (or other upper layer) output functions may also call the ip_output() function to send a packet. The ip_output() function searches for route of the packet and send the packet to Internet through network interface drivers.

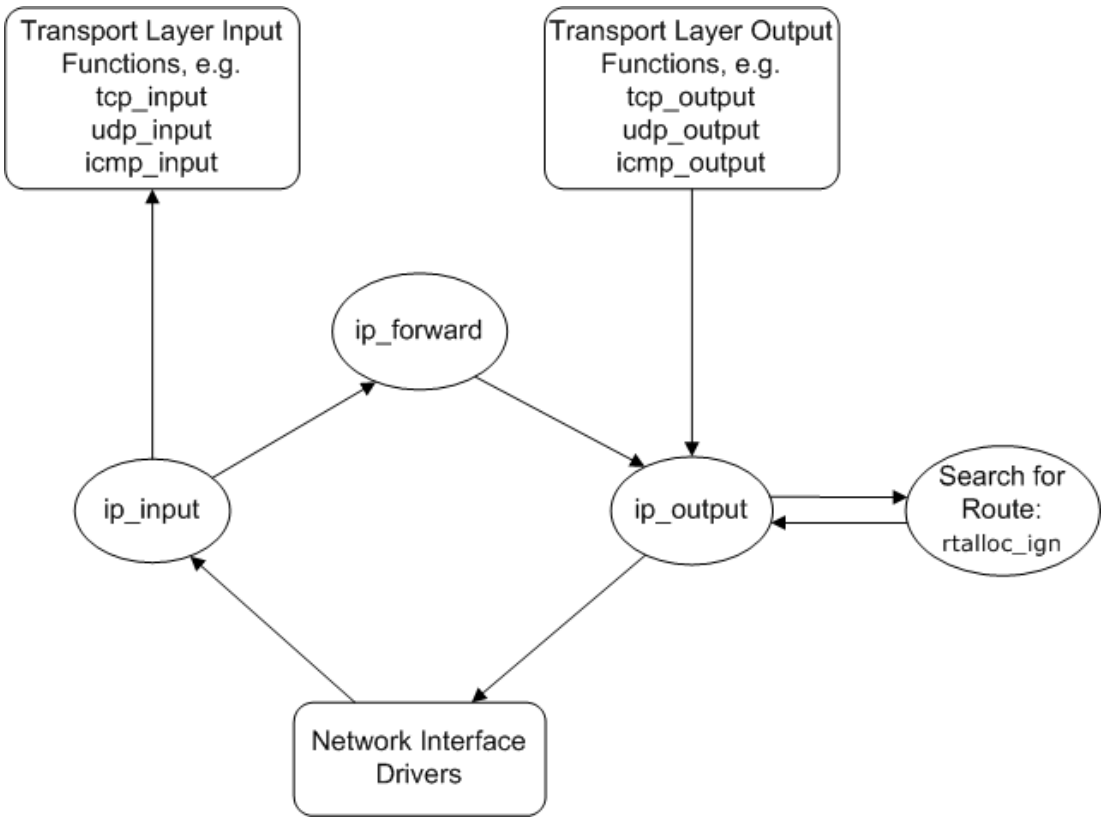


Figure 5-8 IPv4 Packet Input/Output Flow

By default, IPsec is not supported under the FreeBSD kernel. To add IPsec support to the FreeBSD kernel, add IPSEC option to your FreeBSD kernel configuration file (by default, the file is /usr/src/sys/i386/conf/GENERIC) and recompile the kernel. Managing IPsec packets is similar to managing IP packets. Some new functions will be inserted into the original framework. Figure 5-9 illustrates an overview of the process flow of managing IPsec packets under the FreeBSD kernel.

Similar to managing IP packets, all packets are received by each network interface driver and passed to the ip_input() function. Two kinds of packets needs IPsec handling in the ip_input() function which is shown below. Other packets are handled like normal IP packets.

(1) ESP/AH packets which reach an IPsec tunnel endpoint. If the packet reaches its destination address, the ip_ipsec_input() function will be called and then returned to check whether the packet is an ESP/AH packet with correct SA information according to SADB. If the ip_ipsec_input() function returns success, the ip_input() function forwards the packet to the ipsec_common_input() function. The ipsec_common_input() function decrypts and decapsulates the packet into an IP packet, and then call the ip_input() function using the IP packet. This ip_input() function call will also call the ip_ipsec_input() function and then call the ip_ipsec_fwd() function to check SP according to SPD. Finally, the IP packet will be forwarded through normal IP process (the IP packet goes through ip_forward() -> ip_output()).

(2) IP packets which should be encapsulated by ESP/AH and sent to the other IPsec tunnel endpoint. If the packet should be sent to other interfaces or other network devices, it will be passed through ip_ipsec_fwd() -> ip_forward() -> ip_output(). The ip_output() function will pass the IP packet to the ip_ipsec_output() function to check SP according to SPD. If a correct SP is found, the IP packet will be passed to the ipsec4_precess_packet() function. The ipsec4_precess_packet() function check whether there is valid SA related to the SP. If no SA is found, create a new SA via IKE (detailed information is already discussed in Section 5.2 and 5.3). Then, the ipsec4_precess_packet() function encrypts the IP packet into an ESP/AH packet and passes the ESP/AH packet to the ip_output() function. Finally, the ESP/AH packet can be sent to the Internet.

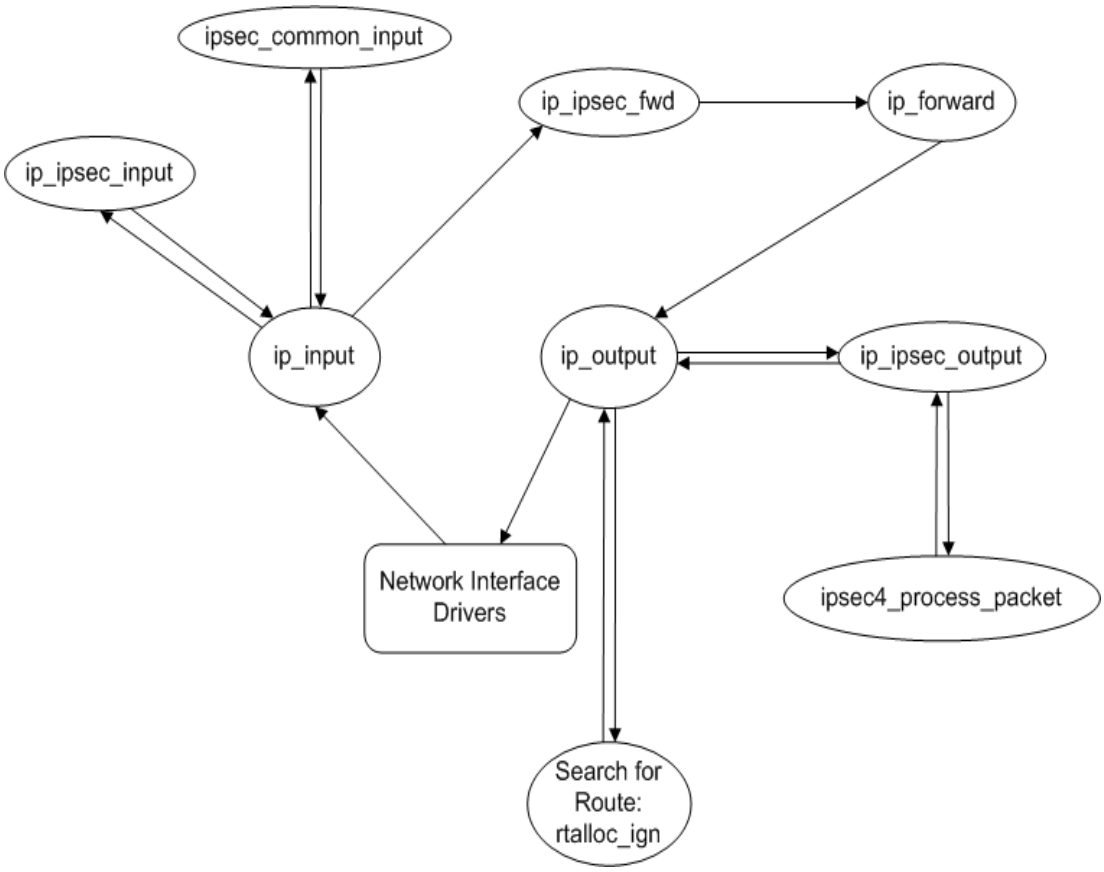


Figure 5-9 IPsec Packet Input/Output Flow

SADB and SPD are in the FreeBSD kernel, and are saved in memory. There is another module to handle PFKEY in the FreeBSD kernel.

5.5 COMPARISON BETWEEN WINDOWS IPSEC AND UNIX IPSEC

The IPsec service needs to be installed on FreeBSD (UNIX) by recompiling the FreeBSD kernel [6] and installing racoon2 [1]. In the Microsoft Windows operating system, however, all that is required comes with the operating system. The IPsec service can be started by configuring an IPsec Policy snap-in in the Microsoft Management Console (MMC) and creating a new connection via the Network Connection Wizard [60].

The topology of IPsec, which is shown in Figure 5-1, is also different for Windows and FreeBSD (UNIX). In Figure 5-1, computer A, B and concentrator can be on the same machine while in UNIX they must be on separate machines. In other words, Windows supports both

Client to Server VPN and Site to Site VPN, while FreeBSD (UNIX) only supports Site to Site VPN.

5.6 SUMMARY

This chapter first introduces the Internet Protocol Security (IPsec), its components and how it protected IP packets. Then, the implementation of IPsec under FreeBSD (both kernel and user space parts) is presented. Finally, this chapter discusses the differences between Microsoft Windows IPsec and UNIX IPsec.

Internet Protocol Security (IPsec) is a suite of protocols for securing IP communications by authenticating and encrypting each IP packet of a data stream. It is a network layer protocol and has been integrated into the IPv6 (next generation network) protocol. It can be divided into four components: encapsulation protocols (AH and ESP), security management (SP and SA), key management (IKE) and algorithms for cryptographic algorithm. Encapsulation protocols (AH and ESP) are the packet structures used to protect IP packets. Other components are used to manage security information and to establish the IPsec tunnel.

The typical topology of an IPsec tunnel is shown in Figure 5-1, and only the red line in the figure is protected by IPsec. In order to start up an IPsec tunnel, SPDB, SADB and IKE services (spmd and iked provides these services under FreeBSD) should be started first. When an IPsec concentrator receives an IP packet, it checks the packet according to SPD. If an SP matches the packet, a new SA will be negotiated during tunnel establishment via IKE according to the SP, or old SA is used after tunnel establishment. Then the IPsec concentrator encapsulates and encrypts the IP packet into an ESP (or AH) packet according to the SA, and sends the ESP (or AH) packet to the other end of IPsec tunnel. The IPsec tunnel server decapsulates and decrypts the ESP (or AH) packet into the original IP packet according to the SA, checks SP and forwards the IP packet to its final destination.

The implementation of IPsec under FreeBSD can mainly be divided into 2 parts: kernel and user space. User space part is responsible for handling key exchange and managing security

policies. The kernel part is responsible for sending and receiving IP packets, encapsulating and decapsulating ESP or AH packets, and checking security policies.

IPsec by itself does not have native ability to cope with mobile clients. An IPsec extension adds mobility support to IPsec, which is discussed in RFC 4555 [53]. However, that solution has some limitations which will be discussed in Section 8.4.

Chapter 6: L2TP/IPsec Interworking

L2TP does not provide strong authentication by itself, instead it often uses IPsec to secure packets by providing confidentiality, authentication and integrity.

This chapter will first introduce why L2TP/IPSec is necessary, how L2TP and IPsec work together, and then will discuss the details of the L2TP/IPsec tunnel establishment and authentication. Finally, this chapter will discuss the implementation of the L2TP/IPsec tunnel under FreeBSD (UNIX).

Advanced readers should also read this chapter carefully to get a better understanding of L2TP/IPsec tunnel and its implementation, so that they can understand the rationale and advantages of the new and novel L2TP/IPsec solution explained in later chapters.

6.1 OVERVIEW OF L2TP/IPSEC TUNNEL

Although IPsec can be used as an independent secure tunnel, it cannot tunnel layer 2 packets. Tunneling layer 2 packets is necessary when several persons inside a moving vehicle are connected to a network at layer 2 and they want to connect to the company network through VPN.

L2TP is used to tunnel Layer 2 packets (usually PPP packets) over public networks. However, it does not provide strong authentication by itself. Both the control and data packets of L2TP protocol are vulnerable to attack: snooping data packets, modifying both control and data packets and launching Denial of Service (DoS) attacks by terminating PPP connections or L2TP tunnels.

To address these threats, L2TP security protocols must provide authentication, integrity and reply protection for control and data packets, and confidentiality protection is desirable [24]. IPsec meets this category. IPsec provides per-packet authentication, confidentiality protection, and integrity and reply protection for network or upper layer packets. Both control and data packets of L2TP are UDP packets which can be protected by IPsec. Moreover, IPsec has

demonstrated strong security features in the last years and it has already been integrated into the next generation network (IPv6). That is why IPsec is often used to secure L2TP packets.

In order to protect L2TP packets (both control and data packets), they should be put inside the IPsec packets. Two protocols can be used to protect data in the IPsec protocol suite: ESP [12] and AH. Compared with ESP, AH does not provide confidentiality. In other words, attackers can easily see the content of the packets transferring over the public Internet even with AH protection. That is why ESP is usually used to protect L2TP packets.

Two modes under IPsec can be used to protect L2TP packets: transport mode and tunnel mode. The IPsec Transport mode only protect the payload of the IP packet (network layer or upper layer protocols of the packet), while tunnel mode protects the entire IP packet (both payload and IP header). In L2TP standard (RFC 3193) defined by Internet Engineering Task Force, the IPsec transport mode must be supported while the IPsec tunnel mode may be supported [24]. To simplify implementation, the IPsec tunnel mode is used in this thesis. Similar algorithms will be used in the IPsec transport mode.

Figure 6-1 shows the packet structure of L2TP/IPsec tunnel in IPsec tunnel mode.

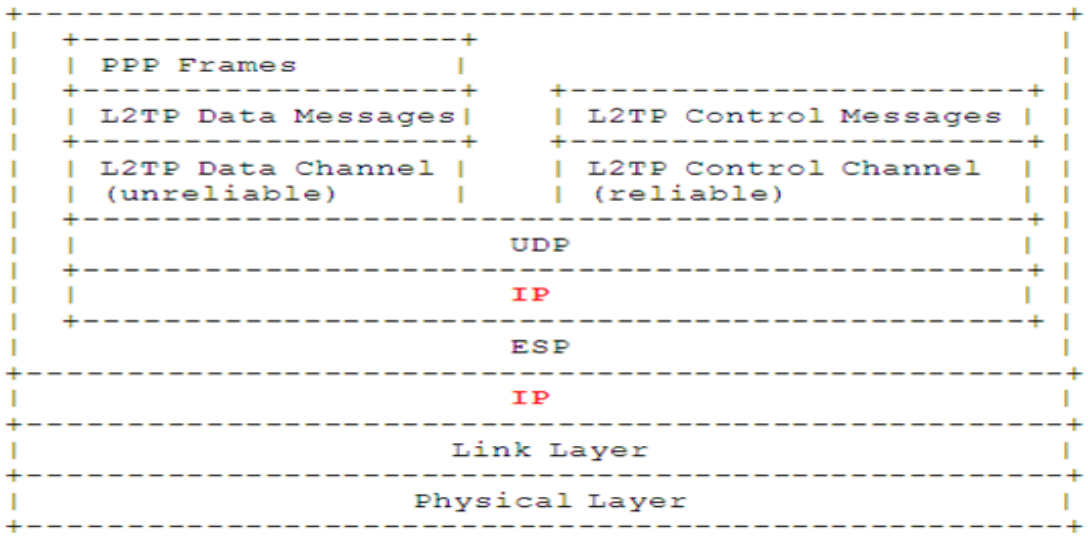


Figure 6-1 Packet Structure of L2TP/IPsec Tunnel in IPsec Tunnel Mode

6.2 L2TP/IPSEC TUNNEL ESTABLISHMENT

As IPsec protects L2TP in an L2TP/IPsec tunnel, IPsec should be negotiated first and then L2TP. The process of setting up an L2TP/IPsec VPN is as follows:

(1) Negotiate IPsec Security Association (SA, shared security information between two network entities) [11], typically through Internet Key Exchange (IKE). Digital certificates should be issued (via Certificate Authority) before this step.

