Distributed Architecture for Resource Reservation Protocol Traffic Engineering (RSVP-TE)

Saloni Neri

A Thesis

in

The Department

of

Electrical & Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science (Electrical & Computer Engineering) at

Concordia University

Montreal, Quebec, Canada

September 2007

Your file Votre référence
ISBN: 978-0-494-52315-5
Our file Notre référence
ISBN: 978-0-494-52315-5

Abstract

Distributed Architecture for Resource Reservation Protocol Traffic Engineering (RSVP-TE)

The explosive growth of the Internet saw a corresponding expansion of information exchange dwarfing the very modest designs of early networking elements. With increasing dependence on this exciting medium the number of users has increased manifold. The exponential growth in the services that needed to be supported has spurred traffic volumes, resulting in the need for huge bandwidth and faster communication. Network infrastructures had to keep pace with these rapid developments but were unfortunately saddled with severe bottlenecks that the only solution was costly upgrades.

Routers, the mainstay of the Internet had to cope with a wide range of protocols. They have undergone several design modifications over the decades. A router essentially must perform two fundamental tasks-compute best routes and forward data packets. The evolution of routers is often described in terms of three generations of architectures. From the first generation with single CPUs and multiple interface cards with a shared bus, the routers evolved to the third generation with a switch fabric.

Reservation signaling protocols have become an indispensable part of the Internet service. Resource reservation protocols were originally designed to signal end hosts and network routers to provide quality of service (QoS) to individual real-time flows. Recently, Internet Service Providers (ISPs) have been using the same signaling

mechanisms to set up provider-level Virtual Private Networks (VPNs) in the form of MPLS Label Switched Path (LSP). Traditional IP router architectures cannot scale to meet these demands, forcing architects to explore alternative designs. However, due to rapid growth of the Internet, the architecture of the third generation routers are not able to meet the expected amount of traffic (i.e., multiple terabits or petabits per second). This development results in emergence of next generation routers with large switching capacity and high speed interfaces.

The traditional centralized software model, where the control card is solely responsible for all routing and management operations is not able to efficiently utilize the new hardware platform architecture of line cards. Distributed architecture is one of the promising trends allowing routers to improve their robustness, scalability and resiliency. In this thesis path calculation and memory resources are proposed to be available on both control and line card in order to perform routing and forwarding tasks.

We investigate the ability to reallocate components of the current MPLS/RSVP-TE architecture on to the line cards in order to share the load between the control card and line card. This allows significant improvement in scalability, resiliency and robustness of the system. New mechanisms for message exchange, synchronization, table and session management, and storage are developed. Performance evaluations in terms of CPU consumption, memory requirements, load balancing and bandwidth utilization for both centralized and the proposed distributed software architectures indicate that the processing time and memory utilization are significantly reduced.

Acknowledgement

I would like to take this opportunity to thank my mentors, advisors and supervisors, Dr. Anjali Agarwal and Dr. Brigitte Jaumard, to whom I am truly indebted for their continuous guidance and financial support in carrying out this research. Their assistance has always been generous and timely. They not only enlightened me on researching this exciting topic but also gave me the strength to be persistent in accomplishing my goals. But above all is what I have academically gained from my association with them.

I gratefully acknowledge the support of KimKhoa Nguyen for his advice, and guidance throughout this research project.

I would further like to express my appreciation to the examiners for their comments for improving my thesis. I would also like to thank Faculty of Electrical and Computer Engineering for their outstanding teaching and guidance.

I would also like to thank former Hyperchip, Inc., for providing me with financial support.

Thanks to my fellow students at University of Concordia, Akanksh Vashisth, Pavel Sinha and Alya Zaidi for making the long work hours fun.

A special whole hearted thanks to my friend Nikhil Mehta for supporting my dream and for believing in me all the way through.

Finally, I dedicate this dissertation to my parents Joe and Ruma Neri for their unconditional love and support. They are and will always be my inspiration and source of pride and strength.

I hope I have made them proud!

Saloni neri

Table of Contents

List of Figures x

List of Tables xii

List of Abbreviations xiii

CHAPTER 1 1

1 Introduction 1

1.1 Key Functions of a Router 4

1.2 Evolution of Router Architectures 6

1.2.1 First Generation Routers: Single Central Processor and a Shared Bus 6

1.2.2 Second Generation Routers: Route Caching 7

1.2.3 Third Generation Routers: Switch-based 8

1.2.4 Next Generation Routers 9

1.2.5 Forwarding and Routing Mechanisms of Next Generation Routers 11

1.3 Motivation for a Distributed MPLS/RSVP-TE Architecture 13

1.5 Thesis Organization 15

CHAPTER 2 17

2 Background Information 17

2.1 Overview of MPLS 17

2.2 Overview of RSVP-TE Protocol 20

2.3 Overview of RTM 25

CHAPTER 3 28

3 Literature Review 28

3.1 Distributed Software Architecture 29

3.2 Industrial Next Generation Routers and Prototype 31

3.2.1 Juniper Next Generation Router 31

3.2.2 Nexabit NX64000 Router 34

3.2.3 Cisco Next Generation Router 34

3.2.4 Hyperchip Prototype 35

3.2.5 Tiny Tera Prototype 37

CHAPTER 4 38

4 Centralized and Distributed Architectures for RTM with RSVP-TE 38

4.1 Centralized RTM Architecture 38

4.2 Partially Distributed RSVP-TE Architecture with Centralized RTM 41

4.3 Fully Distributed RSVP-TE Architecture with Distributed RTM 44

4.4 Summary 46

CHAPTER 5 47

5 Elements for a Distributed and a Scalable Architecture for MPLS/RSVP-TE 47

5.1 A General Framework for Distributed Software Architecture 47

5.2 Table Management 50

5.3 Message Processing 53

5.4 RSVP-TE Data Base Distribution 55

5.5 Resource Reservation with QoS Module 56

5.6 Summary 58

CHAPTER 6 59

6 A New Distributed & Scalable MPLS/RSVP-TE Architecture 59

6.1 Definition of Ingress and Egress Line Card 59

6.2 Detailed description of a RSVP-TE Protocol Message Processing 60

6.2.1 Downstream Processing of Path Message Requests 61

6.2.1.1 Processing a Path Message at LER 62

6.2.1.2 Processing a Path Message at LSR (Transit Router) 65

6.2.2 Upstream Processing of Resv Message Requests 68

6.2.2.1 Processing a Resv Message at LER 69

6.2.2.2 Processing a Resv Message at LSR (Transit Router) 69

6.3 Summary 72

CHAPTER 7 73

7 Performance Analysis of MPLS/RSVP-TE Messages (Distributed vs. Centralized Architectures) 73

7.1 CPU Cycles and Memory Consumption 73

7.2 Load Balancing with Multi-Plane Switch Fabric 86

7.3 Bandwidth Utilization 95

7.4 Advantages of a Distributed/Scalable Architecture 97

CHAPTER 8 100

8 Conclusion and Recommendations 100

8.1 Conclusion 100

8.2 Future Work 101

References 103

List of Figures

Figure 1.1 First Generation Router with Single Central Processor and a Shared Bus..... 7

