

SANDIA REPORT

SAND2003-3457

Unlimited Release

Printed September 2003

Final Report for the Multiprotocol Label Switching (MPLS) Control Plane Security LDRD Project

Thomas D. Tarman, John T. Michalski, Stephen P. Black, Mark D. Torgerson,
Lyndon G. Pierson

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation,
a Lockheed Martin Company, for the United States Department of Energy's
National Nuclear Security Administration under Contract DE-AC04-94AL85000.

Approved for public release; further dissemination unlimited.



Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: <http://www.doe.gov/bridge>

Available to the public from

U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: <http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online>



Final Report for the Multiprotocol Label Switching (MPLS) Control Plane Security LDRD Project

Thomas D. Tarman and Lyndon G. Pierson
Advanced Networking Integration Department

John T. Michalski and Stephen P. Black
Networked Systems Survivability and Assurance Department

Mark D. Torgerson
Cryptography and Information Systems Surety Department

Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-0806

Abstract

As rapid Internet growth continues, global communications becomes more dependent on Internet availability for information transfer. Recently, the Internet Engineering Task Force (IETF) introduced a new protocol, Multiple Protocol Label Switching (MPLS), to provide high-performance data flows within the Internet. MPLS emulates two major aspects of the Asynchronous Transfer Mode (ATM) technology. First, each initial IP packet is "routed" to its destination based on previously known delay and congestion avoidance mechanisms. This allows for effective distribution of network resources and reduces the probability of congestion. Second, after route selection each subsequent packet is assigned a label at each hop, which determines the output port for the packet to reach its final destination. These labels guide the forwarding of each packet at routing nodes more efficiently and with more control than traditional IP forwarding (based on complete address information in each packet) for high-performance data flows.

Label assignment is critical in the prompt and accurate delivery of user data. However, the protocols for label distribution were not adequately secured. Thus, if an adversary compromises a node by intercepting and modifying, or more simply injecting false labels into the packet-forwarding engine, the propagation of improperly labeled data flows could create instability in the entire network. In addition, some Virtual Private Network (VPN) solutions take advantage of this "virtual channel" configuration to eliminate the need for user data encryption to provide privacy. VPN's relying on MPLS require accurate label assignment to maintain user data protection.

This research developed a working distributive trust model that demonstrated how to deploy confidentiality, authentication, and non-repudiation in the global network label switching control plane. Simulation models and laboratory testbed implementations that demonstrated this concept were developed, and results from this research were transferred to industry via standards in the Optical Internetworking Forum (OIF).

Contents

1	Introduction	9
2	MPLS Overview	10
2.1	MPLS architecture.....	10
2.2	Signaling label-switched paths.....	12
3	MPLS Control Plane Threat Analysis	13
3.1	Data Content has Value.....	13
3.2	Traffic Analysis.....	15
3.3	Denial of Service	16
3.4	Threat Summary	17
4	Security Approach	19
4.1	Security Architecture.....	20
5	Security Protocol Implementation	22
5.1	Message Processing	24
5.2	Architecture Simulation	25
5.2.1	Security Design	26
5.2.2	RSVP Process Model	27
5.2.3	Security Monitor	33
5.2.4	Security Code Performance	34
5.3	Prototype Implementation	38
5.3.1	RSVP Daemon for DiffServ over MPLS	38
5.3.2	What is DiffServ?	39
5.3.3	Demonstration Network	42
6	Conclusions and Future Work	46
7	References	47

Figures

Figure 1: Label switching example	10
Figure 2: Nested LSPs and logical adjacency	11
Figure 3: Proof-of-concept architecture	20
Figure 4: Monitored MPLS Network.....	25
Figure 5: OPNET Label Edge Router Model	26
Figure 6: RSVP Model.....	28
Figure 7: Test Network Model	34
Figure 8: Simulated Throughput	35
Figure 9: Simulated Aggregate Load for 50 LSPs (packets/sec).....	36
Figure 10: Simulated Tunnel Delay	37
Figure 11: Simulated CPU Utilization	38
Figure 12: IP Header - Type of Service is circled	40
Figure 13: Network Diagram for MPLS Monitoring Demonstration.....	43

Acronyms

AF – assured forwarding

ATM – Asynchronous Transfer Mode

BGP – Border Gateway Protocol

CR-LDP – constraint routed LDP

DiffServ – differentiated services

DoS – denial of service

DS – diffserv

DSCP – diffserv code point

EF – expedited forwarding

EXP - experimental

IETF – Internet Engineering Task Force

IP – Internet Protocol

IPSec – IP Security

FEC – forwarding equivalence class

LDP – Label Distribution Protocol

LER – label edge router

LRO – label request object

LSP – label switched path

LSR – label switching router

MAC – media access control

MPLS – MultiProtocol Label Switching

OIF – Optical Internetworking Forum

PHB – per-hop behavior

QoS – quality of service

RCPM – RSVP control plane monitor

RFC – Request For Comments

RSVP – Resource Reservation Protocol

RSVP-TE – RSVP with traffic engineering extensions

SLA – service level agreement

SNMP – Simple Network Management Protocol

VPN – virtual private network

1 Introduction

MultiProtocol Label Switching (MPLS) is an emerging network protocol that provides mechanisms for establishing flows through an Internet Protocol (IP) network. These Label Switched Paths (LSPs) allow network operators and users to direct traffic through an explicitly defined network path, allowing default traffic routes to be overridden if network conditions require it (e.g., network congestion). This ability to establish LSPs allows users or network management personnel to assign Quality of Service (QoS) attributes, and to perform traffic engineering.

MPLS LSPs are established in two ways – 1) manually via management access protocols, and 2) automatically via label distribution “control plane” protocols. Prior to this project, the control plane protocols were minimally secured using mechanisms that did not scale well to large networks. As a result, practical deployment of MPLS networks did not address security for these control plane protocols (this problem also applied to photonic switched networks, which use the same MPLS control plane). This lack of security for MPLS control plane protocols is problematic because these protocols affect network state, and if the protocol or the Label Switched Routers (LSRs) are compromised, then the network can be “taken down” by an adversary.

This project developed a security architecture that addressed the shortcomings of the MPLS and optical network control plane protocols. One component of this architecture is an IP Security (IPSec) protection profile that provides confidentiality and integrity protections for control plane flows between LSRs. The second component is a network monitor that monitors control plane protocol actions and compares them with the network policy to determine whether violations are occurring, and if so, which LSR may be causing the violations.

Initially, this project developed a network simulation that included both components of this architecture. This simulation provided early validation of the architecture’s efficacy and scalability, and it also provided a “virtual testbed” for algorithm experimentation. Later, this project migrated the simulation implementation into a testbed of Linux MPLS LSRs, and a working implementation of the architecture was demonstrated by the College Cyberdefenders in August 2003.

The results from this work were primarily transferred to industry via the Optical Internetworking Forum (OIF). The OIF is an industry group that is responsible for developing implementation agreements (specifications) for interoperable optical networking equipment. In this case, Sandia worked with representatives from industry and government to develop IPSec protection profiles for optical control plane protocols, and auditing functions for optical switches. In addition, this project was partially responsible for licensing to industry related technology for securing Asynchronous Transfer Mode (ATM) control plane protocols.

2 MPLS Overview

MPLS is a method for controlling the flow of IP packets (datagrams) and quickly switching them through a network. When standard routers forward IP datagrams, they make routing decisions on a per-packet basis using most of the contents provided by the IP header. However, datagram switching is much faster than routing because it uses a path setup protocol to establish a path through the network. Once the path is established, a small flow identifier is added to the packet, which allows datagram forwarding by switching hardware rather than route lookups.

MPLS borrows many concepts from ATM, including the ideas of path switching, dynamic flow setup, and support for traffic engineering. The following sections describe these architectural concepts in more detail.

2.1 MPLS architecture

MPLS is a set of Internet specifications that describes interoperable mechanisms for implementing IP switching. An example of using MPLS for IP switching is shown in Figure 1. MPLS Label Switching Routers (LSRs) switch IP datagrams over a Label Switched Path (LSP) that traverses an MPLS Domain. The LSP is established manually via management access or via signaling.

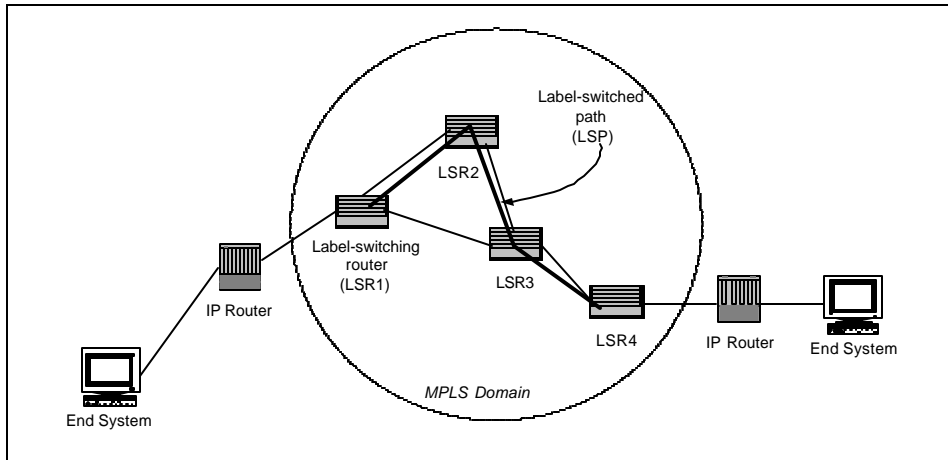


Figure 1: Label switching example

Switching on an LSP is supported by a label affixed to the IP packet by an MPLS ingress LSR (LSR1 in Figure 1), and the label is removed by the egress LSR (LSR4 in Figure 1) when the datagram leaves the MPLS domain. Whereas conventional IP routing looks at the IP header to make forwarding decisions, MPLS forwarding decisions are based on a small label affixed to the packet in a “shim header” between the IP and the link layer protocol (e.g., Ethernet MAC) headers. This label is

assigned to the packet's Forwarding Equivalence Class (FEC), which in practice, is a network prefix. Therefore, all packets belonging to the same FEC (i.e., having the same destination address prefix) are assigned the same label, and are switched along the same LSP. Labels may be unique to the LSR interface (interface label space), or unique to all interfaces on the LSR (platform label space).

LSRs may be physically or logically adjacent. If LSRs are logically adjacent, then they are connected through another LSP or through a special tunnel, resulting in a hierarchy of LSPs, as shown in Figure 2. This hierarchy allows the MPLS network to scale, and also allows end-to-end MPLS switching through different administrative domains.