(2) Negotiate and establish L2TP tunnel with the protection of ESP.

(3) Layer 2 packets can be sent via L2TP/IPsec tunnel.

6.3 L2TP/IPSEC TUNNEL AUTHENTICATION

In general, PPP provides initial authentication, but not per-packet authentication. The authentication in L2TP is optional and is usually not used, as IPsec provides per-packet authentication.

During the setup phase of an L2TP/IPsec tunnel, the authentications will be used in the sequence below:

(1) Negotiate IPsec Security Association (SA), typically through Internet Key Exchange (IKE).

(2) Do PPP initial authentication using PPP Challenge Handshake Authentication Protocol (CHAP) [32].

The setup phase is a relatively short period. The possibility of changing IP addresses in setup phase is very limited. Furthermore, there are several states during setup phase, which add complexity to mobility solutions. Also, it is hard to decide whether authentication failure is caused by changing IP addresses or some other issues. Due to the reasons above, we decided to add mobility support only in data transfer phase.

In data transfer phase (when a packet arrives at a tunnel), the following authentication will be used:

(1) Check Security Association (SA) information in IPsec according to the ESP header and the IP header. The IPsec component fetches and checks the SA from the

Security Association Database (SADB) using the destination address (end point of IPsec tunnel), protocol (ESP), and Security Parameter Index (SPI). The payload is then decrypted according to SA [17].

(2) Check the security policy of the packet by querying the Security Policy Database (SPD). The source and destination addresses in SPD will be checked and they are the addresses of end users (Point A and Point F in Figure 5). Other information will also be checked in SPD such as valid time, relevant SA information and upper layer protocols [17].

(3) Verify that the IP addresses and port values in the L2TP packet match the socket information which was used to setup the L2TP tunnel [24].

6.4 L2TP/IPSEC TUNNEL IMPLEMENTATION

An L2TP/IPsec tunnel can be easily configured under Windows. However, there is no native support for creating an L2TP/IPsec tunnel under FreeBSD. The L2TP and IPsec tunnel can be configured separately under FreeBSD, but not as a whole. This thesis will analyse the implementation of the L2TP and IPsec tunnel under FreeBSD, and will propose and compare some solutions to create an L2TP/IPsec tunnel.

The topology of an L2TP, or IPsec, or L2TP/IPsec tunnel is shown in Figure 6-2. Point B and Point E, which usually use private IP addresses, are called internal interfaces. Point C and Point D, which usually use public IP addresses, are called external interfaces.

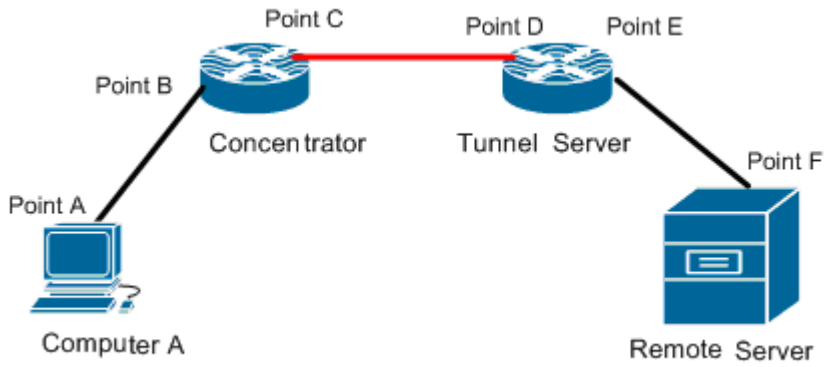


Figure 6-2 IPsec, or L2TP, or IPsec/L2TP Tunnel Topology

IPsec tunnel concentrator listens to IP packets at Point B, encapsulates and encrypts the IP packets into ESP packets and sends them through Point C. L2TP tunnel concentrator (LAC) listens to layer 2 packets at Point B, encapsulates packets into UDP packets and sends them through Point C. L2TP and IPsec concentrator listen on the same network interface. If they handle packets at the same time, only one of the concentrators (L2TP or IPsec) will eventually receive and handle the packets. Under FreeBSD, the L2TP concentrator will receive and handle packets from Point B first, as callback functions (the functions which are defined under user space but are used under the FreeBSD kernel) from L2TP will be called before IPsec functions in the ip_input() functions (see Figure 5-9).

Similar things happen when tunnel servers receive tunnel packets from Point D. There is also a confliction between the L2TP tunnel server (LNS) and the IPsec tunnel server. The L2TP tunnel server will receive and handle tunnel packets first under FreeBSD.

Therefore, if you simply configure L2TP and IPsec using the topology above, the whole system will work like an L2TP tunnel.

Upper and Lower Protocol Solution (ULP solution)

One solution [64] is to regard L2TP as an upper layer protocol and to regard IPsec as a lower layer protocol. For example, the input and output functions of TCP (upper layer protocol) are the tcp_input() function and the tcp_output() function; the input and output functions of IP (lower layer protocol) are the ip_input() function and the ip_output() function. The relationship of these functions is shown in Figure 5-8. However, this solution is not easy to implement, as each packet has to call both input and output functions of upper and lower layers. Security policy checking in IPsec is also a significant problem in this solution, as source and destination addresses are changed to the addresses of Point C and D after L2TP receives and handles a packet from Point B, and these packets will not pass the security policy checking and will not be encrypted. It would be extremely difficult to implement this solution under FreeBSD and so we rejected this solution path. However, under other platforms (such as Linux which uses Openswan [65] for IPsec and OpenL2TP [64] for L2TP), this solution is possible; as L2TP directly calls

IPsec functions. IPsec and L2TP listen to the same port number in an L2TP/IPsec tunnel under that platform [64].

Loopback Interface Solution

The solution we have developed is to add a loopback interface to both concentrator and tunnel server. A loopback interface is a virtual network interface implemented only in software. Any traffic that sends to the loopback interface is immediately received on the same interface. Adding a new interface to concentrator and tunnel server solves the confliction, as L2TP and IPsec can listen on different network interfaces. The topology is shown in Figure 6-3.

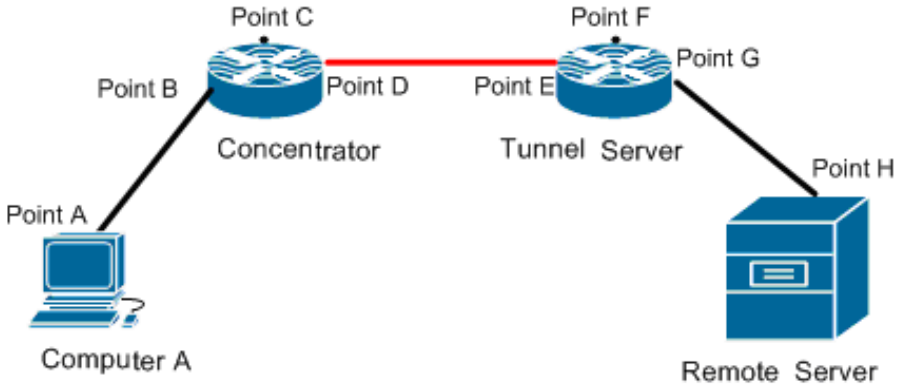


Figure 6-3 Loopback Interface Solution for IPsec/L2TP Tunnel

The L2TP concentrator receives a packet (source: Point A, destination: Point H) from Point B, encapsulates the packet and sends the new packet (source: Point C, destination: Point F) through Point C. The packet will go back to Point C immediately. The IPsec concentrator receives the packet from Point C, encapsulates and encrypts the packet and sends the new packet (source: Point D, destination: Point E) through Point D. The IPsec tunnel server receives the packet from Point E, decrypts and decapsulates the packet into the original packet (source: Point C, destination: Point F) and forwards the packet through Point F. The packet will go back to Point F immediately. The L2TP tunnel server receives the packet from Point F, decapsulates the packet into the original packet (source: Point A, destination: Point H) and forwards the packet to

Point H. Similar algorithm is used for the returning traffic. Table 6-1 shows some configuration of the loopback interface solution.

	Concentrator		Tunnel Server	
	Internel Interface	Externel Interface	Internel Interface	Externel Interface
IPsec	Point C	Point D	Point F	Point E
L2TP	Point B	Point C	Point G	Point F

Table 6-1 Configuration of Loopback Interface Solution

This solution is simple and stable, because a loopback interface provides a stable address to minimize impact of a physical interface going down. The approach taken in this solution could be used on any POSIX (a software interface to different operating systems) [69] operating system.

A routing loop [80] is a common problem with computer networks. It will cause the traffic to be forwarded into an endless loop. It is formed when an error occurs in the operation of the routing algorithm. Loopback interfaces are similar to physical network interfaces. This loopback interface solution does not add any new routing loops to public networks. Routing loops can be avoided by correctly configuring routing protocols such as OSPF [81].

This thesis uses this solution to setup an L2TP/IPsec tunnel.

6.5 SUMMARY

This chapter first introduces why L2TP/IPSec is necessary and how L2TP and IPsec work together, and then discusses the details of the L2TP/IPsec tunnel establishment and authentication. Finally, this chapter discusses the implementation of the L2TP/IPsec tunnel under FreeBSD (UNIX).

Although IPsec is a VPN protocol with strong security, it cannot tunnel layer 2 packets. L2TP tunnels layer 2 packets, but does not provide strong authentication by itself. It is often used with IPsec to provide a secure layer 2 tunnel. ESP (a part of IPsec) is usually used to protect L2TP packets.

During L2TP/IPsec tunnel establishment, IPsec is negotiated first, and then PPP and L2TP. During data transfer phase, SA, SP and L2TP information will be checked.

There is no native support for creating an L2TP/IPsec tunnel under FreeBSD. The L2TP and IPsec tunnel can be configured separately under FreeBSD, but not as a whole. This thesis proposes a loopback solution for this problem.

Chapter 7: Add Mobility Support to L2TP/IPsec Tunnel

L2TP/IPsec is a secure VPN which tunnels Layer 2 packets. In the previous chapter, this protocol has been discussed and a solution has been proposed to create an L2TP/IPsec tunnel under FreeBSD.

This chapter shows how mobility support is added to the L2TP/IPsec tunnel. The new solution tunnels Layer 2 packets without incurring tunnel-re-establishment at handoff, without losing packets during handoff, achieves better security than current mobility solutions for VPN, and supports fast handoff in IPv4 networks. This chapter is organized as follows:

First, a general framework of the mobility solution is introduced, followed by details of the eight modifications required to achieve a fully working system. Finally, the analysis and discussion on the security and the performance of the new solution are provided.

7.1 SOLUTION OVERVIEW

First, the real world problem will be analysed with the topology as shown in Figure 7-1. Consider several persons work together using a PPP network inside a moving vehicle, a L2TP/IPsec concentrator is used to encapsulate PPP packets and to tunnel to the remote server. When the vehicle moves, the public IP address of the L2TP/IPsec concentrator (Point D in Figure 7-1) is changed. Point C is a loopback interface, and its IP address does not change. Other IP addresses inside the vehicle (Point A and B) are private IP addresses, which do not change even if persons move inside the vehicle. The IP addresses outside the vehicle (Point E, F, G and H) do not change, as they are fixed IP addresses. In summary, only the IP address of Point D is changed when the vehicle moves.

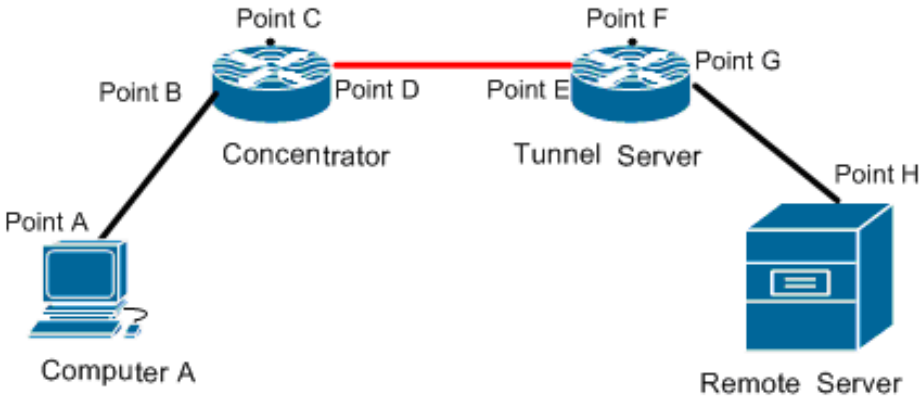


Figure 7-1 Loopback Interface Solution for IPsec/L2TP Tunnel

The goal of this thesis is to add mobility support to existing L2TP/IPsec tunnel so that the tunnel can keep sending packets with low delay, without the cost of re-establishing the tunnel and without losing packets.

The proposed solution is to let the tunnel concentrator communicate with tunnel server directly, without any home agent or foreign agent (like Mobile IP), without involving new headers. ESP is used to protect L2TP packets and to update IPsec information and the reasons will be explained in Section 7.2.

In order to increase the performance, the new solution only modifies the kernel part of IPsec because all the IPsec information for L2TP/IPsec mobility support (ESP, SPD and SADB) is inside the IPsec kernel. In order to reduce the communication between kernel and user space, and make debugging easier, most of the modifications are made to user space code of L2TP. This is because most L2TP tunnel information is saved under user space.

The detailed procedure is described below.

(1) The tunnel concentrator updates IPsec tunnel information and tells the tunnel server directly that its IP address is changed in IPsec from the old IP address to the new IP address.

(2) The tunnel server confirms the IP change, updates IPsec tunnel information and sends back a confirming message.

(3) The tunnel concentrator tells the tunnel server directly, with encryption, that the sequence number of the last data message received. In the ULP solution (see Section 6.4 for details), the IP change information under L2TP should also be sent in that message.

(4) The tunnel server decrypts and verifies the packet. If the packet verification is successful, the tunnel server updates the tunnel information in L2TP, and sends a reply message indicating whether tunnel server has buffered enough packets and the sequence number of the last data message received.

(5) The tunnel concentrator decrypts and verifies the reply packet. If either tunnel server or tunnel concentrator buffers not enough packets, the tunnel concentrator cuts off the user connection. If buffered enough packets, the tunnel concentrator updates the tunnel information in L2TP, sends the missing packets which do not reach tunnel server and asks the tunnel server to send the missing packets which do not reach tunnel concentrator.

(6) The tunnel server sends the missing packets. Finally the VPN can resume operation with new data packets.

There are eight modifications to implement the mobility support. These will be shown in the remainder of this chapter.

7.2 UPDATING IP CHANGE INFORMATION IN IPSEC

For security reasons, IP change information must be encrypted and transferred to the other peer. It is best to add the information to ESP payload, not ISAKMP (Internet Security Association and Key Management Protocol) [29] or IKE (Internet Key Exchange). The reasons are as follows:

(1) Avoid SA reestablishment. SA modification within ISAKMP is accomplished by creating a new SA and deleting the old SA at any time after the new SA is established [29]. Sending information within the ESP payload can modify existing SAs instead of creating new ones.

(2) Reduce the communication between kernel and user space under UNIX. ISAKMP and IKE messages are handled in user space (see Section 5.4) while ESP and SA database are under UNIX kernel. Sending information within ISAKMP and updating SA database (SA database must be changed when handling IP change) needs communication between UNIX kernel and user space. Sending information within ESP and modifying SA database can eliminate the communication between kernel and user space.

(3) ISAKMP is already obsolete in IKEv2.

ESP packets are used to transfer IPsec address change information and two private NOTIFY message types (250 and 251) are written into the Next Header field in the ESP header to signal this activity. Other information, such as SPI, in the ESP header is the same as normal ESP headers in the old IPsec tunnel. In the IPSEC-ADDRESS-UPDATE (message type 250) packet, the ESP payload contains the IP addresses of the end points of the IPsec tunnel when setting up the tunnel and the current addresses. In the IPSEC-ADDRESS-REPLY (message type 251) packet, the ESP payload contains the same information as in the IPSEC-ADDRESS-UPDATE message and the action taken at VPN gateway, i.e. ADDRESS-UPDATE-SUCCEEDED. The encryption method has not changed in these ESP packets.