Figure 1.2 Second Generation Router with Route Cache Architecture 8

Figure 1.3 Third Generation Router with Switch Based Architecture..... 9

Figure 1.4 Components of a Typical LC and CC 10

Figure 1.5 Architecture of Next Generation Router 12

Figure 2.1 MPLS Architecture..... 19

Figure 2.2 Path and Resv Messages..... 22

Figure 2.3 MPLS/RSVP-TE for Next Generation Routers [30] 24

Figure 2.4 Inter-communication of RTM with Routing Protocols 27

Figure 3.1 M-Series Router Architecture..... 32

Figure 3.2 T-Series Router Entities 33

Figure 3.3 Hyperchip Architecture 36

Figure 3.4 Architecture of Tiny Tera..... 37

Figure 4.1 Current RTM Architecture: Distributed on Protocol Basis..... 40

Figure 4.2 Distributed MPLS Architecture with a Centralized RTM..... 43

Figure 4.3 Distribution of LC-RTMs..... 45

Figure 5.1 RSVP-TE Components on the LC..... 50

Figure 5.2 FEC-TO-NHLFE (FTN) Table Structure..... 52

Figure 5.3 ILM Table Structure..... 52

Figure 5.4 Distributing RSVP-TE Data Base in LCs 56

Figure 5.5 Resource Reservation for a Distributed RSVP-TE 57

Figure 6.1 Ingress LC and Egress LC 60

Figure 6.2 Downstream Processing of Path Message 62

Figure 6.3 Processing RSVP-TE Path Message on LER 63

Figure 6.4 Path Message Processing on LSR 68

Figure 6.5 Processing Resv Message (Upstream On demand Mode) 69

Figure 6.6 Processing RSVP-TE Resv Message on LSR 70

Figure 7.1 Centralized Architecture with Multiple Planes in a Switch Fabric 75

Figure 7.2 Distributed Architecture with Multiple Planes in a Switch Fabric 80

Figure 7.3 CPU Resource Consumption 85

Figure 7.4 Memory Consumption 86

Figure 7.5 Load Distribution in Distributed Architecture 94

Figure 7.6 Bandwidth Utilization in Centralized and Distributed Architecture 97

List of Tables

Table 7-1 CPU Resource Consumption 83

Table 7-2 Memory Consumption for Centralized and Distributed Architecture 84

Table 7-3 Wait Time Calculations in Centralized Architecture 91

Table 7-4 Wait Time Calculations in Distributed Architecture 93

Table 7-5: Qualitative Comparison between Centralized and Distributed RSVP-TE .....

Architectures 99

List of Abbreviations

ATM	Asynchronous Transfer Mode
BGP	Border Gateway Protocol
CC	Control Card
eNP	Egress Network Processor
eTM	Egress Traffic Manager
CLI	Command-Line Interface
FEC	Forwarding Equivalence Class
FIT	Forwarding Information Table
FTN	FEC-To-NHLFE
ILM	Incoming Label Mapping
iNP	Ingress Network Processor
iTM	Ingress Traffic Manager
IP	Internet Protocol
LAT	Label Allocation Table
LDP	Label Distribution Protocol
LC	Line Card
LER	Label Edge Router
LIB	Label Information Base
L-ROUTE	Local Route Base
LSP	Label Switch Path
LSR	Label Switch Router
MPLS	Multi-Protocol Label Switching
NHLFE	Next Hop Label Forwarding Entry
OSPF	Open Shortest Path First
QoS	Quality of Service
RSVP	Resource ReSerVation Protocol
RSVP-TE	RSVP Traffic Engineering
RTM	Routing Table Manager
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

Chapter 1

1 Introduction

The explosive growth of the Internet has led to an increase in the number of users who have diverse demands for more reliable and differentiated services. Both large and small Internet Service Providers (ISPs) constantly face the challenges of adapting their networks to accommodate new services and meeting more diverse customer requirements. New services appear, which may require modifications of existing protocols, entirely new protocols, or modifications of the ways in which packets are processed in routers. Examples of such new services are video conference, IPTV and the transition to Internet Protocol version 6 (IPv6). In many situations, software updates are not enough to achieve this goal. Meanwhile, due to extremely high costs, physically replacing or upgrading network infrastructure constantly is not feasible, either.

The traditional way to build routers has been to use a monolithic architecture [1] [2] , where line cards (LCs) are connected to a backplane, and where a CPU-based controller is used to run routing software, to handle network management software and other types of control software.

The LCs and the forwarding functionality are normally referred to as the forwarding plane, while the controller and its functionality are referred to as the control plane.

The continuously growing need for new services and protocols also affects the control plane of the router [3]. The control plane is traditionally a design where a variety of large software functions are integrated into an operating system in a monolithic fashion. This means that adding new services to a router is often a complex task from the control plane point of view. Protocols such as BGP [5], OSPF [6], ISIS [7], MPLS [8], RSVP-TE [10], together with other functions needed in a router, constitute large amounts of complex software. This may consequently result in high demands on control processing. Thus, it is hard to add new functionality in the control plane and at the same time maintain a high degree of reliability and efficient execution of the software.

The monolithic structure of these traditional routers is an architectural limitation when it comes to meeting future requirements. In order to satisfy these requirements, a modular design is required. With modular design, routing software components can run independently on the same or separate CPUs and interact with each other regardless of their respective physical location. This approach produces a robust network that is not vendor specific and that can use modules developed by different manufacturers.

MPLS is the protocol framework on which the attention of a network service provider is focused as it provides a solution to scalability and enables significant flexibility in routing. MPLS is capable of providing controllable quality of service (QoS) features. It is meant to primarily

prioritize Internet traffic and improve bandwidth utilization. When packets enter a MPLS-based network, Label Edge Routers (LERs) give them a label identifier based on its Forward Equivalence Class (FEC). Once this classification is complete and mapped, different packets are assigned to the corresponding Labeled Switch Paths (LSPs), where Label Switch Routers (LSRs) place outgoing labels on the packets. MPLS employs RSVP-TE, an extension of the Resource Reservation Protocol, for establishing LSPs [10] .

RSVP-TE is a receiver-oriented protocol, meaning that label allocation and bandwidth reservation are driven by the receiver node. RSVP-TE allows the establishment of LSPs, taking into consideration network constraint parameters such as available bandwidth and explicit hops. The two main message types used are a Path Message (used to establish a path from the source to the destination) and Resv Message (used to reserve the resources that will be used in an LSP). Path Message is first sent downstream to the destination. Once Path Message is successfully sent and received, Resv Message is sent upstream to the source and an LSP is established to send data to the destination [8][10].

