Securing RSVP and RSVP-TE Signaling Protocols and Their Performance Study

by

Jing Zhi, Ph.D in Eng.

A thesis submitted to the Faculty of Graduate Studies and Research in partial
fulfillment of the requirements of the degree of Master of Science
in information and Systems Science

Department of Systems and Computer Engineering
Carleton University
Ottawa, Ontario
K1S 5B6 Canada

August 2004

©Copyright 2004, Jing Zhi

Library and
Archives Canada

Bibliothèque et
Archives Canada

Published Heritage
Branch

Direction du
Patrimoine de l'édition

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-612-97489-8

Our file Notre référence

ISBN: 0-612-97489-8

The author has granted a non-exclusive license allowing the Library and Archives Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

PAGINATION ERROR.

TEXT COMPLETE.

ERREUR DE PAGINATION.

LE TEXTE EST COMPLET.

Abstract

RSVP and RSVP-TE are signaling protocols used to set up paths and/or support certain Quality of Service (QoS) requirements in IP and MPLS-based networks, respectively. This thesis aims to analyze an authentication mechanism for securing the RSVP and RSVP-TE control messages and subsequently study their performances. This design and implementation, using four commonly used hash algorithms - MD5, RIPEMD160, SHA-1, and SHA-256, not only improves security, but also provides useful information from the performance aspect. The time for authenticating the RSVP and RSVP-TE messages depends on the algorithm used, and increases slightly in the order of MD5, SHA-1, RIPEMD160 and SHA-256. The performance of the RSVP-TE with multiple sessions were measured and discussed.

Acknowledgements

By the time that the thesis is finished, I wish to deeply thank the people who gave me support and encouragement that made me accomplish this thesis.

I would like to thank my co-supervisors Professor Chung-Horng Lung and Professor Anand Srinivasan for their advice and guidance during the work. Their patience in reading the thesis and helpful comments are greatly acknowledged.

I am grateful to colleagues Pramod Dhakal, Yi Lei and Xia Xu from the EION Inc. for their valuable discussions and assistance with experimental difficulties.

The support of NCIT and the EION Inc. for providing funding and a real industrial environment for this research is greatly acknowledged.

Finally, many thanks go to my family for all their love, support and encouragement. Special thanks go to my husband, for his support and constant encouragement, and to my son, George Li, for his support, understanding and tolerance.

Table of Contents

LIST OF FIGURES.....	VIII
LIST OF TABLES	X
GLOSSARY	XI
CHAPTER 1 INTRODUCTION	1
1.1 MOTIVATION	1
1.2 CONTRIBUTIONS	3
1.3 THESIS OUTLINE	4
CHAPTER 2 RESEARCH BACKGROUND	6
2.1 SECURITY IN MPLS-BASED NETWORKS	6
2.1.1 <i>An Overview on MPLS Architecture.....</i>	7
2.1.2 <i>Securing an MPLS Core in an MPLS-Based Network.....</i>	8
2.2 RSVP AND RSVP-TE PROTOCOL OVERVIEW	11
2.2.1 <i>RSVP Messages</i>	12
2.2.2 <i>Soft State.....</i>	14
2.2.3 <i>Basic Operations of RSVP-Enabled Routers</i>	15
2.2.4 <i>RSVP-TE Messages and Basic Operations.....</i>	16
2.2.4.1 <i>RSVP-TE Messages.....</i>	16
2.2.4.2 <i>RSVP-TE Basic Operations.....</i>	16
2.3 SECURING RSVP RESOURCE RESERVATION PROTOCOL.....	17
2.3.1 <i>Securing RSVP with SDS/CD Protocol.....</i>	17
2.3.2 <i>Securing RSVP with RSVP-SQoS protocol.....</i>	21
2.3.3 <i>RSVP Cryptography Authentication (RFC 2747).....</i>	23
2.3.4 <i>Pros and Cons of the Existing RSVP Security Protocols.....</i>	25
2.4 MESSAGE AUTHENTICATION ALGORITHMS AND KEY GENERATION.....	25
2.4.1 <i>The Hash Functions.....</i>	27

2.4.1.1 An overview of hash functions	27
2.4.1.2 MD5.....	29
2.4.1.2 RIPEMD160	34
2.4.1.3 SHA-1	36
2.4.1.4 SHA-256.....	38
2.4.2 Key Generation and Management	41
2.5 INFLUENCE OF THE SECURITY PROTOCOL ON NETWORK PERFORMANCE	43
2.5.1 An Overview of IPSec.....	44
2.5.2 The Performance Measurements	46
CHAPTER 3 DESIGN AND IMPLEMENTATION OF THE AUTHENTICATION PROTOCOL IN RSVP AND RSVP-TE	49
3.1 OBJECTIVES.....	49
3.2 BRIEF INTRODUCTION TO THE EION OPEN IP ENVIRONMENT.....	52
3.3 DATA STRUCTURE OF THE INTEGRITY OBJECT	54
3.4 HASH FUNCTION ALGORITHMS	56
3.5 SYMMETRIC KEY GENERATION, DISTRIBUTION AND SYSTEM AUTHENTICATION CONFIGURATION	58
3.5.1 Generating Keys with a Pseudo-Random Number Generator	58
3.5.1.1 Algorithm for Generating Pseudo Random Numbers	58
3.5.1.2 Length of the Key	59
3.5.1.3 Number of Keys in Key Stack	59
3.5.2 Distribution of Keys in Networks.....	59
3.5.3 System Authentication Configuration.....	61
3.6 MESSAGE REPLAY PREVENTION.....	62
3.7 INTEGRATION OF THE INTEGRITY OBJECT IN RSVP/RSVP-TE PROTOCOLS.....	63
3.7.1 RSVP Protocol.....	63
3.7.2 RSVP-TE Protocol.....	65
3.8 SCOPE AND LIMITATIONS OF THE AUTHENTICATION SCHEMES	67
3.8.1 Scope.....	67

3.8.2 Limitations	68
CHAPTER 4 PERFORMANCE ANALYSIS OF AUTHENTICATED RSVP AND RSVP-TE PROTOCOLS IN NETWORKS.....	69
4.1 EXPERIMENT CONDITIONS AND PROCEDURES	69
4.1.1 Simulation Network Topology	69
4.1.2 Procedures and Methodology of Measuring Time.....	70
4.1.3 Analysis of the Measured Time	72
4.2 PERFORMANCE ANALYSIS OF AUTHENTICATING RSVP MESSAGES	76
4.2.1 Space Complexity of the Authentication Protocol	76
4.2.2 Time for Generating Authentication Keys	77
4.2.3 Time for Generating Message Digest	79
4.2.4 Time for Authenticating Path and Resv Messages.....	80
4.2.5 Discussion on Authenticating RSVP Messages.....	82
4.3 PERFORMANCE ANALYSIS OF AUTHENTICATING RSVP-TE MESSAGES.....	83
4.3.1 Time for Authenticating RSVP-TE Path and Resv Messages.....	83
4.3.2 Connection Setup Time for RSVP-TE Messages.....	86
4.3.3 Discussion on Authenticating RSVP-TE Protocols.....	91
CHAPTER 5 CONCLUSIONS AND FUTURE WORK	92
5.1 SUMMARY AND CONCLUSIONS	92
5.2 FUTURE STUDIES	95
REFERENCES	96

List of Figures

Figure 2.1 The RSVP model in hosts and routers [Braden, et al., 1997].....	12
Figure 2.2 RSVP messages and their flow direction [White, 1997].....	12
Figure 2.3 Insider attack, Scenarios 1-1 and 1-2 [Wu, et al., 1999] (■ malicious routers; □ normal route; same in the following graph).....	18
Figure 2.4 Insider attack, Scenarios 2-1 and 2-2 [Wu, et al., 1999].....	19
Figure 2.5 Insider attack, Scenarios 3-1 and 3-2 [Wu, et al., 1999].....	19
Figure 2.6 The <i>Path</i> messages using Phase 1 and Phase 2 (PATH_FB refers to the feedback message corresponding to the PATH message).....	21
Figure 2.7 Data structure of the INTEGRITY object of the RSVP message.....	23
Figure 2.8 The format of padded message.....	31
Figure 2.9 The MD5 main loops.....	32
Figure 2.10 One round of MD5 operations.....	33
Figure 2.11 The block processing of RIPEMD160.....	35
Figure 2.12 Main field of the ESP header.....	45
Figure 2.13 3DES encryption time vs. input blocks and processing power.....	47
Figure 2.14 HMAC-MD5 authentication time vs. input blocks and processing power...	48
Figure 2.15 HMAC-SHA-1 authentication time vs. input blocks and processing power.	48
Figure 3.1 Illustration of the authentication mechanism.....	51
Figure 3.2 Architecture for MPLS network [EION-DESN-0, 2003].....	53
Figure 3.3 The sequence diagram of processing a RSVP <i>Path</i> message.....	64
Figure 3.4 The sequence diagram of processing a RSVP <i>Resv</i> message.....	65
Figure 3.5 The sequence diagram of processing a RSVP-TE <i>Path</i> message	66
Figure 3.6 The sequence diagram of processing a RSVP-TE <i>Resv</i> message.....	66
Figure 4.1 Simulation network topology of the experiment.....	69
Figure 4.2 Time for authenticating <i>Path</i> message as a function of measuring times.....	72
Figure 4.3 Frequency distribution of the time authenticating a <i>Path</i> message and the fitting with Gaussian normal distribution.....	76
Figure 4.4 Time for generating secret keys as a function of number of keys.....	78
Figure 4.5 Time for generating secret keys as a function of length of keys.....	78

Figure 4.6 Time measured for encoding RSVP Path message (denoted as msg) and time for generating message digest as a function of different hash algorithms.....	80
Figure 4.7 Time for encoding and decoding a RSVP <i>Path</i> message (for R1 for comparison with the same CPU).....	81
Figure 4.8 Time for encoding and decoding a Resv message (From R1 for comparison with the same CPU).....	82
Figure 4.9 Time for encoding and decoding RSVP-TE Path message with different algorithms.....	84
Figure 4.10 The variation of the time for encoding and decoding RSVP-TE Path messages with the number of sessions using MD5.....	85
Figure 4.11 Connection setup time of RSVP-TE as a function of the number of sessions.....	87
Figure 4.12 Average connection setup time of RSVP-TE as a function of the number of sessions with confidence interval for each point	88
Figure 4.13 Average connection setup time of RSVP-TE as a function of number of sessions for authentication algorithm of RIPEMD160.....	89
Figure 4.14 Average connection setup time of RSVP-TE as a function of number of sessions for authentication algorithm of SHA-1.....	89
Figure 4.15 Average connection setup time of RSVP-TE as a function of number of sessions for authentication algorithm of SHA-2.....	90

List of Tables

Table 2.1 Security schemes of routing protocol in IP networks.....	9
Table 2.2 The fields in the data structure of INTEGRITY object in RSVP message.....	24
Table 2.3 Comparisons of SDS/CD, RSVP-SQoS and hop-by-hop (HMAC) security protocols.....	26
Table 2.4 One-way hash functions.....	30
Table 2.5 Performance of optimized assembly language implementations of MD4-like hash functions on a 90 MHz Pentium using a 32-bit flat memory model (i.e., running in native protected mode) [Bosselaers, 2004].....	41
Table 2.6 Security requirements for different information [Schneier, 1996].....	43
Table 2.7 IPSec AH for IP datagram.....	44
Table 2.8 IP ESP for IP datagram.....	46
Table 2.9 IPSec header field sizes (bytes).....	46
Table 3.1 The implementation and copyright status of hash function algorithms.....	57
Table 3.2 Command parameters of the CLI for setting key parameters.....	60
Table 4.1 System parameters of the computers that run the router program.....	70
Table 4.2 Definitions of different time measurements.....	71
Table 4.3 Classification of Components into layers.....	71
Table 4.4 Standard error analysis of the time measured.....	74
Table 4.5 Frequency distribution of the experiment data.....	75
Table 4.6 Sizes of different fields in Integrity Object.....	77
Table 4.7 The time ratio of authentication mechanism in RSVP-TE message for different sessions.....	90

Glossary

3DES	- The triple Data Encryption Standard
AH	- Authentication Header
ANSI	- American National Standards Institute
ATM	- Asynchronous Transfer Mode
BGP	- Border Gateway Protocol
CBC	- Cipher Block Chaining mode
CLI	- Command Line Interface
CR-LDP	- Constraint-based Routing Label Distribution Protocol
DNU	- Degradation of Network Utilization
DQoS	- Denial of QoS Request
ESP	- Encapsulating Security Payload
FIPS	- Federal Information Processing Standards
FR	- Frame Relay
HMAC-MD5	- Keyed Hashing for Message Authentication with MD5
HMAC-SHA-1	- Keyed Hashing for Message Authentication with SHA-1
IETF	- Internet Engineering Task Force
IP	- Internet Protocol
IPsec	- Internet Security Protocol
LDP	- Label Distribution Protocol
LSP	- Label Switch Path
LSR	- Label Switch Routers
MAC	- Message Authentication Code
MD	- Message Digest
MIPS	- Millions of Instructions Per Second
MPLS	- Multiprotocol Label Switching
NIST	- National Institute of Standards and Technology
NTP	- Network Time Protocol
OSI	- International Standards Organization
OSPF	- Open Shortest Path First

PATH_FB	- Feedback information of <i>Path</i> message
QoS	- Quality of Service
RIP	- Routing Information Protocol
RIPEMD	- RACE Integrity Primitives Evaluation - Message Digest
RCV	- the Receiver
RFC	- Request for Comments
RQoS	- Reserved QoS Degradation
RSVP	- Resource Reservation Protocol
RSVP-SQoS	- Resource Reservation Protocol – Securing QoS
RSVP-TE	- Resource Reservation Protocol with Traffic Engineering
SDS/CD	- Selective Digital Signature with Conflict Detection
SHA	- Secure Hash Algorithm
SHS	- Secure Hash Standard
SPI	- Security Parameter Index
TCP/IP	- Transmission Control Protocol/Internet Protocol
UR	- Unnecessary/Suboptimal Reservation
VPN	- Virtual Private Network

Chapter 1 Introduction

This chapter provides a brief introduction of motivations, contributions of the thesis, along with the organization of this thesis.

1.1 Motivation

In recent years, computer networks have had a great impact on the way that people communicate with each other and a variety of new services have been set up. With the emergence of the distributed multimedia applications such as audio/video conferencing, the traditional networks based on the best-effort service delivery model could not satisfy the requirements of the real-time performance needed by these applications [Neogi and Chiueh, 1999; Barzilai and Kandlur, 1998]. A general approach to these applications is to provide end-to-end resource reservation. Resource ReSerVation Protocol (RSVP) [Zhang, et al., 1993] provides one of the solutions in IP networks by setting up network bandwidth reservation. It is used for a host to request a certain Quality of Service (QoS) from the network for a particular application, and for routers to deliver the QoS to the nodes in the network that the data flows [Braden, et al., 1997]. Coupled with traffic control module, the RSVP guarantees service quality of specific IP network applications throughout their lifetime. RSVP-TE is an extension of the RSVP protocol, which is used in MultiProtocol Label Switching-based (MPLS-based) networks to establish and maintain explicitly routed Label Switched Paths (LSPs). MPLS has received tremendous attention in both industry and academia for its ability to support traffic engineering, protection and restoration, and virtual private networks.

Along with the increased connectivity and new services, computer networks have also allowed technically advanced intruders opportunities to carry out a variety of attacks that threaten the integrity of its infrastructure and violate the privacy of its users [Wang, 1994; Molva, 1999]. Corrupted or spoofed reservation requests can lead to theft of services by unauthorized parties or denial of services. In recent years, several research groups have been looking for way of securing the RSVP messages. Different security protocols are proposed [Wu, et al., 1999; Talwar, et al., 2001]. The RSVP protocol is embedded with the following security features [Talwar, et al., 2001]:

- Message Integrity and router (node) authentication;
- User Authentication;
- Non-repudiation;
- Confidentiality;
- Replay;
- Traffic Analysis; and
- Denial of Service.