7.3 UPDATING IP CHANGE AND DATA SEQUENCE NUMBER INFORMATION IN L2TP

Different L2TP information should be exchanged in different L2TP/IPsec tunnel implementations. In the loopback interface solution (see Section 6.4 for details), the IP addresses of the L2TP tunnel (Point B, C, F and G in Figure 7-1) are not changed. Therefore, only the sequence number of the last data message received should be exchanged for re-transmitting lost packets. In the ULP solution (see Section 6.4 for details), the external IP address of the L2TP tunnel concentrator (Point C in Figure 7-2) is changed. Therefore, both IP change information and the sequence number of the last data message received should be exchanged. The port

number of L2TP tunnel is not changed in our situation. If the port number is changed then L2TP tunnel should be re-established.

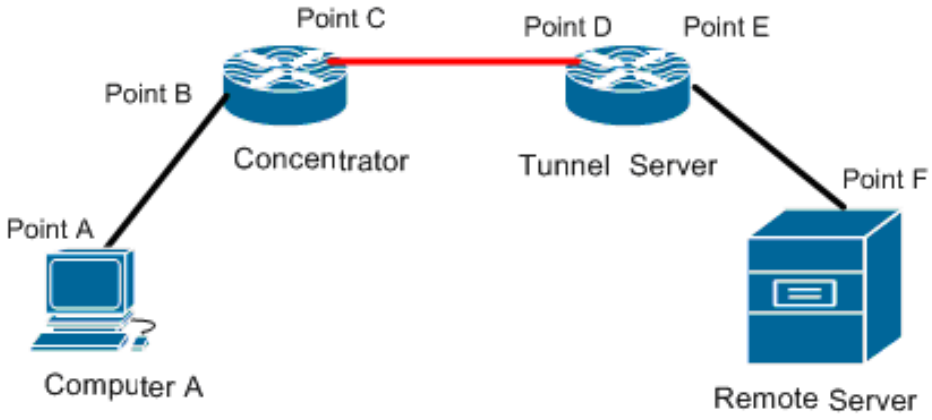


Figure 7-2 IPsec/L2TP Tunnel Topology in ULP solution

The L2TP packet is situated inside the IPsec packet (see Section 6.1). IP change information is updated in IPsec first and then in L2TP. L2TP is responsible for flow control of the VPN tunnel.

In the ULP solution, an L2TP control packet with four AVPs (Attribute-Value Pair) is used for the notification message: L2TP-ADDRESS-UPDATE (Message Type 40), OLD-ENDPOINT-ADDRESS (Attribute Type 42), NEW-ENDPOINT-ADDRESS (Attribute Type 43), and LAST-DATA-NUMBER (Attribute Type 44). In the loopback interface solution, an L2TP control packet with two AVPs is used for the notification message: L2TP-SEQUENCE-UPDATE (Message Type 45) and LAST-DATA-NUMBER (Attribute Type 44).

In the ULP solution, an L2TP control packet is used with five AVPs (Attribute-Value Pair) for the reply message: L2TP-ADDRESS-UPDATE-REPLY (Message Type 46), OLD-ENDPOINT-ADDRESS (Attribute Type 42), NEW-ENDPOINT-ADDRESS (Attribute Type 43), ENOUGH-DATA-BUFFERED (Attribute Type 47) and LAST-DATA-NUMBER (Attribute Type 44). In the loopback interface solution, an L2TP control packet with three AVPs is used for the notification message: L2TP-SEQUENCE-REPLY (Message Type 48),

ENOUGH-DATA-BUFFERED (Attribute Type 47) and LAST-DATA-NUMBER (Attribute Type 44).

7.4 SAVING IP CHANGE INFORMATION IN L2TP AND IPSEC

Both L2TP and IPsec information are saved in a list inside the memory (not a separate database). L2TP information is saved under user space while IPsec information is saved in the FreeBSD kernel.

The IP addresses of the end points of IPsec tunnel when setting up the tunnel and the current addresses (mobile address) are saved in SA database in the IPsec kernel. Similar addresses are saved in the L2TP authentication database in the ULP solution. No modification is needed in the L2TP authentication database in the loopback interface solution.

All the packets will be sent according to the new IP addresses. The old IP addresses are only used to identify the original connection.

7.5 BUFFERING LOST PACKETS

In the L2TP/IPsec tunnel, L2TP is responsible for tunnel and flow control, while IPsec is responsible for security only. The proposed solution requires a buffer function to be added to L2TP. Transmission Control Protocol (TCP) [25] is a good example for buffering and flow control and it is used as a reference for buffering packets in L2TP.

The proposed solution has added a fixed size circular buffer to the L2TP tunnel. TCP requires the receiver to respond with an acknowledgment message as it receives the data. However, packet acknowledgment is not used in an L2TP tunnel because wireless network resources are very limited and packet acknowledgment requires too much network traffic (a response message has to be sent for every data message received) [25].

The size of the circular buffer must be negotiated in the setup phase of the L2TP tunnel. The size has been placed in the “Framing Capabilities” AVP in SCCRP (Start-Control-Connection-Reply) and SCCRQ (Start-Control-Connection-Request) messages. The “Framing Capabilities” AVP is reserved for future use in L2TP specification [28].

A sequence number must be added to the L2TP data message to support lost packet transfer (retransfer packets that could not reach the old IP address). Sequence numbers are only optional in L2TP data message as described in L2TP specification [28].

Both the tunnel server and the tunnel concentrator buffer packets they sent, and save the sequence number of the last message they received. Therefore, decision of whether buffered enough packets can be made according to the following formula:

The sent buffer contains the packet with the sequence number: (the sequence number of the last message they received of the peer+1). If the sequence number of the last message they received of the peer equals to max sequence number size, the sent buffer should contains the packet with the sequence number 0.

7.6 SOLVING AUTHENTICATION PROBLEMS

The proposed solution could not pass the old authentication of the L2TP/IPsec tunnel as described in Section 6.3. The challenge here is to devise a new security system without adding new security vulnerabilities

There are two databases for security information in IPsec: SPD (security policy database) and SADB (security association database). SPD contains user information, such as the IP address of end users (addresses of Point A and F in Figure 7-2, or addresses of Point A and H in Figure 7-1). SADB contains connection and encryption information, such as the end points of IPsec tunnel (addresses of Point C and D in Figure 7-2, or addresses of Point D and E in Figure 7-1). The proposed solution only modifies the SA information (the IP address of Point C in Figure 7-2, or the IP address of Point D in Figure 7-1).

The following modifications have been made to IPsec authentication:

(1) When creating an SA (security association) in SADB, save the tunnel end point addresses as initial tunnel end point addresses and also as current addresses (mobile addresses).

(2) When an IPsec server or concentrator sends IPsec packets, the packets are sent to the mobile addresses of the tunnel. Initial tunnel end point addresses are only checked in IPsec update messages.

(3) When an IPsec server or concentrator receives IPsec packets, the packets are checked according to the mobile addresses of the tunnel.

(4) Before an IPsec server or concentrator sends the IPSEC-ADDRESS-UPDATE packets, the mobile addresses of the SA in SADB are updated. Otherwise, the IPSEC-ADDRESS-UPDATE packets may be sent from old tunnel end point addresses.

(5) When an IPsec server or concentrator receives an IPSEC-ADDRESS-UPDATE packet, the packet is decrypted according to SPI (normal IPsec packets are also decrypted according to SPI (Security Parameter Index)) and then is checked according to the content of the packet. If the packet is an IPSEC-ADDRESS-UPDATE packet and SA can be found according to initial tunnel end point addresses, update mobile addresses of the SA in SADB.

(6) SPD check does not change because the addresses in SPD are the addresses of end users (Point A and Point F in Figure 7-2, or Point A and Point H in Figure 7-1) and these addresses do not change in our solution.

No modification of L2TP authentication is needed in the loopback interface solution. The following modifications were made to L2TP authentication in the ULP solution:

(1) When creating an L2TP tunnel, save the tunnel end point addresses as initial tunnel end point addresses and also as current addresses (mobile addresses) in the L2TP connection list.

(2) For normal L2TP packets, verify that the IP addresses and port values in the L2TP packet match the saved mobile node addresses and port numbers.

(3) For L2TP address update messages, check the initial tunnel end points addresses and update mobile addresses in the L2TP connection list.

7.7 UPDATING ROUTE INFORMATION

The routing table of the tunnel concentrator should be updated before the IPsec update message is sent. The routing table of the tunnel server should be updated after receiving the IPsec update message. If a network interface changes its IP address, all routes related to that interface will automatically be deleted under FreeBSD. A related route has to be added when an IP address is changed.

However, updating route information may not be necessary in an L2TP/IPsec mobility solution. It depends on how the tunnel concentrator gets a new IP address. In the simulation, which will be discussed in the next chapter, the new IP address is got by using the "ifconfig" command. In this situation, all routes related to the IPsec external interface (Point D in Figure 7-1) will be deleted and a new route to the tunnel concentrator has to be added by the L2TP/IPsec mobility solution. In experimentation on real devices, the new IP address is got from a DHCP server [66]. DHCP servers usually assign an IP address and a default route to DHCP client. The default route will route all packets from DHCP client (the tunnel concentrator in the solution) to DHCP server when no other route exists for a given IP packet's destination address. In this case, tunnel packets will be routed to DHCP server and then to the tunnel server. Then, no route information needs to be updated in the experiment on real devices.

7.8 DETECTING IP CHANGE

Detecting IP change is a necessary part of any mobility support. This change can be detected by scanning the interface address periodically or by a system event which shows a possible IP change on the interface. As mentioned in Section 7.7, FreeBSD will delete related routes when an interface changes its IP address. This event triggers the new system to detect the IP change and then to update the L2TP/IPsec tunnel.

Another important note is that the new IP address must be a valid IP address. Otherwise, the L2TP/IPsec tunnel will become invalid on real network devices.

Getting new IP addresses is not a part of this solution, but it is a necessary part of VPN handoff. Other software or systems should be used to get a new IP address when current Internet connection is lost. This will be discussed in Section 8.2, 9.2 and 9.4.

During handoff, tunnel packets will keep sending without reaching their destination. These packets will be buffered in L2TP side. When a new IP address is available, the novel solution will detect the IP change and handle the handoff.

7.9 SOLVING SYNCHRONIZATION PROBLEMS

ESP and SADB are two entities which work together under the FreeBSD kernel. SA information in SADB should not change when handling ESP packets. When the tunnel concentrator changes its IP address and also updates SA information, it is possible to build an ESP packet with its old IP address in the ipsec4_precess_packet() function and send the packet to its new IP address in the ip_output() function (Section 5.4 describes how IPsec processes packets). This synchronization problem occurs when the mobile node changes networks and if not properly handled may cause problems with IPsec or may cause an Intrusion Detection System to suspect that hacking is happening.

A flag is added to the FreeBSD kernel to solve this problem. When the tunnel concentrator wants to update the SA information, it enables the flag so that all the tunnel traffic, except the update message itself, will be dropped in the ipsec4_precess_packet() function. Lost packets are already buffered in L2TP. The tunnel concentrator then updates the SA information and sends the IPsec tunnel update message. After the tunnel concentrator receives the confirmation packet, the flag is disabled so that tunnel traffic will be encrypted and passed on.

7.10 SECURITY CONSIDERATIONS

On consideration, it appears that the new solution does not introduce any new security issues to the L2TP/IPsec tunnel as every packet is protected by IPsec during handoff. The solution adds IP change information to the ESP payload which is encrypted and could not be seen by other online users. These packets may be captured by someone to attempt a replay attack,

however the sequence number defined in the ESP header will cause any replay to be ignored. The L2TP update messages are also protected by ESP and again sequence number checks on ESP packets stop any replay attack from being successful. The vulnerability of the solution to other attacks, such as Denial-of-Service (DoS) attacks, should be the same as the vulnerability in regular IPsec operations.

7.11 PERFORMANCE IMPLICATIONS

Here, the performance of the new approach is compared with that of "IPsec over Mobile IP" [21] first, then "mobility support for IPsec" [19], and then "IKEv2 Mobility and Multihoming Protocol" [53]. A detailed analysis will be shown in Section 8.4. L2TP is used to encapsulate Layer 2 packets and to provide Layer 2 tunneling functions [8, 28]. L2TP traffic will not take into account in this comparison as other solutions do not provide Layer 2 tunneling function.

In the new solution, the traffic involved in the initial tunnel setup process is identical to the traffic of standard L2TP/IPsec tunnel. It is assumed that the mobile node does not handoff in the setup phase, a relatively uncommon occurrence (see Section 6.3). IPsec over Mobile IP solution is also able to handle the handoff but it needs to perform Mobile IP registration with UDP packets (at least 48 bytes) and typically adds one additional roundtrip for Mobile IP registration in setup. In IPv4, the negotiation between Home Agent and Foreign Agent is also needed.

In the process of transferring data, IPsec over Mobile IP solution has one more Mobile IP header (at least 26 bytes in IPv4 [31] and 6 bytes in IPv6 [17]) compared to the new approach. This traffic overhead is large especially when sending small packets. Also, the agent discovery packets in Mobile IP should be sent at least once a second [23]. This adds up to the traffic overhead particularly in a wireless system.

In the process of handoff, IPsec over Mobile IP has to use one roundtrip for Mobile IP update and six roundtrips for IPsec update [11]. However, the new solution uses only 1 round trip for IPsec information update.

In summary, the new solution has a smaller overhead and faster handoff than IPsec over Mobile IP by avoiding double tunneling and connection reestablishment. The above advantage will greatly increase the efficiency and performance especially in resource-constrained wireless networks where mobile nodes move frequently between networks.

Compared with "mobility support for IPsec" [19] and "IKEv2 Mobility and Multihoming Protocol" [53], the new method has better performance. ESP packets are used to update IPsec information instead of ISAKMP or IKE packets. The new method avoids SA reestablishment and reduces the communication between kernel and user space (see Section 7.2).

7.12 SUMMARY

A new and novel mobility support for the L2TP/IPsec tunnel has been proposed in this chapter. The new solution tunnels Layer 2 packets without incurring tunnel-re-establishment at handoff, without losing packets during handoff, achieves better security than current mobility solutions for VPN, and supports fast handoff in IPv4 networks.

The new solution is to let the tunnel concentrator communicate with the tunnel server directly, without any home agent or foreign agent, without involving new headers. The communication is protected by ESP which is strong in security.

The implementation of this solution will have better performance than the alternatives which will be discussed in Section 8.4. It only modifies the kernel part of IPsec, and most of the modifications under L2TP are made to user space code.

Compared to other IPsec mobility solutions, the new solution has better performance and can tunnel Layer 2 packets while other solutions cannot. No packet loss on network transfer is another advantage that other solutions do not have.

Chapter 8: Experiment on Simulator

In the previous chapter, the theory of a novel mobility support for the L2TP/IPsec tunnel has been proposed. The mobility support has good handoff performance, without losing packets during handoff, achieves better security than current mobility solutions for VPN.

In this chapter, the theory is proved by a network simulation in a VMware Workstation environment (VMware Workstation is virtual machine software that will be fully explained in Section 8.1) [67, 68]. Four copies of FreeBSD 7.0 [6] and one Windows XP were installed as virtual machines in VMware on a single computer. The networks between different virtual machines were tested for the L2TP/IPsec handoff time. The test results prove that the proposed mobility support works as intended and has excellent handoff performance. This chapter is organized as follows:

First, VMware Workstation software, the software used in the network simulation, is introduced, followed by an overview of setting up the test environment. Next, the experimental results are shown and analysed. Finally, mobility supports from other people are introduced and a comparison is made between their solution and the solution proposed in this thesis.

8.1 OVERVIEW OF VMWARE WORKSTATION

VMware Workstation [67, 68] is software which allows running multiple virtual computers inside a host computer. Each computer, inside the host machine, is called a virtual machine. Each virtual machine runs a separate operating system such as Microsoft Windows or Linux and these multiple virtual machines can be run at the same time.

On the surface, VMware is similar to an emulator (a machine where the processor is simulated in software) and a screenshot of VMware virtual machines is shown in Figure 8-1. However, there is a crucial difference between a VMware virtual machine and an emulator. While an emulator intercepts and interprets every last instruction using its own mechanisms, a

virtual machine passes most of the work down to the original operating system and the real computing machine sitting beneath it [68].