In the current generation of routers, the RSVP-TE module is neither distributed nor scalable. There are several limitations of the centralized architecture on the ever-increasing number of interconnections between routers. It requires routers to have more CPU cycles, more powerful accompanying hardware resources, and an increased memory size to contain all available routing information. Until recently, the only valid solution to support the increasing Internet traffic was to periodically upgrade the router control card (CC), on which the RSVP-TE module is running, or to replace the whole router with a new one having more powerful hardware resources (e.g.,

CPUs and increased memory size), demanding some service interruptions. An alternate solution is to implement distributed and scalable routers. Taking advantages of the hardware revolution, a more cost-effective approach to deal with the ever-increasing traffic in the networks has been proposed, where a next generation router with large-capacity switching of multiple terabit or even petabit is deployed in the core network [3][18].

In this chapter we intend to review the evolution of routers and give a brief overview of key functions of a router. In the following chapters we focus on a distributed and scalable architecture for MPLS/RSVP-TE routing protocol. We investigate the ability to reallocate components of MPLS/RSVP-TE on to the LCs in order to share the load between the CC and the LC.

1.1 Key Functions of a Router

Broadly speaking, a router must perform two fundamental tasks which are described below.

- **Compute best routes:** The first function is to compute best routes that the data packets would use to traverse through the network to their destinations. The route computation has to take into account various policies and network constraints [13]. For example, the best route can be required to maximize network efficiencies, to minimize bandwidth usage costs and deliver the fastest possible response times to users. In the current generation routers, the route computation is accomplished by a route processor or universally known as a routing engine. Based on the information exchanged between neighbouring routers using routing protocols, the routing process constructs a view of the network topology and computes the

best paths. A routing engine is essentially the "brain" of the router and is responsible for communication with neighbours. This communication enables the route processor to build a route database, or routing table, that allows the forwarding engine to send packets across optimal paths through the network are located. Such communication is achieved by the routing protocols [17]. The routing protocols communicate information about which networks each router can reach and how far away those networks are. These messages are called routing updates. All routing protocols can be classified as Interior Gateway Protocols (IGPs) or External Gateway Protocols (EGPs) [13].

- IGPs run inside an autonomous system (AS) and perform so called intra-domain routing functions. Widely used IGP protocols include Routing Information Protocol (RIP) [14], Open Shortest Path First (OSPF) [6] and Intermediate System-to-Intermediate System (IS-IS) [7].
- EGPs run between ASs. The currently most used EGP is Border Gateway Protocol (BGP) In a core router, the BGP module has to handle a very large number of routes (e.g., some hundreds of thousands of routes) and ASs (e.g., tens of thousands of ASs) [15].

- **Forward data packets:** The second function of the router is to forward data packets received across the network. Forwarding relies on the best route information computed by the route processor. The forwarding function is achieved by forwarding engines. The forwarding engine consults a Forwarding Information Table (FIT) which contains a complete set of forwarding information for all destinations learned by the routing protocols or by the list of static routes. Based on the destination address field of the IP packet header,

the forwarding engine looks up the FIT to find the next hop to forward the packet. As traffic loads grow, the processing resources required for FIT lookups increase.

1.2 Evolution of Router Architectures

The Internet has been in function since the 1970s, and IP routers have gone through several design modifications over the decades. The evolution of routers is often described in terms of three generations of architectures. The different generation of routers [16] [21] [35] [44] will be described briefly below. The latest generation of routers, could be thought of as the next generation routers or distributed routers, which will be described in detail in the next chapters.

1.2.1 First Generation Routers: Single Central Processor and a Shared Bus

The first generation of IP routers was basically made of a single central processor and multiple interface cards interconnected through a shared bus. The CPU runs a commodity real-time operating system and implements the functional modules, including the forwarding engine, the queue manager, the traffic manager, and some parts of the network interface, especially Layer 2/Layer 3 processing logic in software. Figure 1.1 shows the architecture of the first generation routers. An incoming packet at a LC (called ingress line card) [16] is forwarded to the buffer memory through the shared bus. The central CPU extracts the headers of the packet and uses the forwarding table to determine the outgoing LC (called egress interface) and port. The packet is subsequently prioritized by the queue manager and shaped by the traffic manager. Finally, the packet is transferred from memory to the appropriate output port in the egress LC [17].

The performance of these routers depends on the throughput of the shared bus and on the speed of the central processor; therefore they are not able to scale to meet today's bandwidth requirements [17].

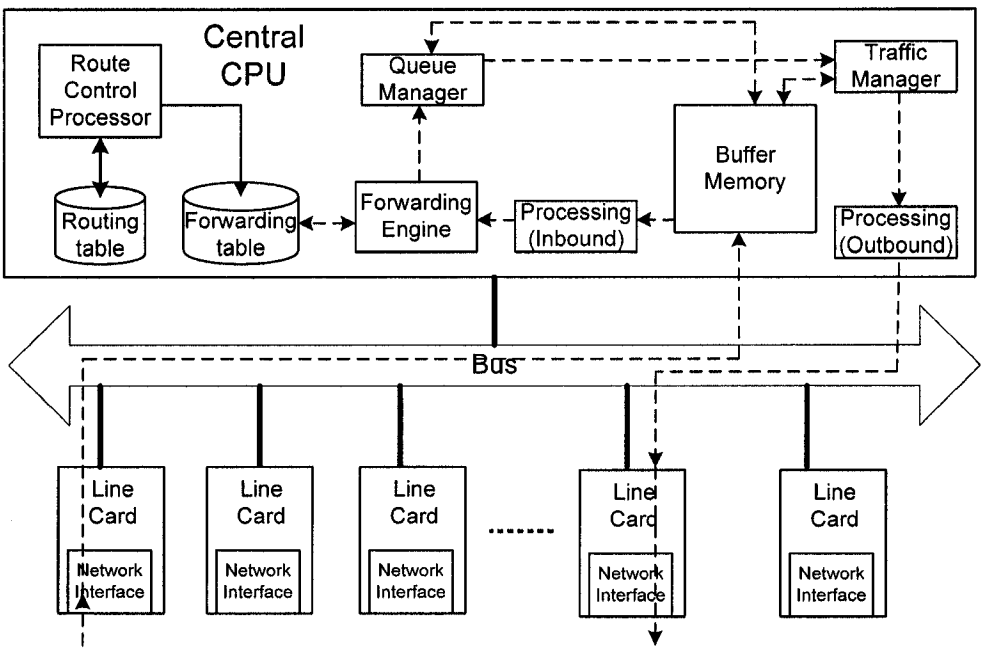

Figure 1.1 First Generation Router with Single Central Processor and a Shared Bus

1.2.2 Second Generation Routers: Route Caching

The second generation router architecture was designed with more intelligence. The router has a central CPU maintaining a central forwarding table and the LCs contain a local cache which is a subset of the master forwarding table based on the routes that were recently used as shown in Figure 1.2. When a LC receives a data packet, it first looks up the local cache for the next hop to forward the packet. If no entry is available in the cache, the LC sends a request to the central CPU [16].