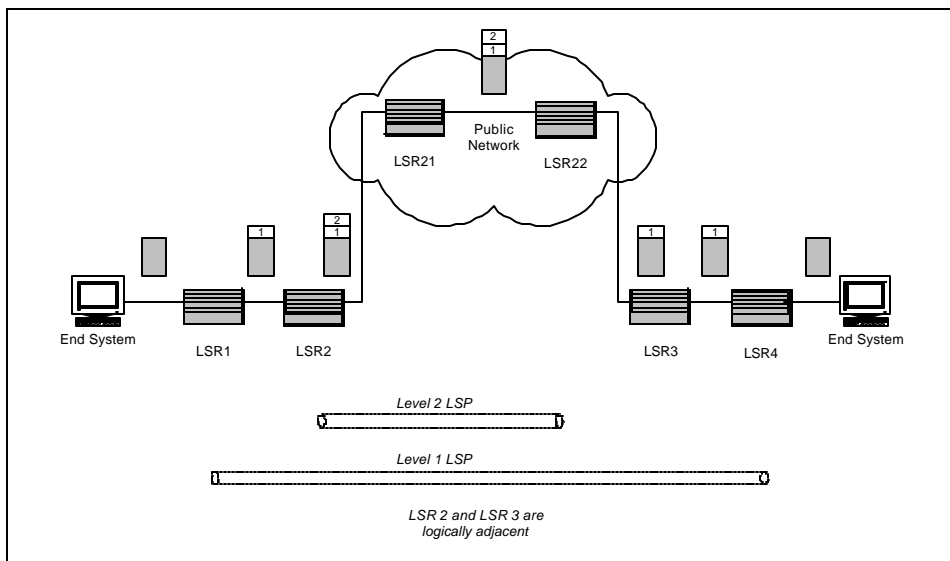


Figure 2: Nested LSPs and logical adjacency

LSPs may be configured manually or via signaling, and like ATM virtual circuits or virtual paths, LSPs allow traffic engineering. This permits a network operator to override the automatic path that a packet would take (i.e., one determined by normal IP routing), and send the packet through one that is configured via management. Standard IP routed networks also support traffic engineering, but it is based on source routing, which has security and efficiency issues. Therefore, MPLS is considered a better solution for engineering traffic flows in a backbone network.

When LSPs are configured via signaling, a label distribution protocol is required to allow LSRs to inform other LSRs of bindings between FECs and labels. The binding between a FEC and a label is normally determined by the downstream LSR and communicated to the upstream LSR via signaling. Once communicated, the signaling protocol is used to maintain the binding and tear it down when it is no longer needed. As described below, multiple protocols for label distribution are

defined by the Internet Engineering Task Force, including the Label Distribution Protocol with extensions for Constraint Routing (CR-LDP) and the Resource Reservation Protocol with extensions for Traffic Engineering (RSVP-TE), with RSVP-TE emerging as the dominant protocol.

2.2 Signaling label-switched paths

MPLS does not specify a particular protocol for disseminating FEC-label bindings in a network of LSRs. However, there are two fundamental approaches that can be taken. The first approach is to take an existing label distribution protocol and extend it to support resource reservations and explicit routing (which in turn supports traffic engineering). The most common protocol that adopts this approach is CR-LDP. The second approach is to take an existing resource reservation protocol and extend it to support explicit routing and labels, and is most commonly represented by RSVP-TE. In this paper, the term LDP is used to refer to CR-LDP, RSVP refers to RSVP-TE, and the term “label distribution protocol” refers to the generic signaling protocol that distributes labels.

In general, a label distribution protocol allows two LSRs to agree on the meaning of labels used to forward traffic between them. This allows the creation, maintenance, and deletion of LSPs within the network.

When RSVP is used to distribute FEC-label bindings, special RSVP extension objects are used to carry MPLS labels in the RSVP PATH and RESV messages. In addition, routing protocols such as Border Gateway Protocol (BGP) can also be used to distribute labels. When BGP is used to distribute a route, it also distributes a label for that route. This method can also work with route reflectors.

3 MPLS Control Plane Threat Analysis

The first step in the development of a security approach in a network environment is to define an appropriate threat model. The following is the description of adversarial threats considered when designing our security architecture.

The network under consideration is owned and controlled by single entity. The ingress/egress nodes are the nodes that are in physical communication with nodes in another network. The transit nodes can only communicate with other network nodes.

The adversary may have three goals for manipulation of signaling messages.

1. The adversary wishes to obtain network messages because the data content has value to the adversary.
2. The adversary wishes to do traffic analysis.
3. The adversary wishes to degrade the capability of the network.

Any money obtained from QoS considerations goes to the owner of the network. An adversary will not realize any direct financial benefit (derived from QoS considerations) from a malicious reroute. For instance, if the adversary manipulates flows so that high quality and value flows are routed through a particular node, the network owner will not pay him for his work. If he helps the overall throughput of the network, he probably is not considered an adversary. His only motivation, from a QoS sense, is to harm the overall throughput of the network.

We assume that the control channel is secured through some sort of link-by-link security feature. An adversary may view and/or delete any control message, but any altered message will be discarded. As such the adversary may sever physically or logically any control link that appears in the network that he can gain access to. We assume that to manipulate further the control messages the adversary must compromise one or more nodes. Note however, that if one can take only the word of the two link ends, it may not be possible to tell if a disruption is generated at one end of a link, in the middle of the link, or the other end of the link.

3.1 Data Content has Value

Suppose that an adversary wishes to attain specific data flows because the data content has value to the adversary. To get hold of the data, the adversary may sit and hope the data happens to pass through his hands or he must devise a way to draw the data to himself.

If the adversary has access to the communication channel at a particular spot in the network, there is a certain probability that the data will pass through the link that

he is monitoring. In order to increase that probability the adversary must either gain access to another and/or better link or the adversary must compromise a node.

If we assume that a node has been compromised, there are several ways to increase the probability that the sought after data passes through his hands. The following is an incomplete list.

1. A node may generate traffic roughly orthogonal to the flow of the desired traffic. If ingress nodes make path decisions based on the actual, available, reported bandwidth, then the extra traffic may cause the ingress node to route around the blockage and through the compromised node. This idea works well provided that the network topology is planar in nature. On the other hand if the topology is non planar, the compromised node may have to swamp several flows to effectively block more logical paths. If route decisions are based on bandwidth that is left over after QoS commitments, then this may not work as described unless the links in the path are saturated and cannot forward any additional traffic. In this case, the ingress node may keep trying other paths until one work.
2. A compromised node may advertise false routing information. It may advertise that it has much more available bandwidth than it actually has. This will tip the scales in its favor.

A particular compromised node may not have the resources to process the desired data. It may need to forward the data to another compromised node or some other node out of the network so that it can be saved and/or processed there. Either the compromised node reroutes the traffic through the other node or else makes a copy of the data and sends the original on its way and the copy to the other node. In the latter case, no signaling rule need be violated in order to copy and pass the data. All that is required is that at some time in the past the node with the desired data was legitimately in communication with the node that it wants to pass the data to. The compromised node must keep the path alive somehow and pass data along the legitimate path when it is needed. In the case that the path is active with legitimate data, the compromised node may append collected data to the legitimate flow. The recipient node may strip it off when received. If the data rate of the desired information is not too high when compared to the legitimate data, this covert channel has little chance of being detected.

The desire of the adversary is to acquire certain data, and as mentioned above, he may also be able to sniff the data channel on which the data normally flows rather than compromise a particular node.

In all cases, the best fix is for the end users to set up a secure tunnel to protect the data. If the end users do not protect the confidentiality of their data, it is not the network's responsibility to do it for them; it does not own the data. (There may be an argument about the network's responsibility to ensure the integrity of the data that it routes.) Unless the end users request confidentiality, a forwarding network has little incentive to provide it. If the confidentiality service is an option, then the end users will likely be willing to pay for it both monetarily and in throughput because of processing delays.

It must be clear that as long as the user plane data remains in the clear, there may be an adversary motivated to manipulate the control data in order to attain user data. Any such manipulations are likely to degrade the overall throughput of the network, which in turn affects the bottom line of the network. It appears that the only way to remove such a temptation to the adversary is to encrypt the data from ingress node to egress node. However, such an endeavor comes at a cost in overall network performance. A network must weigh that cost against the expected degradation in performance associated with reroutes because an adversary desires data content.

Finally, it is not reasonable to assume that an adversary will reroute encrypted traffic so that it can be saved for a later time so that it can be decrypted at a later time. If the adversary has the ability to break a well implemented, solid cipher, it will have the resources to do just about anything else to the network.

3.2 Traffic Analysis

Traffic analysis is an attempt to gain information about the network in which the traffic flows. This information is independent of the data that flows may contain. Typical analysis will seek to understand how traffic flows through the network, determine critical nodes, and critical network paths, etc. When the network has a fairly flat nodal job description, global traffic analysis makes little sense. Often the form and function of every node is known to begin with. For instance, if the job of every node in the network is to route and forward traffic, little can be gained by watching the network to see what happens. Further, to accomplish this type of analysis one must have access to data produced throughout the network. That can be accomplished with many sensors or with access to a collection point such as a network monitor.

When discussing traffic analysis it is important to understand what network the adversary is attempting to analyze. An adversary may have no interest in analyzing the network that employs MPLS. He may really be interested in analyzing a network whose end users reside outside of the MPLS network, yet use the MPLS network to assist in their communications. The adversary may desire to monitor the target network traffic as it passes through the MPLS network. If he must forward the information to another node, then (generally) he forwards less information than is contained in the original data. In any event, the user-plane traffic need not be rerouted.

In some instances the adversary may not have access to the flows that he desires to analyze. In this case, he must draw the flows to himself. This can be accomplished with exactly the same means that he might use to draw data to himself because the data content has value. He may do this even if the data portion of the flows is encrypted, because the goal of traffic analysis has nothing to do with the data encrypted or not. If the end users desire that their traffic not be analyzed they must specify explicit routes and hope that these routes do not happen to contain a compromised node and that no link in between is being monitored.

The MPLS network may not care that an adversary desires to analyze traffic from another network except that reroutes are likely to negatively affect network performance. Since, at worst, a compromised node need only forward condensed flow information, it need not reroute flows it already has. It is only in the case that a node needs flows that it does not already have that it would attempt to subvert control information. There are several options to prevent traffic analysis, however these all come at a price. As with the confidentiality problems above, the MPLS network may not have any real incentive to remove the problem.

3.3 Denial of Service

An adversary that desires to harm the network by degrading its performance has many options. These options include physical attacks on the equipment, user-plane attacks, as well as control plane attacks. We will not discuss physical attacks further.

If the adversary has access to the channel, he has the ability to modify and/or delete any data found in the channel. This is true regardless of any cryptographic security mechanisms applied to the data that is being sent. With access to the channel the adversary is able to sever the link he has access to. Presumably, when a link is restricted or shut down, traffic will, if possible, be rerouted around the trouble spot. Nodes close to the downed link will see an increased amount of traffic and congestion. Even though this type of attack is difficult to mitigate, the effects are fairly local in nature.

If the adversary has access to the user-plane channel he may also have access to the control-plane channel. We assume that the data in the control-plane channel is secured, so the adversary cannot interject unauthenticated messages into the channel. However, all received control messages must pass through computationally intensive authentication processes. By sending a flood of bogus control messages each of which have to be processed the adversary may be able to shut down the control-plane of the nodes he has access to. If a node is unable to process legitimate control messages it will eventually lose its ability to correctly process user-plane data.

Because the intent of MPLS is to efficiently switch datagrams through the network, cryptographic security means probably will not be applied to the switching information in the packet headers. This means that an adversary with access to the channel will be able to modify the switching information to send packets down wrong LSPs. The adversary may modify information deeper in the packet that only the egress node would be able to detect, or information that only the end user could be able to detect. In any of these cases, the corrupt information may pass through the network before it is detected as corrupt. Which in turn may initiate a resend of the information and thus the network's overall performance is degraded. The effects of these attacks are felt throughout the network as corrupt packets are forwarded through the network.