Current efforts to make the RSVP protocol more secure are not only far from complete, but also not yet thorough. Since the specification of the RSVP protocol proposed by Zhang, et al. [Zhang, et al., 1993] became an IETF standard in 1997 [Braden, et al., 1997], less attention has been paid to this area. To the author's knowledge, no such work has yet been reported for RSVP-TE. This thesis will investigate ways to secure

RSVP in IP networks and RSVP-TE in the MPLS framework in the EION Open IP Environment [EION/CSP-00, 2000].

The security services with RSVP assure the integrity and authenticity of services in networks. However, the overhead required by the RSVP message flow, the traffic scheduling procedure and its security mechanisms actually degrade the performance of the network [Barzilai, et al., 1998; Neogi and Chiueh, 1999]. Tests showed [Neogi and Chiueh, 1999] that RSVP's control messages do not incur significant overhead in terms of processing delay and bandwidth consumption, but the overhead of the real-time packet scheduling is noticeable in the presence of a large number of real-time connections. Accordingly, it is expected that securing the RSVP messages may also bring the overhead and performance side effects to the network. However, to the author's knowledge, presently there is no report on the influence of security protocols on the performance of RSVP and RSVP-TE.

This work aims to implement the integrity and authentication of the RSVP and RSVP-TE messages using a one-time-use sequence number, authentication key and keyed hash algorithms. The influence of the security protocol on the performance of RSVP and RSVP-TE protocols will be discussed in terms of time and space complexity.

1.2 Contributions

The main contributions of the thesis are as follows:

- Design and implement an authentication mechanism for both RSVP and RSVP-TE signaling protocols based on RFC2747 [Baker, et al., 2000]. The mechanism

helps the study and comparison of the security issues of the signaling protocols in both IP and MPLS networks.

- Design and implement the EION Command Line Interface (CLI) for all MPLS protocols, including CLIs for RSVP, RSVP-TE protocols and the authentication mechanism. The CLIs facilitate network configurations and efficiently support the realization of the various network scenarios for experiments.
- Divide four commonly used hash function algorithms, MD5, RIPEMD160, SHA-1 and SHA-256, into functional components and empirically study their performance. This study facilitates the comparison between different algorithms.
- An Empirical study of the performance of the authenticated signaling protocols - RSVP and RSVP-TE in terms of time and space complexity. The study enables quantitative understanding of two most commonly used signaling protocols.
- Examine the security assurance issues in MPLS-based networks. The examination covers the potential threats of the MPLS networks.

1.3 Thesis outline

The thesis is organized as follows:

- Chapter 2 reviews the research background and the related work done for RSVP in IP networks and RSVP-TE in MPLS networks; summarizes authentication mechanisms and a variety of authentication algorithms; presents key management techniques, although this is not the focus of this thesis.

- Chapter 3 details the design and implementation of the authentication mechanism in securing the RSVP-enabled and RSVP-TE-enabled router; and analyzes the scopes and limitations of the authentication protocol.
- Chapter 4 shows the performance measurements of the authenticated RSVP and RSVP-TE protocols.
- Chapter 5 presents the summary, conclusions; and discusses areas for future work.

Chapter 2 Research Background

RSVP is a resource reservation protocol used in IP network. Its extension, RSVP-TE, is a signaling protocol used in MPLS-based networks. This chapter reviews the research background and the related work conducted for RSVP and RSVP-TE and their security; summarizes the authentication mechanisms and a variety of authentication algorithms; and briefly describes key generation techniques.

2.1 Security in MPLS-Based Networks

According to International Standards Organization (ISO), the basic network architecture is composed of 7 layers – the physical layer, data link layer, network layer, transport layer, session layer, presentation layer and application layer. In consideration of the security issue, different security mechanisms have been exploited to various layers.

Multiprotocol Label Switching (MPLS) protocol is a recently developed technology that offers many additional services to applications using IP [Rosen, et al., 2001; Awduche, et al., 1999]. It exploits a label swapping mechanism with network layer routing, which requires separate design of signaling and routing protocols. Especially, nowadays, with the increasing deployment of VPNs, MPLS-based services are in great demand. Hence the security of the MPLS architecture is an important research topic.

For the security of the MPLS architecture itself, less research has been undertaken to date. The existing opinions are of diverse. However, the following are some of the general concerns:

- Are there big holes that could lead to a security compromise in MPLS?
- How can confidentiality, integrity and authentication be provide by MPLS if it is not used with IPsec?
- How can MPLS implementation influence the current level of security?
- Do the MPLS-based VPNs provide the same security level as Layer 2 VPNs?
- How can MPLS core infrastructure be secured in MPLS-based networks?

In this section, not all of the questions will be answered. Instead, I will mainly examine the security of the MPLS-based core networks.

2.1.1 An Overview on MPLS Architecture

The MPLS architecture relies on the concepts of Label Switching Routers (LSRs), Label-Switched Paths (LSPs), and labeled packets, where LSRs forward labeled packets on LSPs. It provides a framework for efficient signaling, routing, forwarding, and switching of traffic flow through the network.

MPLS is used to perform the following functions [IEC, 2004]:

- Manage traffic flow of various applications by specifying different mechanisms.
- Maintain independence of the Layer-2 and Layer-3 protocols.
- Map IP addresses to simple, fixed-length labels used by different packet-forwarding and packet-switching technologies.
- Interface with existing signaling and routing protocols, such as, resource reservation protocol (RSVP) and open shortest path first (OSPF).

In an MPLS network, data is transmitted on label-switched paths (LSPs) that consist of a sequence of labels at every node along the path from the source to the destination. LSPs are established in two ways: control-driven or data-driven. The labels are distributed using label distribution protocol (LDP) or signaling protocol RSVP-TE or piggybacked onto routing protocols like border gateway protocol (BGP) or OSPF. Each data packet encapsulates and carries the labels during transmission from source to destination. Switching of data could be faster in MPLS because the fixed-length labels are inserted at the beginning of a packet or a cell. MPLS has addressed current network backbone requirements effectively to meet a diversity of needs for Internet users.

2.1.2 Securing an MPLS Core in an MPLS-Based Network

The security of the MPLS core is of great concern and ways of enhancing security have been proposed [Cisco, 2003; Molva, 1999]. The methods of securing the MPLS core are as follows:

- Securing management

This includes the issues of access control, secure configuration management, and storage [Cisco, 2003];

- Routing authentication

Routing protocols are responsible for maintaining network connectivity for the TCP/IP traffic. Attacks on routing protocols can cause legitimate traffic to flow on unsecure paths and various types of security exposure may happen such as eavesdropping and denial of service. The routing protocols include Routing Information Protocol (RIP), Open Shortest Path First (OSPF), and Border Gateway Protocol (BGP) [Molva, 1999;

Papadimitratos and Haas, 2002; Kent and Atkinson, 1998]. The security schemes of the routing protocols are briefly summarized in the following table.

Table 2.1 Security schemes of routing protocol in IP networks

Security scheme	Routing Protocols		
	RIP	OSPF	BGP
Authentication (A)	RIPv2 Password-based A(MD5) ^[a]	OSPFv2 password-based or cryptographic A(MD5) ^[a]	Password-based MD5 digest TCP option ^[b]
Identification	-	Digital signature ^[a,b]	Digital signature ^[b]
Non-repudiation	-	MAC ^[b]	-
Integrity	-	MAC ^[b]	-
Authentication	IPsec ^[c]		
Integrity			
Replay detection			
Encryption			
Intrusion detection	Tools of intrusion detection systems; Traffic counter		

[a] – Molva, 1999;

[b] – Papadimitratos and Haas, 2002

[c] – Kent and Atkinson, 1998

In MPLS core networks [Osborne and Simha, 2003], network layer routing has two functional components: control and forwarding. The control component is for construction and maintenance of the forwarding table. Each router has one or more routing protocols (e.g. OSPF, BGP) to exchange routing information and procedures to convert routing information to a forwarding table. The forwarding component actually

forwards labeled packets from input to output across a switch or a router. Thus the security issue of the OSPF and other routing protocols used in IP network also affects the security of the MPLS network. In addition, since the forwarding table and the labels carried in the packet are two sources of information for MPLS forwarding, the security of the forwarding table and the labels are also of concern in MPLS-based networks. As shown in Table 2.1, using MD5 authentication and IPSec were proposed for the routing protocols manipulating these tables [Molva, 1999; Papadimitratos and Haas, 2002].

- LDP authentication

LDP is a Label Distribution Protocol for MPLS-based networks. It can be secured with MD5 authentication across the MPLS cloud to prevent hackers from introducing bogus routers [Cisco, 2003].

- Signaling protocol authentication

MPLS-based networks uses two signaling protocols: Constraint-based Routing Label Distribution Protocol (CR-LDP) and Reservation Protocol with Traffic Engineering (RSVP-TE). Both of these protocols are essential to traffic engineering. To date, little research on its authentication mechanism has been reported, especially for RSVP-TE.

- IPsec over MPLS

MPLS itself does not provide encryption, integrity or authentication services [Cisco, 2003]. To secure the MPLS core, IPsec is used on top of the MPLS infrastructure to:

- Encrypt part or all traffic.
- Authenticate the endpoints.

- Integrate the traffic, without changing the route through the core, and thus hindering bogus packets.
- Detect replay of messages.

The MPLS networks provide a high level of security, comparable to ATM or FR networks. A combination of security measures, such as security management, routing authentication, signaling authentication and IPsec with MPLS, will enhance the security of the MPLS-based network. In this thesis, only RSVP and RSVP-TE are discussed.

2.2 RSVP and RSVP-TE Protocol Overview

RSVP protocol was first proposed in 1993 [Zhang, et al., 1993]. Due to its usefulness, much work has been done since the Internet Engineering Task Force (IETF) standardized RSVP as Request for Comments (RFC) 2205, released in 1997 [Braden, et al., 1997].

RSVP is a signaling protocol used by a host to request specific QoS from the network for data flow on the Internet and by a router to deliver QoS requests to all nodes along the path(s) of the data flow. It is also used for routers to establish and maintain the requested service. The basic RSVP model in hosts and routers is shown in Figure 2.1.

The QoS parameters, such as bandwidth, are processed by a traffic control module that includes: (1) a packet classifier, (2) admission control, and (3) a packet scheduler. After being analyzed and processed by the traffic control module, the reserved information such as bandwidth and buffer space is sent to RSVP for processing reservation.

Figure 2.1 The RSVP operational model in hosts and routers [Braden, et al., 1997]

2.2.1 RSVP Messages

RSVP is receiver-oriented; the receiver of the data flow is responsible for the initiation of the resource reservation. Figure 2.2 illustrates the RSVP message flow during reservation setup.

Figure 2.2 RSVP messages and their flow direction [White, 1997]

This example shows a multicast session that involves one sender, S1, and three receivers RCV1 – RCV3. There are mainly six types of RSVP messages, as shown in Figure 2.2: *Path*, *PathTear*, *PathErr*, *Resv*, *ResvTear*, and *ResvErr*. Of the six message types, *Path* message, which originates from the traffic sender, and *Resv* message, which originates from the traffic receiver are the two fundamental message types used in RSVP.

The formats of both messages are briefly described below (RFC 2205) [Braden, et al., 1997].

- *Path* Message

The primary role of the *Path* message is to install a reverse routing state in each router along the path. The *Path* message also provides receivers with information about the sender traffic and the end-to-end path to make appropriate reservation requests [Braden, et al.]. The format of a *Path* message is as follows:

<Path Message> :: = <Common Header> [<INTEGRITY>] <SESSION>

<RSVP_HOP> <TIME_VALUE> [<POLICY_DATA> ...] [<sender descriptor>]

where the Common Header, and the objects such as INTEGRITY, and SESSION, are specified in RFC 2205, and will not be repeated here. The thesis is focused on the INTEGRITY object, which is used for an authentication to check the integrity and authenticity of the RSVP *Path* messages.

Periodically, the RSVP process at a router scans the path state to create a new *Path* message to be forwarded to the neighboring router along a path.

- *Resv* Message

The primary role of the *Resv* message is to carry reservation requests to the upstream routers along the path from the receiver to the sender. The format of a *Resv* message is as follows:

<Resv Message> ::= *<Common Header>* [*<INTEGRITY>*] *<SESSION>*

<RSVP_HOP> *<TIME_VALUE>* [*<RESV_CONFIRM>*] [*<SCOPE>*]

[*<POLICY_DATA>* ...] *<STYLE>* *<flow descriptor list>*

where the Common Header, and the objects such as INTEGRITY and SESSION., are specified in RFC 2205, and will not be repeated here. Similar to the *Path* message, the INTEGRITY object ensures the integrity and authenticity of the RSVP *Resv* message.

2.2.2 Soft State

In RSVP [Braden, et al., 1997], the soft state, which was first used by Clark [Clark, 1988], is used to adapt the resource reservation to changes in network routes, multicast group membership, and occasional link and node failures. The reservation state is set up through timely refreshes between senders and receivers for the reservation connection. That is, the sender first periodically sends a *Path* message to establish and update the path state; and then the receiver periodically sends a *Resv* message to establish and update the reservation state. In accordance with the changes in network, the reservation state will be automatically invalidated after a period of time unless the *Resv* signal is refreshed.

The strategy of the reservation setup characterizes the simplicity in management of the reserved resources and recycling of the resource utilization. However, the periodical

refreshing of the control messages of *Path* and *Resv* between both senders and receivers results in additional overhead to the network [Neogi and Chiueh, 1999].

2.2.3 Basic Operations of RSVP-Enabled Routers

As shown in Figure 2.1, the traffic-control module implements QoS service models defined by the Integrated Services Working Group. Basically, a reservation request initiates the following two general actions:

- Make a reservation on a link

During a reservation setup, an RSVP QoS request from the application is first passed to two local decision modules – Admission Control and Policy Control (see Figure 2.1). The admission control decides whether the intermediate router has enough resources for the request. The policy control determines whether the user has administrative permission to make the reservation. If both conditions are satisfied, QoS parameters are set in the packet classifier and in the link layer interface (for example, in the packet scheduler) to get the desired QoS. If either fails, the RSVP returns an error message to the application process that started the request. RSVP itself transfers and maintains the QoS and policy control parameters as opaque data, delivering them to the appropriate traffic control and policy control modules for interpretation.

- Forward the request upstream

A reservation request is guided upstream along the reversed route to reach the sending router. The reservation request may differ from the request received from downstream for two reasons:

- The traffic control module may modify the QoS parameter hop-by-hop; and
- Reservations from multiple downstream branches must “merge” as reservations travel upstream.

2.2.4 RSVP-TE Messages and Basic Operations

RSVP-TE, the “Extensions to RSVP for LSP Tunnels,” is a signaling protocol that adds a number of extensions to the original RSVP to support MPLS traffic engineering applications. The key point is that RSVP-TE supports the establishment and maintenance of explicitly routed label switched paths (LSPs), which could be routed away from network failures and congestion. It also supports resource reservations along the path to meet the requirements of traffic flow.

2.2.4.1 RSVP-TE Messages

RSVP-TE supports all RSVP messages, but adds new objects to *Path* messages and *Resv* messages. Explicit route and label request objects are added to *Path* message to support explicitly routed LSP tunnel setup. Record route object is added to both *Path* and *Resv* message to support loop detection and allow network administrators to observe actual route traversed by an established LSP tunnel. Session attribute object is added to *Path* message to support LSP preemption and smooth reroute. A label object is added to *Resv* message to support downstream-on-demand label distribution and binding.

2.2.4.2 RSVP-TE Basic Operations

The RSVP-TE signaling protocol model is used to set up LSP tunnels [Awduche, et al., 2001]. It uses downstream-on-demand label distribution. First, a request to bind labels

to a specific LSP tunnel is initiated by an ingress node through the RSVP *Path* message. For this purpose, the RSVP *Path* message is added with a LABEL_REQUEST object. Labels are allocated downstream and distributed upstream by means of the RSVP *Resv* message. Thus, the RSVP *Resv* message is extended with a special LABEL object. The procedures for label allocation, distribution, binding, and stacking are described in IETF RFC 3209 [Awduche, et al., 2001].