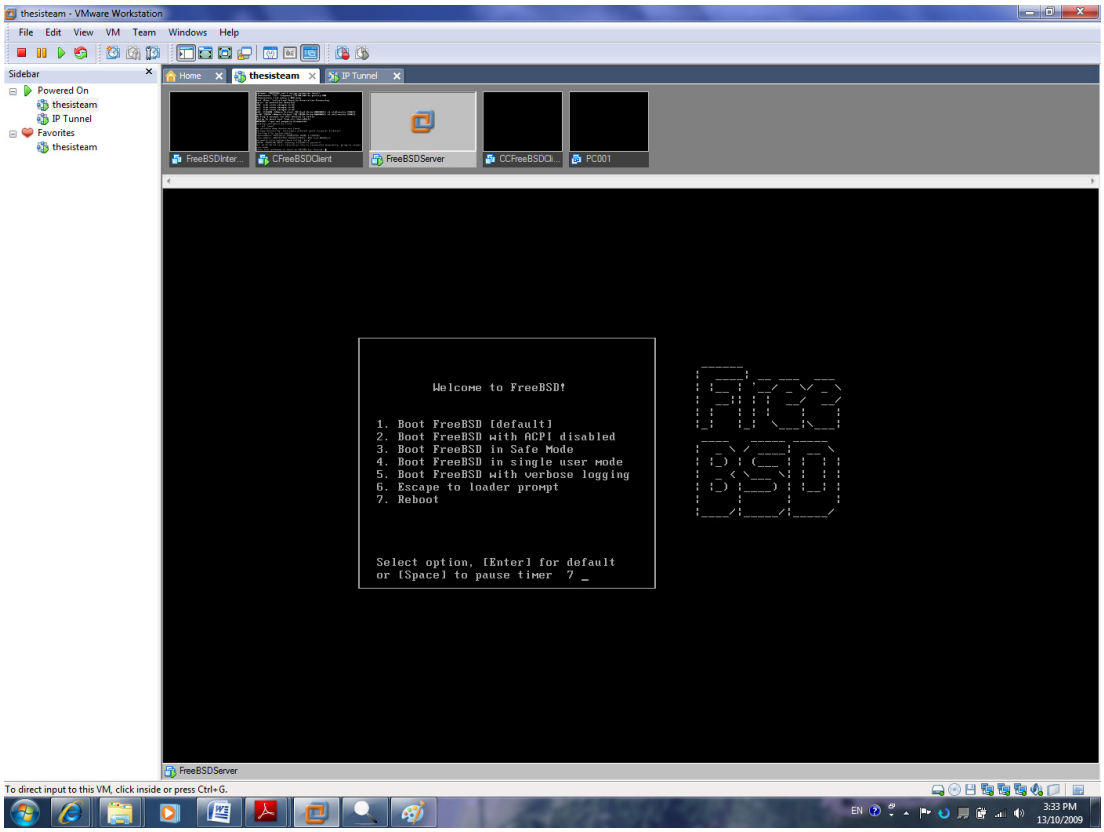


Figure 8-1 Screenshot of Virtual Machines

After the VMware Workstation software is installed, different virtual machines can be created on a host machine. This can be done by using the new virtual machine wizard to configure the size of virtual hard disks and to install the operating system. Other hardware configurations, such as Ethernet adapters and CD-ROM disks, can be set after the operating system is installed and when the virtual machine is turned off. Figure 8-2 shows how to change the hardware settings. After all the steps mentioned above, the virtual machine can be used like a real, physical PC.

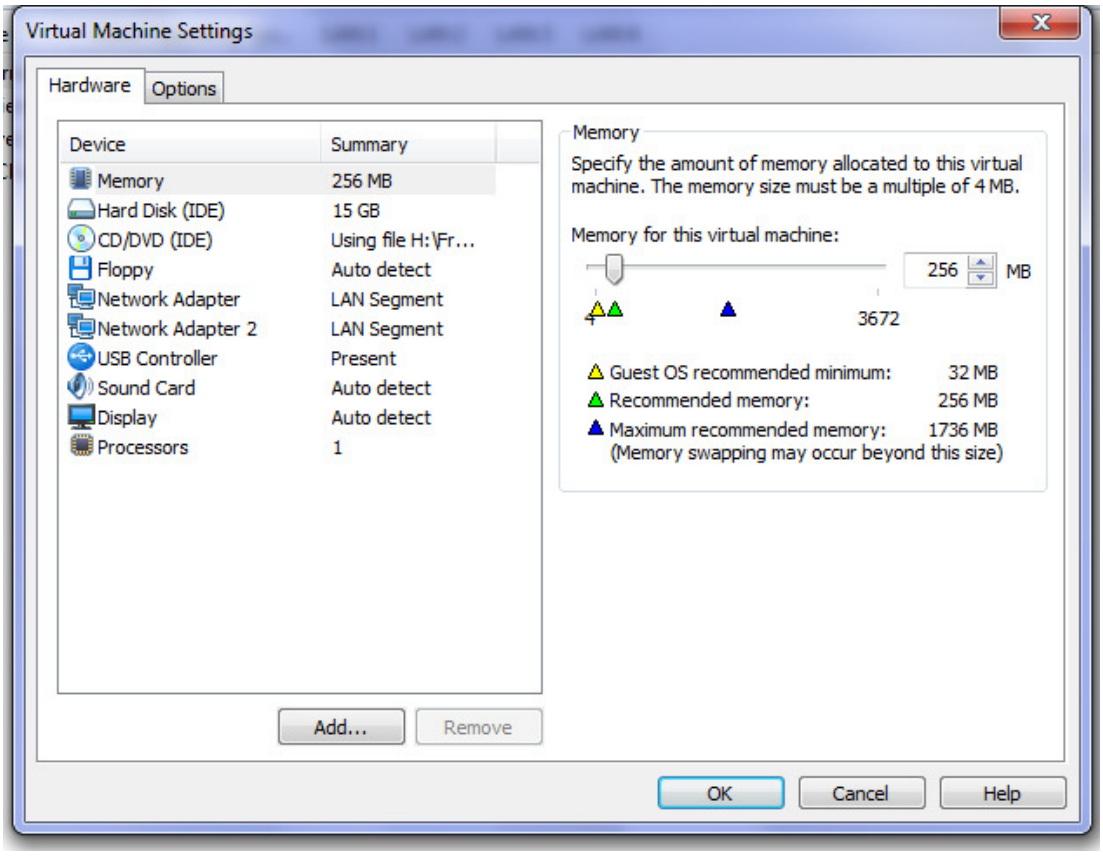


Figure 8-2 Change Hardware Settings in VMware

Next, the network system of VMware will be introduced. The type of network interface card configuration determines the networking relationship among different virtual machines and the host machine. There are four types of network interface card configuration: bridged, host-only, NAT and LAN networking.

A bridged network interface of a virtual machine is on the same level as your host machine's network. In other words, the virtual machine can obtain an IP address on the same subnet as the host machine and can access public networks through the router to which the host machine connects. The host machine and the virtual machines must be on the same network in this case. Setting up an L2TP/IPsec tunnel cannot use a bridged network, as an L2TP/IPsec tunnel has to run on different networks.

Host-only networks create an entire virtual Ethernet local area network inside the host machine. In other words, virtual machines under a host-only network can communicate with each other, but there is no native ability to communicate between virtual machines and the host

machine (or outside networks). VMware tools can be used to transfer files between the host machine and virtual machines. L2TP/IPsec tunnels can be used under host-only networks.

NAT networking is similar to host-only networking, except that VMware uses network address translation (NAT) [52] to communicate between virtual machines and the outside networks. NAT networking is not necessary in an L2TP/IPsec tunnel and will not be used.

LAN networking is the advanced version of host-only networking. LAN networking enables the communication between virtual machines, while disabling the communication between virtual machines and the host machine. Network interfaces within the same LAN can communicate with each other, while network interfaces from different LANs cannot. LAN networking is used in the L2TP/IPsec tunnel simulation, as different networks are used in an L2TP/IPsec tunnel.

The "team" feature of VMware is a relatively new and important feature of VMware and has been available since version 5 [67] (the L2TP/IPsec simulation is done in version 6.5). A whole virtual lab can be configured within a team, so that all the traffic from other virtual labs or virtual machines is blocked by VMware. Furthermore, LAN networking can be configured in a team. It makes the communication between virtual machines much easier. Each virtual machine can have many LAN network interfaces and each LAN network interface can communicate with network interfaces under the same LAN. Figure 8-3 shows the team and LAN configuration of the L2TP/IPsec tunnel simulation.

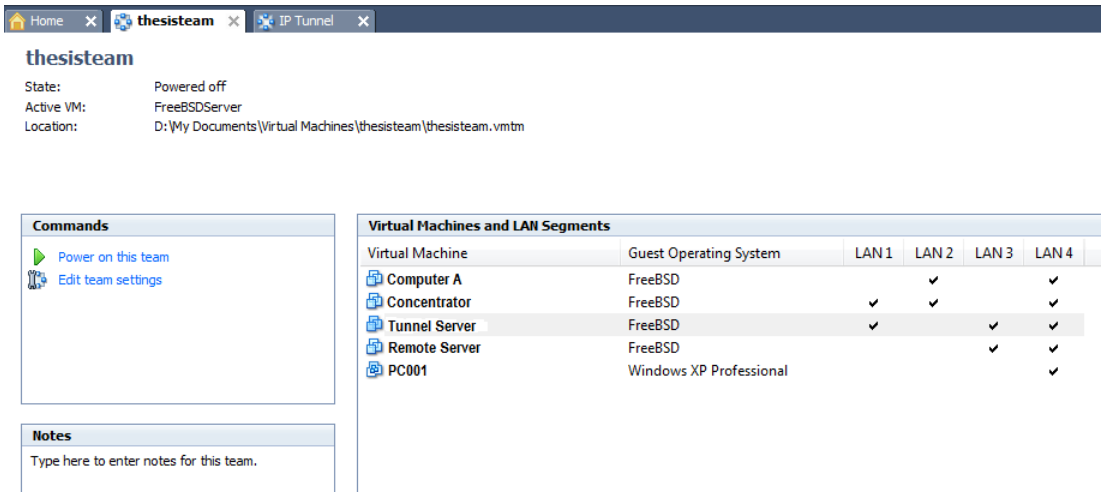


Figure 8-3 Team and LAN Configuration of L2TP/IPsec Tunnel Simulation

Transferring files between virtual machines and the host machine is a necessary part of using VMware Workstation software. When new software has to be installed on a virtual machine, or when some source code must be modified in a virtual machine, file transfer between the virtual machine and the host machine is an easy way to achieve this goal. There are several methods to transfer files:

- VMware tools:

  This is the easiest way for file transferring. After VMware tools (software) for specific operating systems, such as Windows or Linux, are installed, files can be transferred between virtual machines and the host machines by simply dragging and dropping. However, this method does not work in a FreeBSD virtual machine. FreeBSD in this thesis is used as a command line operating system and files cannot be dragged and dropped directly to FreeBSD virtual machines after VMware tools for FreeBSD are installed.

- Flash disks or other external storage devices:

  As shown in Figure 8-2, external storage devices can be configured when a virtual machine is turned off. Then files can be transferred directly from external storage devices to virtual machines under Windows and Linux. Under FreeBSD, “mount” and “cp” commands have to be used to notify operating system a file system is ready and to copy the files. However, this method is not convenient in the L2TP/IPsec tunnel simulation, as different virtual machines cannot share an external storage device and the hardware settings has to change every time a file transfer is needed.

- FTP: File Transfer Protocol (FTP) is a network protocol used to exchange files over a TCP/IP based network [70]. Files can be transferred by setting up an FTP server, and connecting to the FTP server from other virtual machines (FTP clients). This is a convenient way to transfer files to or from FreeBSD virtual machines under command line.

8.2 SETTING UP TEST ENVIRONMENT

The goal of the simulation is to develop and setup an L2TP/IPsec tunnel, and to test the handoff time of the L2TP/IPsec tunnel. The L2TP/IPsec tunnel was established and tested among different virtual machines in VMware.

Two software setups were used: the first setup was FreeBSD 7.0 with unmodified L2TP/IPsec and the second setup was the implementation of the novel L2TP/IPSec solution.

The topology of L2TP/IPsec tunnel simulation is shown in Figure 8-4.

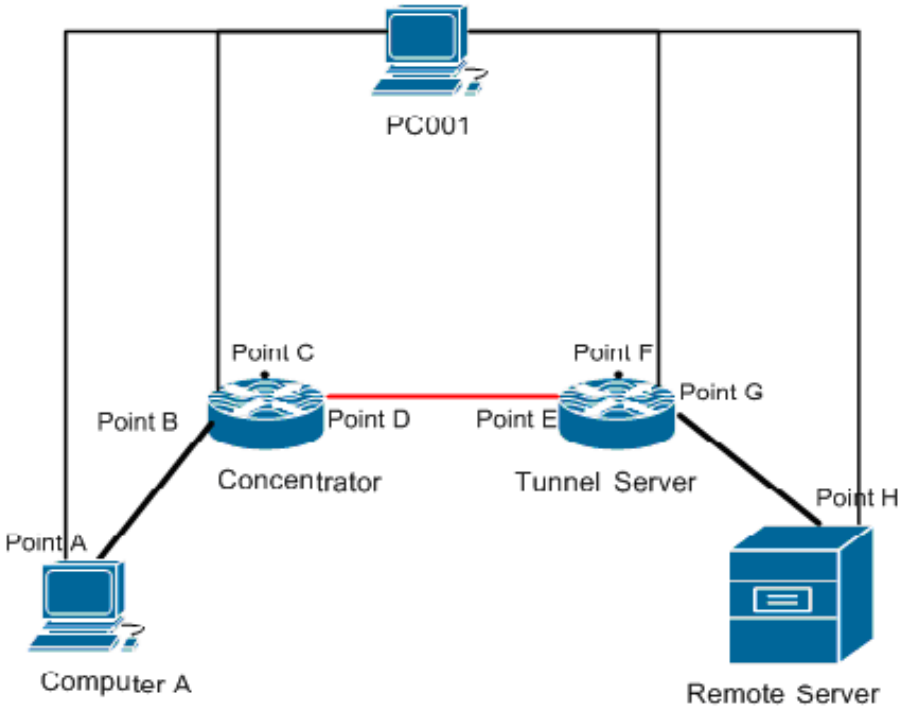


Figure 8-4 Topology of L2TP/IPsec Tunnel Simulation

Four copies of FreeBSD 7.0 [6] and one Windows XP were installed as virtual machines in VMware on a host machine (Windows 7). Three networks were used to establish the L2TP/IPsec tunnel (the networks between Point A and B, between Point D and E, and between Point G and H in Figure 8-4), and one network was used for file transferring. LAN networking (see Section 8.1) was used on all virtual machines, so that there was no network connection between virtual machines and the host machine. Each network belonged to a LAN. The Windows XP virtual machine was not a part of the L2TP/IPsec tunnel, and it was only used for

file transfer between virtual machines and the host machine. An FTP server (Serv-U [71]) was set up on LAN 4 (see Figure 8-3) on the Windows XP virtual machine (PC001 in Figure 8-3), so that files could be transferred between the host machine (Windows 7) and the Windows XP virtual machine by simply dragging and dropping, and then the files could be further transferred to other virtual machines by FTP service inside LAN 4.

No routing protocol (the protocol to automatically exchange route among different routers or network devices) was used in this simulation. Only static routes were used and so tunnel traffic would not go through PC001 which only provided file transfer service. The concentrator and the tunnel server should have a static route to the other tunnel endpoint (Point E and Point D in Figure 8-4). The route to the network behind the concentrator and the tunnel server (the networks between Point A and B, and between Point G and H in Figure 8-4) will be created automatically after a successful L2TP/IPsec tunnel establishment. A default route to Point B had to be set on Computer A in Figure 8-4. A similar route to Point H had to be created on Remote Server in Figure 8-4.

Software needed to be installed on the concentrator and the tunnel server included MPD (version 5.2) [2] and racoon2 (version racoon2-20071227e) [1]. The FreeBSD kernel also needed to be modified and recompiled. These matters have already been discussed in Chapter 4, 5 and 6. The detailed settings of the L2TP/IPsec tunnel will be shown in Appendix B. Software, which sent a stream of UDP packets at a regular heartbeat interval, had to be developed and installed on Computer A. No software needed to be installed on Remote Server. The program tcpdump [72] was used to capture packets being transmitted or received over a network interface. Tcpdump is installed on FreeBSD by default. Wireshark [73] was used to capture packets and view the packets received by tcpdump or wireshark itself.