Since the switched packets are not authenticated en route, the adversary with channel access may generate locally valid looking packets to the full capacity of the channel. The falsely generated packets may progress through the network in any number of ways.

Thus far in this section we have described situations where the adversary is able to disrupt network operation without ever compromising a node. Further, the DoS mechanisms described above are independent of the security features applied at the control-plane. Because, there is no hope that a secure control plane will be able to prevent all denial or degradation of service attacks, one must be careful to understand the overall impacts of security mechanisms applied to the control-plane.

If a node is compromised, the adversary will certainly have the ability to accomplish any DoS attack describe thus far, but will also have additional abilities. If an ingress node is compromised, it may generate traffic of any sort, at will. It may flood the control plane of the entire network with bogus messages. It may reserve very high bandwidth, very high priority, and very explicit circuitous routes. On the more subtle side, a compromised ingress node may just bump up the bandwidth reservation requirements of all valid flows by a certain percentage. It may choose the worst, but still within tolerance paths. One compromised ingress node may completely shut down the entire network. This can happen independent of any additional control-plane security mechanisms applied. These must be hardened to the highest level.

Compromised transit nodes may wreak havoc as well. If the route set-up messages are not globally authenticated, a transit node may mimic an ingress node and generate very high bandwidth, very high priority, and very explicit circuitous routes. This will have just as serious of an effect on the network as when an ingress node generates the false requests. However, there are security mechanisms that can be applied to stop this behavior, if the network is willing to pay the price. A network monitor may be able to track down the origin of the bad behavior, but simple monitoring will not prevent that behavior.

3.4 Threat Summary

This project addresses the detection/isolation of MPLS label switched routers (LSRs) that have been compromised. A compromised LSR can be used to do the following:

1. Establish unauthorized label switched paths (LSPs). An LSP that advertises connectivity to an IP subnet can be re-routed in manner that allows examination of traffic.
2. Advertise false routing information or LSP/label mappings. Not only does this facilitate re-routing, but it also corrupts correct routing of LSPs, resulting in denial of service.

Also important is a consideration of what is *not* in the threat model. This project is not concerned with protection of the inter-LSR control plane channels, as this

protection is provided by conventional IPSec mechanisms, as described in the Architecture section below. However, even with IPSec mechanisms in place on the control plane connections, compromise of LSR's is still possible, e.g., unauthorized management access using inadequately protected passwords. In addition, erroneous LSR operation may also occur if the LSR has been inserted correctly, but configured incorrectly.

4 Security Approach

This project developed mechanisms to allow global (within a BGP routing domain) monitoring of LSP creation and deletion events. With global monitoring, the following can be detected:

1. LSP establishment that is contrary to domain policy. If an LSR is compromised, and it attempts to establish LSPs in an unauthorized manner, then the neighboring LSRs will report LSP establishment events to a global LSP monitor. The monitor will compare these events against a policy database and a routing topology database to determine if the LSP is unauthorized.
2. Label advertisements that are contrary to domain policy and/or current network state (this is closely related to item # 1). When an LSR receives a label advertisement, it will notify the global LSP monitor. The monitor will compare these advertisements against the LSP policy database and the topology database to determine deviations.
3. Collusion between two or more LSPs (and identify which LSPs are involved). This approach relies on the correlation of events from uncompromised routers to determine if an intrusion has occurred. If two or more LSPs are colluding, then they can report false events to the monitor in an attempt to mask the unauthorized behavior. The monitor will compare LSP and label distribution protocol (LDP) events to determine whether unauthorized activity is occurring, and which LSPs are involved.

This project developed a proof-of-concept architecture for monitoring MPLS control plane events. This architecture was implemented and simulated in the OPNET network simulation package. New algorithms for detection of unauthorized behavior with respect to LSP policy and routing topology state were also developed and implemented in the OPNET simulation. In addition, algorithms for detection of collusion between multiple LSRs were examined during the initial study phase, with emphasis given to solutions of the “Byzantine Generals Problem”.

An OPNET simulation model was developed and used to evaluate the system according to the following metrics:

1. Network traffic overhead. The overhead associated with the extra encapsulation headers for IPsec control channel protection, key exchange protocol messages, and LSP/LDP event notifications.
2. Latency associated with establishment of legitimate LSPs. If the monitor must be in the loop, then a delay will be incurred in the establishment of LSPs.

3. Load on LSP monitor. Incoming event messages from a network of LSRs to a centralized LSP monitor will cause processing load on the monitor. As the number of LSRs increases, so will the load.

4.1 Security Architecture

The architecture for the proof of concept system is illustrated in Figure 3.

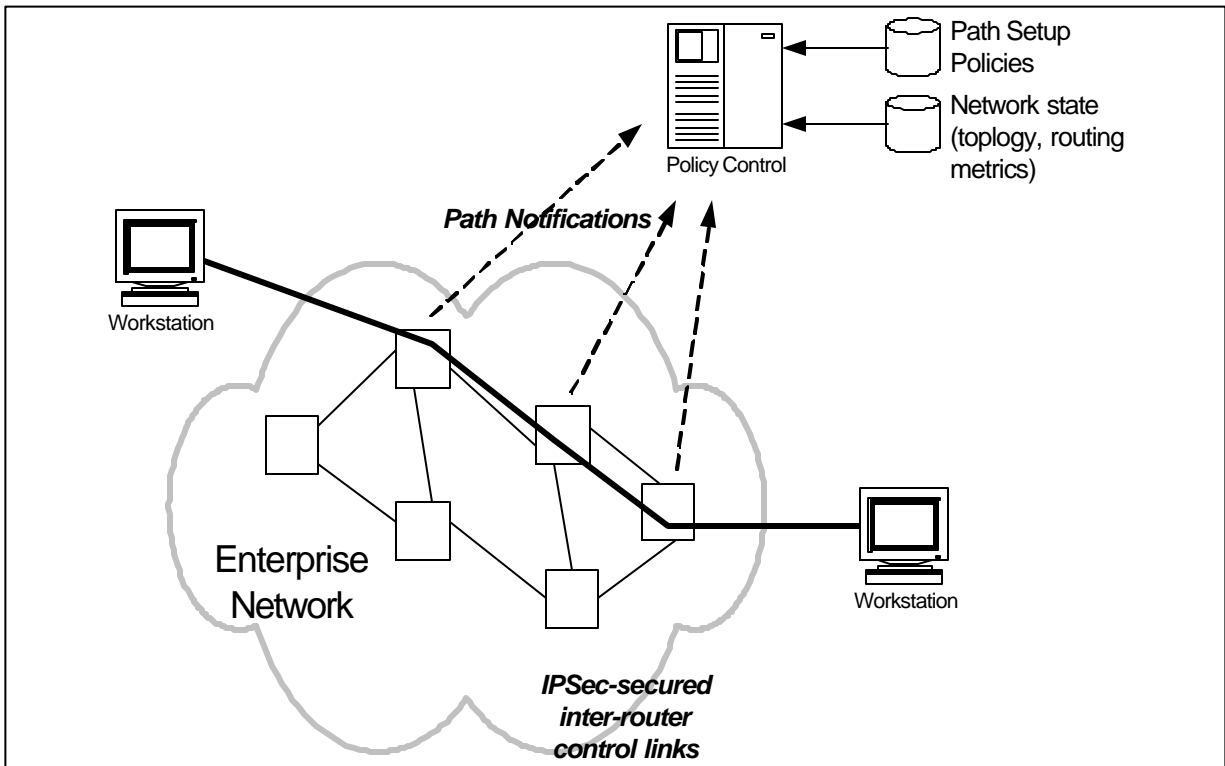


Figure 3: Proof-of-concept architecture

In this architecture, the control channels that link each of the LSRs are secured using IPsec. Also, each LSR is configured with the address of the Policy Control monitor, and sends path notifications to it over an IPsec-protected channel.

A Policy Control monitor is configured to receive path notification messages from the LSRs over IPsec-secured links. Upon arrival of the path notifications, the monitor examines a Path Setup Policy database and a Network Topology database to determine whether the incoming path notifications are consistent with policy. (Note: traffic engineering may require that a path be setup in a manner that is *not* consistent with the current network topology. In this case, only the Path Setup Policy database is consulted.)

The Policy Control monitor uses the incoming path notifications to determine whether an LSP is authorized, and if not, it determines which LSR is compromised.

If multiple LSRs are compromised and colluding, then the monitor detects which LSRs are involved.

5 Security Protocol Implementation

Our security mechanism provides a method of monitoring Multi-Protocol Label Switching (MPLS) for proper operation in an adversarial environment. MPLS is a set of standards for IP switching which are designed to provide low-latency, and differentiated transfer services of IP datagram flows across an MPLS domain. MPLS does not provide security mechanisms that assure proper network operation. An MPLS control plane signaling protocol called Resource Reservation Protocol with Traffic Engineering (RSVP-TE) governs the setup and control of these flows. This MPLS control plane signaling protocol has been specified in the Internet Engineering Task Force (IETF) in a series of Request For Comments (RFCs) and will become an important set of protocols for IP data transfer in enterprise core networks as well as the Internet.

This MPLS control plane protocol is susceptible to exploitation by adversaries attempting to create Denial of Service (DOS) situations and unauthorized Quality of Service (QoS) gains. This RSVP-TE Control Plane Monitor (RCPM), which resides on participating Label Edge Router's (LERs) and a standalone monitor platform, *can provide detection and isolation of Label Switched Routers (LSRs) that have been compromised*. A compromised LSR can be used to do the following:

1. Establish unauthorized Label Switched Paths (LSP's). An LSP that advertises connectivity to an IP subnet can be re-routed in a manner that allows examination of traffic.
2. Advertise false routing information or LSP/label mappings. Not only does this facilitate re-routing, but it also corrupts correct routing of LSP's, resulting in denial of service.
3. Modify quality of service attributes to either increase or decrease the authorized quality of service for a specified flow. This can also create a denial of service scenario by disallowing the establishment of a LSP due to an excessive service subscription request.
4. Tear down authorized Label Switch Paths by injecting unauthorized path and reservation (RESV) tear down messages.

The RCPM is not concerned with protection of the inter-LSR control plane channels, as this protection can be provided by conventional IPSec mechanisms, although even without this protection the RCPM can detect modification of LSP control messages that occur on any RCPM participating LSR to LSR inter-link connection. However, even with IPSec mechanisms in place on the inter-link control plane connections, compromise of LSR's are still possible, e.g., adversarial compromise of control plane code residing on an LSR, unauthorized management access using inadequately protected passwords. In addition, erroneous LSR operation may also occur if the LSR has been inserted correctly, but configured incorrectly. The RCPM function can detect these types of adversarial attacks and mis-configurations.