The advantage of this architecture is the increased throughput because of the forwarding cache. However, the shared bus is still a potential bottleneck because it does not allow more than one data packet to go across at the same time. Due to these drawbacks, this architecture can neither scale to high capacity links nor provide complex traffic pattern-independent throughput [17].

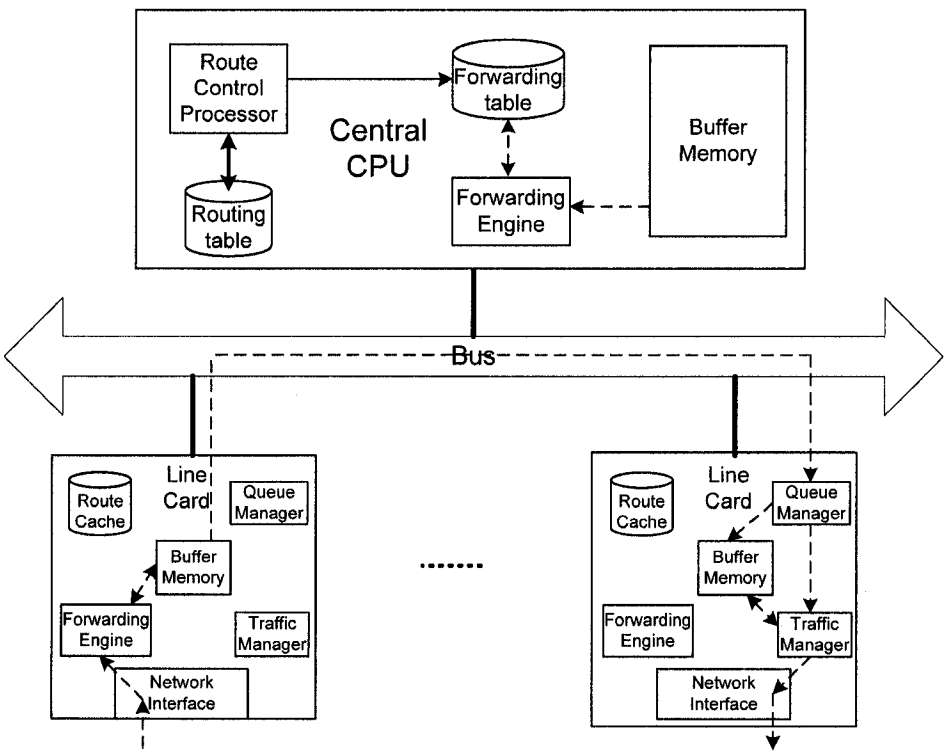

Figure 1.2 Second Generation Router with Route Cache Architecture

1.2.3 Third Generation Routers: Switch-based

The third generation or the current generation router was introduced to solve bottlenecks of the second generation. The shared bus was replaced by a switch fabric which allows multiple packets to be simultaneously transferred across, hence increasing the performance as shown in Figure 1.3 [16]. The switch fabric is basically a crossbar connecting multiple cards together thus providing large bandwidth for transmitting packets among LCs [3][21].

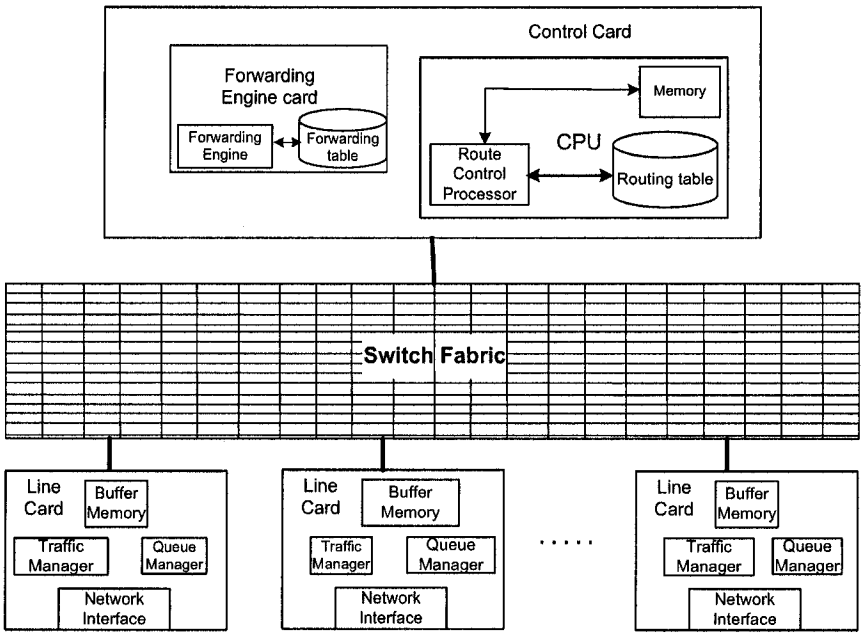

Figure 1.3 Third Generation Router with Switch Based Architecture

There are three main bottlenecks which can potentially be experienced in a first and second generation router: processing capacity, memory bandwidth, and internal bus bandwidth. Hence the switching architecture has been deployed in the third generation routers in order to replace the internal bus. However, due to rapid growth of the Internet, the architecture is not able to meet the expected amount of traffic (i.e., multiple terabits or petabits per second). The third generation routers are not scalable to support a large number of LCs. These issues led to the next generation routers, which we will be describe in the next section.

1.2.4 Next Generation Routers

The architecture of the next generation routers is essentially switch-based, with a switching capacity of petabits per second, satisfying different QoS requirements [4]. First commercial

terabit routers that appeared in the market were Avici's TSR [18][20], Juniper T1600 [12] and Nexabit [42]. Some prototypes have also been introduced like HyperChip's PBR1280 [19] and Tiny Tera [43]. Next generation routers have the following advanced features:

Line Cards

The Line Card (LC) provides one or more interfaces to external devices (such as other routers) and connects these interfaces to the switch fabric, as shown in Figure 1.4. The ingress Traffic Manager (iTM) receives frames from the ingress Network Processor (iNP) with the respective destination addresses, and queue numbers. The iTM's main role is to forward these frames to the switch fabric as well as to provide balanced traffic distribution among the switch fabric planes. The egress Traffic Manager's (eTM) role in the LC is to receive packets from the switch fabric planes directly connected to its LC, to perform frame re-assembly and to perform frame sequencing of re-assembled frames. Once these operations are completed, frames are either sent to the local CPU or forwarded to the egress Network Processor (eNP) for additional forwarding treatment and per-egress-port output queuing depending on the destination of the internal frame being re-assembled. Nowadays, a network processor can handle flows at OC-48 or OC-192 line rate or even faster [3].