Using RSVP-TE to establish LSP tunnels enables the allocation of resources along the path. For example, a bandwidth can be allocated to an LSP tunnel with standard RSVP reservations and the Integrated Services service classes.

2.3 Securing RSVP Resource Reservation Protocol

There are three main security concerns for RSVP: 1) Message integrity and node authentication; 2) User authentication; 3) Secure data stream [Braden, 1997]. In the following, a summary of the message integrity and node authentication concern is presented, including security protocols of SDS/CD [Wu, et al., 1999], RSVP-SQoS [Talwar, et al., 2001] and hop-by-hop authentication. The latter two concerns (User authentication and Secure data stream) are beyond the scope of this thesis.

2.3.1 Securing RSVP with SDS/CD Protocol

Wu et al. [Wu, et al., 1999] have performed a security analysis on RSVP. The key vulnerabilities of the distributed resource reservation setup process in RSVP were described. Possible threats were found. According to Wu et al., four target attacks were classified, based on the assumption that at most one router is malicious at a time:

- Unnecessary/Suboptimal Reservation (UR): the reserved resources are different from what the user wants.
- Degradation of Network Utilization (DNU): due to interference from the attacker, the network resources can not be fully utilized.
- Reserved QoS Degradation (RQoSD): the attacker has unauthorized use of the resources reserved successfully.
- Denial of QoS Request (DQoSR): the attacker is able to intercept or drop all or part of the reservation message so that the reservation setup can fail or be maliciously delayed.

Wu et al. primarily dealt with intruders that are malicious routers on the reservation path. These intruders were classified into two categories: *Insider_{RSVP}*, an RSVP-enabled router on the reservation path, and *Outsider_{RSVP}*, an RSVP-disabled router also on the reservation path. The following attack scenarios may lead to the attack on RSVP (see Figure 2.3 – 2.5):

Scenario 1-1

Scenario 1-2

Figure 2.3 Insider attack, Scenarios 1-1 and 1-2[Wu, et al., 1999] (■ malicious router; □ normal router; same in the following graphs)

Scenario 2-1

Scenario 2-2

Figure 2.4 Insider attack, Scenarios 2-1 and 2-2 [Wu, et al., 1999]

Scenario 3-1

Scenario 3-2

Before Attacked

After Attacked

Figure 2.5 Insider attack, Scenarios 3-1 and 3-2 [Wu, et al., 1999]

In scenarios 1-1 and 1-2, shown in Figure 2.3, the TSpec (*Path*) parameter was modified. This led to the wrong decision of the receiver. If the attacker converts TSpec to a lower value, the QoS that the receiver expects will be degraded; on the other hand, if TSpec is maliciously changed into a higher value, the reservation might be discarded due to local policies because the service appears to be more expensive. The service utilization of the network will also be degraded. In scenarios 2-1 and 2-2, shown in Figure 2.4, the parameter AdSpec (*Path*) was modified. This parameter is to be passed to the local traffic control module for AdSpec (*Path*) update. The change led the receiver to reserve either a higher or lower level QoS. In scenarios 3-1 and 3-2, shown in Figure 2.5, parameters Rspec (*Resv*) and TSpec (*Resv*) were modified. An unauthorized RSVP-enabled router increase or decrease change in parameter may result in incorrect reservation and QoS degradation.

In order to protect the RSVP parameters, Wu et al. proposed secure RSVP protocol with Selective Digital Signature and a Conflict Detection (SDS/CD) mechanism. The main features of the SDS/CD mechanism is shown as below, on the assumption that no malicious routers cooperate with each other [Wu, et al., 1999]:

- Sender, A, selectively and digitally sign the TSpec (*Path*) of the *Path* message with its private key.
- Upon receiving a *Path* message, the receiver, B, verifies the integrity of the TSpec (*Path*) using the sender's public key, and then makes resource reservations accordingly.
- The receiver B also sends back a message piggybacked with the AdSpec (*Path*) that it just received. At this time, both the AdSpec (*Path*) and the Rspec (*Resv*) in the *Resv* message are selectively and digitally signed using B's private key.

- Upon receiving the RSVP message, the intermediate router C (see Figure 2.4) verifies whether the piggybacked AdSpec (*Path*) is equivalent to or lower than the last AdSpec (*Path*) that it forwarded to B. If they are different, a conflict is identified and the local Policy Decision Point (PDP) is notified. Then, the PDP will determine whether the message will be dropped.

As shown in Figures 2.3-2.5, the SDS/CD protocol can successfully detect intruding routers and check the integrity of the resource reservation message.

2.3.2 Securing RSVP with RSVP-SQoS protocol

Talwar et al. [Talwar, et al., 2001] designed a secure RSVP protocol, RSVP-SQoS, to secure RSVP transmission of QoS parameters in three phases. Figure 2.6 indicates a network that consists of subnetworks, S_k ($k=1, 2$, and 3).

Figure 2.6 The Path messages using Phase 1 and Phase 2 (*PATH_FB* refers to the feedback message from the neighboring receiver router to the sender router)

Definitions of the three phases: Phase 1, Phase 2, and Phase 3 can be briefly summarized as follows:

Phase 1 is an intra-domain PATH protocol, where the Sender, A, digitally signs the *Path* messages, attaches the timestamps, and then sends the message towards E_1 , within the same subnetwork of S_1 . E_1 then sends back a $PATH_FB_1$ to its upstream neighbor. The sender router, on receiving $PATH_FB_1$, verifies whether the AdSpec equals the one forwarded downstream. Upon identifying a conflict, it will send an alarm to the local policy decision point.

For a receiver, for example E_1 , it will check the $PATH_FB_2$ to see whether the AdSpec equals the one forwarded last time. In case of a conflict, it will raise an alarm to the local policy decision point.

Phase 2 is called an inter-domain PATH/RESV protocol, which uses keyed hashing to authenticate the *Path* and *Resv* messages with a secret key mechanism at (ingress, egress) pair of nodes between two subnetworks.

Phase 3 is designed as an intra-domain RESV protocol. The RSVP parameters were digitally signed and verified with a mechanism similar to the one for *Path* messages.

In this securing mechanism, any core router has a chance to compare whether its history is consistent with the local observation of the $PATH_FB$ message. In addition, it verifies the consistency of the digital signature with the *Path* message. So, any malicious tempering by some core router will be detected. On the other hand, if an ingress router is tempered, it will be detected when the corresponding feedback message reaches egress of the previous subnetwork. For the egress tempering, it will be detected when the

corresponding feedback message reaches the node just before the egress. The unauthorized router can only be undetected when the routers in a subnetwork cooperate with each other, which is unlikely.

2.3.3 RSVP Cryptography Authentication (RFC 2747)

A security scheme with cryptographic authentication for securing RSVP signaling protocol was defined [Zhang, et al. 1993] and accepted as RFC 2747 by IETF in 1997 [Braden, et al., 1997].

The main purpose of the scheme is to transmit an INTEGRITY object in the RSVP message, where an authenticating digest of the message is computed with a secret authentication key and a keyed-hash algorithm. This scheme offers protection against forgery or message modification. In addition, the one-time use of a sequence number tagged with the INTEGRITY object allows the message receiver to identify playback and, thus, prevent replay attacks. This scheme does not provide confidentiality, and the reasons for this will be explained in Chapter 3.

The format of RSVP's INTEGRITY object to provide hop-by-hop integrity and authentication of RSVP message is described below. Figure 2.7 shows the data structure of the INTEGRITY object of the RSVP message.

Flags	0 (Reserved)	
Key Identifier		
Sequence Number		
Keyed Message Digest		

Figure 2.7 Data structure of the INTEGRITY object of the RSVP message.

The data structure of each field shown in Figure 2.7 is summarized in Table 2.2

Table 2.2 The fields in the data structure of INTEGRITY object in RSVP message

No.	Field	Meaning
1	Flag	Enable/Disable integrity handshake message
2	Reserved	Equals 0
3	Key Identifier	48-bit number, unique for a given sender; used to identify the key number
4	Sequence Number	Unsigned 64-bit number monotonically increasing, unique
5	Keyed Message Digest	The message authentication code with cryptographic keys

It was suggested that the Key Identifier could be the combination of the address (either IP or MAC) of the sending interface and a key number. The sequence number can be any monotonically increased sequence. There are two important properties that MUST be satisfied by the generation procedure: 1) Unique or one-time, for the lifetime of the integrity key. A receiver can distinguish between the new and replayed message. 2) The sequence number monotonically increases, modulo 2^{64} . The generation of the sequence number can be done using one of the following schemes:

- Simple sequence number
- Sequence number based on a Real Time Clock
- Sequence number based on a Network Recovered Clock

The implementation of the Keyed Message Digest was not specified in the RFC 2747, besides a suggestion of HMAC-MD5 or HMAC-SHA-1.

2.3.4 Pros and Cons of the Existing RSVP Security Protocols

The current RSVP security protocols, described in sections 2.2.1, 2.2.2, and 2.2.3, have contributed to the securing of the RSVP protocol. Each approach has Pros and Cons. Table 2.3 summarizes a comparison of the security protocols/schemes. Although the security issue of RSVP has been raised long before, more studies are needed in this field.

2.4 Message Authentication Algorithms and Key Generation

The RSVP message integrity and authentication use cryptographic mechanism and its security is based on the underlying cryptographic algorithms. Cryptographic hash functions are one of the important tools in cryptography for applications such as digital signatures of messages, message authentication, and key derivation. They are characterized by simplicity, public availability, and fast operation.

The hash functions in the real world are built on *compression function*. The inputs to the compression function are a message block and the output of the previous blocks of text. That is,

$$h_i = f(M_i, h_{i-1}) \quad (\text{Eq. 2-1})$$

where M_i is the message block and h_{i-1} is the output of the previous block of text.

This hash value h_i , along with the next message block, becomes the next input of the compression function. The hash of the entire message is the hash of the last block.

In this thesis, a variety of hash functions, such as MD5, RIPEMD160, SHA-1 and SHA-256 were selected to implement message authentication of RSVP messages.

The different hash algorithms mentioned above will now be reviewed and analyzed.

Table 2.3 Comparisons of SDS/CD, RSVP-SQoS and hop-by-hop (HMAC) security protocols

Securing Protocol of RSVP message			
RSVP-SQoS		SDS/CD	HMAC
Possible attacks	Spoofing, modification, replay, malicious injecting	Denial of Quality of network service: 4 types	Spoofing, modification, replay, Data origin authentication
Security mechanisms	Phases 1&3: digital signature	Selective digital signature and	HMAC
	Phase 2: HMAC	conflict detection	Replay detection
	Replay detection Conflict detection		Data origin authentication
Factors influencing time complexity	Digitally signed message	Selectively digitally signed message	Key management
	Integrity hash function		Hash function
	Timestamp verification		Timestamp verification
	Feedback message verification		
Features/Pros	Flexible	Handling different scenarios of attacks	Protection from outsider attack
	Protection from both insider and outsider attack		Hop-to-hop protection
	More scalable than SDS/CD	Protection from insider attack	Integrity handshake that ease the restart or recovery from a failure
Limitation/Cons	Overhead larger than SDS/CD	Cannot handle dropping attack	Overhead larger than RSVP-QoS

2.4.1 The Hash Functions

2.4.1.1 An overview of hash functions

A hash function H is a transformation that takes a variable-size input (m) and returns a fixed-size string, which is called the hash value (h). That is,

$$h = H(m) \quad (\text{Eq.2-2})$$

Hash functions have a variety of general computational uses, however, when employed in cryptography, the hash functions are usually chosen to have some additional properties.

The basic requirements for a cryptographic hash function are:

- The input can be of any length,
- The output has a fixed length,
- $H(x)$ is relatively easy to compute for any given x ,
- $H(x)$ is one-way, and
- $H(x)$ is collision-free.

A hash function H is said to be one-way if it is hard to invert, where "hard to invert" means that given a hash value h , is computationally unfeasible to find an input x such that $H(x) = h$.

If, given a message x , it is computationally unfeasible to find a message y not equal to x such that $H(x) = H(y)$, then H is said to be a weakly collision-free hash function.

A strongly collision-free hash function H is one for which it is computationally unfeasible to find any two messages x and y such that $H(x) = H(y)$.

The first constructions for hash functions (e.g. SNEFRU) were based on block cipher (such as DES) [Merkle, 1990]. However, their software performance was poor since they are 2 to 4 times slower than the corresponding block cipher. The most popular hash functions currently used in a wide variety of applications are from the MD4-family. The cryptographic hash function MD4 was first made public in 1991 [Rivest, 1991] and later accepted as an RFC [Rivest, 1992a]. Its main contribution is to make optimal use of the structure of 32-bit processors. The use of serial operations and the favorable treatment of little-endian architectures show that MD4 is tuned towards software implementations. Due to the vulnerabilities of MD4 (namely collisions in two rounds out of three) [Boer and Bosselaers, 1992; Dobbertin, 1996], independent research was carried out to develop strengthened version of MD4; so came the MD5 [Rivest, 1991, Rivest, 1992b] and RIPEMD [Bosselaers and Preneel, 1995]. MD5 is slightly slower than MD4, but it was quickly implemented into products and became probably the most widely used hash function, in spite of the fact that the compression function of MD5 is not collision resistant [Boer, 1992]. Research shows that although MD5 violates one of the design principles – collision resistance, this does not pose a threat for standard applications of MD5 [Dobbertin, 1996; Ragab and Ismail, 2001]. An alternative for MD5 is the Secure Hash Algorithm (SHA-1) designed by NSA and published by NIST [FIPS, 1995]. The weakness of SHA-1 was subsequently removed with improved versions of SHA-256, SHA-512, etc. with longer digest lengths.

The proposed RIPEMD was developed in the framework of the EU project RIPE (RACE Integrity Primitives Evaluation). It is a strengthened version of MD4 that essentially consists of two parallel versions of MD4 [Dobbertin, et al., 1996]. RIPEMD

had collisions in two out of three rounds [Dobbertin, 1997]. Strengthened RIPEMD160 was designed in 1996 to remove this weakness in RIPEMD [Dobbertin, et al., 1996].

All these algorithms are based on MD4 and were regarded as MD4-family algorithms. Table 2.4 summarizes the general features of the hash functions in the MD4-family.

Since these MD4-family algorithms are based on the MD4 algorithm, the designs are basically similar. A comparative description of some commonly used algorithms such as MD5, RIPEMD160, SHA-1 and SHA-256 will be given in the following sections.

The description consists of three critical components and sub-components as follows:

- Description of an algorithm;
 - Padding and initialization of chaining variables;
 - One-way hash functions;
 - Block processing;
- Security issue; and Performance issue.

2.4.1.2 MD5

MD5 is an improved version of MD4 [Rivest, 1992b]. It produces 128-bit hash values and is similar in design to MD4 [Rivest, 1991].

1. Description of MD5

There are 3 main steps for the message digest processing for the MD5 algorithm. Each step is briefly described as follows.