The RCPM has developed mechanisms to allow global (within a BGP routing domain) monitoring of LSP creation and deletion events. With global monitoring, the following can be detected:

1. LSP establishment that is contrary to domain policy. If an LSR is compromised, and it attempts to establish LSPs in an unauthorized manner, then the neighboring LSRs will report LSP establishment events to the global LSP monitor. The monitor will compare these events against a policy database to determine if the LSP is unauthorized.
2. Collusion between two or more LSPs (and identify which LSPs are involved). This approach relies on the correlation of events from uncompromised routers to determine if an intrusion has occurred. If two or more LSPs are colluding, then they can report false events to the monitor in an attempt to mask the unauthorized behavior. The monitor will compare LSP and Label Distribution Protocol (LDP) events to determine whether unauthorized activity is occurring, and which LSPs are involved.
3. Unauthorized LSP tear down request. The RCPM can detect and track the path and reservation tear down message as they propagate through the network.
4. Unauthorized quality of service request that are contrary to the appropriate Forward equivalency Class (FEC) mappings assigned at the Ingress of the network. The RCPM can detect tampering of the assigned QoS attributes of an LSP.
5. Traffic engineered routes that are unauthorized or have been modified from their initial path. Detection of explicit route and record route object modifications.

The RCPM has a reporting function that is installed on participating Label Switch Routers. This reporting function sends a copy of the LSP setup request message that is being transmitted through the network in real time as part of the LSP setup procedure. The LSP setup messages are comprised of two parts; a PATH setup message that originates at the Ingress of a network on a LER and is transmitted through the network hop by hop being processed by each appropriate LSR until it terminates at the Egress, and a RESV message, which is a response to a PATH message, that originates at the Egress of the network on an Label Edge Router (LER) and terminates at the originating ingress LER. Messages sent to the monitor provide a real time monitoring capability for the monitoring software that resides on the monitor platform. The monitor platform can provide LSP tamper detection for an entire domain of LSP routers. Figure 4 shows the communication process between participating LSP routers and the Monitor.

5.1 Message Processing

As the LSP signaling process begins a path message is sent from the “Ingress router” LER of the originating node to the “Egress router” LER of the terminating node. As the LER processes this message and sends it out to the next hop router, it also sends this same message to the monitor through a protected IPsec or equivalent VPN tunnel that each participating router establishes with the monitor. The VPN tunnels are needed to prevent/detect manipulations of message traffic to the monitor. The setup and control of these tunnels are easier to maintain than the link-to-link establishment that would be needed for a traditional implementation of inter-link protection as discussed earlier. As the monitor receives the first message sent from the Ingress router it identifies the *originator* of the message, the *message type*, *PATH* or *RESV* message, and *which LSP* this message pertains too. It then extracts the message object fields that will be used for comparison and temporarily stores these field values. As additional messages arrive from routers reporting the setup of this LSP, the monitor compares each appropriate message pair from each router and makes a real-time or near real time decision on the validity of the message object fields. When the monitor receives the last monitor message for the LSP being monitored it releases the resources that were needed for processing the incoming message, thus allowing resources to be utilized by other incoming messages.

For example, as seen in Figure 4, the first message sent to the monitor would be from the “Ingress router A” port 2 as it forwards the LSP setup request downstream to the next hop router LSR “B” port 1. The LSR B port 1 would then receive this message from the upstream LER “A” port 2 and forward this message to the monitor it would then process the message before sending it out on port 2 to the monitor and also forwarding it downstream to the terminating “Egress router C” port 1. When LER “C” receives the message it also forwards a copy of the message to the monitor and then processes the terminating PATH message. The “Egress router” LER “C” port 1 then launches an associated RESV message upstream to the “Ingress route” that originated the initial path message. This RESV message is transmitted to the “Ingress router” LER “A” in the reverse order that it was received. Along the way, each node sends a copy of this RESV message to the monitor for security processing. The monitor receives each one of the subsequent messages and just like the PATH message processing it uses a compare algorithm that determines if any of the message object fields have been changed. If field tampering is detected, it can then be reported immediately or logged for later review.

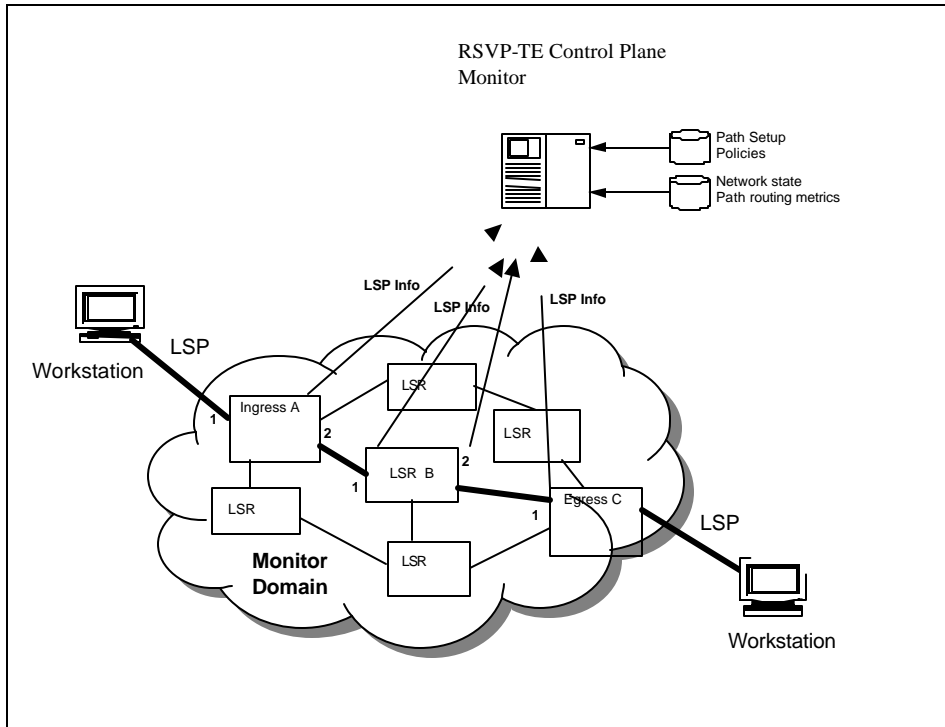


Figure 4: Monitored MPLS Network

5.2 Architecture Simulation

To validate the protocol design a node model was constructed using a network communications simulator. The modeling software used to provide the simulation was OPNET Modeler release 8.0.c and is copyright protected by OPNET Technologies, Inc.

OPNET simulations are event-driven. The simulation kernel maintains a single global event list, time in the simulation advances when an event occurs. All objects access a shared simulation clock and all events are scheduled on the list in time order. When an event is completed it is removed from the list. An event becomes an interrupt when it reaches the head of the event list and is delivered by the simulation kernel to the designated module. The module can obtain any data associated with the event when the interrupt occurs.

A node is a collection of interconnected modules in which data is manipulated as defined by the modules themselves. Modules represent the internal capabilities of a node such as data creation, transmission, processing, internal routing, reception, storage and queuing. Within the OPNET Node editor a variety of different modules can be accessed. The modules are used to model some aspect of node behavior. A single node model is usually comprised of multiple modules. The modules are connected together by packet streams and statistic wires. Packet streams are used to transport data between the modules while statistic wires allow one module to

monitor a varying quantity within another module. The ability to integrate the use of modules, packet streams, and statistic wires allow the developer to create highly realistic simulations of node behavior.

5.2.1 Security Design

As previously mentioned in section 3.0 Security Protocol Implementation, MPLS is governed by a control protocol called resource Reservation Protocol with Traffic Engineering (RSVP-TE). This protocol is used to setup all Label Switched Paths within the MPLS domain and therefore presumed to be subject to adversarial compromise. To build a security mechanism that could detect unauthorized changes and manipulations within this protocol, modifications were needed to the source code of the protocol itself.

The OPNET simulation software has a module called “RSVP” that represents this signaling protocol. This “RSVP” module is contained within many represented communication objects in OPNET’s “object palette”. The objects chosen for our security implementation include, Ethernet Workstations, Ethernet Switches, Ethernet Servers, LER IP Routers, and LSR IP Routers. Within these object models are process models that represent specific supported functions within the object. One of these process modules represents the signaling protocol that implements MPLS. It is referred to as RSVP. Figure 5 shows the LER IP node module and all its associated process models.

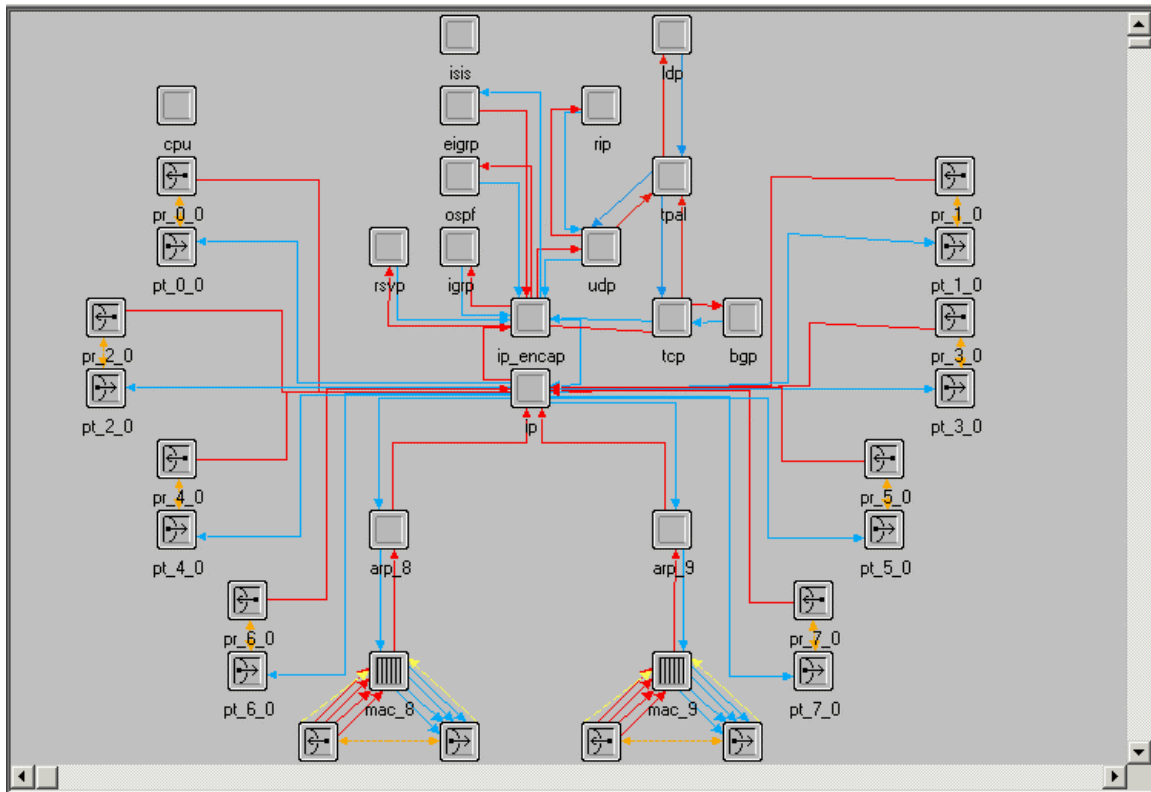


Figure 5: OPNET Label Edge Router Model

All modifications to the RSVP signaling control plane software are in reference to the RSVP process model except for the Modifications located within the ip_encap process block. Modifications within this process block allow RSVP packets, originating from each router node interface, to be routed to the monitor station for processing. All RSVP packets originating from the RSVP process module are identified by “type’ field located within the packet itself. This type identification triggered the ip_encap process block to disregard a standard IP routing table lookup for information pertaining to the routing of the RSVP packet. Because RSVP packets utilize a field that is populated by IP addresses of each RSVP participation node, route lookup is not needed to forward the packet to its next hop. Because a copy of this packet was sent to the monitor it had to use the standard route “lookup” process to allow the packet to reach its destination (the monitor). To accomplish this function a new type identifier was created and inserted within the RSVP packets destined to the monitor station. This allowed the ip_encap process to discriminate between the RSVP standard forwarding packet and the RSVP route lookup packet destined to the security monitor.