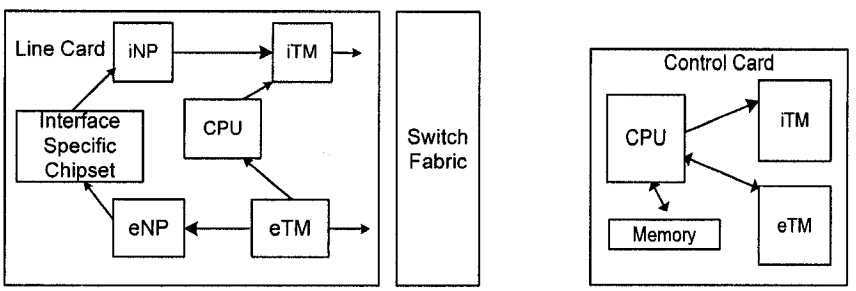

Figure 1.4 Components of a Typical LC and CC

Control Card

The control card (CC) architecture is very similar to the one of a LC. The basic difference between both cards lies in the processing power and storage capabilities of CC being far superior, and there are no line interfaces. The CC has one ingress Traffic Manager (iTM) chip and one egress Traffic Manager (eTM) chip as shown in Figure 1.4. These chips provide an interface between the local processor and the switch fabric planes. The iTM and eTM chips are exactly the same as the ones used in the LCs.

Switch Fabric

The router's cards are interconnected by a scalable switch fabric. The switch fabric is distributed into identical and independent switching planes. In our research, we assume using a switch fabric model provided by Hyperchip [19] [22] which consists of multiple planes. In such a model, each switching plane has bandwidth to handle a full OC-48 port or equivalent. A switch fabric port, which is equivalent to one frontal interface chassis slot, connects a LC or a CC to one of the four planes. Each switching plane is made of identical switching cards, so called matrix cards.

1.2.5 Forwarding and Routing Mechanisms of Next Generation Routers

Forwarding and routing mechanisms in a next generation router are illustrated in Figure 1.5. Data packets come in by the iNP of the ingress LC which contains a FIT table that is used to determine the path to the destination. Packet classification is also done by the iNP. Packets are then forwarded through the iTM where traffic engineering policies are applied. They then travel through the switch fabric to the egress LC. The eTM and eNP of the egress LC forward the

packet to the next hop in towards the destination. Control packets, on the other hand, are filtered by the iNP of the ingress LC and forwarded directly to the CC or the CPU of the LC where they will be processed by the routing protocol modules. The iTM and eTM chipsets located on the CC are responsible for managing flows of control packets. Control packets may also be sent out to external routers in the network through appropriate LCs.

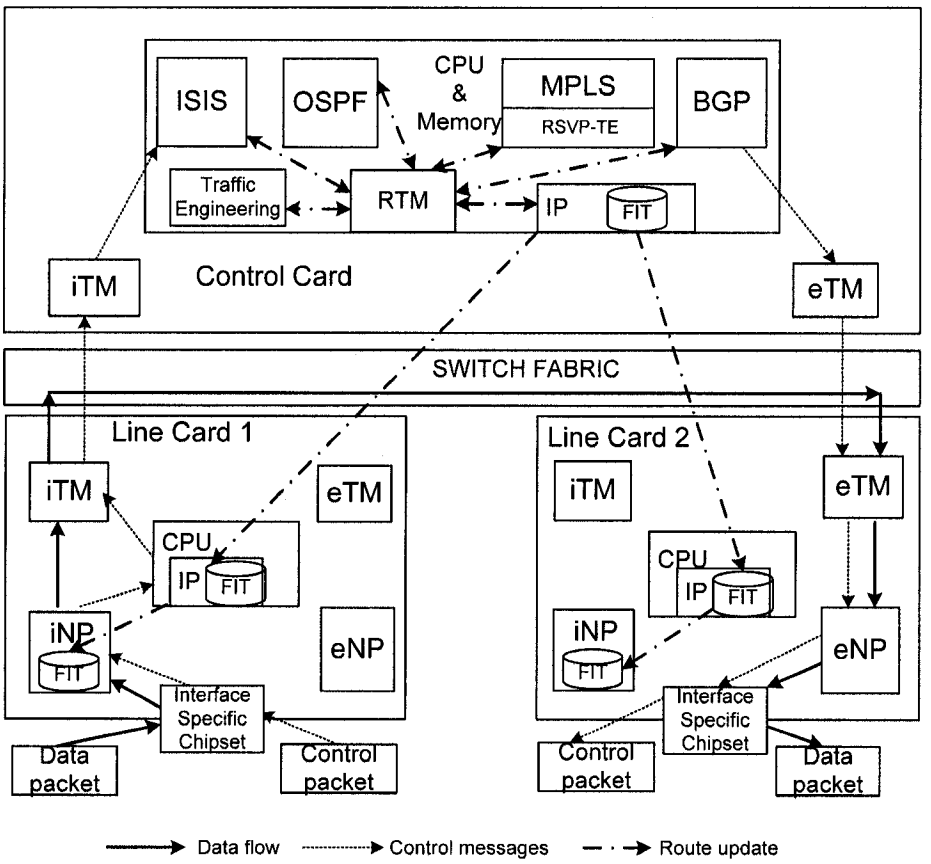

Figure 1.5 Architecture of Next Generation Router

As we can see from Figure 1.5, routing engine handles a set of routing protocols like IS-IS, OSPF, BGP and MPLS that run together and interchange information such as routes or labels. The advantage of such an architecture is the ease of management since all the routing protocols run together on the same CC. The synchronization and message exchange mechanisms are also

quite simple to implement. However, the main issue of such legacy systems is their monolithic code base with all forwarding and routing processes competing for the same CPU and memory resources. Consequently, as the demanding packet forwarding process consumes almost all the CPU capacity, the other functions are left starving for CPU cycles. Clearly this type of routers can only be used for small and medium size networks.

Thus we investigate the ability to relocate RSVP-TE signaling protocol [10] on to the LC. This approach will let routers to have more CPU cycles, more powerful accompanying hardware resources, and an increased memory size to contain all available routing information.

1.3 Motivation for a Distributed MPLS/RSVP-TE Architecture

One of the primary requirements for next generation routers is to obtain the advantages of the additional resource available on LCs. In general, next generation routers have to exchange control messages with hundreds of peers. Keeping pace with the growing requirement of bandwidth, a large number of LCs needs to be added to the router platform. This imposes several challenges to the operation of routing protocols. Current generation routers provide terabit throughput, while next generation routers, assuming a distributed architecture, will reach petabit throughput per set of thousand LCs.