The mobility support requires the addition of about 1500 lines of code in kernel space and about 400 lines of code in user space. The details of these modifications will not be shown in this thesis for IPR (Intellectual Property Rights) issues. The principle of these modifications is already discussed in Chapter 7. The modifications do not change the architecture of the original

IPsec and L2TP implementations. The architecture of these implementations is already discussed in Section 4.4, 5.4 and 6.4.

The "ifconfig" command was used to change the IP address of the outside network interface of the tunnel concentrator. It is assumed that the execution time of this command is very short. The complete command used is shown below:

ifconfig <interface name> inet <new IP address>

Note that other ways of acquiring a new IP address such as DHCP may take much longer. The aim of the simulation is to test just the new L2TP/IPsec approach and so the "ifconfig" command was used to eliminate the delays in getting a new IP.

8.3 EXPERIMENTAL RESULTS ANALYSIS

Two software setups were used. The first setup was FreeBSD 7.0 with unmodified L2TP/IPsec and the second setup added the implementation of the novel L2TP/IPSec solution. The two setups were tested in the virtual networks (see Figure 8-4) by sending a stream of UDP packets at a regular heartbeat interval from Computer A to the Remote Server. The regular periods tested range from 5ms to 50ms and so simulated the majority of streamed traffic (see figure 8.5).

First the unmodified software was tested. L2TP/IPsec tunnels do not natively support an IP change so the time delay involved is simply the average time for initializing a new L2TP/IPsec tunnel. In our virtual environment, this time was 1.56 seconds. Such a break, and the packet loss involved, would be unacceptable for many protocols and users.

Next the modified software was tested and the results are shown in Figure 8-5. The result was quite exciting: the average handoff time for an IP change in the L2TP/IPsec tunnel was only 0.08 second which was far more acceptable for many protocols and users. The handoff time is mainly caused by three things:

(1) The time to get a new IP address. The "ifconfig" command is used to get a new IP address. The execution time of this command is very short.

(2) Packet buffering in operating system. Windows XP buffers network traffic. This can be proved by the experiment described below. Initially, the operating system of Remote Server in Figure 8-4 was Windows XP and other configurations were the same as current configurations. Packets sent out of Point G and Packets received on Point H were recorded. The experimental result showed that if Tunnel Server sent packets every 0.3 seconds, Remote Server received packets every 0.6 seconds and no obvious delay was shown on Remote Server during VPN handoff. Windows XP doubled the time to receive a packet (several tests were executed to prove this fact). If every computer used FreeBSD, no buffering was detected and a delay was shown during VPN handoff. From this experiment, it can be concluded that Windows XP always buffers and delays packets while FreeBSD does not. FreeBSD machines were used to exclude the buffering factor from influencing the test result.

(3) The time for VPN negotiation. This was the major delay in the simulation. The main goal of the novel L2TP/IPSec solution is to minimize the time for L2TP/IPsec tunnel negotiation after an IP address change.

Compared to the unmodified software, 95 percent of the handoff time is saved. In addition the new solution does not lose any packets.

It is hard to compare this experimental result with results from other mobility support solutions to IPsec or L2TP [19, 21]. The handoff time is not mentioned in these papers. But given the algorithms described above, we believe that the solution proposed in this thesis will have significantly better performance (see Section 7.11).

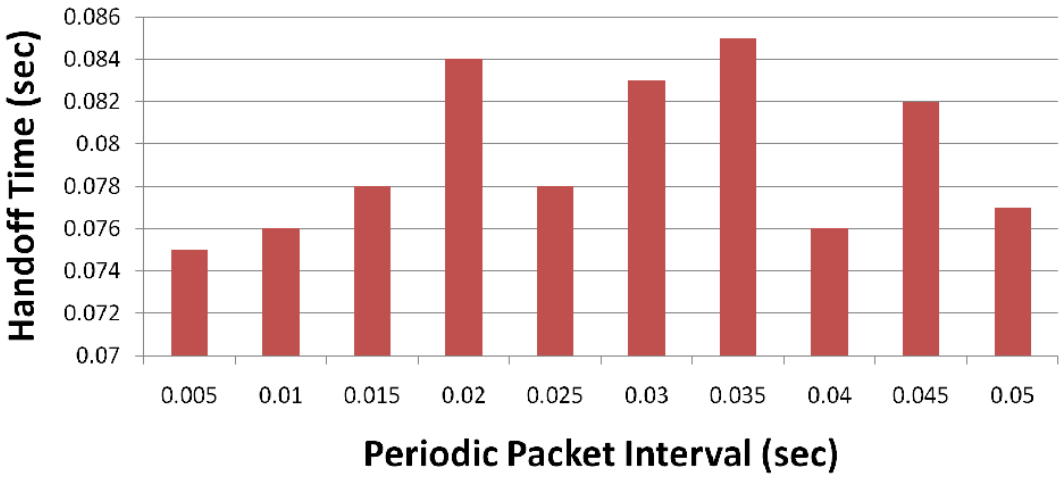


Figure 8-5 Test Result for Handoff Time in IPsec/L2TP Tunnel

8.4 OTHER SOLUTION ANALYSIS

The most common solution to mobility problems using VPN is to run tunnels, such as L2TP or IPsec, over Mobile IP (double tunneling) [21, 22]. Some solutions [19, 53] add mobility support only to IPsec. This section will review these solutions and compare them with the mobility solution proposed in this thesis.

8.4.1 Berioli and Trotta's Solution

Berioli and Trotta of Institute of Communications and Navigation German Aerospace Centre have proposed an IP mobility solution over a Mobile IP network [21]. The main idea of their solution is to run IPsec over Mobile IP. IP packets are protected by IPsec using ESP between the Mobile Node and the VPN Gateway. A Mobile IP tunnel will be used to redirect ESP packets between Foreign Agent and Home Agent. When the MN roams into another foreign network, only the Mobile IP binding needs to be updated and nothing is changed in the IPsec tunnel.

Figure 8-6 shows how IP packets are transferred through the IPsec over Mobile IP tunnel. Screening routers (routers that control traffic to specific networks) are used to redirect VPN traffic and protect networks.

A packet sends from Host A to Mobile Node (MN):

(1) Host A sends an IP packet to the home address.

(2) The internal screening router redirects the packet to the VPN Gateway (VPN-GW) according to the policy.

(3) The VPN-GW receives the packet destined to the home address. It checks IPsec policies and encrypts the packet into an ESP packet in tunnel mode. A new packet from VPN-GW address to the home address is created and sent to the HA.

(4) The HA encapsulates the ESP packet in a Mobile IP packet (from home agent to foreign agent) and sends the packet.

(5) FA receives the Mobile IP packet, decapsulates the packet and sends the obtained ESP packet to the Mobile Node.

(6) MN decrypts the ESP packet and gets the original packet from Host A.

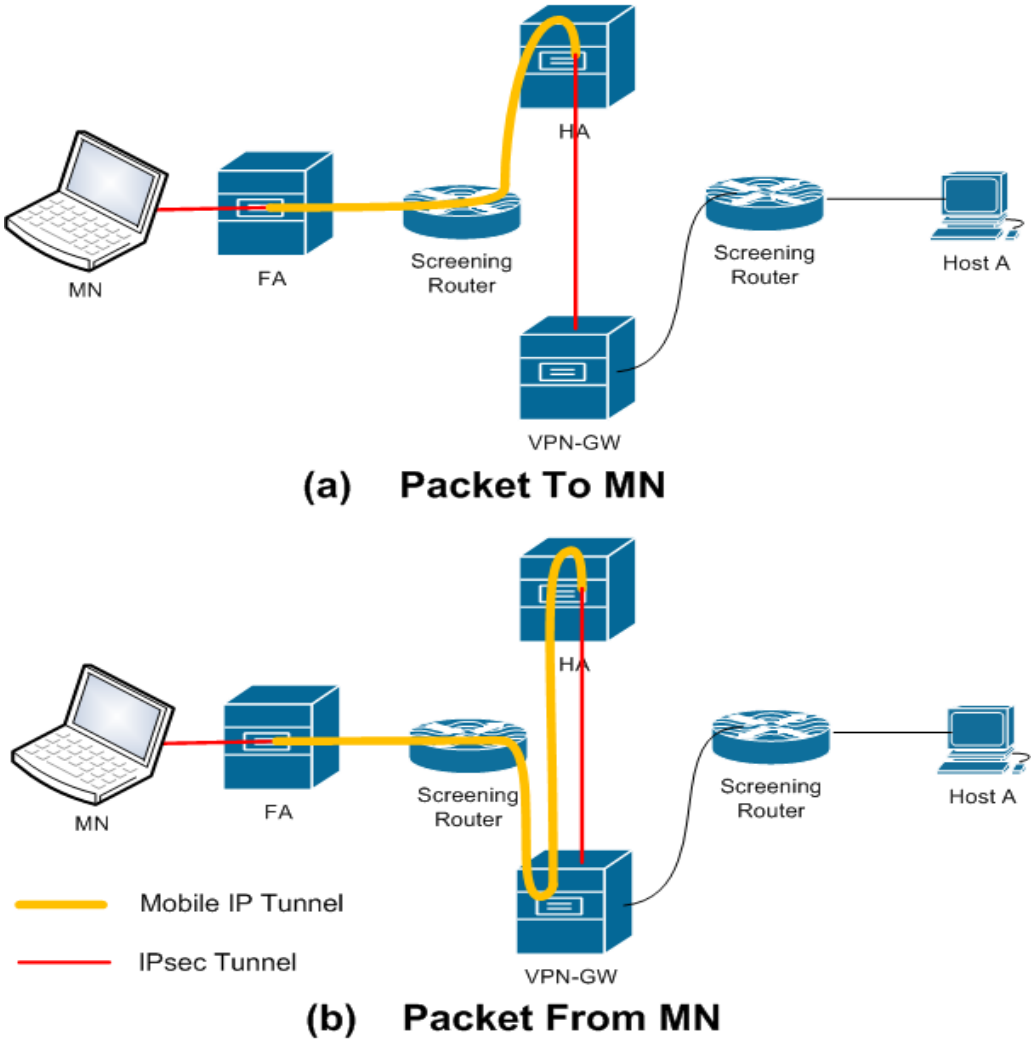


Figure 8-6 System Operation of Berioli and Trotta's Solution

A packet sends from MN to Host A:

(1) MN encapsulates the packet into an ESP packet. This packet has the home address as sender and VPN-GW address as recipient. The packet is sent to the FA.

(2) The FA receives the packet, encapsulates the packet into a Mobile IP packet with FA as source and HA address as destination, and sends the packet.

(3) The packet reaches the external screening router and is redirected to the VPN-GW and then to HA.

(4) HA decapsulates the Mobile IP packet and forwards the embedded ESP packet to VPN-GW.

(5) VPN-GW decrypts the ESP packet and sends the obtained packet to Host A.

The biggest problem of this solution is the large external equipment requirement. At least one HA, one FA and two screening routers are used. Another problem is the large overhead (20% as mentioned in the paper). Furthermore, this solution does not have ability to tunnel Layer 2 packets.

Compared to this solution, the novel L2TP/IPsec mobility solution proposed in this thesis requires no external equipment, has smaller packet overhead and can tunnel Layer 2 packets.

8.4.2 **Comstock's Solution**

Comstock has proposed a mobility solution to tunnel Layer 2 packets over a Mobile IP network [21]. L2TP is used to tunnel Layer 2 packets. LNS will be installed on Home Agent and LAC will be installed on Foreign Agent. When a packet is received at Home Agent, LNS encapsulates the packet and sends the new packet to Foreign Agent according to Mobile IP information. LAC at the Foreign Agent decapsulates the packet and forwards the original packet to Mobile Node. For returning traffic, Mobile Node sends the Layer 2 packet to the Foreign Agent. The Foreign Agent may use the same L2TP tunnel for the returning traffic, or create another Layer 2 session to Correspondent Node directly. In summary, an L2TP tunnel is created between Home Agent and Foreign Agent. Figure 8-7 shows this solution.

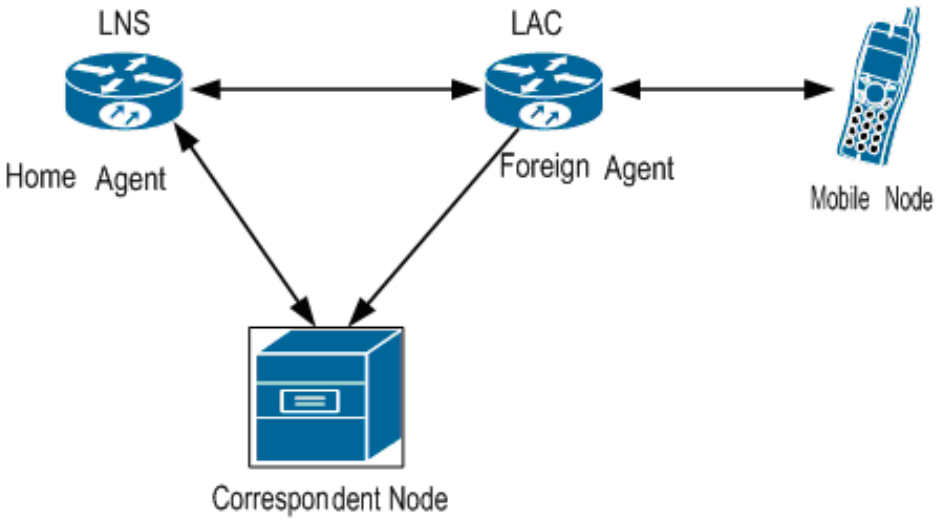


Figure 8-7 Comstock's Solution

The major problem of this solution is security. Both Mobile IP and L2TP do not protect packet confidentiality. This solution cannot be used as a VPN protocol when security is a concern. The security of the L2TP/IPsec solution proposed in this thesis is much better than the Comstock's Solution.

8.4.3 **Kim and Srinivasan's Solution**

Kim of AT&T Labs-Research and Srinivasan of Georgia Institute of Technology have proposed a simple mobility support to IPsec tunnel mode [19]. The main idea behind their solution is to change the tunnel endpoint IP address of the mobile host at the IPsec VPN gateway via a secure signalling. Client to Server VPN is used in this solution. The topology of this solution is shown in Figure 8-8.

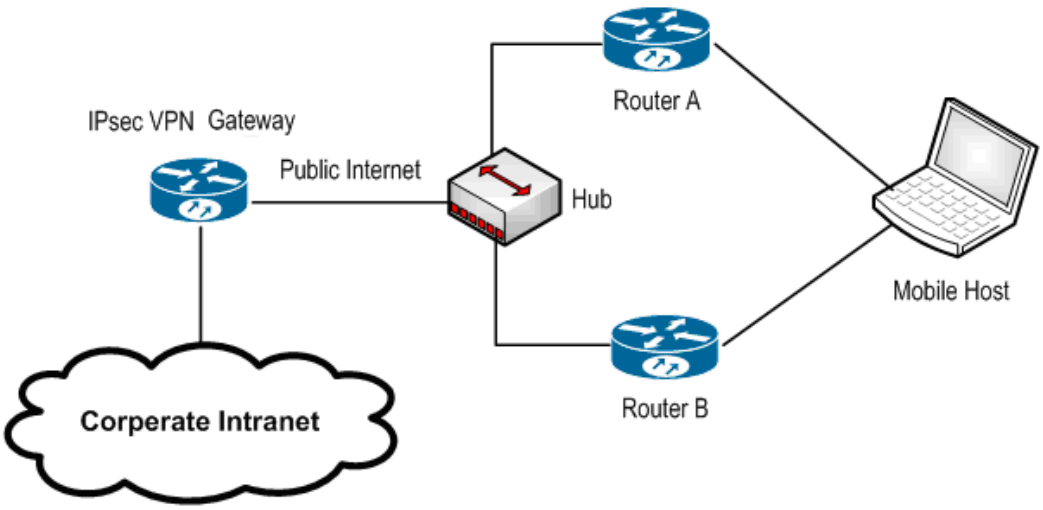


Figure 8-8 Topology of Kim and Srinivasan's Solution

The two IPsec tunnel endpoints are the Mobile Host and the IPsec VPN Gateway. Two different routers are used to assign the Mobile Host IP addresses from different networks. Mobility is then simulated by switching between the two networks. When the Mobile Host changes its IP address, it updates the SADB and informs the IP change by sending an ISAKMP message to the IPsec VPN Gateway. The IPsec VPN Gateway checks the message, updates the SADB and sends an ISAKMP message for reply. After this step, IPsec data packets can be transferred through the tunnel again. This solution also changes the rule for security checking of each data packet. It removes the dependence of identifying a Security Association on the outer header destination address so that the same security parameters can be used in the new network.