5.2.2 RSVP Process Model

The RSVP process model is comprised of all the logic modules needed to emulate the signaling protocol. Different logic states are defined to allow the simulators event based structure to process events in a user defined manner. Figure 6 shows all the processing logic blocks associated with the RSVP process model.

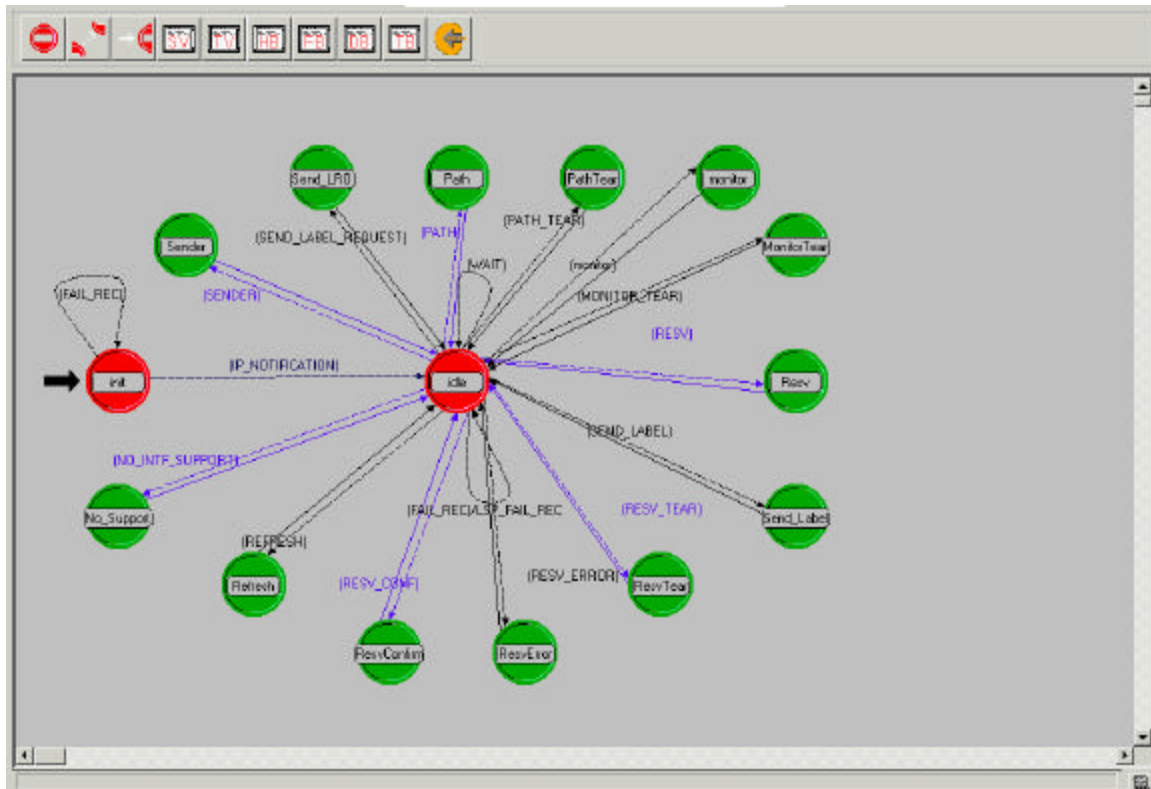


Figure 6: RSVP Model

Within OPNET there are several layers that are associated with each object. These layers, described below provide the designer with a different function. The following is a quick description for use of RSVP in OPNET’s MPLS simulations:

1. Path, Send_Label, Resv, and Send_LRO are all of the “states” within the rsvp “process model” that have modified code. The rsvp process model also has two new state blocks added, a monitor block and a monitor tear block.
2. There are only three objects that use RSVP - they are the Routers, Servers, and Workstations.
3. The levels of the OPNET program are as follows: Project → Object → Node Model → Process Model → State Block. The state block is where the code resides that has been modified.
4. The node models that are used within the objects are as follows, and each contain the modified code which resides in the rsvp process located in the node model:

<u>Object</u>	<u>Node Model</u>
Label Edge Routers	ethernet2_slip8_ler

Workstation
Server

workstation_adv
server_adv

To accomplish our goal of building a secure MPLS signaling protocol modifications/additions to some of the software based process modules needed to be made. The procedures for the modifications were built based on the sequential order of the signaling protocol itself. The first module to be modified was the Send_LRO module. The Send_LRO module is responsible for the initiation of a signaling request. This request is initiated from the Ingress router and launches the PATH setup message for a Label Switch Path (LSP).

The **Send LRO** state is entered when the corresponding Node is an Ingress LER for at least one LSP. In this state, set up all the LSPs originating from this node:

1. Find all the LSPs originating from this node.
2. For all LSPs get the constraints and Explicit route.
3. Create the PATH message with LABEL_REQUEST object.
4. Create a corresponding Path state.
5. Invoke route query to find the list of outgoing interfaces for the path message, and copy them into the path state.
6. Execute Path refresh sequence. This sequence sends path messages to all outgoing interfaces defined in step 2.

This PATH message is sent node-to- node from the Ingress to the Egress of the network. This PATH message contains the following “fields of information” referred to as objects, that pertain to the setup of the LSP.

Label Request Object----- request a label binding, along with label type and label range be associated with the LSP

Explicit Route Object----- predefines an explicit hop-by-hop route from the Ingress router through the network to the Egress, independent of standard IP routing protocols.

Record Route Object----- this field is populated by each router that is participating in the LSP setup. The IP address of each subsequent node is recorded on its way to the Egress router.

Session Attribute Object----- defines the session type of the LSP Ipv4 or IP v6 ,and preemption capability of the specified for the LSP

CoS Flowspec Object----- defines specific attributes of the requested data flow such as inter-packet delay and minimum bandwidth the LSP needs to provide.

It's these object fields the MPLS security monitor protects. To provide this protection each link associated with the setup of LSP reports path information that it processes and forwards to downstream routers. It also reports all resv message information that it receives from upstream routers to the security monitor.

The "PATH" state block provides the following message processing activity. Upon the reception of a PATH setup message, the message type is determined. If the message is "RSVP with label request" a matching path state is found, or if there is none, it is created. If there is a matching state, its content is updated. Any change in forwarding path state will trigger immediate generation of updated Path messages.

To maintain a local node consistency, the modification of Path states can also result in the modification of reservation states, and possibly traffic control states, and reservation update. In that case reservation refresh messages are sent out immediately. There is one path state for each (sender, destination, incoming interface) triple. Since RSVP supports multicast applications, the node may receive more the same session will be used for forwarding Path messages downstream. It will be a state created by a message that arrived on an interface expected by routing. This state will have its *local_only* flag set to off, and will be referred to as "forwarding_state".

Three variables of the type *RsvpT_Path_State** are used to process the message:
path_state_ptr denotes a general path state that was found
matching_path_state_ptr denotes a matching state for an arrived path message
forwarding_path_state_ptr denotes a forwarding path state for the session and sender for the message.

Two flags are used to identify whether the reception of the message changes existing states. In a case of a change refresh messages will be sent out immediately. The flags are:

1. *path_refresh_needed* - if turned on, path state will be refreshed.
2. *resv_refresh_needed* - if turned on, resv state will be refreshed

EVENT SEQUENCE: (based on RFC 2209):

1. Find a matching path state and determine whether it is a forwarding state. If a matching state does not exist, create it.
2. Update the soft state lifetime timer of the matching state.
3. Contact routing to find expected incoming and outgoing interfaces for the session. Determine whether the matching state is a forwarding state.
4. If forwarding path state is new or modified, execute path refresh sequence to maintain local state node consistency; this sequence may turn on resv refresh flag, which will trigger the execution of reservation refresh sequence.

Within the Send_LRO and PATH block software function prototypes were created and code modifications were made to create the PATH message security reporting capability. The following shows a typical code block that performs the forwarding of a PATH setup message to the security monitor.

```
/* send path message to monitor */  
  
printf ("SENDING OUT INITIAL PATH MESSAGE TO MONITOR \n\n")  
  
d = d+1;  
  
printf ("This is the [%d] path message send \n", d);  
  
packet_fields pointer->sender_nd_objid = node_objid;  
  
printf ("The node object that sent this is: %d \n", node_object);  
  
moitor_addr = ip_address_create ("192.0.2.2");  
  
rsvp_monitor_path_send (path_state_ptr, RsvpC_Monitor_path,  
                        OPC_INT_INVALID, iface_table_ptr, refresh_timer, LTRACE_PATH ||  
                        LTRACE_MESG, monitor_addr);
```

Along with protecting the PATH setup message the security component would not be complete without also protecting the subsequent RSVP "RESV" signaling messages. The RESV message contains the following data objects that are associated with the setup of a LSP.

Label Object----- Performs the upstream on demand label distribution process

Record Route Object----- Returns the LSPs path to the sender of the PATH message, also use to find hop-by-hop route back to Ingress

Session Object----- Uniquely identifies the LSP being established

Style Object----- Specifies the reservation style and filter type of the LSP

The RESV function is associated within the Resv and Send_Label process blocks.

The **Resv** state processes received reservation message and resv request from a local application.

EVENT SEQUENCE:

1. If a message was received, determine the interface on which it arrived.
2. Make sure that there is a matching path state. If there is no matching path state, send RERR message.
3. Check for incompatible styles with existing resv states for the same session.

4. Process flow descriptor from the message or from the down call to make reservations. Each flow descriptor is processed separately. a) if there is a matching path state for the message, add its incoming interface into the list of interfaces for which resv refresh will be performed. b) Find or create a reservation state and update it. Any change will turn *new_or_modified* flag on. c) If the new or modified flag is on, execute update traffic control sequence. This may modify the existing traffic control states and the reservations in place. If the result is to modify traffic control states, this sequence will turn on resv refresh needed flag.
5. If resv refresh needed flag is on, execute resv refresh sequence (e.g refresh the states) for each PHOP in the list of interfaces created in 4.
6. Drop the resv message and return.

The **send label** state is entered after an Egress LER of an LSP receives RSVP Path message along with Label_Request object.