Routing protocols like BGP [5], OSPF [6] and ISIS [7] should be distributed on the LCs in order to take advantages of the next generation distributed router platform to provide additional processing and memory resources on LCs. In the similar manner signaling protocols such as LDP [26] and RSVP-TE should also be distributed on the line cards. No research has been done on routing protocols. However work is in progress in [45] to distribute the LDP protocol. In this thesis we investigate the ability of distributing MPLS/RSVP-TE module on the LCs.

In addition, the model proposed in this thesis has the following advantages:

- **Performance:** parallel processing is enabled in the proposed architecture and waiting queue can therefore be avoided. LCs are able to process the routes separately without having to wait for reply from the CC.
- **Scalability:** Scalability is improved since some of the control tasks, particularly the signaling, are processed by LCs. The CC processes only the most complicated tasks, the tasks that need human interactions or the tasks used to inter-operate different LCs [23] .
- **Resiliency:** If the CC is responsible for all control tasks, the system will completely shutdown if the CC fails. The distributed architecture proposed allows the LCs to maintain connections with the peer routers [25]. The CC restarts only when it fails. The fully distributed architecture we propose for signaling protocols such as RSVP-TE is able to self restore without the help of the CC.
- **Availability:** The time to recover from failures is reduced. Since complete restart is not required, problems pertaining to the CC will not slow down process on LCs.

1.4 Contributions of this Thesis

We consider the next-generation router architecture as a starting point for a distributed framework. In this thesis we consider that the RTM is distributed on the LCs to serve RSVP-TE signaling protocol to increase the scalability and to reduce the load of the CC.

The major contributions of this thesis are as follows:

- **A Novel Scalable and Distributed Architecture for MPLS/RSVP-TE:** The system design of a distributed architecture for MPLS/RSVP-TE is proposed with respect to the RFC's

specifications. Different distributed solutions are explored in order to eliminate centralized scheduling and bottlenecks on the CC.

- **Synchronization Mechanisms:** Various synchronization mechanisms are used to process RSVP-TE downstream and upstream signaling messages. Synchronization between LCs, synchronization between routing tables and MPLS tables, distributed label provision, distributed table access and update have also been dicussed.
- **Performance Evaluation:** Mathematical models have been designed for the proposed MPLS/RSVP-TE with respect to the number of CPU cycles, memory requirements, load balancing and bandwidth utilization. These models imply that the processing time can be significantly reduced via the distribution of workload and reduce potential bottlenecks experienced on the CC when the number of requests increase.

While we kept the resiliency issues in mind, we did not investigate it further in this thesis and left it for future research.

1.5 Thesis Organization

This thesis consists of eight chapters including necessary background, motivation and the objectives of the research effort in Chapter 1. Chapter 2 provides an overview of MPLS and RSVP-TE protocols. In addition it provides an overview of the Routing Table Manager, which is the most important module of a router. Chapter 3 provides related information and previous research done by other researchers in the same area. It includes a review of recent literature on software architecture for distributed routers. It also includes the new generation routers in the industry. Chapter 4 discusses the benefits of distributed Routing Table Manager on the LCs and

motivations of using this approach in the research proposal. Chapter 5 covers the efficient elements needed for a distributed and scalable RSVP-TE architecture. Chapter 6 describes each element in detail for a distributed and scalable architecture. Chapter 7 gives the performance evaluation in terms of CPU consumption, memory usage and bandwidth utilization in centralized and distributed architecture. Finally, Chapter 8 provides conclusions and summary of the research effort as well as the recommendations for future research.

Chapter 2

2 Background Information

Resource reservation protocols were originally designed to signal end hosts and network routers to provide quality of service to individual real-time flows. More recently, Internet Service Providers (ISPs) have been using the same signaling mechanisms to set up provider-level Virtual Private Networks (VPNs) in the form of MPLS Label Switched Path (LSP).

In core routers, the Routing Table Manager (RTM) [13] module plays an important role by managing all best routes coming from various sources. Possible sources are the different routing protocols, such as Open Shortest Path First (OSPF) [6], Intermediate System to Intermediate System (IS-IS) [7], Border Gateway Protocol (BGP) [5], and RSVP-TE [10]. This chapter provides an overview of MPLS/RSVP-TE and the RTM.

2.1 Overview of MPLS

In a MPLS [8] network, packets are forwarded hop-by-hop based on a fixed-length label. This label, called Label Switched Path (LSP), determines the route that the packet will take to the destination. Thus, the routing process is done only in edge routers, so called Label Edge Router

(LER), then the packet is simply switched over transit routers, so called Label Switch Router (LSR); in consequence, the forwarding speed is improved. In traditional IP based networks, all packets from a given source to a given destination travel on a best route determined by a routing protocol. Hence the additional services such as VPN are not enabled. In addition, nodes on the best route can become critical points due to the overload, while other nodes in the network can be inefficiently utilized. Based on labels, MPLS provides flow management, traffic engineering, quality of service (QoS), VPNs (Virtual Private Networks) and Any Transport over MPLS (AToM).

A MPLS label is a 20-bit identifier, added to the MPLS data packets to forward them over a network. Packets share the same forwarding criteria, so called forwarding equivalence class (FEC), e.g., that experience the same delay, and carry the same label. Therefore, a LSP is a combination of a FEC and a label. In order to define the LSPs among routers in a network, signaling protocols like LDP [26] [25] and RSVP-TE [10] are deployed.

In a next generation router, MPLS architecture is divided into two main architectural blocks as shown in Figure 2.1 [30].

- **Control plane:** Performs functions of identify reachability to destination prefixes. Therefore the control plane contains all the Layer 3 routing information. In addition, all protocol functions that are responsible for the exchange of labels between neighboring routers and RSVP-TE module.
- **Data plane:** Performs the functions relating to forwarding data packets. These packets can be either Layer 3 IP packets or labeled IP packets. The information in the data plane, such as

label values are derived from the control plane. Information exchange between neighboring routers creates mappings of IP destination prefixes to labels in the control plane, which is used to forward data plane labeled packets.