(i) Padding and Initialization of Chaining Variables

Table 2.4 One-way hash functions

Name	Designer(s)	Version	Block size (bit)	Digest size (bit)	Round	Cryptanalysis	References
SNEFRU	Merkle	1990	512-Digest size	128 256	—	Biham, Shamir, 1993	Schneier, 1996
N-Hash	Miyaguchi, Ohta, Iwata	1990	128	128	n	Biham, Shamir, 1991	Miyaguchi, et al., 1990
MD2	Rivest	1989	512	120	4	Rogier, Chauvaud, 1995	Kaliski, 1992
MD4	Rivest	1990	512	128	3	Dobbertin, 1996a	Rivest, 1991
MD5	Rivest	1992	512	128	4	Dobbertin, 1996b	Rivest, 1992b
SHA-0	NIST/NSA*	1991	512	160	4	Chabaud, Joux, 1998	NIST, 1993
SHA-1	NIST/NSA	1993	512		4	—	NIST, 2002
SHA-256	NIST/NSA	2000	512		64	—	NIST, 2002
SHA-512	NIST/NSA	2000	1024		80	—	NIST, 2002
RIPEMD	The RIPE Consortium*	1990	512	128	4		Dobbertin, 1997
RIPEMD160	Dobbertin, Basselaers, Preneel	1996	512	160	5	—	Dobbertin, 1996c
HAVAL	Zheng, Pieprzyk, Seberry	1993	1024		3,4,5	Kasselman, Penzhorn, 2000	Zheng, et al., 1993

MD5 processes the input message in 512-bit blocks, divided into 16 32-bit sub-blocks. The resulting digest of the algorithm is a set of 4 32-bit blocks, which concatenate to form a single 128-bit hash value. The original message is padded with 64 bits, representing the length of the message, and then followed by a 1-bit and as many zeros as required, being of a multiple of 512 bits, shown in Figure 2.8.

Figure 2.8 The format of padded messages

The four 32-bit variables for hash values are called chaining variables, which are initialized into a , b , c , and d variables.

(ii) One-Way Hash Functions

There are 4 one-way nonlinear bit-wise functions in this algorithm:

$$F(x, y, z) = (x \wedge y) \vee ((\neg x) \wedge z) \quad (\text{Eq. 2-3})$$

$$G(x, y, z) = (x \wedge z) \vee (y \wedge (\neg z)) \quad (\text{Eq. 2-4})$$

$$H(x, y, z) = x \oplus y \oplus z \quad (\text{Eq. 2-5})$$

$$I(x, y, z) = y \oplus (x \vee (\neg z)) \quad (\text{Eq. 2-6})$$

where $\oplus$ is XOR, $\wedge$ is AND, $\vee$ is OR, and $\neg$ is NOT; x , y , and z are 32-bit words.

These functions are designed so that if the corresponding bits of x , y , and z are independent and unbiased, thus, each bit of the result will also be independent and unbiased. The function F is bit-wise conditional. The function H is the bit-wise parity operator.

(iii) Block Processing of MD5

The main loop is composed of 4 rounds of the transformation operation functions. Each round iterates the operation on the four variables 16 times. The basic operations are:

$$FF(a,b,c,d,M_j,s,t_i) \text{ denotes } a = b + ((a + F(b,c,d) + M_j + t_i) \lll s) \quad (\text{Eq. 2-7})$$

$$GG(a,b,c,d,M_j,s,t_i) \text{ denotes } a = b + ((a + G(b,c,d) + M_j + t_i) \lll s) \quad (\text{Eq.2-8})$$

$$HH(a,b,c,d,M_j,s,t_i) \text{ denotes } a = b + ((a + H(b,c,d) + M_j + t_i) \lll s) \quad (\text{Eq.2-9})$$

$$II(a,b,c,d,M_j,s,t_i) \text{ denotes } a = b + ((a + I(b,c,d) + M_j + t_i) \lll s) \quad (\text{Eq.2-10})$$

where M_j represents the j^{th} sub-block of the message (from 0-15), and $\lll s$ represents a left circular shift of s bits.

Figure 2.9 shows the MD5 main loops and Figure 2.10 shows one MD5 operations.

Figure 2.9 The MD5 main loops ($\boxplus$ represents add modulo 2^{32})

Figure 2.10 One round of MD5 operations

In each round shown in Figure 2.9, there are going to be 16 operations on the variables of a , b , c , and d operated as shown in Figure 2.10. In step i , t_i is the integer part of $2^{32} \cdot \text{abs}(\sin(i))$, where i is in radians. The digest of the last round is the message digest of the entire message.

2. Security of MD5

Compared with MD4, the MD5 algorithm adds a fourth round and the function G is changed so as to make it less symmetric. Each step in MD5 adds on the results of the previous step. Also, the order in which message sub-blocks are accessed in rounds 2 and 3 is changed, to make these patterns less alike. Its ability to withstand brute-force attacks depends on its length of hash code, which is 2^{64} or about 2×10^{19} evaluations of the function. Tom Berson attempted to use differential cryptanalysis of a single round of MD5, however, his attack was ineffective against all four rounds [Berson, 1992]. On the other hand, a more successful attack by den Boer and Bosselaers produced collisions

using the compression function in MD5, but this does not permit attacks against MD5 in practical application [den Boer and Bosselaers, 1994].

3. Performance of MD5

MD5 is intended for high-speed software implementation and removes the vulnerabilities of MD4. A collection of performance data was presented in Table 2.5 in Section 2.4.1.4.

2.4.1.2 RIPEMD160

RIPEMD160 is a 160-bit cryptographic hash function, designed by Dobbertin, Bosselaers, and Preneel [Dobbertin, et al., 1996]. It is used as a secure replacement for the 128-bit hash functions of MD4, MD5, and RIPEMD. It is expected to remain secure for the next ten years or more.

1. Description of RIPEMD160

(1) Padding and the Initialization of Chaining Variables

The RIPEMD160 algorithm uses the same padding scheme as MD5. It has five 32-bit words chaining variables that hold the 160-bit hash value. The chaining variables are initialized at the beginning of the process.

(2) One-Way Hash Functions

$$f1(x,y,z) = x \oplus y \oplus z \quad (\text{Eq. 2-11})$$

$$f2(x,y,z) = (x \wedge y) \vee ((\neg x) \wedge z) \quad (\text{Eq. 2-12})$$

$$f3(x,y,z) = (x \vee \neg y) \oplus z \quad (\text{Eq. 2-13})$$

$$f4(x,y,z) = (x \wedge z) \vee (y \wedge (\neg z)) \quad (\text{Eq. 2-14})$$

$$f5(x,y,z) = x \oplus (y \vee \neg z) \quad (\text{Eq. 2-15})$$

where $\oplus$ is XOR, $\wedge$ is AND, $\vee$ is OR, and $\neg$ is NOT; x, y, z are 32-bit words.

(3) Block Processing of RIPEMD160

The main loop is composed of two parallel 5-round transformation operation functions. Each round carries out 16 iterations of operations on the five variables. The basic operation is as follows:

$$A = (A + f(B,C,D) + X + K) \lll s + E \quad (\text{Eq. 2-16})$$

where $C = C \lll 10$, and $\lll s$ denotes cyclic shift (rotation) over s positions.

During the operations of the message blocks, the order of the message words is also permuted. Figure 2.11 shows the RIPEMD160 main loops.

Figure 2.11 Block processing of RIPEMD160

2. Security and Performance of RIPEMD160

The ability of RIPEMD to withstand brute-force attacks depends on its length of hash value, which is 2^{80} evaluations of the function. The significant increase in security comes at the cost of reduced performance, as shown in Table 2.5 in Section 2.4.1.4.

2.4.1.3 SHA-1

The Secure Hash Algorithm (SHA) was developed by NIST and is specified in the Secure Hash Standard (SHS, FIPS 180). SHA-1 is a revision of SHA and was published in 1993 [FIPS 180-1, 1993]. It is also described in the ANSI X9.30 (part 2) standard. SHA-1 produces a 160-bit (20 bytes) message digest. Although slower than MD5, this larger digest size makes it stronger against brute force attacks.

1. Description of SHA-1

(i) Padding and the Initialization of Chaining Variables

The SHA-1 algorithm uses the same padding scheme as MD5. It has five 32-bit words chaining variables that hold the 160-bit hash value. The chaining variables are initialized in the beginning of the process, e.g. in variables H_0 , H_1 , H_2 , H_3 , and H_4 , initialized as $H_0 = 67452301$; $H_1 = \text{EFCDAB89}$; $H_2 = 98BADCFE$; $H_3 = 10325476$; and $H_4 = \text{C3D2E1F0}$.

(ii) One-Way Hash Functions

The SHA-1 algorithm uses a sequence of functions $f(0)$, $f(1)$, ..., $f(79)$. Each $f(t)$, where $0 \leq t \leq 79$, operates on three 32-bit words x , y , and z and produces a 32-bit word output. $f(t;x,y,z)$ is defined as follows: for words x , y , and z

$$f(t;x,y,z) = (x \wedge y) \vee ((\neg x) \wedge z) \quad (0 \leq t \leq 19) \quad (\text{Eq. 2-17})$$

$$f(t;x,y,z) = x \oplus y \oplus z \quad (20 \leq t \leq 39) \quad (\text{Eq. 2-18})$$

$$f(t;x,y,z) = (x \wedge y) \vee (x \wedge z) \vee (y \wedge z) \quad (40 \leq t \leq 59) \quad (\text{Eq. 2-19})$$

$$f(t;x,y,z) = x \oplus y \oplus z \quad (60 \leq t \leq 79) \quad (\text{Eq. 2-20})$$

A sequence of constant words $K(0), K(1), \dots, K(79)$ is used in SHA-1. These are given by

$$K(t) = 5A827999 \quad (0 \leq t \leq 19)$$

$$K(t) = 6ED9EBA1 \quad (20 \leq t \leq 39)$$

$$K(t) = 8F1BBCDC \quad (40 \leq t \leq 59)$$

$$K(t) = CA62C1D6 \quad (60 \leq t \leq 79).$$

where $\oplus$ is XOR, $\wedge$ is AND, $\vee$ is OR, $\neg$ is NOT and t refers to the number of steps.

(iii) Block Processing of SHA-1

The message digest is computed using the message padded as described above. To generate the message digest, the 16-word blocks $M^{(1)}, M^{(2)}, \dots, M^{(n)}$ are processed in order. The bit-wise operations below have the same meanings as the above, except that S^n means left circular shift of n bits. The processing of each $M^{(i)}$ involves 80 steps.

To process $M^{(i)}$, the following procedures are processed:

- Divide $M^{(i)}$ into 16 words $W_{(0)}, W_{(1)}, \dots, W_{(15)}$, where $W_{(0)}$ is the left-most word
- For $t = 16$ to 79 , let $W_{(t)} = S^1(W_{(t-3)} \oplus W_{(t-8)} \oplus W_{(t-14)} \oplus W_{(t-16)})$
- Let $A = H_0, B = H_1, C = H_2, D = H_3, E = H_4$
- For $t = 0$ to 79 do

$$\text{TEMP} = S^5(A) + f(t;x,y,z) + E + W_{(t)} + K(t);$$

$$E = D; D = C; C = S^{30}(B); B = A; A = \text{TEMP};$$

- Let $H_0 = H_0 + A$, $H_1 = H_1 + B$, $H_2 = H_2 + C$, $H_3 = H_3 + D$, $H_4 = H_4 + E$.

After processing $M^{(n)}$, the message digest is the 160-bit string represented by the 5 words: $H_0 H_1 H_2 H_3 H_4$.

2. Security and Performance of SHA-1

SHA-1 is called secure because it is computationally not feasible to find a message that corresponds to a given message digest, or to find two different messages that produce the same message digest [Eastlake and Jones, 2001]. Any change to a message in transit will, with very high probability, result in a different message digest, and the signature will fail to verify. The performance parameters are shown in Table 2.5 in Section 2.3.1.4.

2.4.1.4 SHA-256

1. Description of SHA-256

SHA-256 was specified in the Federal Information Processing Standard, FIPS PUB 180-2 Secure Hash Standard, in August 2002.

(i) Padding and the Initialization of Chaining Variables

The SHA-256 algorithm uses the same padding scheme as MD5. It has eight 32-bit words chaining variables that hold the 256-bit hash value. The chaining variables are initialized in the beginning of the process, e.g. in variables $H_0 - H_7$, initialized as $H_0 = 0x6A09E667$; $H_1 = 0xBB67AE85$; $H_2 = 0x3C6EF372$; $H_3 = 0xA54FF53A$; $H_4 = 0x510E527F$; $H_5 = 0x9B05688C$; $H_6 = 0x1F83D9AB$; $H_7 = 0x5BE0CD19$.

(ii) One-Way Hash Functions

SHA-256 uses six logical functions. Each function operates on 32-bit words, which are represented as x , y , and z . The result of each function is a new 32-bit word.

$$Ch(z, y, x) = (x \wedge y) \oplus (\neg x \wedge z) \quad (\text{Eq.2-21})$$

$$M_{aj}(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (z \wedge y) \quad (\text{Eq.2-22})$$

$$\Sigma_0(x) = S^2(x) \oplus S^{13}(x) \oplus S^{22}(x) \quad (\text{Eq.2-23})$$

$$\Sigma_1(x) = S^6(x) \oplus S^{11}(x) \oplus S^{25}(x) \quad (\text{Eq.2-24})$$

$$\sigma_0(x) = S^7(x) \oplus S^{18}(x) \oplus R^3(x) \quad (\text{Eq.2-25})$$

$$\sigma_1(x) = S^{17}(x) \oplus S^{19}(x) \oplus R^{10}(x) \quad (\text{Eq.2-26})$$

where S^n means right rotation of n bits, and R^n means right shift of n bits.

(iii) Block Processing of SHA-256

The SHA-256 hash computation uses functions and constants defined in previous sections. Addition (+) is performed modulo 2^{32} . After preprocessing is completed, each message block, $M^{(1)}, M^{(2)}, \dots, M^{(N)}$, is processed in order, using the following steps:

For $i=1$ to N , repeat the following steps:

- Prepare the message operation and working variables

$$W_t = \begin{cases} M_t & 0 \leq t \leq 15 \end{cases} \quad (\text{Eq.2-27})$$

$$\begin{cases} \sigma_1(W_{t-2}) + W_{t-7} + \sigma_0(W_{t-15}) + W_{t-16} & 16 \leq t \leq 63 \end{cases} \quad (\text{Eq.2-28})$$

- Initialize the eight working variables, a, b, c, d, e, f, g , and h , with the $(i-1)^{\text{st}}$ hash

values: $a = H_0^{i-1}, b = H_1^{i-1}, c = H_2^{i-1}, d = H_3^{i-1}, e = H_4^{i-1}, f = H_5^{i-1}, g = H_6^{i-1}, h = H_7^{i-1}$

- For $t=0$ to 63:

$$T_1 = h + \Sigma_1(e) + Ch(e, f, g) + K_t + W_t \quad (\text{Eq.2-29})$$

$$T_2 = \Sigma_0(a) + M_{aj}(a, b, c) \quad (\text{Eq.2-30})$$

$$h = g \quad (\text{Eq.2-31})$$

$$g = f \quad (\text{Eq.2-32})$$

$$f = e \quad (\text{Eq.2-33})$$

$$e = d + T_1, \quad (\text{Eq.2-34})$$

$$d = c \quad (\text{Eq.2-35})$$

$$c = b \quad (\text{Eq.2-36})$$

$$b = a \quad (\text{Eq.2-37})$$

$$a = T_1 + T_2 \quad (\text{Eq.2-38})$$

After repeating a total of N times (i.e., after processing $M^{(N)}$), the resulting 256-bit message digest of the message, M , is

$$H_0^{(N)} H_1^{(N)} H_2^{(N)} H_3^{(N)} H_4^{(N)} H_5^{(N)} H_6^{(N)} H_7^{(N)}.$$

2. Security and Performance of SHA-256

SHA-256 is a 256-bit hash and is meant to provide 128 bits of security against brute-force collision attacks. The following table gives an idea of the performance of the different MD4-like hash functions. The implementations are written in 80x86 assembly language and are optimized for the Pentium processor. It is assumed that both code and

data reside in the on-chip caches. Under these conditions, the cycles are independent of the clock speed, and the throughput scales with the clock speed.

Table 2.5 Performance of optimized assembly language implementations of MD4-like hash functions on a 90 MHz Pentium using a 32-bit flat memory model (i.e., running in native protected mode) [Bosselaers, 2004]

Algorithm	Cycles	Mbit/sec	Mbyte/sec	Relative performance
MD4	241	191.2	23.90	1.00
MD5	337	136.7	17.09	0.72
RIPEMD	480	96.0	12.00	0.50
SHA-1	837	55.1	6.88	0.29
RIPEMD-160	1013	45.5	5.68	0.24

From Table 2.5, it can also be seen that the performance of algorithm MD5, SHA-1 and RIPEMD160 decreases.