EVENT SEQUENCE:

1. Get the Path state made when this PATH Message was received.
2. Get the Information about LSP for which this PATH Message was sent.
3. Make a reservation decision by finding the reservation parameters from LSPs TE Parameters.
4. Form the Label Object.
5. Make and Send Reserve request message along with the LABEL object

Function prototypes have been designed and inserted in the appropriate locations in the code block to forward all RESV message traffic origination and forwarding through the MPLS nodes participating in the setup of an LSP. The following displays a typical code block that forwards a RESV message to the security monitor.

```

/* send the packet to the monitor */

printf ("SENDING OUT INITIAL RESV MESSAGE TO MONITOR \n\n")

packet_fields pointer->sender_nd_objid = node_objid;

printf ("The node object that sent this is: %d \n", node_object);

moitor_addr = ip_address_create ("192.0.2.2");

rsvp_monitor_send (packet_fields_ptr, my_ip_address, monitor_addr,
                  out_intf_index, next_hop_address);

```


5.2.3 Security Monitor

The overall security approach utilizes a link-by-link analysis of signaling information. It starts with the first PATH message generated from the Ingress router and ends when the Egress router receives the final RESV message. Any message objects that are deleted, changed, or manipulated are detected and reported at the security monitor. The code that performs this message signaling verification resides in a newly created logic module called "Monitor" and is part of the RSVP process model as seen in Figure 6.

The code's primary structure is built around temporarily storing LSP PATH and RESV object information that has been sent by each link of the MPLS participating routers. It starts the analyses by first determining the LSP message type, either PATH or RESV. This allows the code to direct packets to the appropriate section in code for processing.

For PATH message processing the code stores the node (router) ID and the LSP object ID used for status reporting. It then stores the CoS flowspec object which contains the QOS bandwidth variable and the QOS time delay variable. It also stores the explicit route object, which contains directed path information for the LSP. As each subsequent router link reports its LSP signaling PATH information the monitor compares the information shared between any two interconnected router links to determine if the information that was sent was the same information that was received. Information that has been tampered with by a man-in-the-middle is quickly detected.

For RESV message processing the code also stores the router ID and the LSP object ID. This is used for status reporting but also used to correlate PATH and RESV messages that belong to the same LSP. The code then stores the label that has been assigned by the downstream router and reported by the subsequent upstream router.

Our security approach to LSP control plane tamper detection is dependent on hardening all Ingress and Egress routers associated with MPLS path setup. Along with hardening the edge devices IPsec tunnels are created from each LER and LSR device to the distant Monitor. This protects the reported signaling information from compromise prior to reaching the Monitor. If this procedure is followed then the security code can capture tampering even if there is adversarial code on routers reporting false information and routers participating in collusion. This can all be detected because there is a high level of certainty that the first report originating from the Ingress router has not been tampered with. Figure 7 shows a MPLS network scenario, which is the first level in the OPNET program, the Project level. This is where Nodes are placed and are connected to each other to form a network. There are several workstations, servers, switches, and routers, each connected with either a red or black line. These lines represent either a 100 Base T Ethernet connection (red), or a T1 or T3 connection (black). Workstations look like

workstations, while servers are the gray rectangles, switches are the gray squares, and the MPLS routers are the blue and green cylinders. The other lines that can be seen in various colors represent Label Switch Paths.

Just above the network within the Project there are 5 icons, these are attribute boxes that are user interfaces to access specific network parameters that influence the behavior of the project objects. All of the objects within a project are selected from a pallet that allows you to select the objects as either generic or specific (i.e. Cisco) products. For example you can select a generic workstation, which is connected to a 3Com Switch, which in turn is connected to a Cisco router.

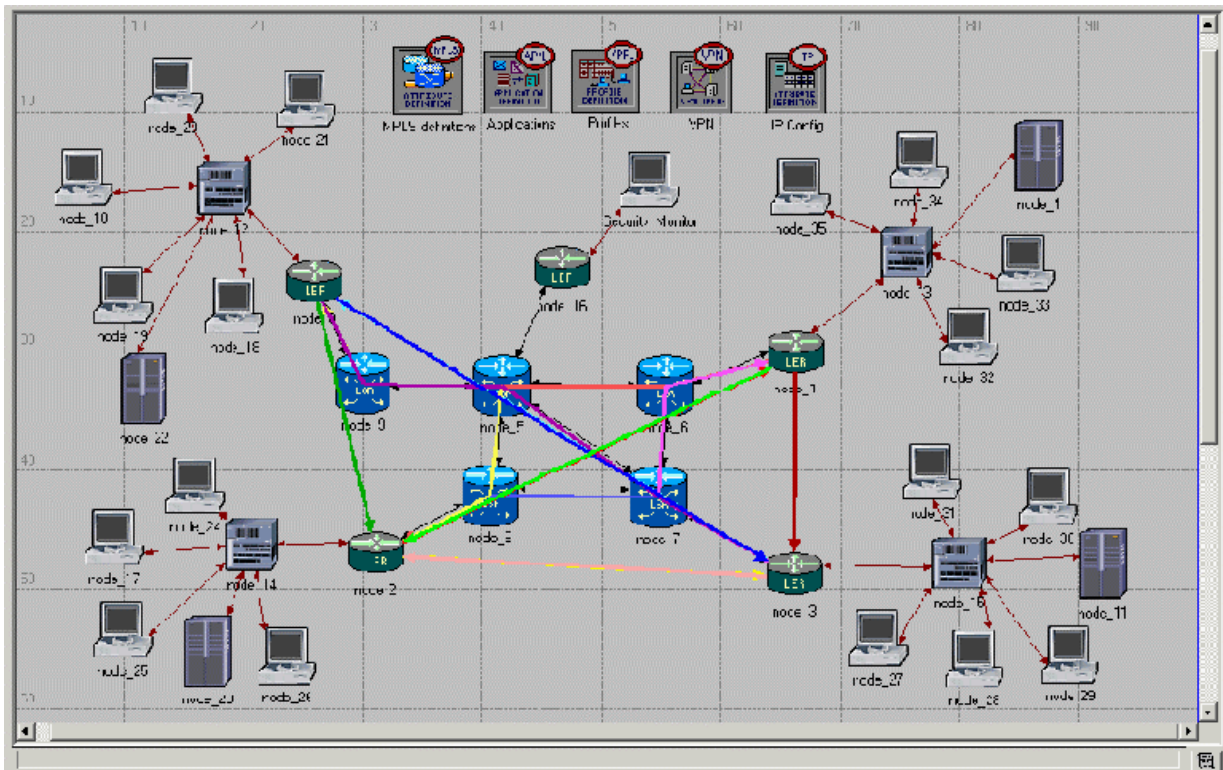


Figure 7: Test Network Model

To add to the assurance that reported information to the monitor has not been tampered with encryption in the form of IPsec tunnels are also included in the overall security architecture.

5.2.4 Security Code Performance

Whenever a new security practice or approach is introduced into a network system it is important to understand the impact of the new security implementation to the overall network under test. To help characterize the system impact to our security

approach the following statistics, link congestion, VPN tunnel delay, and processor load where gathered to help measure and understand the system impact.

5.2.4.1 Link Congestion

Within the LSP security monitor architecture each LER or LSR link that is participating in the security monitoring of LSP's must report all LSP signaling requests to the security monitor. When this occurs a packet containing the signaling information, either PATH or RESV is routed through the network to the distant security monitor. These packets create additional packet loading on the network links they traverse. This additional loading is a direct result of the implementation of the security architecture and has been simulated to gather information on its impact.

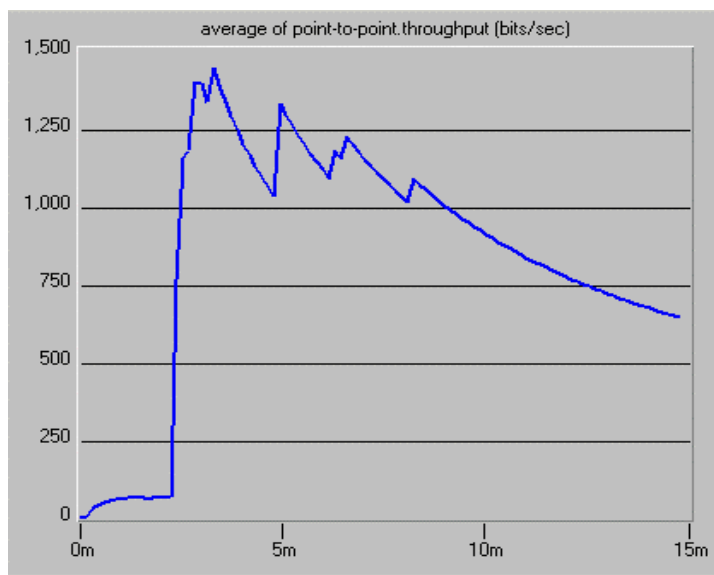


Figure 8: Simulated Throughput

The size of the packet and the volume of packets represent the overall impact to the underlying network. Figure 8 shows the Bits/second link rate seen at the Security Monitor. When this value is divided by the packet/second rate it reveals an average packet size of a LSP reporting message, Figure 9 shows the overall link capacity encountered when 50 LSPs are created through our network scenario as seen in Figure 7.

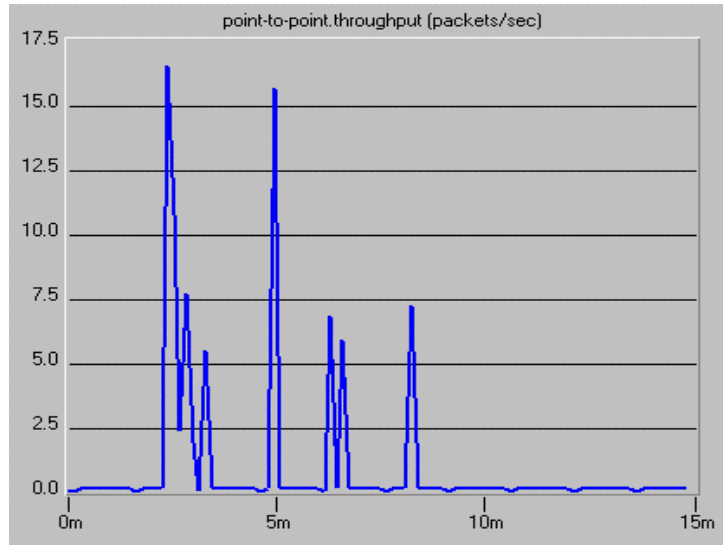


Figure 9: Simulated Aggregate Load for 50 LSPs (packets/sec)

5.2.4.2 VPN Tunnel Delay

Within the LSP security monitor architecture VPN's are created from each participating LER or LSR to the security monitor device. These VPN's are needed to prevent tampering with the control plane signaling information being sent to the monitor. LSP information originated at the label switch devices (LER, LSR) is encapsulated into an encrypted tunnel. The impact to the data routed through an IPsec tunnel is the encryption delay into the tunnel and the decryption delay encountered when exiting the tunnel. Simulations were run to determine the overall average tunnel delay encountered by LSP reporting data. The actual data rate was based on the packet/second rate shown in the previous link congestion Figure 9. Figure 10 shows the average per packet tunnel delay.