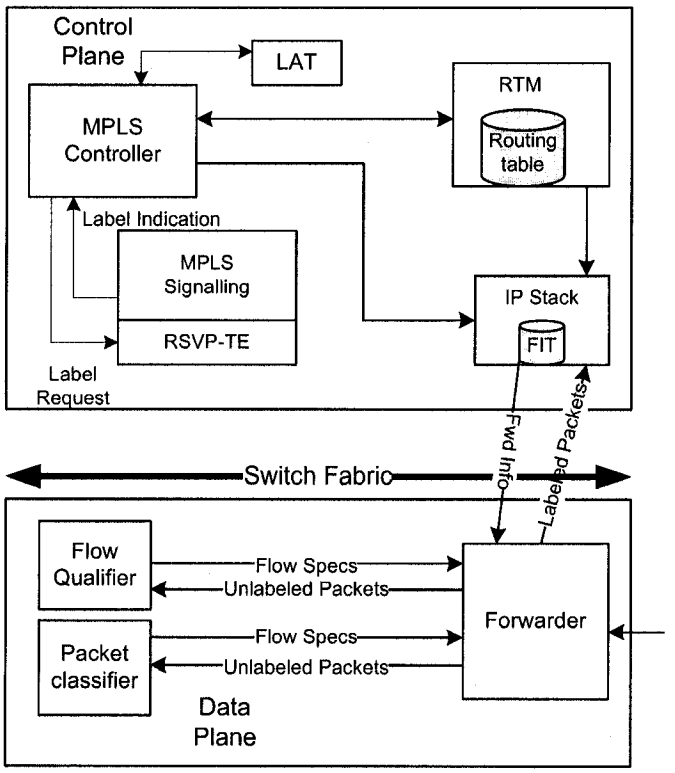

Figure 2.1 MPLS Architecture

In recent router products including next generation routers, the MPLS and the RSVP-TE module are neither distributed nor scalable. These routers handle all RSVP-TE processes on the CC. Indeed, there are no MPLS and RSVP-TE module running on any LC. The centralized architecture of a MPLS module consists of the following modules as shown in Figure 2.3 [30].

- **eTM, eNP (Forwarder):** Is responsible for classifying and processing unlabeled packets, making MPLS forwarding decision for labeled packets and forwarding outgoing packet.

- **iTM, iNP (Flow Qualifier):** Is in charge of qualifying unlabeled packets, associating the unlabeled packets with a FEC.
- **iTM, iNP (Packet Classifier):** Is similar to Flow Qualifier and can be used alternatively. The difference is that Packet Classifier, filters packets according to the Traffic Engineering which can be specified by user, while Flow Qualifier, gets filter information from the FIT.
- **FIT (Forwarding Information Table):** Used for forwarding MPLS data packets. FIT is controlled and updated by MPLS Controller.
- **Label Allocation Table (LAT):** Used by MPLS Controller to manage label spaces and to keep track of all allocated labels.
- **Label Information Base (LIB):** Used to forward data packets. The LIB contains information for labeling a data packet, changing the current label, or removing label when the packet reaches the destination.
- **MPLS controller:** Is the heart of the MPLS system. It is responsible for managing FIT, using LAT to manage label space and to track all allocated and ready-to-use labels, and interacting with MPLS signaling module to establish LSPs.

2.2 Overview of RSVP-TE Protocol

RSVP-TE extends RSVP by supporting additional objects allowing the establishment of explicitly routed label switched paths using RSVP as a signaling protocol. The result is the instantiation of label-switched tunnels that can be automatically routed away from network failures, congestion, and bottlenecks.

Description of RSVP-TE Messages

RSVP-TE defines seven signaling messages. This section gives a brief overview of the seven signaling messages: Path, Resv, PathErr, ResvErr, PathTear, ResvTear, and ResvConf used by the RSVP-TE protocol [9] [10] [11].

- **Path Message and Resv Message**
  - **Path Message** is sent by each sender along the unicast or multicast routes provided by the routing protocol(s). A Path Message is used to store the path state in each node. The path state is used to route reservation-request messages in the reverse direction.
  - **Resv Message** is sent by each receiver host toward the senders. This message follows in the reverse directions the routes that the data packets use, all the way to the sender hosts. A reservation-request message must be delivered to the sender hosts so that the hosts can set up appropriate traffic-control parameters for the first hop.

RSVP-TE takes a "soft state" approach to manage the reservation state in routers and hosts. RSVP soft state is created and periodically refreshed by Path and Resv Messages. The state is deleted if no matching Refresh Message arrive before the expiration of a "cleanup timeout" interval. State may also be deleted by an explicit "teardown" message, described in the next section. At the expiration of each "refresh timeout" period and after a state change, RSVP-TE scans its state to build and forward Path and Resv Refresh Message to succeeding hops.

Figure 2.2 shows the flow of Path and Resv Message from router R1 to R5.

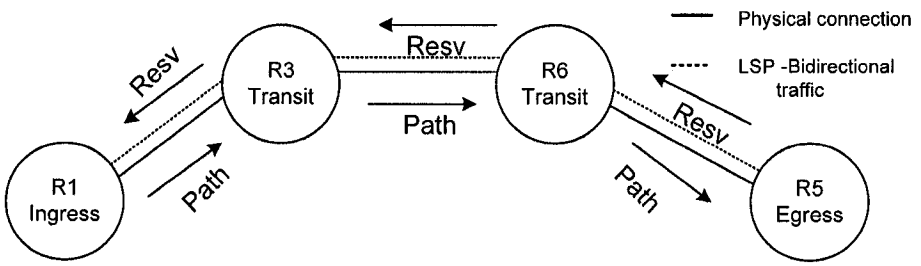

Figure 2.2 Path and Resv Messages

- **Tear messages (Path Tear and Resv Tear)**

RSVP-TE "teardown" messages remove path or reservation state immediately. There are two types of RSVP-TE teardown messages, PathTear and ResvTear. A teardown request may be initiated either by an application in an end system (sender or receiver), or by a router as the result of state timeout or service preemption. Once initiated, a teardown request must be forwarded hop-by-hop without delay. A teardown message deletes the specified state in the node where it is received.

  - A **PathTear** Message travels towards all receivers downstream from its point of initiation and deletes path state, as well as all dependent reservation state, along the way.
  - A **ResvTear** Message deletes reservation state and travels towards all senders upstream from its point of initiation. A PathTear and ResvTear Messages may be conceptualized as a reversed-sense Path Message (Resv Message, respectively).

- **Error Messages (Path Error and Resv Error)**

There are two Error Messages PathErr and ResvErr that are described below.

- **PathErr** (path error) messages report errors in processing Path Messages. They travel upstream towards senders and are routed hop-by-hop using the path state. At each hop, the IP destination address is the unicast address of a previous hop. PathErr messages do not modify the state of any node through which they pass; they are only reported to the sender application.
- **ResvErr** (reservation error) messages report errors in processing Resv message. ResvErr messages travel downstream towards the appropriate receivers, routed hop-by-hop using the reservation state. Information carried in error messages are admission failure, bandwidth unavailable, Service not supported, bad flow specification and ambiguous path.

- **Confirmation Message (ResvConf)**

Reservation-request acknowledgment messages are sent as the result of the appearance of a reservation-confirmation object in a reservation-request message. This acknowledgment message contains a copy of the reservation confirmation. A reservation-request acknowledgment message is forwarded to the receiver hop by hop (to accommodate the hop-by-hop integrity-check mechanism).

Detailed on processing of these messages is explained in Chapter 6. Essentially to establish a LSP path across an ingress and egress interface we require five basic signaling messages, Path, Resv, PathErr, ResvErr, Resv ResvConf Message. Therefore we consider only these five signaling messages in our thesis.

The modules required for processing these messages are shown in Figure 2.3 and are explained below.