2.4.2 Key Generation and Management

The security of a symmetric cryptosystem is based on two aspects: the strength of the algorithm and the length of the key. Assume that the strength of the algorithm is perfect. That means there is no better means to break the cryptosystem than to try every possible key in a brute-force attack. The brute-force attack is a known-plaintext attack. It requires a small amount of cipher text and corresponding plaintext. Its complexity is 2^k , where k is the length of the key. For example, if the key is 56-bits long, there are 2^{56} possible keys. Assume a computer can try a million keys in a second, then it would take 2285 years to find a correct key. Thus, two parameters determine the speed of a brute-force attack: the number of keys to be tested and the speed of each test.

Key management is another aspect of a secure cryptosystem. Its goal is to keep keys secret. Attackers can likely break a key management, but would find it difficult to break a cryptographic algorithm. This thesis disregards the key management issue, which is a challenging and important research area.

Cryptographic keys are random-bit strings generated by some automatic process from a reliably random source or a cryptographically secure pseudo-random-bit generator. A sequence of random numbers, e.g. $R_1, R_2, R_3, \dots$, must have two important statistical properties, uniformity and independence. Numerous methods can be used to generate the random values. The followings are some of the important considerations:

- The method should be portable to different computers, and ideally to different programming languages.
- The method should have a sufficiently long cycle that represents the length of the random-number sequence before repeating the sequences.
- The random numbers should be replicable. Given the starting point (or conditions), it should be possible to generate the same set of random numbers.
- Most importantly, the generated random numbers should closely approximate the ideal properties of uniformity and independence.

The following tests can be performed to ensure these desirable properties:

- Frequency
- Runs
- Autocorrelation
- Gap

- Poker test

So, with a good key generator, the required length of a key relies on the security requirements and the cost estimation. Table 2.6 gives an example of the security requirements for different information. At least 128 bits of key length is usually required.

Table 2.6 Security requirements for different information [Schneier, 1996]

Type of Traffic	Lifetime	Minimum Key Length
Tactical military information	Minutes/hours	56-64 bits
Product announcements, mergers	Days/weeks	64 bits
Long-term business plans	years	64 bits
Trade secrets	decades	112bits
H-bomb secrets	>40 years	128bits
Identities of spies	>50years	128bits
Personal affairs	>50years	128bits
Diplomatic embarrassments	>65years	At least 128bits
U.S. census data	100years	At least 128bits

2.5 Influence of the Security Protocol on Network Performance

It is well known that network security and performance are two of the factors that need consideration in network designing. At this moment, after an extensive search of current publications, not much empirical work has been done in this field. Elkeelany etc. analyzed the IPSec protocol with encryption and authentication using space complexity and time complexity (Elkeelany, et al., 2002). What follows is a summary of the research results from Elkeelany etc.

2.5.1 An Overview of IPSec

IPsec is designed to provide cryptographically-based security for IPv4 and IPv6 to protect IP layer paths (Kent and Atkinson, 1998; Molva, 1999). It has two security protocols, IP Authentication Header (IPSec AH) and IP Encapsulating Security Payload (IPSec ESP).

1. IP Authentication Header (IPSec AH)

IPSec AH realizes data origin authentication and data integrity for IP datagram. Replay detection is an option. IP header includes three fields:

- Security Parameter Index (SPI), is a random value used with the destination address to find the Security Association for the datagram;
- Security Number, is used to detect replayed datagram to provide data integrity;
- Authentication data, is the protected field of the original datagram, obtained from the security hash function.

There are two working modes for AH: transport mode and tunnel mode. The two modes are included in Table 2.7.

Table 2.7 IPSec AH for IP datagram

		Transport mode	Tunnel mode
IP datagram format		Only include AH	Include AH and a new IP header
Protection Scope		End-to-end protection of original datagram	The end-to-end path can be a fraction of original one.
Security Connection		Through security association, a triple of SPI, destination IP, and security protocol, prior the transmission of protected IP datagram	
Position	IPv4	After the original IP header and before TCP header	After the new IP header and before the original IP header
	IPv6	After all the IP header field required for intermediate node processing (hop-by-hop extension fields) and before the first end-to-end field (end-to-end extension field)	Same as for IPv4

2. IP Encapsulating Security Payload (IPSec ESP)

IPSec ESP can be used alone or in combination with IP AH. IPSec ESP provides data confidentiality in addition to data origin authentication, data integrity and replay detection services. Data confidentiality is the main objective of ESP and the others are optional functionalities.

Figure 2.12 Main fields of the ESP header

The ESP header, shown in Figure 2.12, contains the security parameter index, sequence number and an optional authentication data field that is dealt with as similar fields of the AH. Additionally, the payload field contains the data that will be protected. The Padding field fills the payload field to get the input size that is required by the encryption. The padding can also be used as a mean to maintain the secrecy of the actual length of the protected IP datagram. ESP also has two working modes: transport mode and tunnel mode. The two modes are summarized in Table 2.8.

Table 2.8 IP ESP for IP datagram

		Transport mode	Tunnel mode
IP datagram format		Only include AH	Include AH and a new IP header
Encryption payload includes	IPv4	Upper layer protocol (TCP) information, user data and the padding	Original IP header, upper layer protocol (TCP) information, user data, and the padding
	IPv6	those for IPv4 plus end-to-end extension fields	
Protection Scope		Confidentiality assured only for upper layer information	Entire IP datagram by confidentiality services
Security Connection		Through security association, which is a triple of SPI, destination ip, and security protocol, prior the transmission of protected IP datagram	
Position	IPv4	After the original IP header and before TCP header	After the new IP header and before the original IP header
	IPv6	After all the IP header field required for intermediate node processing (hop-by-hop extension fields) and before the first end-to-end field (end-to-end extension field)	Same as for IPv4

2.5.2 The Performance Measurements

1. Space Complexity

Table 2.9 shows the field sizes of the IPSec header.

Table 2.9 IPSec header field sizes (bytes)

Protocol	IPSec AH			IPSec ESP		
	Fixed	Variable	Total	Fixed	Variable	Total
Transport mode	12	12	24	10	12	22
Tunnel mode	12	20	44	10	20	42

From Table 2.9, it can be seen that a total of 24/22 bytes for AH/ESP respectively used for each packet. This overhead is needed for each authentication request transmitted through a security association in transport mode. An additional 20 bytes of IP header are also needed for tunnel destination, which results in a total of 44/42 bytes for AH/ESP, respectively, in tunnel mode.

2. Time Complexity

For IPSec ESP encryption, the triple Data Encryption Standard (3DES) in Cipher Block Chaining (CBC) mode is used (Madson and Doraswamy, 1998). The DES algorithm requires the longest time. Figure 2.13 shows the variation of the computed 3DES encryption time with input blocks and processing power in Millions of Instructions Per Second (MIPS).

Figure 2.13 3DES encryption time vs. input blocks and processing power [Madson and Doraswamy, 1998]

With 500 MIPS, the time complexity of 3DES for the encryption rate is about 4 Mbps.

For authentication, either HMAC-MD5 or HMAC-SHA-1 algorithm is used. Figures 2.14 and 2.15 show the variation of the computed algorithm computation time (digest time) in microseconds with input blocks and processing power in MIPS.

Figure 2.14 HMAC-MD5 authentication time vs. input blocks and processing power [Madson and Doraswamy, 1998]

Figure 2.15 HMAC-SHA-1 authentication time vs. input blocks and processing power [Madson and Doraswamy, 1998]

From the results, the author summarized that MD5 and SHA-1 can be used for IPSec authentication. Compared with SHA-1, MD5 has a higher throughput with 96 bits of message digest and is sufficient for authentication purposes. Since 3DES encryption speed is very slow, it should only be used for critical users, and not for regular traffic flow.

Chapter 3 Design and Implementation of the Authentication Protocol in RSVP and RSVP-TE

This chapter describes the design and implementation of the authentication protocol for the EION Open IP Environment to secure RSVP and RSVP-TE control messages for IP and MPLS networks, respectively.

3.1 Objectives

This section aims to give the main design considerations of the authentication mechanism to be implemented. The reasons that only authentication, but not encrypting, is considered for securing an RSVP or RSVP-TE message are first described. In addition, the possibility of using IPsec is discussed. Then, the authentication mechanism of a message is illustrated in a general way. Lastly, the arrangement of this chapter is described.

As discussed in Chapter 2, RSVP raised three main security concerns: 1) Message integrity and node authentication; 2) User authentication; 3) Secure data streams [Braden, 1997]. These can be achieved with authentication services. In some cases, encryption of the RSVP messages was also considered as an option to provide confidentiality when the RSVP messages must not be visible to outside parties, e.g. for RSVP-SQoS schemes [Talwar, 2001]. However, the encryption brings problems when enciphering all messages that require authentication with a secret key [Davies and Price, 1989]. There are two considerations to this problem, which will be given below.

First, within a network system, some routers have authentication services while others do not. In this situation, the transmission must be made in plaintext, with an associated

authentication field. Some routers can check the authentication field that they hold; however, other routers just rely on general alarm, e.g. check sum field, when an authentication failure is detected.

Second, in hop-by-hop communications, the receiver router has a heavy load of processing responsibilities. Thus, it cannot afford to decipher all received messages. From the performance measurement of IPSec [Braden, 1997], which was described in Section 2.6, the author concluded that 3DES encryption throughput is very slow. It should only be used for critical users, not for regular traffic flow.

As for RSVP, the authentication protocol is used in a large network in a hop-by-hop manner. The encryption of the RSVP messages would waste resources and bring an extra heavy load to the network. Therefore, authentication by full encryption may be inappropriate.

Since IPSec is a standard authentication mechanism, as reviewed in Section 2.6, another question is whether it is appropriate to use IPSec for RSVP messages. IPSec is defined with the security association that is based on the destination address. However, RSVP is a hop-by-hop communication. And the RSVP messages are not clearly defined with source or destination as a router must forward *Path* and *PathTear* messages using the same source address. Moreover, there may be non-RSVP systems among RSVP-enabled routers, which are not visible to senders. Therefore, the security association could not be set up in these cases. The use of IPSec in authenticating RSVP messages becomes inappropriate.

The initial work and discussion of the authentication of RSVP protocol was done by IETF [Baker, et al., 2000]. As shown in RFC 2747, the data structure of the RSVP

Path Msg in the figure. First, the *Path* message from Router 1 is encoded with the key and calculated into a Message Digest-1, using a hash function algorithm. Then the Message Digest-1, together with the *Path* message, is sent to Router 2. Once Router 2 receives both *Path* message and Message Digest-1, using the same algorithm and key as Router 1, Router 2 calculates its own Message Digest-2 with the *Path* message and the key. Then it is the responsibility of Router 2 to compare whether Message Digest-2 and Message Digest-1 are identical. If they are the same, the *Path* message is secure; otherwise, the *Path* message may have been modified, or inserted from an unknown entity.

The following sections are arranged as follows: Section 3.2 describes a brief introduction of the architecture of the EION Open IP Environment where RSVP and RSVP-TE are implemented. Section 3.3 gives the data structure definition of the INTEGRITY Object. Sections 3.4, 3.5, and 3.6 respectively, present the design and implementation of different hash function algorithms, the key generation and distribution method, and message replay prevention. Section 3.7 illustrates the integration of INTEGRITY Object in RSVP/RSVP-TE protocols in order to authenticate RSVP/RSVP-TE messages. Section 3.8 discusses the scope and limitations of the authentication schemes.

3.2 Brief Introduction to the EION Open IP Environment

The EION Open IP Environment is a real-time software system that provides building blocks for IP communication networks, which situate themselves between the operating system and the application/service layer of Internet devices. MPLS protocol is one of the

components in the EION Open IP Environment. An MPLS network has its own unique control plane operating Label Switch Routers (LSRs). Figure 3.2 shows the architecture of the MPLS network. If an IP packet is received, Forwarding Engine Packet Classifier forwards the packet to IP Forwarder, then to IP protocol. When resources like bandwidth reservation are needed, RSVP or RSVP-TE signaling protocol will become involved. If an MPLS packet is received, Forwarding Engine Packet Classifier will forward it to MPLS Forwarder, and the packet will be treated as an MPLS packet. The labels of MPLS packets will be distributed either by Label Distribution Protocol (LDP) or signaling protocol RSVP-TE.

Figure 3.2 Architecture for MPLS network [EION-DESN-0, 2003]

A Label Switched Path (LSP) is a specific path through an MPLS network that a data packet follows, based on its MPLS labels. Label Switch Paths (LSPs) are established by network operators for a variety of purposes, such as to guarantee a certain level of performance, reroute network congestion, or create IP tunnels for network-based virtual private networks. Multiple LSP tunnels can be set up between an ingress router and an egress router to facilitate traffic engineering.

In traditional IP networks, the resource reservation path is set up by RSVP *Path* and *Resv* messages through periodic refreshing of the soft state between senders and receivers, as discussed in Section 2.2 of Chapter 2. Authentication of RSVP messages is achieved by the embedded INTEGRITY object within RSVP message sequences.

In MPLS-based networks, the extension of RSVP, RSVP-TE, is designed to set up LSP tunnels, which also facilitates traffic engineering. Authentication of RSVP-TE messages is implemented by the INTEGRITY object in RSVP-TE messages.

3.3 Data Structure of the INTEGRITY Object

The data structure format of the INTEGRITY object in RSVP messages was defined in RFC 2747 [Baker, 2000], and summarized in Section 2.3.3 of this document. In RSVP/RSVP-TE messages, the detailed format in C is the following:

```
typedef struct{
    RSVP_OBJ      hdr;
    Byte          flag;
    Byte          resv;
    KEY_ID        k_id;
    Byte          seqnum[8];
    union{
        char      kmac_md5[16];
        char      kmac_sha1[20];
        char      kmac_sha2[32];
        char      kmac_ripemd160[20];
    }
}
```

```

    }u;
} ROBJ_INTEG, * integobj_pt;

```

Some of the fields used in this work are described next.

- flag

The flag is an 8-bit field with the bit 0 defined as a handshake flag, and other bits set to 0. If 0th bit equals 1, an integrity handshake mechanism is invoked. The integrity handshake mechanism is used for the receiver to “handshake” with the sender to obtain the starting sequence number for a live authentication key at the restart or initialization period. If it equals 0, this mechanism is suppressed. In this thesis, the flag is set to 0. The handshake mechanism has not been implemented yet.

- k_id (KEY_ID):

Key Identifier is used to identify the key that is currently used and it is unique for a given sender. Key Identifier is a 48-bit field. In this work, a key identifier is composed of a 32-bit sender’s interface and 16-bit key number, representing the specific key used. A number of keys are created by a random number generator, from which one will be used each time. The key generation mechanism and its security consideration will be detailed in Section 3.5.

- Sequence Number (seqnum):

The Sequence Number is a 64-bit field with numbers monotonically increasing so that any playback of the RSVP message will be detected. In this work, the system time is used as a Sequence Number of *Path* and *Resv* messages. Details of the mechanism for prevention of replay are given in Section 3.6.

- Keyed Message Digest (kmac_xx):

The digest is required to be a multiple of 4 bytes long. In this thesis, the digests of four algorithms, MD5, RIPEMD160, SHA-1, and SHA-256, are 16-bytes, 20-bytes, 20-bytes and 32-bytes long, respectively. A detailed description of the algorithms will be presented in Section 3.4. (Here, xx represents one of the four algorithms: md5, sha1, sha2 and ripemd160 used in the data structure).

From the data structure of the INTEGRITY object, the Security Association between the sender and the receiver is obtained by sharing the authentication key and the following parameters:

- Authentication algorithm and algorithm mode
- Associated sending interface
- Source address in the header
- Latest sending sequence number

3.4 Hash Function Algorithms

As reviewed in Chapter 2, the security strength of a symmetric cryptosystem is based on two aspects: the strength of the algorithm and the length of the key. In this section, the issues concerning the strength of the hash function algorithms are discussed.