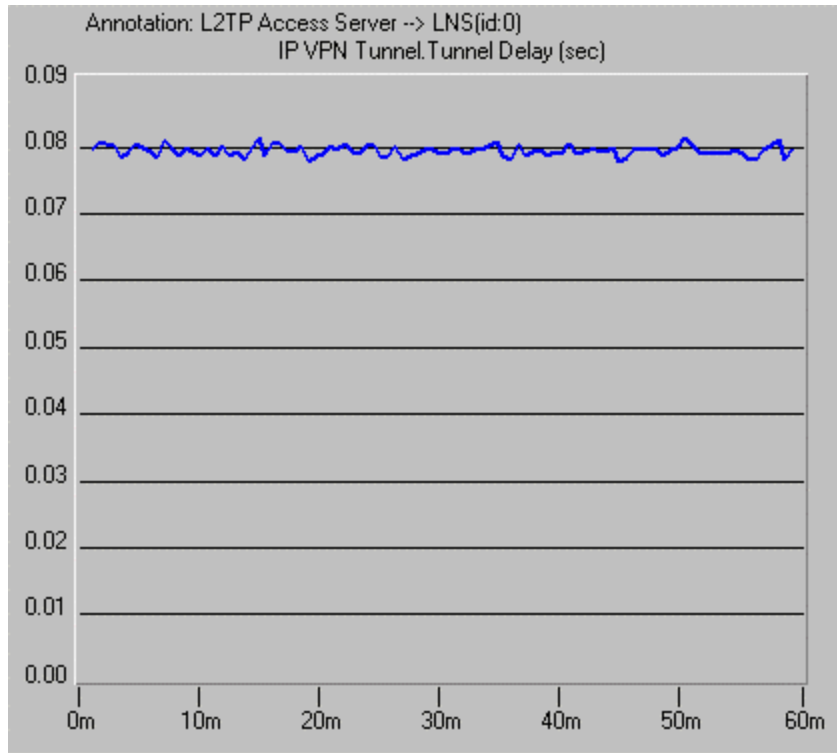


Figure 10: Simulated Tunnel Delay

5.2.4.3 Processor Load

Another important statistic to gather about our security implementation is the impact to the computer processor on the security monitor that is hosting the tamper detection software. Statistics were gathered on the processor when 50 LSP's , and their associated reporting packets, were generated based on the network scenario as shown in Figure 7. Figure 11 shows the overall CPU utilization for the monitor node based on our network scenario.

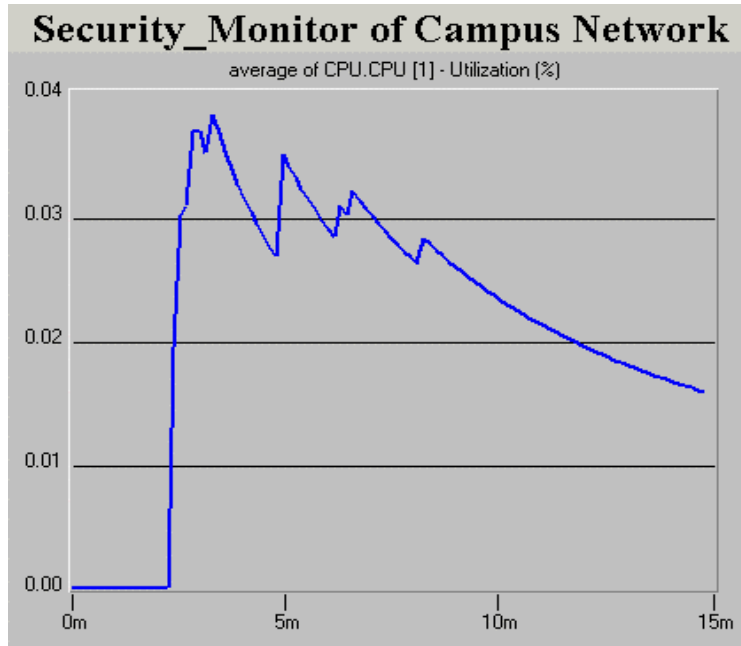


Figure 11: Simulated CPU Utilization

5.3 *Prototype Implementation*

To help validate the architecture and its supporting security code a demonstration network was created as a proof of concept for many of the previously described attributes.

5.3.1 RSVP Daemon for DiffServ over MPLS

The MPLS Linux kernel code that is used for the demonstration network is an extension of the “MPLS for Linux” project on Source Forge, where DiffServ and RSVP-TE support was added. The Source Forge MPLS project concentrated on using LDP to setup LSPs in a network, whereas for this research we need RSVP-TE support instead. The program we officially used is titled “RSVP-TE daemon for DiffServ over MPLS under Linux” and is a Ph.D. project from INTEC Broadband Communication Networks (IBCN), an organization within the Faculty of Applied Science at the Gent University in Belgium.

The current release of the daemon offers DiffServ over MPLS support, the use of multiple routing tables and LSP byte and packet counters. The daemon also offers an RSVP API that allows developers to build custom applications that can interact with the daemon. LSP rerouting and LSP protection switching is also supported. Netfilter is the part of the Linux Kernel that is used to classify the outgoing packets, the quality of service, and queuing for differentiating between flows. The RSVP daemon is responsible for the RSVP signaling, maintenance of the MPLS states,

allocation/installation of the MPLS labels during LSP setup, and freeing and removing labels on LSP tear down.

For the demonstration network the daemon was compiled and run on six PCs running Red Hat Linux 7.1, kernel version 2.4. The only support that is offered for the daemon is either included in the source code or on the IBCN MPLS website. Table 1 presents some of the executables and configuration files used for the daemon. These specific files are referenced in scripts, described later, that were written to enable a monitoring process on the network.

Executable	Description
rsvpd	MPLS daemon
rapirecv_auto	Listens for MPLS packets
rtest2	Sends out MPLS requests
Tunnel	Maps protocols to LSPs
label.conf	Configuration file that defines the label ranges for each interface
lspconfig	Configuration file that defines an explicit or loose LSP.

Table 1. Executables and configuration files for running DS/MPLS

5.3.2 What is DiffServ?

The importance of DiffServ is that it is a type of traffic control architecture for providing different types of service (ToS) for network traffic, which provides a Layer 3 Quality of Service (QoS) solution. A router at the edge of the network identifies packets based on the IP precedence or on the DiffServ Code Point (DSCP) fields in the header. Network devices that support DiffServ use DSCP code points in the IP header to select a per-hop behavior (PHB) for a packet. In the case of the daemon the type of service is based on the value located in the EXP field in the MPLS header.

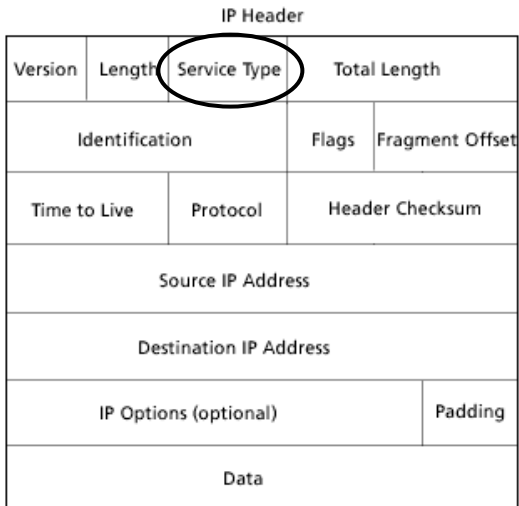


Figure 12: IP Header - Type of Service is circled

The ToS field describes one entire byte (eight bits) of an IP packet. The six most significant bits of the ToS byte are now called the DiffServ field. The last two bits in the DiffServ field are used as Early Congestion Notification (ECN) bits. IP precedence uses three bits, while DSCP, an extension of IP precedence, uses six bits to select the PHB for the packet at each network node. For example if a packet is marked with a DSCP value of 000000 the PHB determines traditional “best effort” service from a DS-compliant node. If a packet arrives at a DS-compliant node and its DSCP value is not mapped to a PHB, it will get mapped to the default “best effort” PHB. The following Table shows the encoding for the ToS bytes.

Precedence Bits – DS5, DS4, DS3 XXX00000	Type of Service Bits – DS2, DS1, DS0 000XXX00
111 = Network Control = Precedence 7	0000 = Normal Service
110 = Internetwork Control = Precedence 6	1000 = Minimize delay
101 = CRITIC/ECP = Precedence 5	0100 = Maximize Throughput
100 = Flash Override = Precedence 4	0010 = Maximize Reliability
011 = Flash = Precedence 3	0001 = Minimize Monetary Cost
010 = Immediate = Precedence 2	Early Congestion Notification Bits 000000XX Currently Unused
001 = Priority = Precedence 1	
000 = Routine = Precedence 0	

Table 2. Original IPv4 ToS byte

There are specific definitions for the DSCP values that define the per-hop behavior for a packet. Expedited Forwarding (EF) PHB provides for a guaranteed bandwidth service, while Assured Forwarding (AF_{xy}) PHB defines a method by which behavior aggregates can be given different forwarding assurances. The EF PHB (DSCP value of 101110) provides for a low-loss, low-latency, low-jitter, and assured bandwidth service [2], ideal for applications such as video streaming and Voice over IP.

The Assured Forwarding (AF_{xy}) PHB defines four AF_x classes, AF1, AF2, AF3, and AF4. Each class is assigned a certain amount of buffer space and bandwidth, dependent on the service level agreement (SLA) with the Service Provider. Within each AF_x class there are three different drop precedence values. Therefore, if there is congestion at a node on a specific link, and packets of a particular AF_x class need to be dropped, packets in AF_{xy} will be dropped such that AF_{x1} <= AF_{x2} <= AF_{x3}. For a better understanding the following table shows the encoding for the AF_{xy} classes and the drop precedence.

Drop Precedence	Class 1	Class 2	Class 3	Class 4
Low	AF11 = 001010	AF21 = 010010	AF31 = 011010	AF41 = 100010
Medium	AF12 = 001100	AF22 = 010100	AF32 = 011100	AF42 = 100100
High	AF13 = 001110	AF23 = 010110	AF33 = 011110	AF43 = 100110

Table 3: DiffServ AF Code Point Table

The daemon uses DSCP to assign quality of service attributes, which are then mapped to the EXP field in the MPLS header, and then is assigned to an LSP. Since the EXP field is only three bits long the daemon has a translation from DSCP to DiffServ/MPLS. What this means is that the Layer 3 QoS has been translated to Layer 2. DS/MPLS can then only support eight different types of service for an LSP, with one type of service dedicated to an explicitly routed LSP (E-LSP). The other seven are shortest path LSPs (L-LSP) that are setup in the network according to IP routing tables, and the DSCP signaling is encoded during LSP setup. Traffic can be mapped on the LSPs based on the destination address, protocol, destination port, and port ranges of the IP packets.

5.3.3 Demonstration Network

The MPLS demonstration network was setup with two “workstations”, two MPLS edge routers, three MPLS core routers, and a “monitor”. The two workstations are configured to communicate with each other across the network through the MPLS routers. The edge routers are where the LSPs are setup and torn down, while the core routers offer two different routes for the LSPs to be directed. Figure 13 is a graphical representation of the network showing how each computer is interfaced, along with the IP addresses, and arbitrary Spanish names given to differentiate between the PCs. The core MPLS routers are represented by the light green boxes, the edge routers are in light blue, and the monitor is in light yellow.