This solution has many weaknesses. First, removing outer header destination check introduces new security vulnerabilities. When several persons with different privileges connect to the same IPsec VPN Gateway, they may use their original identity to access the IPsec VPN server and spoof their IP addresses so that the packets are decrypted and forwarded to the resources that are off limits (see Section 4 and 5 of [16]). Second, exchanging ISAKMP messages for IP change information has worse performance than exchanging ESP messages. This is already discussed in Section 7.11. Third, ISAKMP is already obsolete in IKEv2 [11].

Compared to this solution, the novel L2TP/IPsec mobility solution proposed in this thesis does not introduces new security vulnerabilities, has better performance and can tunnel Layer 2 packets.

8.4.4 **Eronen's Solution**

Eronen of Nokia has proposed the IKEv2 Mobility and Multihoming Protocol (MOBIKE) [53]. This protocol is an extension of IKEv2 and it allows the IP addresses of the tunnel endpoints in IPsec tunnel mode to change. Only when IKE exchange has progressed far enough (finish creating child SA, in other words, IPsec can transfer data packets), MOBIKE can handle the IP change. All the messages exchanged in this solution are a part of IKE message exchange. The protocol exchanges are shown below:

(1) After IKEv2 finishes the first exchange (IKE_SA_INIT), a MOBIKE_SUPPORTED notification should be included in the IKE_AUTH exchange. Data packets can be transferred through IPsec after this exchange.

(2) When a tunnel endpoint changes the IP address, one INFORMATIONAL exchange will be used to update the IP change information. NAT information may also be included in this exchange to ensure that the IP addresses have not been modified by NATs.

(3) If IPsec system requires "return routability" check, the check should be done before step 2.

The security in MOBIKE is strong. First, a "return routability" check can be used to verify the addresses provided by the peer. Second, a "NAT prohibition" feature can be used to ensure that IP addresses have not been modified by NATs, IPv4/IPv6 translation agents, or other similar devices.

The problem of this solution is synchronization. When a tunnel endpoint changes its IP address and also updates SA information, it is possible to build an ESP packet with its old IP address and send the packet to its new IP address. This may cause big trouble. A lock has to be added to UNIX or Linux kernel to solve the synchronization problem. The detailed analysis of the synchronization problem and the solution has already been discussed in Section 7.9.

Compared to this solution, the novel L2TP/IPsec mobility solution shown in this thesis has better performance, as it uses ESP to update IP change information (see Section 7.11). The new solution can tunnel Layer 2 packets, while the MOBIKE solution cannot. Furthermore, the MOBIKE solution does not handle the synchronization problem, while the new solution has solved the problem. Although this solution has stronger security ("return routability" and "NAT prohibition"), these features may not be necessary as the IP change messages are fully protected by ESP.

8.5 SUMMARY

In this chapter, first, the VMware Workstation software is introduced and then the novel L2TP/IPsec solution is proved in a VMware simulation. Four copies of FreeBSD and one Windows XP were installed as virtual machines in VMware on a single computer. The network between these virtual machines is used to create an L2TP/IPsec tunnel and to test the handoff time. Our experimental setup measured this delay at 0.08 seconds compared to 1.56 seconds for the existing LT2P/IPSec solution.

Next the experimental result is analysed. The new solution makes a mobile L2TP/IPSec solution practically viable as a delay of 0.08 seconds with no packet loss will not be noticeable, even for VoIP traffic. In comparison the existing L2TP/IPsec solution has a delay of 1.56 seconds, plus packet loss over this period, and this will be annoying to the users.

Finally, four concurrent studies related to VPN mobility support are reviewed and compared to the new L2TP/IPsec solution. It was shown that the new solution has better performance and very good security.

In the next chapter, the new L2TP/IPsec solution will be tested on real computers and the test result will be analysed and discussed.

Chapter 9: Experiment on Real Devices

In the previous chapter, the theory of the novel mobility support for the L2TP/IPsec tunnel has been proved by a network simulation in VMware. The handoff time was at 0.08 seconds compared to 1.56 seconds for the existing LT2P/IPSec solution.

In this chapter, the same theory is proved on real devices. Six FreeBSD machines and two wireless access points were used to test the handoff time of the L2TP/IPsec tunnel in the new solution. The test result was 4.8 seconds which included 4.6 seconds for getting a new IP address and only 0.2 seconds for VPN negotiation. The test result proves that the novel mobility support decreases the VPN negotiation time dramatically which is the goal of this thesis, but the time to get an new IP address needs to be shortened. Several methods have been proposed to solve this problem. This chapter is organized as follows:

First, an overview of VPN handoff time is presented followed by an introduction about how to get an IP address from different networks. Next, different kinds of handoff solutions are overviewed and DHCP is discussed. Finally, the experiment on real devices is shown and the result is analysed.

9.1 OVERVIEW OF VPN HANDOFF TIME

In the real world, the time for VPN handoff is mainly caused by four things:

(1) The time to get a new IP address. Although lower layer connections (layer 1 and layer 2) can be got from different sources: for example UMTS [78], Wireless LAN and WiMAX [77], DHCP or other similar protocols must be used to assign an IP address for the mobile client. This will be fully discussed in Section 9.2. This time is the main delay in the experiment on real devices.

(2) Packet buffering in operating system. This is already discussed in Section 8.3. This can be ignored in FreeBSD.

(3) Packet transmission time. It is the amount of time it takes a bit to go from the start of the link to its destination node. It depends on the physical medium and the length of the link. This time becomes noticeable when the tunnel concentrator and the tunnel server are far from each other, and several exchanges need to be performed during handoff.

(4) The time for VPN negotiation. The main goal of the novel L2TP/IPSec solution is to minimize the time for L2TP/IPsec tunnel negotiation after an IP address change.

9.2 GETTING AN IP ADDRESS FROM DIFFERENT NETWORKS

Connections can be got from different sources: for example UMTS [78], Wireless LAN, and WiMAX [77] etc. Different processes can be used to make a lower layer connection (Layer 1 and Layer 2).

In WiMAX, the Base Stations (BSs) periodically broadcast Mobile Neighbour Advertisement control messages. The Mobile Node scans and synchronizes with the neighbouring Base Stations for channel information (signal strength and other information for base stations). The Mobile Node makes a decision about which BS(s) it should connect to according to channel information. After that, real handoff begins in a WiMAX network [74].

Similar processes are used in UMTS [78] or other telecommunication areas. In UMTS, first discovery phase is used to discover base stations and channel information. Then, evaluation phase is used for the mobile nodes to decide whether a handoff is worthwhile. Finally, in execution phase, a real handoff happens [75].

After a lower layer connection is made, IP addresses will be assigned by routers behind base stations. DHCP is a popular protocol to assign IP addresses. Mobile IP can also be used to assign IP addresses. In Mobile IP, the care-of address is assigned by DHCP or a foreign agent's advertisements [23]. DHCP will be discussed in Section 9.4 as its special position in a wireless network.

In telecommunication technologies, a mobile node may apply a new connection while the current connection is still valid. In this way, the handoff time is shortened. In a Wireless LAN,

however, the operating system will search for a new wireless access point only when the current wireless access point is completely down for some time.

Wireless access points are only switches with wireless functions. In order to get an IP address, a DHCP server must be configured behind wireless access points.

9.3 HANDOFF BETWEEN DIFFERENT NETWORKS

Handoff between different kinds of wireless networks is a popular research area. One of the examples is Unlicensed Mobile Access [76]. It can be used to seamlessly roam and handoff between wireless LANs and UMTS [78]. However, these solutions only support handoff between 2 kinds of networks.

The novel L2TP/IPsec mobility solution supports handoff between any kinds of wireless networks, because this solution only needs to get an IP address from a DHCP server (behind a base station or a wireless access point) or other sources. At least, the solution can be used within UMTS, Wireless LAN or WiMAX.

9.4 DHCP

Dynamic Host Configuration Protocol (DHCP) [66] provides configuration parameters to Internet hosts (DHCP clients). The main function of a DHCP server is to assign IP addresses, default gateways or to provide other configuration parameters to its DHCP clients. DHCP uses UDP packets to exchange DHCP information. Two ports are assigned to DHCP service: 67 and 68.

The basic operations of requesting a new IP address via DHCP are described as follows:

(1) DHCP discovery. When a DHCP client (a computer or other network devices using DHCP configuration at its network interface) connects to a network, it broadcasts a DHCPDISCOVER message on the physical subnet to discover available DHCP servers.

(2) DHCP offer. When a DHCP server receives a DHCPDISCOVER message, it checks available IP addresses from an IP address pool and responds with a

DHCPOFFER message that includes an available network IP address and other configuration parameters in DHCP options.

(3) DHCP request. A DHCP client may receive DHCP offers from multiple servers, but it will accept only one DHCP offer. The DHCP client then broadcasts a DHCPREQUEST message that must include "server identifier" to indicate which server it has selected. The IP address it requests should also be included in the DHCPREQUEST message.

(4) DHCP acknowledgement. The DHCP servers receive the DHCPREQUEST messages from clients. If the DHCPREQUEST message does not belong to the DHCP server, the DHCP server deletes the offer to the client. If the DHCPREQUEST message belongs to the DHCP server, the DHCP server commits the binding for the client and responses with a DHCPACK message containing the configuration parameters.

(5) When the DHCP client receives the DHCPACK message, it checks the configuration parameters and configures these parameters.

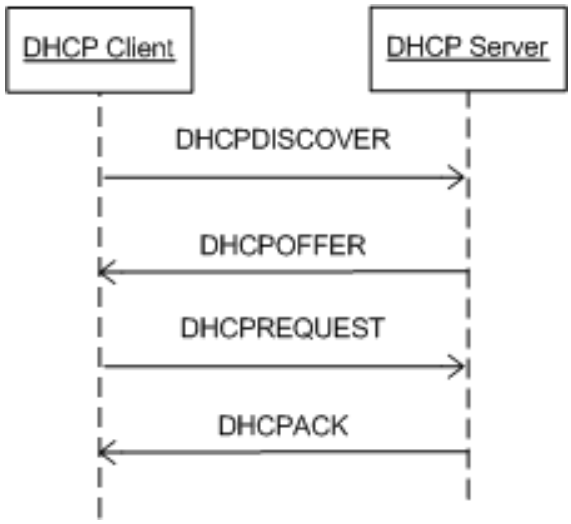


Figure 9-1 Sequence Diagram of Messages Exchanged between a DHCP Client and Server when Allocating a New Network Address

By default (under Windows, FreeBSD and Linux), the operating system will only scan for new IP addresses after current connection is lost for some time. Therefore, a force scan for

new connections when current connection is lost should be performed by software in the tunnel concentrator in the experiment on real devices. This software was developed by the author of this thesis.

When the operating system scans for new connections, by default, the DHCP client will not scan for new IP addresses at first. The DHCP client will first request the previous IP address by sending the DHCPREQUEST message. If the old IP address cannot be used, the process for requesting a new IP address will be used (starts by sending a DHCPDISCOVER message). The time for requesting the previous IP address is about 15 seconds in Windows and about 25 seconds in FreeBSD (lab test result). In the experiment on real devices, a new network address will be assigned to the tunnel concentrator, then, the old IP address cannot be used. Waiting for 25 seconds to scan for a new network is not acceptable in a VPN handoff.

One solution is to force DHCP clients to renew IP addresses. Then, when a connection is lost, the DHCP client will scan for a new IP address from the DHCPDISCOVER message instead of trying the old IP address. Under Linux, “dhclient -r” can be used for this function. Under FreeBSD, no command supports this function. In the experiment, the “lease” file (the file to store DHCP lease information) was deleted after a connection was lost. The FreeBSD could not find the old lease so that it would scan for a new IP address from the DHCPDISCOVER message.

The other solution is to design a fast DHCP. In this new protocol, the DHCP client will only try the old IP address when there is a DHCP server with same MAC address on adjacent networks. The time to try the old IP address should also shortened by simplifying the process involved. Currently, there is little work on fast DHCP. This thesis will not discuss this topic any further but this might become an interesting research area.

9.5 SETTING UP TEST ENVIRONMENT

In the experiment, six FreeBSD machines and two wireless access points were used to test the handoff time of the L2TP/IPsec tunnel in the new solution. Two FreeBSD machines were used as DHCP servers and the other four FreeBSD machines were used for L2TP/IPsec tunnel.

Two DHCP servers were responsible for forwarding packets and assigning IP addresses to the two wireless access points and the wireless workstations behind. The two DHCP servers would assign IP addresses within different networks to wireless access points and wireless workstations behind so that a wireless workstation would connect to a different network when connected to a different wireless access point. The topology of the experiment is shown in Figure 9-2.

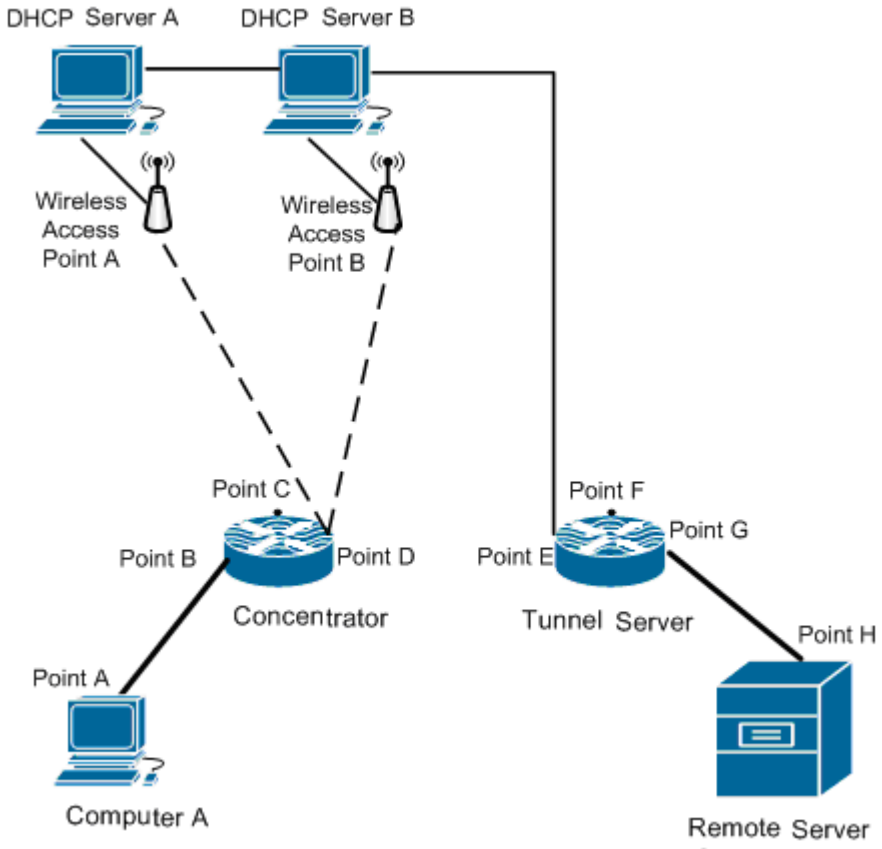


Figure 9-2 Topology of Experiment on Real Devices

The software configurations and the code change in FreeBSD are similar to the simulation discussed in Chapter 8. The differences are as follows:

(1) New software was installed on tunnel concentrator to delete the "lease" file when the old wireless connection was lost and to scan for new networks immediately (see Section 9.4).

(2) A default route was set at the tunnel concentrator by DHCP servers so that updating route information in the new solution was not necessary (see Section 7.7).

(3) The driver for the wireless adapter on the tunnel concentrator must be properly installed.

Some photos about the experiment on real devices are displayed in Figure 9-3, 9-4 and 9-5.