As discussed in Section 2.4.1, the strength of the hash function algorithms depends on the length of digest, complexity of the hash function and the hash function operations of the algorithm. There are a variety of hash function algorithms published in the literature [Schneider, 1996]. The strength and cryptanalysis of the MD-4 family hash functions were summarized in Table 2.4.

In RSVP message authentication, the authentication algorithm was required to be altered in case of a security breach [Baker, 2000]. Due to the fact that MD4 and RIPEMD have lost their significance in application (see discussion in Section 2.4.1), there is a need to select algorithms with different lengths of digest and different algorithm complexities. Therefore, MD5, RIPEMD160, SHA-1 and SHA-256 were selected as typical algorithms in this thesis for RSVP and RSVP-TE message authentication algorithms. The free shareware nature of these algorithms in implementation is also one of the factors to choose. MD5 is the most popular hash function used today although one of its properties, collision resistant, is breached [Boer, 1992]. Research showed that it did not pose a threat in actual applications [Dobbertin, 1996; Ragab and Ismail, 2001]. SHA-1 conforms to the FIPS PUB 180-1 Secure Hash Standard (SHS) with 20 byte long digest, while SHA-256 conforms to the FIPS PUB 180-2 Secure Hash Standard (SHS) with 32 byte long digest [NIST, 2002]. RIPEMD is developed in the framework of the EU project RIPE (RACE Integrity Primitives Evaluation) with 20 byte long digest. Among these algorithms, no breach of RIPEMD160, SHA-1 and SHA-256 were found to date.

In this thesis, the standard implementations were used for the experiments. Information about the source code used is collected in Table 3.1.

Table 3.1 The implementation and copyright status of hash function algorithms

Algorithm	Implemented by	Copyright
MD5	RSA Data Security, Inc	RSA Data Security, Inc, 1991-2
RIPEMD160	Antoon Bosselaers	Katholieke Universiteit Leuven, 1996 [Bosselaers, 1996]
SHA-1	Christophe Devine	Christophe Devine, 2001-3
SHA-256	Christophe Devine	Christophe Devine, 2001-3

3.5 Symmetric Key Generation, Distribution and System Authentication Configuration

As reviewed in Chapter 2, the security of a symmetric cryptosystem is based on two aspects: the strength of the algorithm and the length of the key. In this section, the issues concerning keys are discussed.

3.5.1 Generating Keys with a Pseudo-Random Number Generator

In this thesis, keys are generated using a pseudo random number generator with a deterministic method. A description of the adopted random number generator and the properties of the keys will be presented.

3.5.1.1 Algorithm for Generating Pseudo Random Numbers

An algorithm for generating pseudo-random numbers was selected from a Florida State University Report: FSU-SCRI-87-50 [Marsaglia and Zaman, 1987]. The algorithm was slightly modified by F. James to produce an array of pseudo random numbers [James, 1990]. The algorithm is a combination of a Fibonacci sequence and an "arithmetic sequence" (using subtraction). It passes all of the tests for random number generators and has a period of 2^{144} [Marsaglia and Zaman, 1987]. This random number generation mechanism was ported to the RSVP and RSVP-TE authentication protocol for generating integer sequences.

The random number sequences were created by the two seeds, and are of sufficient length to complete an entire calculation. During the initialization routine, the seeds were selected. The seed variables (IJ and KL) can have values between: $0 \leq IJ \leq 31328$, and $0 \leq KL \leq 30081$. For example, if several different groups are working on different

parts of the same calculation, each group could be assigned its own IJ seed. This would leave each group with 30000 choices for the second seed. That is to say, this random number generator can create 900 million different subsequences – with each subsequence having a length of approximately 10^{30} [James, 1990].

In this thesis, the seeds are IJ = 3891 and KL = 2489. They are given during system configuration and will be described in Section 3.5.3.

3.5.1.2 Length of the Key

According to the random number generation algorithm, the length of the key can be obtained up to 10^{30} and can be system configured (shown in Section 3.5.3). In this thesis, the key length is altered in order to study the performance of the authentication protocol in the network. Keys with lengths of 8 bytes, 16 bytes, 32 bytes and 64 bytes were generated during experiments.

3.5.1.3 Number of Keys in Key Stack

According to RFC 2747, any key has a finite key lifetime, and will become invalid beyond that period of time. So, it was suggested that a number of keys (each has its own key id) be generated for changing keys when the current key exceeds its lifetime. On the other hand, it is also one of the security parameters for each security association to consistently use keys. In this thesis, the key number is in the range of 30-1100.

3.5.2 Distribution of Keys in Networks

As reviewed in Chapter 2, key management, including key generation, storage, distribution/transfer, and deletion, is an important and challenging issue in cryptosystems. In this thesis, the authentication algorithms use the secret key mechanism, and the key is

distributed in the network through system configuration by the operator with Command Line Interface (CLI). Other factors of keys such as more secure distribution and storage are not within the scope of this study.

The EION CLI belongs to the system management plane, which allows the network operator to configure the routers when necessary. The CLIs for MPLS protocols, including RSVP, RSVP-TE and authentication configuration, are designed and implemented in this thesis, but are not the main purpose of this thesis and will not be detailed here. The following steps were followed to set the key parameters for RSVP.

```
Router> enable

Router# config terminal

Router(config)# router rsvp

Router(config)# security key <number_key> <key_length> <seed1>
                  <seed2> <key_id> <cr>
```

The words in italics were input by an operator. The meanings of the parameters in the bracket < > are shown in Table 3.2.

Table 3.2 Command parameters of the CLI for setting key parameters

Command parameters	Purpose
number_key	The number of keys in the key stack
key_length	The Length of the key in bytes
seed1, seed2	Two seeds for generating keys
key_id	Identifies the key used in the security association
cr	Carriage key

For RSVP-TE, the CLI configuration commands are:

```
Router> enable

Router# config terminal

Router(config)# router rsvp

Router(config)# te security key <number_key> <key_length>
                        <seed1> <seed2> <key_id> <cr>
```

The parameters have similar meanings to those in the RSVP protocol. The network operator then sets up the key parameters for each router in the network so that they share the same secret key.

3.5.3 System Authentication Configuration

The authentication algorithm was also selected through CLI operation by the network operator. The following steps were undertaken in order to configure the authentication algorithm in RSVP.

```
Router> enable

Router# config terminal

Router(config)# router rsvp

Router(config)# security algorithm <alg_type> <cr>
```

where the *alg_type* is the type of the hash function algorithm used. It is one of MD5, RIPEMD160, SHA-1, or SHA-256.

For RSVP-TE, the steps for configuring the authentication algorithm are:

```
Router> enable
```

```
Router# config terminal
```

```
Router(config)# router rsvp
```

```
Router(config)# te security algorithm <alg_type> <cr>
```

where the `alg_type` is one of MD5, RIPEMD160, SHA-1, or SHA-256.

3.6 Message Replay Prevention

For detecting the replay, a sequence number is attached to every packet. A sequence number can be any monotonically increasing number. In this thesis, several methods have been adopted. A discussion of these methods follows.

A Simple Count Number as sequence number was used first. This method uses a count number as the sequence number. For each message sent out (including both *Path* and *Resv* messages), the sequence number of the sender router is increased by 1, the receiver router updates its sequence number if the received sequence number is larger than the one it currently possesses, and discards the packet if the received sequence number is smaller than the one it currently possesses. The method is simple. However, the sender sends a *Path* message every 30 seconds in this system. Before the first *Resv* message is sent back to the sender, the sender will send a second *Path* message based on the original sequence number that it has plus the increment. Then the receiver router may receive a packet with a sequence number smaller than the one it currently possesses. Thus, this method will not work in this situation. This problem can be solved by prolonging the refreshing time of the soft state so that the sender can send the subsequent *Path* message after it gets the *Resv* message from the receiver. This means that the sequence number of the sender is updated before it sends another *Path* message. But another problem arises

with this approach. The refresh time can not be enlarged without limit and only for this purpose because if the refresh time is too large, the soft state may be easily torn down. Hence, this method was discarded.

The other method was to use the system time of the router. *Path* message and *Resv* message keep their own sequence numbers. The initial value of the sequence number is the router start time. It is updated when the router sends out a message, and checked when the router receives a message. If a replay is detected, the packet is discarded.

When using time, the synchronization of the time in the network is crucial. In this thesis, within the small network, the system time was set up by the system operator. However, a Network Time Protocol (NTP) is suggested when synchronizing the time in the network.

3.7 Integration of the INTEGRITY Object in RSVP/RSVP-TE Protocols

In order to authenticate RSVP/RSVP-TE messages, the INTEGRITY object is integrated into the object sequences of both RSVP and RSVP-TE messages. According to RFCs 2205 and 3209, the INTEGRITY object is located after the COMMON HEADER object and before the SESSION object. The objects after the INTEGRITY object are, therefore, authenticated.

3.7.1 RSVP Protocol

Figure 3.3 shows the sequence diagram of the *Path* message of RSVP [EION-DESN-1, 2003] after integrating the message authentication code.

Figure 3.3 The sequence diagram of processing a RSVP Path message

Figure 3.4 shows the sequence diagram of *Resv* message of RSVP [EION-DESN-1, 2003] after integrating the message the authentication code.

Figure 3.4 The sequence diagram of processing a RSVP *Resv* message

3.7.2 RSVP-TE Protocol

Figure 3.5 shows the integration of the authentication code into the RSVP-TE [EION-DESN-2, 2003] protocol. And Figure 3.6 shows the integration of the authentication code into the RSVP-TE [EION-DESN-2, 2003] protocol.

Figure 3.5 The sequence diagram of processing a RSVP-TE Path message

Figure 3.6 The sequence diagram of processing a RSVP-TE Resv message

3.8 Scope and Limitations of the Authentication Schemes

3.8.1 Scope

This design of authenticating the RSVP messages conforms to the RFC 2747 and RFC 2104 [Krawczyk, et al., 1997]. The main scope of this implementation realizes:

- Authenticate the *Path* and *Resv* messages of RSVP in checking any possibility of modification, insertion, or any other forgery
- Authenticate the sender router for non-repudiation by checking the sender interface
- Check the replay of RSVP messages
- Authenticate RSVP messages in a hop-by-hop manner. The security association is flexible. It can be used in a network where the security feature is heterogeneously defined. The security authentication only works for those who have security association hop-by-hop while not affecting other components in the network
- Integrate the authentication code in the TurboRouter software under the EION Open IP Environment
- Achieve authentication implementation using free shareware authentication algorithms. Thus, it may be less expensive than other network products.
- Study the influence of the authentication protocol on the network performance, which will be presented in the next chapter, as the main purpose of this implementation

In this thesis, the authentication code is extended to RSVP-TE messages in setting up LSP tunnels and resource reservation. The scope of the authentication service in RSVP-TE protocol is as follows:

- Authenticate the *Path* and *Resv* messages of RSVP-TE in checking for modifications, insertions, or any other forgeries, including label spoofing.
- Authenticate the sender router by checking the sender interface.
- Check the replay of RSVP-TE messages.
- Authenticate the messages on the explicit route. In MPLS-based networks, the routers on the LSP tunnel should all be authentication-enabled in order to secure the path.
- Integrate the authentication code in the TurboRouter software under the EION Open IP Environment.
- Achieve this authentication implementation by using free shareware authentication algorithms. Thus, it may be less expensive than other network products.
- Study the influence of the authentication protocol on the performance of the MPLS-based networks, presented in the next chapter, as the main purpose of this implementation.

3.8.2 Limitations

This design and implementation of the authentication service has the following limitations:

- The key management protocol is not fully considered, as described in Section 3.5.2.
- The replay prevention mechanism is limited by the time synchronization method. System time often varies from host to host; NTP time, on the other hand, has the extra requirement of installing NTP.
- The hop-by-hop manner of authentication cannot set up a stable security association, and cannot provide point-to-point security.

Chapter 4 Performance Analysis of Authenticated RSVP and RSVP-TE Protocols in Networks

This chapter presents a performance analysis of authenticated RSVP and RSVP-TE protocols and their space and time complexities. Section 4.1 describes the experiment conditions. And Sections 4.2 and 4.3 display the results of performance measurements on RSVP and RSVP-TE protocols.

4.1 Experiment Conditions and Procedures

This section introduces the experimental conditions and procedures for performance analysis of RSVP and RSVP-TE protocols. It includes the simulation network topology, the procedures to empirically measure various processing times, and the methodology used to analyze the experimental data.

4.1.1 Simulation Network Topology

In this thesis, three routers are used to form a simple line topology, as shown in Figure 4.1.

Figure 4.1 Simulation network topology for the experiment

R0, R1, and R2 all represent either IP routers or label switched routers (LSRs). The incoming and outgoing IP-address interfaces of the routers are also shown in Figure 4.1. The parameters of the computers that run these router programs are shown in Table 4.1.

Table 4.1 System parameters of the computers that run the router program

Routers	Computer type	Speed	Memory	Operating system
R0	Pentium III	410.745 MHz	261MB	Redhat Linux 8.0
R1	Pentium III	799.807 MHz	261MB	Redhat Linux 8.0
R2	Intel Celeron	1933.586 MHz	260MB	Redhat Linux 8.0

4.1.2 Procedures and Methodology of Measuring Time

In this thesis, various processing times were measured. For the convenience of the description of time measurements and according to different functionalities in the authentication scheme, the authentication protocol can be dissected into the components listed below:

- Authentication Key Generation
- Message Digest Computation
- *Path* Messages Encoding
- *Resv* Messages Encoding
- *Path* Messages Decoding
- *Resv* Messages Decoding
- RSVP-TE Message Connection Setup (or LSP Setup)

The corresponding time measurements were performed and are described in Table 4.1.

Table 4.2 Definitions of different time measurements

Time measured	Definition
Time for generating keys	Δ_K , Delay in time when the key generation starts and ends
Time for computing message digest	Δ_D , Delay in time when certain algorithms start and end
Time for Path message encoding	Δ_P , Delay in time when the authentication encoding function is called and returned; a period of encoding <i>Path</i> message
Time for Resv message encoding	Δ_R , Delay in time when the authentication encoding function is called and returned; the period of encoding <i>Resv</i> message
Time for Path message decoding	Δ_{Pd} , Delay in time when the authentication decoding function is called and returned; a period of decoding <i>Path</i> message
Time for Resv message decoding	Δ_{Rd} , Delay in time when the authentication decoding function is called and returned; a period of decoding <i>Resv</i> message
Time RSVP-TE connection setup	Δ_T , Delay in time when the sender sends a <i>Path</i> message and receives a <i>Resv</i> message

According to the definitions of the time measured, the components described previously can be classified into the following four layers. A layer that is less dependent on other components with regard to time measurements situated at a lower layer, as shown in Table 4.2.

Table 4.3 Classification of Components into layers

Layer	Components
1	Key Generation
2	Message Digest Computation
3	<i>Path</i> Messages Encoding, <i>Resv</i> Messages Encoding, <i>Path</i> Messages Decoding, <i>Resv</i> Messages Decoding
4	RSVP and RSVP-TE Message Connection Setup (or LSP Setup)

For authenticating RSVP and RSVP-TE messages, time was measured from layer 1 to layer 4. The results were also presented in the same order, respectively, in Sections 4.2 and 4.3.

RSVP-TE is a signaling protocol in MPLS-based networks. It is used to set up multiple sessions and LSPs in the MPLS protocols of the EION Open IP environment. In this thesis, the time for authenticating RSVP-TE messages was measured with an increase in the number of sessions. Due to the limitation of the current RSVP-TE implementation, up to 28 sessions were established for the measurements.

During the entire measurements, the experiment was set up in a way that was similar to the realistic usage of the routing system in the field, except that there was no background data traffic. There were only control messages, including OSPF and RSVP or RSVP-TE messages. The process scheduling was based on the kernel scheduling. No process priority was changed during the experiment. The time values were obtained using the system function: *gettimeofday()*. Each experiment was repeated about ten times. The average values were presented as the results in this thesis.