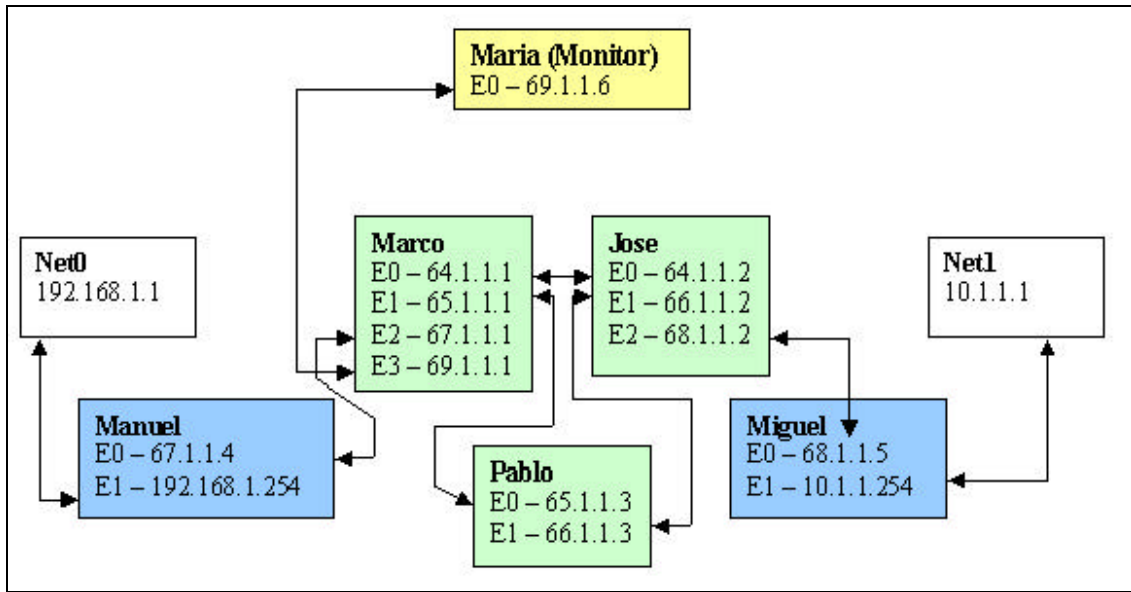


Figure 13: Network Diagram for MPLS Monitoring Demonstration

A few scripts and a couple of programs have been written that will enable the user to remotely start the RSVP daemon and the logging of packets from one computer at the edge of the network, these are the routers Manuel or Miguel. A utility called netcmd was used to send commands remotely from an edge router to each of the other routers so that a user could start/stop/begin-logging MPLS by typing in only one command on one computer, instead of having to repeat that same command on each of the routers. For example, using the command netcmd start_mpls, would start rsvpd and rapirecv_auto on each of the routers. Then using netcmd mpls_logger, this would begin the logging process on each of the routers, and RSVP packets would be sent to the monitor when LSPs are setup across the network. The following table shows each of the scripts, located in /bin on all machines, and their corresponding description.

Script/Program	Description
rsvp_monitor	Located on Maria (monitor), logs the incoming packets from the MPLS routers and makes comparisons between PATH and RESV messages to detect changes.
monitor_mpls	Starts ethereal with a bunch of options including the -f option to filter all RSVP and MPLS packets.
start_mpls	Starts the RSVP daemon on all local machines.
mpls_logger	Located on all the MPLS routers the program uses PCAP to log incoming packets, and then netcat is used to send the packets to the monitor.
kill_mpls	Stops the MPLS rsvp daemon.

Table 4. MPLS scripts for logging packets to the monitor.

The logging program, `mpls_logger`, which resides on each of the MPLS routers, uses PCAP to sniff incoming/outgoing packets on each of the interfaces and then sends a copy of the packet to a specific port on the monitor for processing. During logging, only RSVP packets are sniffed and sent to the monitor. Recall that there are only two different types of RSVP messages, PATH and RESV, so in the case of an LSP being setup across Manuel \approx Marco \approx Jose \approx Miguel there would be six packets for each type of RSVP message being sent to the monitor, for a total of twelve packets in the LSP that is setup. If an LSP were setup from Manuel to Miguel, through Pablo, then there would be 16 total RSVP packets sent to the monitor.

5.3.3.1 Monitor Description

When packets are logged on the MPLS routers and sent to the monitor, it has to “listen” for incoming packets on the specific ports that these packets are to be arriving. It is then tasked to put these packets in order, not in how they have arrived, but in the sequence for an LSP being setup across the routers. It is then the monitor’s responsibility to process the packets to ensure the control plane setup for an LSP hasn’t been compromised, and monitor LSP refresh messages that maintain the LSP.

The `rsvp_monitor` program listens on eleven specific open ports (eleven ports for the different interfaces on the MPLS routers) for incoming packets from the routers. It then uses PCAP functions to read the packet information and compare the packets. The first comparison checks to make sure that what one router has sent to another is what the other has received. So for example, if Manuel sent a packet to Marco, the monitor would receive a packet from Manuel that was sent to Marco, and would also receive a packet from Marco that was received from Manuel, then compare the two packets to make sure they are identical (checking for “man-in-the-middle” attacks). This comparison process occurs for every link on an LSP, so for the LSP example in

the previous section there would be 6 comparisons for the twelve packets sent to the monitor for the LSP.

After the comparisons have been completed the information is then passed to a different function in the program that arranges the PATH and RESV packets for the LSP. Once all PATH and RESV packets have been received the program matches each PATH message to its corresponding RESV message, checking for any changes in the fields, then prints to the display if the LSP setup was successful. Then the RESV messages are compared link by link to ensure labels or LSP ID's haven't been changed.

The monitor was originally designed to display text messages to the console indicating when an LSP has been setup and display alerts when unexpected changes are made to the RSVP messages. This type of display can get to be a bore for a user of the program, just watching text scrolling on a console, so Open GL was used to display a graphical representation of the network, and graphically indicate warnings to the user when message fields have been changed. The demo shows five routers connected by links and when a change has been detected a link will flash red and a warning message will appear.

5.3.3.2 Adversary Scenarios

The same adversary scenarios that were used in the computer simulation were also attempted for the demonstration network, the first being changing the label field in the MPLS header. To do this the `mpls_logger` program was modified (or "hacked" in the adversarial world) so that when a RSVP packet was sniffed and about to be sent to the monitor the adversary code would read in the header information and change the necessary fields. The monitor would then receive its copy of the changed packet and would then have to determine if the field has been changed when it parses through the headers.

Another type of adversarial attack is changing the quality of service attributes in the EXP field in the MPLS header. From the previous discussion about DiffServ there were specific definitions for the levels of QoS. In this case the adversary can manipulate the EXP field and change this attribute so that high priority is assigned to every packet/flow in the network, demanding higher bandwidth through the LSP. This would cause non-MPLS packets to get dropped, thus creating a denial of service. The adversary could do the opposite and assign a low priority to the packets, thus if a video streaming application was using the LSP it would experience a degradation in its service.

The last adversarial attack is to divert traffic to another node by changing the Explicit Route Object in the MPLS header. To accomplish this, another script is used to change the `lsconfig` file which defines the LSP to be explicit or loose and which routers to take. This method has proven to be the most difficult of any of the other attacks because it relies on properly changing the configuration file during LSP setup, if the file is not formatted properly then the LSP never gets setup.

6 Conclusions and Future Work

This project successfully developed an architecture for securing and monitoring control plane communications between MPLS LSRs and optical network switches. This security architecture addressed many threats that were identified during a security analysis that was conducted early in the project. The threats addressed by this architecture include modification of control plane messages by a malicious “man in the middle” and/or a compromised LSR.

This architecture was successfully implemented as an OPNET simulation. This simulation generally showed that the architecture was feasible, and that it could scale to sizes that are appropriate for enterprise networks (e.g., Sandia’s networks). The OPNET implementation was successfully migrated to a real Linux-based implementation for laboratory experimentation and demonstrations.

This architecture also contributed to security specifications in the Optical Internetworking Forum. Specifically, this project contributed to OIF specifications for using IPSec to secure control plane communications between optical switches, and auditing events in an optical network to detect possible malicious behavior.

Finally, this project was also responsible for transfer of network control plane monitoring technology to Bytex Corporation. Bytex licensed network sensor and assessment software from Sandia to incorporate into their next-generation B-6000 network monitoring appliance/firewall.

At this point, the technology has been proven and transferred to industry. Therefore, future work will likely involve further implementation and deployment of the technology. Specific examples include:

- ?? deployment into Sandia’s enterprise networks (e.g., using the Bytex B6000 product)
- ?? deployment into other networks for which Sandia has design responsibilities (e.g., external customer networks)
- ?? development of enhanced MPLS sensors (e.g., sensors that perform sophisticated filtering)
- ?? development or evaluation of reference implementations of the OIF specifications

7 References

1. [P. Van Heuven et al.](http://dsmpls.atlantis.rug.ac.be), "RSVP-TE daemon for DiffServ over MPLS under Linux," <http://dsmpls.atlantis.rug.ac.be>.
2. V. Jacobson *et al.*, "An Expedited Forwarding PHB," IETF RFC-2598.
3. J. Heinanen *et al.*, "Assured Forwarding PHB Group," IETF RFC-2597.
4. D. Awduche *et al.*, "RSVP-TE: Extensions to RSVP for LSP Tunnels," IETF RFC-3209
5. E. Rosen *et al.*, "Multiprotocol Label Switching Architecture," IETF RFC-3031.

DISTRIBUTION:

1	M/S 0451	S.G. Varnado, 6500
1	M/S 0785	R.L. Hutchinson, 6516
1	M/S 0785	S.P. Black, 6515
5	M/S 0785	J.T. Michalski, 6516
1	M/S 0785	W.F. Young, 6516
1	M/S 0801	A.L. Hale, 9300
1	M/S 0801	M.R. Sjulín, 9330
1	M/S 0801	W.F. Mason, 9320
1	M/S 0806	P.C. Jones, 9322
1	M/S 0806	C.D. Brown, 8322
1	M/S 0806	L. Stans, 9336
1	M/S 0806	J.P. Brenkosh, 9336
1	M/S 0806	J.M. Eldridge, 9336
1	M/S 0806	A. Ganti, 9336
1	M/S 0806	S.A. Gossage, 9336
1	M/S 0806	T.C. Hu, 9336
1	M/S 0806	B.R. Kellogg, 9336
1	M/S 0806	L.G. Martinez, 9336
1	M/S 0806	M.M. Miller, 9336
1	M/S 0806	J.H. Naegle, 9336
1	M/S 0806	R.R. Olsberg, 9336
1	M/S 0806	L.G. Pierson, 9336
1	M/S 0806	T.J. Pratt, 9336
1	M/S 0806	J.A. Schutt, 9336
1	M/S 0806	J.D. Tang, 9336
10	M/S 0806	T.D. Tarman, 9336
1	M/S 0806	L.F. Tolendino, 9336
1	M/S 0806	J.S. Wertz, 9336
1	M/S 0806	D.J. Wiener, 9336
1	M/S 0806	E.L. Witzke, 9336
1	M/S 0813	R.M. Cahoon, 9327
1	M/S 0813	R.A. Suppona, 9327
1	M/S 1206	J.V. Vonderheide, 5933
1	M/S 0323	LDRD Office (D. L. Chavez), 1011
1	M/S 9018	Central Technical Files, 8945-1
2	M/S 0899	Technical Library, 9616
1	M/S 0161	Patent and Licensing Office, 11500