Figure 9-3 Wireless Access Point

Figure 9-4 Computer A and Tunnel Concentrator

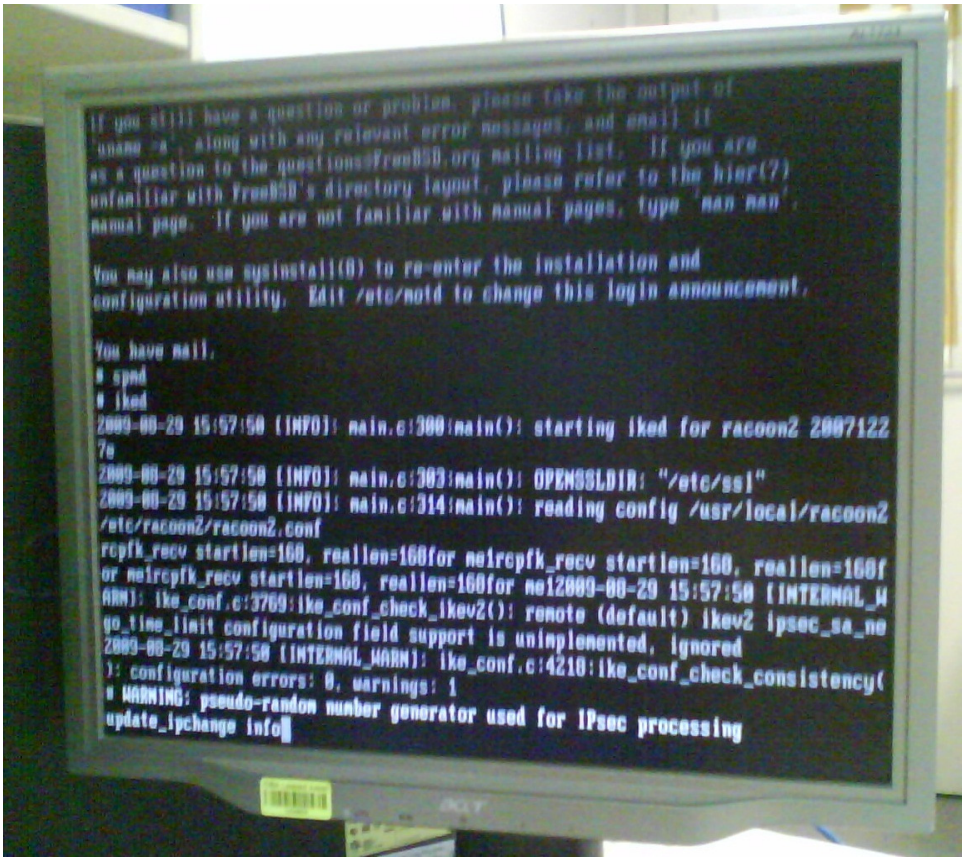


Figure 9-5 Tunnel Server

9.6 EXPERIMENTAL RESULTS ANALYSIS

The L2TP/IPsec tunnel handoff was tested (see Figure 9-2) by sending a stream of UDP packets at a regular heartbeat interval from Computer A to the Remote Server. The regular period tested was 100ms. Tcpdump [72] was used to capture packets being transmitted or received over each network interface. At first, the Tunnel Concentrator got an IP address from Wireless Access Point A. When the Tunnel Concentrator and the Computer A moved out of the range of Wireless Access point A, the Tunnel Concentrator would lose current connection, delete current "lease" file, scan and connect to Wireless Access Point B to get an IP address. Then, the L2TP/IPsec tunnel started negotiation. Finally, after tunnel negotiation, normal L2TP/IPsec tunnel traffic could be transmitted normally between Computer A and the Remote Server.

The test result was a bit disappointing. The total handoff time was 4.8 seconds which includes only 0.2 seconds for VPN negotiation. In practice a 4.8 second delay is absolutely unacceptable for a VPN tunnel user even if there is no packet loss.

The 0.2 seconds for VPN negotiation was bigger than the 0.08 seconds in VMware simulation (see Chapter 8). This is because the traffic transmission time (see Section 9.1) used in VPN negotiation increases dramatically on real devices, and in 4.8 seconds (rather than 0.08 seconds) a larger number of lost packets have to be resent in VPN negotiation.

Most of the time was spent on getting an IP address from wireless access points (4.6 seconds). This time included lower level (physical and data link layer) negotiation and DHCP negotiation. Lower level negotiation is beyond the scope of this thesis as it involves technologies on a hardware level. DHCP negotiation can be improved as discussed in Section 9.4.

Another way to improve VPN handoff in a wireless LAN is to let a mobile node apply for a new connection while current connection is weak. This involves broadcasting signal strength and other information in a wireless access point, and making a decision about which wireless access point should the wireless device connects to. This feature has already been implemented in protocols such as WiMAX and UMTS. However, the solution in this thesis aims only to modify wireless access points and the operating system using a wireless adapter and so cannot have the abilities of WiMAX and UMTS. It would be possible to apply the techniques

used in WiMAX and UMTS to Wi-Fi [79] in wireless LAN but this is a major change and would be a thesis in its own right.

Implementing the novel L2TP/IPsec mobility solution proposed in this thesis on telecommunication systems should improve speeds as a mobile node can apply for a new connection while the current connection is active. The implementation of the new mobility solution on telecommunication systems is not done in this thesis due to the limited study time allowed a Master student.

From the discussion above, the test result proves that the novel mobility support decreases the VPN negotiation time dramatically which is the goal of this thesis. However, the time to get an IP address (4.6 seconds) in a wireless LAN is too large to be used in the real world. Several improvements have been proposed to solve this problem. The new mobility solution can also be used in other telecommunication systems where it will have better performance than in a wireless LAN.

9.7 SUMMARY

In this chapter, the new and novel mobility solution to L2TP/IPsec tunnels was implemented and tested on real devices using a wireless LAN. The L2TP/IPsec tunnel performed a successful handoff; the test result showed a very short time in VPN negotiation (0.2 seconds) and a long time in getting a new IP address (4.6 seconds). This means that the mobility solution is successful with a short negotiation time, but the speed of acquiring a new IP address using DHCP must be improved before the new solution is practically useful.

Six FreeBSD machines and two wireless access points were used to test the handoff time of the L2TP/IPsec tunnel in the new solution. New software was developed and used on the tunnel concentrator to apply for a new IP address through wireless access points when the current IP address was unable to use. The handoff happened when the tunnel concentrator moved out of the range of current wireless access point and into the range of another.

The long delay of 4.6 seconds for DHCP to provide a new IP address is mainly caused by slow DHCP negotiation. One improvement solution is to design a new, fast DHCP. The other

solution is to let a mobile node apply for a new connection while the current connection has weak signal levels in a wireless LAN. These solutions haven't been implemented in this thesis and may become interesting research topics.

The new mobility solution can also be implemented in wireless telecommunication systems. Better performance will be achieved as a mobile node can apply for a new IP from a new network while a current connection is active.

Chapter 10: Conclusions

10.1 CONTRIBUTION

In this thesis the mobility issue in virtual private networks has been addressed and solved using a new and novel method for the combination of L2TP and IPsec. A popular solution to this problem is to run VPN tunnels over Mobile IP (MIP), but Mobile IP itself has significant problems in security and the solution is inefficient due to double tunneling.

L2TP is a Layer 2 VPN and has a great range of applications as it can transfer almost all kinds of Internet packets: IP packets, non-IP packets and Layer 2 packets. It has very good security when teamed with IPsec as IPsec has demonstrated strong security features and has already been integrated into the next generation network (IPv6). The proposed solution has particular application when multiple users within one moving vehicle must use a Layer 2 VPN tunnel to connect to other parts of the network such as a corporate network.

The main idea behind the novel solution is to let the tunnel concentrator communicate with the tunnel server directly about the IP change information without involving new headers. A high level of security is assured during changeover as the IP change information is protected by ESP. Two exchanges are used to first update IP change information in IPsec, and then in L2TP. Lost packets are buffered in L2TP and will be sent after updating IP change information in L2TP. Compared to other solutions described in Section 8.4, the new solution has no packet loss, better performance and very good security, as ESP packets are highly protected and can be handled faster than IKE or ISAKMP messages.

The new solution has been implemented under FreeBSD (a UNIX like operating system). The detailed analyses of L2TP and IPsec implementation have been described in Chapter 4 and 5. Two kinds of L2TP/IPsec tunnel implementations (Upper and Lower Protocol solution and Loopback interface solution) have been proposed and discussed in Chapter 6. The novel mobility

solution does not change the framework of existing L2TP/IPsec tunnel implementations and is easy to migrate to other platforms.

The new mobility solution has been tested under a VMware simulation and on real network devices. In VMware, the experimental setup measured the L2TP/IPsec tunnel handoff time at 0.08 seconds compared to 1.56 seconds for the existing LT2P/IPSec solution. The new solution makes a mobile L2TP/IPSec solution practically viable as a delay of 0.08 seconds will not be noticeable, even for VoIP traffic. In comparison a delay of 1.56 seconds, plus packet loss over this period, will be annoying to the users.

In the experiment using real network devices, the test result was a little disappointing. The total handoff time was 4.8 seconds, however, only 0.2 seconds was used for VPN negotiation and other 4.6 seconds was required to get a new IP address from the wireless network. The main goal of this thesis is to minimize the VPN negotiation time, not minimize IP acquisition time, so we can claim that the novel approach has been successful. The long delay is mainly caused by slow DHCP negotiation and the mobile device only starts to apply for a new IP address when the current connection is invalid. Solutions to this delay are outside the scope of this thesis though two possible solutions have been proposed. One solution is to design a fast DHCP, and the other is to let a mobile node apply a new connection while the current connection is weak.

The proposed mobility solution has no lost packets, avoids tunnel reestablishment, does not involve running IPsec over Mobile IP and is able to handle layer 2 packets. The solution does not cause any new security vulnerability to the L2TP/IPsec tunnel and has good handoff performance making it possible to cope with more rapid movement between networks.

10.2 FUTURE WORK

With the lessons learnt we aim to do equivalent work under IPv6 and create a real physical network for telecommunication systems. The existing L2TP/IPsec approaches will not cope with the rapid movement between wireless nodes but we believe our modified system will

perform successfully under these conditions when a faster method of allocating IP addresses is found.

The work reported in this thesis also opens up some interesting research opportunities. One is to design and implement a fast DHCP. In this new protocol the DHCP client may only try the old IP address when there is a DHCP server with same MAC address on adjacent networks. The other is to design and implement a new mechanism to let a mobile node apply for a new connection while the current connection is weak in a wireless LAN. Currently, only telecommunication systems such as WiMAX and UMTS have this feature. The modification needed for this mechanism has been discussed in Section 9.6.

Bibliography

[1] "Racoon2", Racoon2 website, http://www.racoon2.wide.ad.jp/w/, accessed June 21, 2009.

[2] "MPD", MPD project from sourceforge, http://mpd.sourceforge.net, accessed June 21, 2009.

[3] K. Hamzeh, G. Pall, W. Verthein, J. Taarud, W. Little and G. Zorn, "Point-to-Point Tunneling Protocol (PPTP)", IETF RFC 2637, July 1999.

[4] "OpenSSL", OpenSSL project website, http://www.openssl.org, accessed October 8, 2008.

[5] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol", IETF RFC 5246, August 2008.

[6] "FreeBSD The Power To Serve", FreeBSD official website, http://www.freebsd.org/, accessed January 2008.

[7] Q. Li, T. Jinmei and K. Shima, "IPv6 advanced protocols implementation", Oxford: Elsevier Science, 2007.

[8] J. H. Carmouche, "IPsec virtual private network fundamentals", Indianapolis: Cisco Press, 2007.

[9] M. Feilner, "OpenVPN: building and integrating virtual private networks", Birmingham: Packt, 2006.

[10] Anad R. Prasad and Neeli R.Prasad, "802.11 WLANs and IP Networking: security, QoS and mobility", Boston: Artech House, 2005.

[11] Ed. C. Kaufman, "Internet Key Exchange (IKEv2) Protocol", IETF RFC 4306, December 2005.

[12] Errata and S. Kent, "IP Encapsulating Security Payload", IETF RFC 4303, December 2005.

[13] S. Kent, "IP Authentication Header", IETF RFC 4302, December 2005.

[14] J. Kempf, J. Arkko and P. Nikander, "Mobile IPv6 Security", Wireless Personal Communications, Netherlands: Springer, vol. 29, 2004.

[15] D. Johnson, C. Perkins and J. Arkko, "IP Mobility Support for IPv6", IETF RFC 3775, June 2004.

[16] B. Aboba and W. Dixon, "IPsec-Network Address Translation (NAT) Compatibility Requirements", IETF RFC3715, March 2004.

[17] N. Doraswamy and D. Harkins, "IPSec: The New Security Standard for the Internet, Intranets, and Virtual Private Networks", Prentice Hall PTR, 2003.

[18] P. Loshin, “IPv6: Theory, Protocol, and Practice, 2nd ed.”, United States of America: Elsevier, 2003.

[19] B. Jim, S. Srinivasan, “Simple Mobility Support for IPsec Tunnel Mode”, Vehicular Technology Conference, vol. 3, 2003.

[20] E. Fonsell, "Security in IP Mobility Solutions", HUT. report, Seminar on Internetworking, 2003.

[21] M. Berioli and F. Trotta, “IP Mobility Support for IPsec-based Virtual Private Networks: an architectural solution”, Global Telecommunications Conference, Vol. 3, pp.1532-1536, December 2003.

[22] David R. Comstock, Solna, “Mobile terminating l2tp using Mobile IP data”, US Patent 6452920, 2002.

[23] C. Perkins, "IP Mobility Support for IPv4", IETF RFC 3344, August 2002.

[24] B. Patel, B. Aboba, W. Dixon, G. Zorn and S. Booth, “Securing L2TP using IPsec”, IETF RFC 3193, November 2001.

[25] D. E. Comer, “Internetworking with TCP/IP Vol. 1: Principles, Protocols, and Architecture”, London: Prentice-Hall International, 2000.

[26] C. Rigney, S. Willens Livingston, A. Rubens Merit and W. Simpson Daydreamer, “Remote Authentication Dial In User Service (RADIUS)”, IETF RFC 2865, June 2000.

[27] K. Hamzel, G. Pall, W. Verthein, J. Taarud, W. Little and G. Zorn, "Point-to-Point Tunneling Protocol (PPTP)", IETF RFC 2637, 1999.

[28] W. Townsley, A. Valencia, A. Rubens, G. Pall, G. Zorn and B. Palter, "Layer Two Tunneling Protocol (L2TP)", IETF RFC 2661, August 1999.

[29] D. Maughan, M. Schertler, M. Schneider and J. Turner, “Internet Security Association and Key Management Protocol (ISAKMP)”, IETF RFC 2408, 1998.

[30] R. Thayer, N. Doraswamy and R. Glenn, “IP Security Document Roadmap”, IETF RFC 2411, November 1998.

[31] C. Perkins, “Minimal Encapsulation within IP”, IETF RFC 2004, October 1996.

[32] W. Simpson, “PPP Challenge Handshake Authentication Protocol (CHAP)”, IETF RFC 1994, August 1996.

[33] “From Frame Relay to IP VPN: Why to Migrate, Why to Out-Task”, Cisco Website, http://www.cisco.com/warp/public/cc/so/neso/vpn/vpnsp/vpnmi_wp.pdf, accessed on 24/07/2009.

[34] M. H. Behringer and M. J. Morrow, “MPLS VPN Security”, Cisco Press, June 2005.

[35] C. Lewis and S. Pickawance, “Selecting MPLS VPN Services”, Cisco Press, 2006.

[36] O. Kolesnikov and B. Hatch, “Building Linux Virtual Private Networks (VPNs)”, New Riders, 2002.

[37] G. Zorn, “Microsoft PPP CHAP Extension, Version 2”, IETF RFC2759, January 2000.

[38] G. Pall and G. Zorn, “Microsoft Point-To-Point Encryption (MPPE) Protocol”, IETF RFC3078, March 2001.

[39] W. Simpson, “The Point-to-Point Protocol (PPP)”, IETF RFC1661, July 1994.

[40] J. Postel, "Internet Protocol", IETF RFC 791, September 1981.

[41] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", IETF RFC 2460, December 1998.