4.1.3 Analysis of the Measured Time

Figure 4.2 shows the variations in the times for authenticating a *Path* message with the measured times. Time is also presented in Table 4.4. The standard error analysis was performed on the measurements. The average, variance, standard deviation, and confidence interval were calculated as follows.

- Average

For N values of a quantity x given by $x_1, x_2, \dots, x_N$, the average or mean is given by

$$\langle x \rangle = \{x_1 + x_2 + x_3 + \dots + x_N\} / N = \Sigma x_i / N \quad (\text{Eq. 4-1})$$

Figure 4.2 Time for authenticating Path message as a function of measuring times

- Deviation

The deviation, d_i , of any reading, x_i , is its difference from the average (or mean),

$$d_i = x_i - \langle x \rangle \quad (\text{Eq. 4-2})$$

- Variance:

The variance is a measure of how spread out a distribution is. It is computed as the average squared deviation of each number from its mean.

$$\sigma^2 = \sum (x_i - \langle x \rangle)^2 / N \quad (\text{Eq. 4-3})$$

- Standard deviation

The standard deviation, σ , is the square root of the mean of the squares (rms) of the deviations, d_i , is given in Eq.4-4 and shown in Table 4.4:

$$\Sigma = \{[d_1^2 + d_2^2 + d_3^2 + \dots + d_N^2] / N\}^{1/2} = \{\sum d_i^2 / N\}^{1/2} \quad (\text{Eq. 4-4})$$

- Confidence interval

A confidence interval is a range of values that has a certain probability of containing a parameter being estimated. For example, the 95% confidence interval is constructed in such a way that 95% of such intervals will contain the parameter.

Table 4.4 Standard Error analysis of the time measured

No.	Time (μsec)	Deviation	No.	Time (μsec)	Deviation
1	872	-1	22	873	0
2	873	0	23	877	4
3	872	-1	24	877	4
4	873	0	25	873	0
5	875	2	26	872	-1
6	872	-1	27	873	0
7	873	0	28	873	0
8	872	-1	29	872	-1
9	871	-2	30	873	0
10	872	-1	31	873	0
11	874	1	32	872	-1
12	873	0	34	874	1
13	872	-1	35	872	-1
14	871	-2	36	874	1
15	873	0	37	872	-1
16	877	4	38	872	-1
17	873	0	39	872	-1
18	872	-1	40	876	3
19	871	-2	41	872	-1
20	873	0	42	870	-3
21	874	1			
Average			872.9		
Variance			2.46		
Standard Deviation			1.57		

Table 4.5 Frequency distribution of the experiment data

Data Interval (x)	Time (μsec)	number	Frequency P(x)
870-871	870	1	0.024
871-872	871	3	0.073
872-873	872	15	0.366
873-874	873	13	0.318
874-875	874	4	0.098
875-876	875	1	0.024
876-877	876	1	0.024
877-878	877	3	0.073

For this measurement, the frequency distribution of the data is obtained, and shown in Table 4.5. A normal distribution is described as the following:

$$P(x) = C \times \exp \{-(x_i - \langle x \rangle)^2 / 2\sigma^2\} \quad (\text{Eq. 4-5})$$

where C is a constant, $\langle x \rangle$ is the average, and σ is the standard deviation. A fitting of the experiment data ($P_e(x)$) with the normal distribution data ($P_n(x)$) is shown in Figure 4.3. A value of $\upsilon = \sum [P_e(x) - P_n(x)]^2$ is calculated. With the minimum υ value of 0.030, a C value of 0.35 and a σ value of 1.02 are obtained.

The confidence interval for the average is thus computed based on the sample average (M) and estimated standard deviation (δ) as: $M - t\delta \leq \mu \leq M + t\delta$, where t is obtained by the table in [RVLS, 2004], depending on the degree of freedom (DF) and the level of confidence (LC). Here, $DF = N-1 = 40$, where N is the sample number; $LC = 0.95$; and $z = 2.0211$. M and δ are shown in Table 4.3. Thus, the confidence interval for the average is: $869.43 \leq t \leq 876.37$.

Figure 4.3 Frequency distribution of the time authenticating a Path message and the fitting with Gaussian normal distribution.

4.2 Performance Analysis of Authenticating RSVP Messages

4.2.1 Space Complexity of the Authentication Protocol

As shown in Section 3.3, an RSVP Integrity Object has a fixed size for fields of RSVP_OBJ, flag, a reserved bit, Key Identifier, Sequence Number. Their sizes are 4, 1, 1, 6, and 4 bytes, respectively. The message digest is algorithm-dependent. In this work, the authentication uses Keyed Hashing for Message Authentication (HMAC) [RFC 2104] with Message Digest algorithms: MD5, RIPEMD-160, and Secure Hash Algorithm (SHA-1 and SHA-256). The generated digests have sizes of 16, 20, 20, and 32 bytes, respectively for HMAC-MD5, HMAC-RIPEMD-160, HMAC-SHA-1, HMAC-SHA-256. Thus, the total sizes of the Integrity Objects are 32, 36, 36, and 48 bytes, respectively, by combining hash algorithms of MD5, RIPEMD-160, SHA-1, and SHA-256.

Table 4.5 shows the variations in the size (in bytes) of different fields in the Integrity Object for different hash function algorithms. So the overheads of 32, 36, 36, and 48 bytes, respectively, for MD5, RIPEMD160, SHA-1, and SHA-256 are needed for each authentication request message transmitted through a security association.

Table 4.6 Sizes in byte of different fields in Integrity Object

Alg Fields	MD5	RIPEMD-160	SHA-1	SHA-256
RSVP_OBJ	4	4	4	4
Flag	1	1	1	1
Reserved	1	1	1	1
Key Identifier	6	6	6	6
Sequence Number	4	4	4	4
HMAC-Alg	16	20	20	32
Total	32	36	36	48

4.2.2 Time for Generating Authentication Keys

The secret authentication keys are generated with a pseudo-random number generator. The key generation time also influences the performance of the RSVP and RSVP-TE. Figure 4.4 and Figure 4.5 show the times for generating secret authentication keys as functions of the number of keys and the lengths of the keys, respectively.

Figure 4.4 Time for generating secret keys as a function of the number of keys

Figure 4.5 Time for generating secret keys as a function of the length of the keys

It can be seen that the time for generating the secret key increases markedly with the number of keys, but increases only slightly with an increase in the length of the key. Thus, the increase in length of the keys does not add much overhead in terms of key generation time, but can improve the security strength.

4.2.3 Time for Generating Message Digest

The time for authenticating the RSVP *Path* message and generating its message digest was measured (for R_0 and thereafter, if not otherwise specified), as shown in Figure 4.6.

Figure 4.6 Time measured for encoding RSVP Path message (denoted as msg) and time for generating message digest as a function of different hash algorithms
(Number of key – 30; Length of key – 8 bytes)

It can be seen that MD5 has the shortest time of generating the message digest and encoding the whole *Path* messages. SHA-256 adds the most time of all of the algorithms studied. As discussed in Chapter 2, SHA-256 has the longest hash value, at 32 bytes, and the most complicated block processing procedures among these hash algorithms. Thus SHA-256 is the most secure, but brings more overhead in authenticating RSVP messages. The time for generating the message digest is 94 μ sec, 137 μ sec, 118 μ sec, and 213 μ sec, respectively, for MD5, RIPEMD160, SHA-1 and SHA-256.

4.2.4 Time for Authenticating *Path* and *Resv* Messages

This section reports the time measurement results for authenticating RSVP *Path* and *Resv* messages, i.e. layer 3 components, as defined in Table 4.2, followed by a discussion of the performance of the authentication mechanism in RSVP protocol.

1. Time for Authenticating *Path* Messages

Figure 4.7 shows the time for encoding and decoding *Path* messages.

It can be seen that the time for encoding and decoding a *Path* message varies for different algorithms; and the decoding time is shorter than the encoding time. This occurred when encoding a *Path* message, because extra operations, such as encoding the RSVP_OBJ header and generating a sequence number, were undertaken.

From the results shown in Figure 4.7, the MD5 requires the shortest time for encoding, while the SHA-256 requires the longest. This may occur because MD5 leads to the shortest message digest value (16 bytes) and the SHA-256 creates the largest digest value (32 bytes). Comparing RIPEMD160 with SHA-1, although both have the same length of message digest, SHA-1 shows a better performance than RIPEMD160. The measurement

results for the first three algorithms, shown in Figure 4.7, are in accordance with the results obtained in Table 2.5.

*Figure 4.7 Time for encoding and decoding a RSVP Path message
(for R1 for comparison with the same CPU)*

2. Time for Authenticating *Resv* Messages

It can be seen that the times for encoding and decoding *Resv* messages are almost the same, about 387 μsec, and do not affect different algorithms in this experiment environment.

*Figure 4.8 Time for encoding and decoding a Resv message
(for R1 for comparison with the same CPU)*

4.2.5 Discussion on Authenticating RSVP Messages

As discussed in Chapter 2, in IP networks, the reservation state is set up through timely refreshes between senders and receivers for the reservation connection. The RSVP message connection setup time can be defined as the delay between the time the sender first detects a *Path* message and the time the sender receives a *Resv* message. The connection setup time for RSVP messages can be affected by the time measured for layers 1 to 3, as defined in Table 4.2. They are the time for key generation, message

digest computation, *Path* message encoding and decoding, and *Resv* message encoding and decoding. From the measurement results, the time required for authenticating RSVP messages varies with different authentication key parameters and authentication algorithms. For a certain key used, the connection setup time could increase in an order of MD5, SHA-1, RIPEMD160, and SHA-256 if other factors in the network can remain the same in different algorithms.

4.3 Performance Analysis of Authenticating RSVP-TE Messages

The authentication mechanism, as described in Chapter 3, is integrated into the RSVP-TE protocol to authenticate the signaling messages of RSVP-TE. Compared with RSVP messages, the additional objects of EXPLICIT_OBJ, LABEL_REQUEST_OBJ, and RECORD_ROUTE_OBJ are added to the RSVP-TE *Path* and *Resv* messages. As is known, the time for generating authentication keys and message digest is only dependent on the authentication mechanism itself, and is independent of the protocols of RSVP or RSVP-TE. Thus, for RSVP-TE protocol, only the time for authenticating RSVP-TE *Path* and *Resv* messages and the time for RSVP-TE connection setup were measured, which is shown below.

4.3.1 Time for Authenticating RSVP-TE *Path* and *Resv* Messages

Figure 4.9 shows the time required for encoding and decoding RSVP-TE *Path* messages with different algorithms.

Figure 4.9 Time required for encoding and decoding RSVP-TE Path message with different algorithms

From Figure 4.9, it can be seen that the time required to encode and decode RSVP-TE *Path* messages shows little variation between the algorithms, which is different from that of RSVP message shown in Figure 4.7.

Figure 4.10 shows the time required for encoding and decoding RSVP-TE *Path* messages with the increase in the number of sessions using MD5. The experimental data fluctuated slightly within 20 μsec for encoding time, but remained same for decoding time. Thus, the time measured is independent of the number of sessions for MD5.

Figure 4.10 Variations in the time for encoding and decoding RSVP-TE Path messages with the number of sessions using MD5

The time required to encode and decode RSVP-TE *Path* messages with other algorithms displayed similar behavior. As shown in Chapter 3, the time measurement for encoding and decoding *Path* message only concerns the specific size of the *Path* message and the algorithms used, which is not influenced by the number of sessions. Similar results are obtained for authenticating the *Resv* messages and the same reason applies. The measurement results for *Resv* messages will not be shown here.

4.3.2 Connection Setup Time for RSVP-TE Messages

The connection setup time (Δ_T) for RSVP-TE messages in setting up an LSP in MPLS-based network is defined as the delay between the time the sender first detects a *Path* message and the time the sender first receives a *Resv* message. The first *Path* and *Resv* messages of a session were measured. In this experiment, the connection setup time was measured both with and without the authentication mechanism in RSVP-TE messages, designated Δ_A and Δ_{NA} , respectively. When there is only one session and the authentication mechanism is disabled in the network, the connection setup time was $\Delta_{NA} = 57.3$ msec for RSVP-TE. The connection setup time was $\Delta_A = 61.7$ msec when the authentication mechanism was enabled with MD5. The time ratio (ϕ) of the authentication mechanism versus the mechanism without authentication was calculated as:

$$\phi = \Delta_A / \Delta_{NA} \quad (\text{Eq. 4-6})$$

The time ratio for setting up an LSP by RSVP-TE messages is 1.08 when there is only one session in the network. Multiple sessions were then set up, and only one LSP was established in each session. The connection setup time of an RSVP-TE message was measured for different number of sessions. Figure 4.11 shows the average time for the connection setup of an RSVP-TE message as a function of the number of sessions under the condition of authentication mechanism and without authentication mechanism. It can be seen that the connection setup time increases with an increase in the number of sessions with authentication and without authentication. The time ratio, ϕ , was calculated and shown in Table 4.7 for MD5 at some of session numbers. Least square fits of the

experimental values with polynomials are also shown as the solid lines in this figure. In case of the authentication, the fitting equation is:

$$y = -0.0837x^2 + 6.6139x + 68.739 \quad (\text{Eq. 4-7})$$

with the R-squared value of 0.99. In case of the non-authentication, the fitting equation is:

$$y = -0.0612x^2 + 5.4318x + 33.753 \quad (\text{Eq. 4-8})$$

with R-squared value of 0.98.

The estimated time ratio, ϕ_{cal} , of the connection setup time was calculated from the fitting curves using Eq. 4-7 and Eq. 4-8, respectively, and was also shown in Table 4.7 for MD5. The estimated time ratio qualitatively decreases with an increase in the number of session. The confidence intervals of each data were calculated and shown in Figure 4.12.

Figure 4.11 Average connection setup time of RSVP-TE as a function of the number of sessions and the fitting curves

Figure 4.12 Average connection setup time of RSVP-TE as a function of the number of sessions with 95% confidence interval for each point

The connection setup time was also measured for other authentication algorithms. Figures 4.13, 4.14, and 4.15 show the average connection setup time of RSVP-TE messages with the authentication algorithms of RIPEMD160, SHA-1, and SHA-256. The average connection setup times using different authentication algorithms show similar behaviors with the increase in the number of sessions. The least square fits were also applied to these algorithms. Results of the estimate time ratios, ϕ_{cal} , together with the measure time ratios, ϕ , are shown in Table 4.7 for different authentication algorithms at a selective number of sessions.

Figure 4. 13 Average connection setup time of RSVP-TE as a function of number of sessions for authentication algorithm of RIPEMD160

Figure 4. 14 Average connection setup time of RSVP-TE as a function of number of sessions for authentication algorithm of SHA-1

Figure 4.15 Average connection setup time of RSVP-TE as a function of number of sessions for authentication algorithm of SHA-2

Table 4.7 The time ratio of authentication mechanism in RSVP-TE message for different sessions

No. of sessions	Processing overhead (ϕ) (%)							
	MD5		RIPEMD160		SHA-1		SHA-256	
	ϕ	ϕ_{cal}	ϕ	ϕ_{cal}	ϕ	ϕ_{cal}	ϕ	ϕ_{cal}
2	1.63	1.84	1.95	2.23	1.80	2.05	1.89	2.03
5	1.89	1.68	2.09	1.84	1.87	1.62	1.89	1.77
10	1.60	1.54	1.67	1.55	1.46	1.32	1.70	1.57
15	1.47	1.47	1.42	1.42	1.23	1.23	1.44	1.49
20	1.32	1.42	1.26	1.36	1.08	1.23	1.44	1.47
25	1.38	1.38	1.41	1.35	1.29	1.28	1.48	1.47
28	1.48	1.36	1.45	1.36	1.48	1.34	1.54	1.49

The measured data from Table 4.7 was obtained under the same experimental conditions and reflects quantitatively the change of the time ratio with the number sessions using different authentication algorithms. By closely checking the measured (ϕ) and calculated (ϕ_{cal}) time ratio values, a critical session number (n_{cr}) was observed. When n is smaller than n_{cr} , both calculated and measured time ratio decreases with increasing n . When n is larger than n_{cr} , the time ratio increases slightly with increasing n . These n_{cr} 's are > 28 for MD5, 26 for RIPEMD160, 20 for SHA-1 and 22 for SHA-256.

4.3.3 Discussion on Authenticating RSVP-TE Protocols

As reviewed in Section 2.2.4, different from RSVP, where the sender sends the *Path* messages and the receiver sends the *Resv* messages along a hop-by-hop path, the RSVP-TE is used to set up LSP on an explicit route. The ingress sends out the route information in a *Path* message to request label binding from the egress. Upon receiving the *Path* message, the RSVP-TE *Resv* message distributes labels upstream along the explicit route. By authenticating the RSVP-TE *Path* messages and *Resv* messages, the explicit route information (included in EXPLICIT_ROUTE object, i.e., ERO), the labels of the LSP, and resource reservation parameters such as bandwidth are authenticated. This contributes to the security assurance of the MPLS core networks.

It should be pointed out that the maximum number of sessions that can be set up within the simulation network is 28. This is limited by the current implementation of RSVP-TE. More work should be carried out on setting up more sessions so that the network capacity with the authenticated RSVP-TE protocol can be studied.

Chapter 5 Conclusions and Future Work

5.1 Summary and Conclusions

The EION Open IP Environment [EION/CSP-00, 2000] is a portable and real-time software system that provides high performance IP communications building blocks, and sits between the operating system and the application/service layer of various Internet-enabled devices. Within the MPLS framework in the EION Open IP Environment, various protocols, such as MPLS Forwarding Engine, Label Distribution Protocol, and Signaling Protocols (RSVP-TE), were developed. In this thesis, in order to secure the MPLS-based network, an authentication mechanism was first designed and implemented for the signaling protocol of RSVP in IP networks and then to extended to, RSVP-TE.

The authentication mechanism was designed based on the RFC 2747 [Baker, et al., 2000]. Within the authentication protocol, the key generation mechanism, hash function algorithms, and replay prevention mechanism were specifically designed and implemented with current shareware implementations. The design and implementation realized the following functionalities:

For RSVP protocol:

- Authenticated the *Path* and *Resv* messages of RSVP in checking for modifications, insertions, or any other forgeries.
- Authenticated the sender router for non-repudiation by checking the sender interface.
- Checked the replay of RSVP messages.

- Authenticated RSVP messages in a hop-by-hop manner. The security association is flexible. It can be used in a network where the security feature is heterogeneously defined. The security authentication only works for those who have hop-by-hop security association while not affecting other components in the network.
- Achieved this authentication implementation by using free shareware authentication algorithms. Thus, it may be inexpensively applied to other network products.

Similarly, for RSVP-TE protocol:

- Authenticated the *Path* and *Resv* messages of RSVP-TE in checking for modifications, insertions, or any other forgeries, including label spoofing.
- Authenticated the sender router for non-repudiation by checking the sender interface
- Checked the replay of RSVP-TE messages.
- Authenticated the messages on the explicit route. In MPLS-based networks, the routers on the LSP tunnel should all be authentication-enabled in order to secure the path.
- Achieved this authentication implementation by using free shareware authentication algorithms. Thus, it will not be expensive to add it to other network products.

The performance analysis on the authentication protocol itself was performed via RSVP protocol. In terms of the space complexity, the overheads of 32, 36, 36, and 48 bytes, respectively, for MD5, RIPEMD160, SHA-1, and SHA-256 are needed for each authentication request. The time for generating message digest was measured, being 94, 137, 118, and 213 μ sec, respectively, for MD5, RIPEMD160, SHA-1 and SHA-256 in a specific environment. A secret key mechanism was designed for the authentication protocol. Results showed that the time for generating keys depended more on the number of keys in the key stack than on the length of the keys.

For a specific key, the time for authenticating a RSVP *Path* message relied on the performance of the hash function algorithms, which degraded slightly in the order of MD5, SHA-1, RIPEMD160 and SHA-256. However, authenticating a *Resv* message was independent of the hash function algorithms, being almost the same at about 387 μ sec in this experimental environment. Further investigation is still needed in this area.

In concern with the security assurance in MPLS-based networks, while authenticating the RSVP-TE *Path* messages and *Resv* messages, the explicit route information, the labels of the LSPs, and resource reservation parameters such as bandwidth, were authenticated. This contributes to securing the MPLS core networks. The time for authenticating the RSVP-TE *Path* messages and *Resv* messages is around 400 μ sec.

While studying performance in terms of connection setup time of RSVP-TE messages, more than one session was set up to establish multiple LSP tunnels. The time was measured with and without the authentication protocol. In one session, the ratio of the connection setup time with and without the authentication protocol was about 1.08 for MD5. Results from more sessions indicated that the ratio of the connection setup time

with and without the authentication mechanism qualitatively decreases for the session number smaller than a critical value. These critical session numbers are > 28 for MD5, 26 for RIPEMD160, 20 for SHA-1 and 22 for SHA-256. Larger than the critical numbers, the time ratio is a slightly increase for all the algorithms.

5.2 Future studies

Future studies will focus on the following aspects:

1. Design and implementation of the authentication protocol

- Improve the sequence number generation method.
- Implement a key management protocol.
- Implement an integrity handshake mechanism.

2. Performance analysis of authenticated RSVP protocol

- Measure the connection setup time for RSVP messages that has been qualitatively discussed in Section 4.2.5.
- Study the IP network capacity with the authenticated RSVP protocol.

3. Performance analysis of authenticated RSVP-TE protocol

- Set up more than 28 sessions in an MPLS-based network.
- Study the capacity of an MPLS-based network with authenticated RSVP-TE protocol.

References

- Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and Shallow, G., (2001), "RSVP-TE: Extensions to RSVP for LSP Tunnels", *RFC 3209*.
- Awduche, D., Malcolm, J., Agogbua, J., M. O'Dell and J. McManus, (1999), "Requirements for traffic engineering over MPLS", *RFC 2702*.
- Baker, F., Lindell, B., Talwar, M., (2000), "RSVP Cryptographic Authentication", *RFC 2747*.
- Barzilai, Tsipora P., and Kandlur, Dilip D., (1998), "Design and implementation of an RSVP-based quality of service architecture for an integrity services internet", *IEEE Journal on Selected Area in Communications*, Vol. 16, No. 3, pp. 397-413.
- Berson, Thomas A., (1992), "Differential Cryptanalysis Mod 2^{32} with Application to MD5", *Advances in Cryptology -- Proceedings of Eurocrypt '92*, R. Rueppel, Ed., *Lecture Notes in Computer Science*, Springer Verlag, Vol. 658, pp. 71-80.
- Biham, E. and Shamir, A., (1991), "Differential cryptanalysis of Feal and N-Hash", *Advances in Cryptology - Eurocrypt'91*, LNCS 547, Springer-Verlag, pp. 1-31.
- Biham, E. and Shamir, A., (1993), *Differential Cryptanalysis of the Data Encryption Standard*, Springer-Verlag.
- Braden, R., Zhang, L., Berson, S., Herzog, S., (1997), "Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specifcaiton", *RFC 2205*.

- Boer, den, B., (1992), "An attack on the last two rounds of MD4," *Advances in Cryptology, proc. Crypto '91, LNCS 876*, J. Feigenbaum, Ed., Springer-Verlag, pp. 194-203.
- Boer den, B. , and Bosselaers, A., (1994), "Collisions for The Compression Function of MD5," *Advances in Cryptology, proc. Crypto '93, LNCS 765*, T. Helleseeth, Ed., Springer-Verlag, pp. 293-304.
- Bosselaers, A., (1996), "RIPEMD-160 software written by Bosselaers, Antoon", available at <http://www.esat.kuleuven.ac.be/~cosicart/ps/AB-9601/>.
- Bosselaers, A. and Preneel, B., Eds., (1995), "RIPE, Integrity Primitives for Secure Information Systems, Final Report of RACE Integrity Primitives Evaluation (RIPE-RACE 1040)," *Lecture Notes in Computer Science*, Springer-Verlag, Vol. 1007.
- Chabaud F. and Joux A., (1998), "Differential Collisions in SHA-0", *Advances in Cryptology - Crypto'98, LNCS 1462*, Springer-Verlag, pp. 56-71.
- Cisco, (2003), "Security of the MPLS Architecture", White paper, available at http://www.cisco.com/warp/public/cc/pd/iosw/prodlit/mxinf_ds.htm
- Clark, D.D., (1988), "The design philosophy of the DARPA internet protocols", *Proc. ACM. SIGCOMM'88*, Computer Communication Review, Vol. 18, No.4, pp.106-114.
- Davvies, D.W., and Price, W.L., (1989), *Security for Computer Networks - An Introduction to Data Security in Teleprocessing and Electronic Funds Transfer*, John Wiley & Sons Ltd.
- Dobbertin, H., (1996a), "Cryptanalysis of MD4," *Fast Software Encryption, LNCS 1039*, D. Gollmann, Ed., Springer-Verlag.

Dobbertin H., (1996b), "The Status of MD5 after a Recent Attack", *CryptoBytes* Vol.2, No.2, pp. 1-6.

Dobbertin, H., Bosselaers, A., and Preneel, B., (1996c), "RIPEMD-160: A Strengthened Version of RIPEMD", *Fast Software Encryption, LNCS 1039*, D. Gollmann, Ed., Springer-Verlag, pp. 71-62.

Dobbertin, H., (1997), "RIPEMD with Two-Round Compress Function is Not Collision-Free", *Journal of Cryptology*, Vol. 10, No. 1, pp. 51-70.

Dobbertin, H., (1998), "Cryptanalysis of MD4", *Journal of Cryptology*, Vol.11, No.4, pp. 253-271.

Bosselaers A., (2004), <http://www.esat.kuleuven.ac.be/~bosselae/ripemd160.html#What>

Eastlake, D., 3rd, Jones, P., (2001), "US Secure Hash Algorithm 1 (SHA1)", *RFC 3174*.

EION/CSP-00, (2000), "CSP-Ace_OpenIP2_1_Dev_Doc.pdh", *Common System Plane, Developer's Guide*, Open IP Environment 2.0.2, Eion Inc.

EION-DESN-0, (2003), *MPLS Network Architecture*, the EION Inc.

EION-DESN-1, (2003), *RSVP Module Design Document*, the EION Inc.

EION-DESN-2, (2003), *MPLS – RSVP-TE Design*, the EION Inc.

FIPS, (1993), "Secure Hash Standard", *United States of American, NIST, Federal Information Processing Standard (FIPS) 180-1*.

FIPS, (1995), "Secure Hash Standard", *NIST, US Department of Commerce, FIPS 180-1*.

FIPS, (2002), "Secure Hash Standard", NIST, FIPS 180-2.

<http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf>

- IEC, (2004), "Multiprotocol Label Switching (MPLS)", *International Engineering Consortium*, available at <http://www.iec.org/online/tutorials/mpls/topic01.html>.
- James, F., (1990), "A Review of Pseudo-C random Number Generators", available at <http://astronomy.swin.edu.au/~pbourke/analysis/random/>.
- Kaliski, B., (1992), "The MD2 message digest algorithm", *RFC 1319*.
- Kasselman P. and Penzhorn W., (2000), "Cryptanalysis of Reduced Version of HAVAL", *Electronics Letters*, Vol. 36, No. 1, pp. 30-31.
- Kent, S., Atkinson, R., (1998), "Security Architecture for the Internet Protocol", *RFC 2401*.
- Krawczyk, H., Bellare, M., and Canetti, R., (1997), "HMAC: Keyed-Hashing for Message Authentication", *RFC 2104*.
- Madson, C. and Doraswamy, N., (1998), "The ESP DES-CBC Cipher Algorithm with Explicit IV", *RFC 2405*.
- Marsaglia, George, and Zaman, Arif, (1987), "Toward a Universal Random Number Generator", *Florida State University Report: FSU-SCRI-87-50*.
- Merkle, R., (1990), "One way hash function and DES", *Advances in Cryptography, Proc. Crypto '89, LNCS 435*, G. Brassard, Ed., Springer-Verlag, pp.428-446.
- Miyaguchi, S., Ohta, K., and Iwata, M., (1990), "128-bit hash function (N-hash)", *NTT Review*, vol. 2 (no. 6), pp. 128-132.
- Molva R., (1999), "Internet security architecture", *Computer Networks*, Vol. 31, pp.787-804.

Neogi Anindya and Chiueh Tzi-cker, (1999), "Performance analysis of an RSVP-capable router", *IEEE Network*, Vol.13, Issue: 5, pp.56-63.

NIST, (1993), "FIPS 180".

NIST, (1991), "A proposed federal information processing standard for digital signature standard (DSS)", available at <http://www.zone-h.org/files/33/FIPS-DSSDoc.txt>.

NIST, (1995), "Secure Hash Standard", available at <http://www.itl.nist.gov/fipspubs/fip180-1.htm>

NIST, (2002), "FIPS 180-2: Secure Hash Standard (SHS)", available at <http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf>.

Osborne, E., and Simha A., (2003), *Traffic Engineering with MPLS*, Cisco Press.

Papadimitratos, P., and Haas, Z.J., (2002), "Securing the Internet Routing Infrastructure", *IEEE Communications Magazine*, Vol.40, No.10, pp. 60-68.

Ragab, A. H. M., and Ismail, N. A., (2001), "An efficient message digest algorithm (MD) for data security", *Proceedings of IEEE Region 10th International Conference on Electrical and Electronic Technology*, Vol. 1, pp. 191-197.

Rivest, R., (1991), "The MD4 message digest algorithm," *Advances in Cryptology, Proc.Crypto '90, LNCS 537*, S. Vanstone, Ed., Springer-Verlag, pp. 303-311.

Rivest, R., (1992a), "The MD4 message digest algorithm", *RFC 1320*.

Rivest, R., (1992b), "The MD5 message digest algorithm", *RFC 1321*.

Rogier, N. and Chauvaud, P., (1995), "The compression function of MD2 is not collision free", *workshop record, 2nd Workshop on Selected Areas in Cryptography (SAC'95)*, Ottawa, Canada.

- Rosen, E. and Viswanathan, Callon, A., R., (2001), "Multiprotocol Label Switching Architecture", *RFC 3031*.
- RVLS, (2004), http://www.ruf.rice.edu/~lane/stat_sim/conf_interval/
- Schneier, Bruce, (1996), *Applied Cryptography, Protocols, Algorithms, and Source Code in C*, John Wiley & Son, Inc., Chapter18.
- Talwar, V., Nahrstedt, K., and Nath, S. K., (2001), "RSVP-SQOS: A secure RSVP protocol", *2001 International Conference on Multimedia and Expo*, pp.579-582.
- Tschofenig, H., (2003), "RSVP security properties", *Internet draft* (working in progress).
- Wang, G., (1994), "Inter-network Security", *Singapore ICCS'94 Conference Proceedings*, Vol. 3, pp.14-18.
- White, Paul P., (1997), "RSVP and integrity services in the internet: a tutorial", *IEEE Communications Magazine*, Vol. 35, Issue: 5, pp.100-106.
- Wu, T.-L., Wu, S.F, Gong F.M., (1999), "Securing QoS: Threats to RSVP messages and their countermeasures", *The Seventh International Workshop on Quality of Service (IWQoS)*, pp.62-64.
- Zhang, L., Deering, S., Shenker, S., and Zappala, D., (1993), "Rsvp: A new resource reservation protocol", *IEEE Network*, Vol. 7, Issue: 5, pp. 8-18.
- Zheng, Y., Josef P., and Jennifer S., (1993), "HAVAL - A One-Way Hashing Algorithm with Variable Length of Output", *Advances in Cryptology - AUSCRYPT'92, Lecture Notes in Computer Science*, Springer-Verlag, Vol.718, pp.83-104