[42] J. Lau, Ed., M. Townsley, Ed. and I. Goyret, Ed., “Layer Two Tunneling Protocol – Version 3 (L2TPv3)”, IETF RFC 3931, March 2005.

[43] W. Simpson (Ed.), “The Point-to-Point Protocol (PPP)”, IETF RFC 1661, July 1994.

[44] G. Zorn, “PPP LCP Internationalization Configuration Option”, IETF RFC 2484, January 1999.

[45] G. McGregor, “The PPP Internet Protocol Control Protocol (IPCP)”, IETF RFC 1332, May 1992.

[46] W. Simpson, “PPP Challenge Handshake Authentication Protocol (CHAP)”, IETF RFC 1994, August 1996.

[47] A. Valencia, M. Littlewood and T. Kolar, “Cisco Layer Tow Forwarding (Protocol) “L2F””, IETF RFC 2341, May 1998

[48] E. Rosen, A. Viswanathan and R. Callon, “Multiprotocol Label Switching Architecture”, IETF RFC 3031, January 2001.

[49] U. Black, “PPP and L2TP: remote access communications”, Prentice Hall PTR, 2000.

[50] “FreeBSD Man Pages of Netgraph”, FreeBSD Kernel Interfaces Manual, http://www.freebsd.org/cgi/man.cgi?query=netgraph&sektion=4, accessed on 22/09/2009.

[51] P. Srisuresh and K. Egevang, “Traditional IP Network Address Translator (Traditional NAT)”, IETF RFC 3022, January 2001.

[52] R. Rohit, P. Srisuresh, R. Raghunarayan, N. Pai and C. Wang, “Definitions of Managed Objects for Network Address Translators (NAT)”, IETF RFC 4008, March 2005.

[53] P. Eronen, Ed, “IKEv2 Mobility and Multihoming Protocol (MOBIKE)”, IETF RFC 4555, June 2006.

[54] V. Bollapragada, M. Khalid and S. Wainner, “IPSec VPN Design”, Cisco Press, April 2005.

[55] C. Bonatti, Ed., S. Turner, Ed. and G. Lebovitz Ed., “Requirements for an IPsec Certificate Management Profile”, IETF RFC 4809, February 2007.

[56] E. Rescorla, “Diffie-Hellman Key Agreement Method”, IETF RFC 2631, June 1999.

[57] V. Manral, “Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)”, IETF RFC 4835, April 2007.

[58] S. Sakane, K. Kamada, M. Thomas and J. Vilhuber, "Kerberized Internet Negotiation of Keys (KINK)", IETF RFC 4430, March 2006.

[59] Q. Li, T. Jinmei and K. Shima, "IPv6 Core Protocols Implementation", Elsevier Science, 2007.

[60] R. Allen, L. E. Hunter and B. J. Dinerman, "Windows Server 2003 networking recipes", Apress, 2006.

[61] D. Piper, "The Internet IP Security Domain of Interpretation for ISAKMP", IETF RFC 2407, November 1998.

[62] D. Harkins and D. Carrel, "The Internet Key Exchange (IKE)", IETF RFC 2409, November 1998.

[63] R. Deal, "The Complete Cisco VPN Configuration Guide", Cisco Press, 2006.

[64] "openl2tp.org a linux l2tp solution for enterprise vpn and isps", openl2tp project website, http://www.openl2tp.org/, accessed on 22/08/2009.

[65] P. Wouters and K. Bantoft, "Openswan: Building and Integrating Virtual Private Networks", Packt Publishing, 2006.

[66] R. Droms, "Dynamic Host Configuration Protocol", IETF RFC 2131, March 1997.

[67] S. S. Warren, "The VMware Workstation 5 Handbook", Charles River Media, 2005.

[68] B. Ward, "The Book of VMware: The Complete Guide to VMware Workstation", William Pollock, 2002.

[69] "IEEE POSIX Certification Authority", IEEE website, http://standards.ieee.org/regauth/posix/, accessed on 12/10/2009.

[70] J. Postel and J. Reynolds, "FILE TRANSFER PROTOCOL (FTP)", IETF RFC 959, October 1985.

[71] "Serv-U FTP Server", Serv-U software website, http://www.serv-u.com, accessed on 14/10/2009.

[72] "TCPDUMP/LIBPCAP public repository", official site for tcpdump, http://www.tcpdump.org, accessed on 14/10/2009.

[73] "Wireshark: Go deep", Wireshark official website, http://www.wireshark.org, accessed on 14/10/2009.

[74] L. Chen, X. Cai, R. Sofia and Z. Huang, "A Cross-Layer Fast Handover Scheme For Mobile WiMAX", Vehicular Technology Conference, VTC-2007 Fall, IEEE 66th, 2007.

[75] S. Shannon, "Computer Networking and Networks", Nova Science Publishers, 2006.

[76] Y. Zhang, L. T. Yang, J. Ma, "Unlicensed Mobile Access Technology: Protocols, Architectures, Security, Standards and Applications", Taylor & Francis Ltd., 2008.

[77] "WiMax.com Connecting the WiMAX Community", WiMAX official website, accessed on 20/10/2009.

[78] C. Kappler, "UMTS networks and beyond", John Wiley & Sons, 2009.

[79] "IEEE 802.11 Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications", IEEE Standards Association website, http://standards.ieee.org/getieee802/download/802.11-2007.pdf, accessed on 20/10/2009.

[80] "Routing loop problem", Wikipedia website, http://en.wikipedia.org/wiki/Routing_loop_problem, accessed on 28/02/2010.

[81] J. Moy, "OSPF Version 2", IETF RFC 2328, April 1998.

[82] "Triangular routing", Wikipedia website, http://en.wikipedia.org/wiki/Triangular_routing, accessed on 28/02/2010.

Appendices

Appendix A: Publication lists

Journal

[J1] C. Xu and P. Radcliffe, "Building Secure Tunnel from PPP Wireless Network", *Wireless Personal Communications,* DOI 10.1007/s11277-009-9894-x, 2009.

Conference

[C1] C. Xu and P. Radcliffe, "A novel mobility solution based on L2TP/IPsec tunnel", *2009 IEEE Sarnoff Symposium,* 2009.

Appendix B: L2TP/IPsec Tunnel Configuration

It is hard to find an example configuration of L2TP or IPsec tunnel using FreeBSD on the Internet. This appendix will show an example configuration of setting up an L2TP/IPsec tunnel under FreeBSD.

The topology of the L2TP/IPsec tunnel is shown in Figure B-1. The IP addresses of the network interfaces are shown in Table B-1. Loopback interface solution (see Section 6.4) is used in this example.

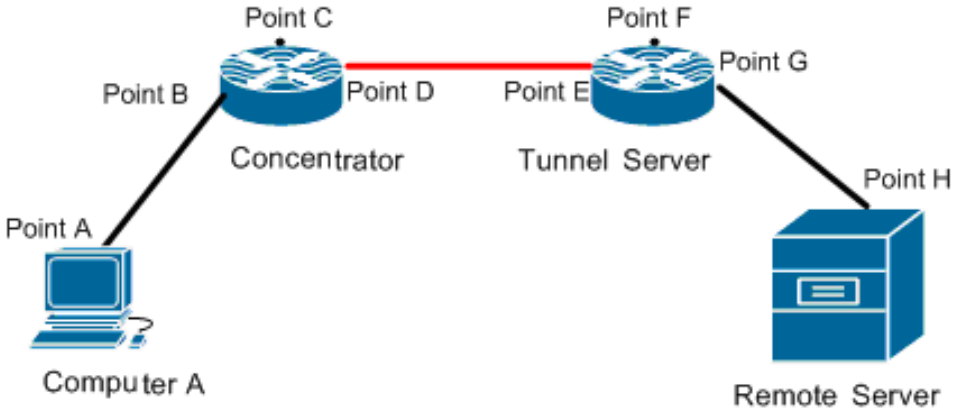


Figure B-1 L2TP/IPsec Tunnel Topology

Network Interface	IP Address
Point A	192.168.4.4
Point B	192.168.4.12
Point C	192.168.5.2
Point D	192.168.1.10
Point E	192.168.1.9
Point F	192.168.5.1
Point G	192.168.3.20
Point H	192.168.3.11

Table B-1 Network Interface Addresses of L2TP/IPsec Tunnel L2TP/IPsec Tunnel

Appendix B.1: L2TP Configuration

MPD is used to configure L2TP. Detailed explanation of MPD configuration is shown in "MPD 5.4.a1 User Manual" in [2].

The configuration of LNS is shown below:

mpd.conf:

```
default:
    load complete_lns

complete_lns:
    create bundle template B1
    set ipcp range 192.168.3.20/32 192.168.4.12/32
    set iface route 192.168.4.0/24

    create link template L1 l2tp
    set link action bundle B1
    set link no pap
    set link yes chap
    set link mtu 1460
    set link keep-alive 10 75
    set link max-redial 0
    set l2tp peer 192.168.5.2
    set l2tp self 192.168.5.1
    set link enable incoming
```

mpd.secret:

```
MyLogin   "kkk"
PeerLogin "kkk"
xu        "kkk"
```

The configuration of LAC is shown below:

mpd.conf:

```
default:
    load complete_lac

complete_lac:
    create bundle static B1
    set ipcp range 192.168.4.12/32 192.168.3.20/32
    set iface route 192.168.3.0/24

    create link static L1 l2tp
    set link action bundle B1
    set link no pap
    set link yes chap
    set auth authname "xu"
    set link mtu 1460
```

```
set link keep-alive 10 75
set link max-redial 0
set l2tp self 192.168.5.2
set l2tp peer 192.168.5.1
set link enable incoming
open
```

mpd.secret:

```
MyLogin   "kkk"
PeerLogin "kkk"
xu        "kkk"
```

Appendix B.2: IPsec Configuration

Racoon2 is used to configure IPsec. The FreeBSD kernel needs to be recompiled to support IPsec. Detailed steps of setting up an IPsec tunnel are shown in "FreeBSD Handbook" of [6]. However, the configuration in "FreeBSD Handbook" is based on the old version of Racoon2. The example below shows the configuration of Racoon2. The configuration of IPsec tunnel concentrator and IPsec tunnel server are symmetric. Only server side configuration of IPsec will be shown.

racoon2.conf:

```
## Edit vals.conf for your environment
include "/usr/local/racoon2/etc/racoon2/vals.conf";

# interface info
interface
{
  ike {
      192.168.1.9 port 500;
  };
  spmd {
      unix "/var/run/racoon2/spmif";
  };
  spmd_password "/usr/local/racoon2/etc/racoon2/spmd.pwd";
};

# resolver info
resolver
{
  resolver off;
};

include "/usr/local/racoon2/etc/racoon2/default.conf";


## Tunnel mode IKEv2 or IKEv1 (initiator and responder)
```

```
include "/usr/local/racoon2/etc/racoon2/tunnel_ike.conf";
```

vals.conf:

```
# $Id: vals.conf.in,v 1.13 2007/12/27 01:08:52 mk Exp $
setval {
### Directory Settings ###
  # Preshared key file directory : specify if you want to use preshared keys
  PSKDIR       "/usr/local/racoon2/etc/racoon2/psk";

  # Cert file directory : specify if you want to use certs
  CERTDIR      "/usr/local/racoon2/etc/racoon2/cert";

### Preshared Key Setting ###
  # Preshared Key file name
  # You can generate it by pskgen.
  PRESHRD_KEY         "xu.psk";

### Certicate Setting ###
  # Your Public Key file name
  MY_PUB_KEY     "my_pub.pem";

  # Your Private Key file name
  MY_PRI_KEY     "my_pri.pem";

  # Peer's Public Key file name
  PEERS_PUB_KEY   "peers_pub.pem";

### Tunnel Mode Settings ###
  # Your Network Address or Host Address (host-to-host tunnel mode)
     MY_NET        "192.168.5.1/32";
  # Peer's Network Address or Host Address (host-to-host tunnel mode)
     PEERS_NET     "192.168.5.2/32";

  # Your SGW Address
  MY_GWADDRESS    "192.168.1.9";

  # Peer's SGW Address
  PEERS_GWADDRESS          "192.168.1.10";

  # Application Version String
  CP_APPVER      "Racoon2 iked";

### Scripts
     ## IKEv2
  IKESAUP_SCR         "/usr/local/racoon2/etc/racoon2/hook/ikesa-up";
  IKESADOWN_SCR    "/usr/local/racoon2/etc/racoon2/hook/ikesa-down";
  CHILDUP_SCR         "/usr/local/racoon2/etc/racoon2/hook/child-up";
  CHILDOWN_SCR      "/usr/local/racoon2/etc/racoon2/hook/child-down";
  IKESAREKEY_SCR   "/usr/local/racoon2/etc/racoon2/hook/ikesa-rekey";
  CHILDREKEY_SCR   "/usr/local/racoon2/etc/racoon2/hook/child-rekey";
  MIGRATION_SCR     "/usr/local/racoon2/etc/racoon2/hook/migration";
  ## IKEv1
  PH1UP_SCR   "/usr/local/racoon2/etc/racoon2/hook/ph1-up";
```

```
  PH1DOWN_SCR        "/usr/local/racoon2/etc/racoon2/hook/ph1-down";
};
```

default.conf:

```
# default section
default
{
  remote {
      acceptable_kmp { ikev2;  };

      ikev2 {
          logmode normal;
          kmp_sa_lifetime_time infinite;
          kmp_sa_lifetime_byte infinite;
          max_retry_to_send 3;
          interval_to_send 10 sec;
          times_per_send 1;
          kmp_sa_nego_time_limit 60 sec;
          ipsec_sa_nego_time_limit 40 sec;
          kmp_enc_alg { aes128_cbc; 3des_cbc; };
          kmp_prf_alg { hmac_md5; hmac_sha1; aes_xcbc; };
          kmp_hash_alg { hmac_sha1; hmac_md5; };
          kmp_dh_group { modp3072; modp2048; modp1024;  };
          kmp_auth_method { psk; };
          random_pad_content on;
          random_padlen on;
          max_padlen 50 bytes;
      };
  };

  policy {
      ipsec_mode tunnel;
      ipsec_level require;
  };

  ipsec {
      ipsec_sa_lifetime_time infinite;
      ipsec_sa_lifetime_byte infinite;
  };

  sa {
      esp_enc_alg { aes128_cbc; 3des_cbc; };
      esp_auth_alg { hmac_sha1; hmac_md5; };
  };
};
ipsec ipsec_ah_esp {
  ipsec_sa_lifetime_time 28800 sec;
  sa_index { ah_01; esp_01; };
};
ipsec ipsec_esp {
  ipsec_sa_lifetime_time 28800 sec;
  sa_index esp_01;
};
```

```
sa ah_01 {
  sa_protocol ah;
  ah_auth_alg { hmac_sha1; hmac_md5; };
};
sa esp_01 {
  sa_protocol esp;
  esp_enc_alg { aes128_cbc; 3des_cbc; };
  esp_auth_alg { hmac_sha1; hmac_md5; };
};
```

tunnel_ike.conf:

```
remote ike_tun_remote {
  acceptable_kmp { ikev2;  };
  ikev2 {
      my_id ipaddr "192.168.5.1";
      peers_id ipaddr "192.168.5.2";
      peers_ipaddr "192.168.1.10" port 500;
      kmp_enc_alg { aes192_cbc; aes128_cbc; 3des_cbc; };
      kmp_prf_alg { hmac_md5; hmac_sha1; aes_xcbc; };
      kmp_hash_alg { hmac_sha1; };
      kmp_dh_group { modp2048; };
      ## Use Preshared Key
      kmp_auth_method { psk; };
      pre_shared_key "${PSKDIR}/${PRESHRD_KEY}";
  };
  selector_index ike_tun_sel_in;
};

selector ike_tun_sel_out {
  direction outbound;
  src "${MY_NET}";
  dst "${PEERS_NET}";
  policy_index ike_tun_policy;
};

selector ike_tun_sel_in {
  direction inbound;
  dst "${MY_NET}";
  src "${PEERS_NET}";
  policy_index ike_tun_policy;
};

policy ike_tun_policy {
  action auto_ipsec;
  remote_index ike_tun_remote;
  ipsec_mode tunnel;
  ipsec_index { ipsec_esp; };
  ipsec_level require;
  peers_sa_ipaddr "192.168.1.10";
  my_sa_ipaddr "192.168.1.9";
};
```