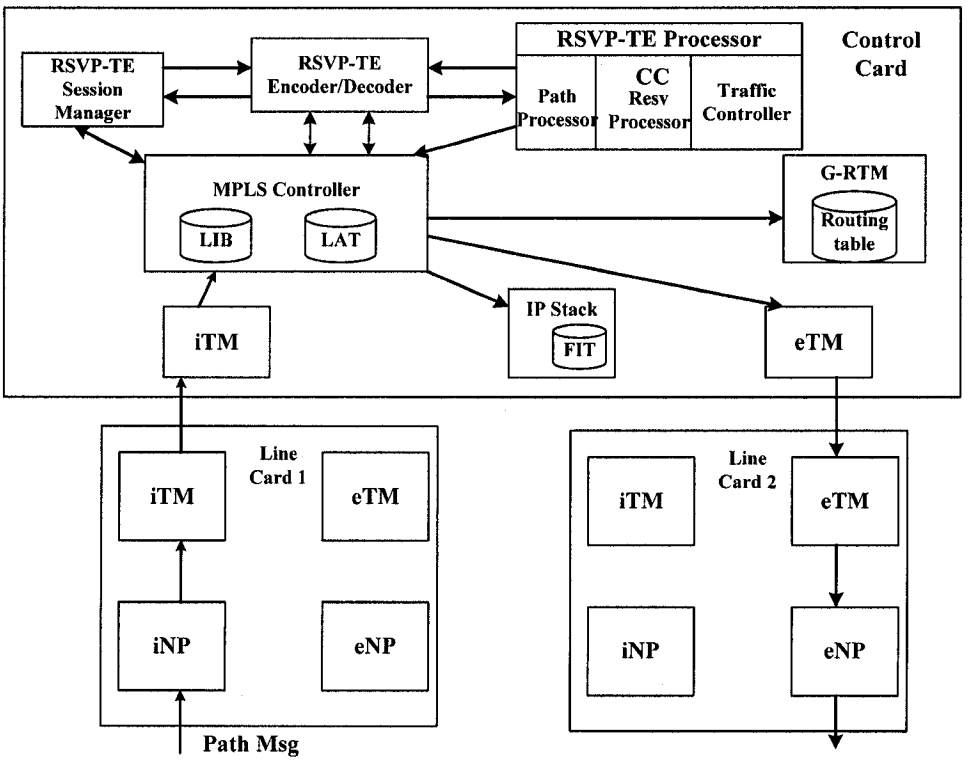

Figure 2.3 MPLS/RSVP-TE for Next Generation Routers [30]

MPLS Controller. With respect to the RSVP-TE process, the MPLS Controller is responsible for:

- Initiating, accepting, rejecting or closing an RSVP-TE session.
- Requesting label mappings and resource reservations from other LSRs by sending appropriate requests to the RSVP-TE process.
- Processing label-related and resource reservation-related notifications received from other LSRs by the RSVP-TE process, and sending information back.

RSVP-TE Processor. It is responsible for processing RSVP-TE messages coming from both the local API (MPLS Controller) and the underlying IP stack. The RSVP-TE processor includes:

- The RSVP-TE Path Processor (PSB), which is responsible for the Path direction processing.

- The RSVP-TE Resv Processor (RSB), which is responsible for the Resv direction processing.
- The RSVP-TE Traffic Controller (TCSB), which is responsible for Traffic Control related operations, including:
  - Creating/modifying/deleting Traffic Control State blocks.
  - Summarizing reservations committed for different senders.
  - Interacting (via the MPLS Controller) with the Traffic Control mechanisms implemented in the Layer 2 Interface module.

RSVP-TE Session Manager. It is responsible for:

- Creating outgoing and incoming RSVP-TE sessions.
- Notifying the MPLS Controller about newly available RSVP-TE sessions.
- Removing RSVP-TE sessions that have no PSBs, RSBs, and TCSBs attached.

RSVP-TE Encoder/Decoder. It is responsible for:

- Encoding RSVP-TE requests generated by the MPLS Controller into RSVP-TE messages and passing them to the underlying IP stack.
- Decoding incoming RSVP-TE messages and passing appropriate notifications to the Session Manager for soft-state management and to the MPLS Controller.

2.3 Overview of RTM

The Routing Table Manager (RTM) [13] is not a routing protocol in itself, but rather a link between the routing protocols. It gathers information from different sources such as the static routes configured by the system user and the dynamic routes. Based on all the route information from different routing protocols, RTM module will determine the Forwarding Information Table (FIT) that will contain the overall best routes.

The main functionalities of RTM are to retrieve all configured static routes, to compute the overall best route for any known subnets or host destinations, and then to generate the FIT table used by the IP stack and the hardware components responsible for forwarding IP packets, as shown in Figure 2.4. It dynamically updates the IP stack with the latest revision of the FIT table, and keeps it up-to-date whenever changes occur. This is achieved through an interface called routing sockets [13].

The RTM module is also responsible for ensuring route redistribution from one routing protocol to another. In addition, it also filters route information being redistributed.

The RTM also manages:

- Routing tables generated by underlying unicast and multicast protocols,
- Static routes,
- Routing tables on a per VPN basis,
- Best-effort and QoS based routing tables,
- Asynchronous notification to users about changes to the routing tables,
- Intra-domain and inter-domain next hop and explicit route resolution (for OSPF and BGP respectively) by looking up the routing tables,
- Support for LSP hierarchies (i.e., mapping from IP to MPLS, transit LSPs within MPLS domains, and mapping from MPLS to IP)

In general, the RTM can also be responsible for traffic engineering constraint-based path calculations. However, in a distributed architecture where routing and signaling protocols run entirely on the LCs, this task can be achieved by appropriate routing and signaling modules.

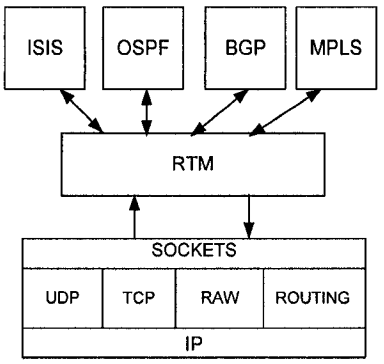

Figure 2.4 Inter-communication of RTM with Routing Protocols

In the next chapter we discuss the previous research conducted on MPLS/RSVP-TE centralized routers. Limitations of the centralized architecture have been discussed. It also discusses the advantages of the industrial next generation routers.

Chapter 3

3 Literature Review

Previous research on router architectures spans over several research areas, such as classification/lookups, internal switch architectures, software modularization and packet forwarding extensions, resource management and QoS [4]. Many of these areas are of interest to our research on distributed router architectures, and a broad understanding of these areas is useful for the distribution of RSVP-TE protocol and the overall system design included in our work.

We have taken into consideration the research done in the chosen field to make a closer description of the state of the art in the areas that are most related to the topic of distributed router architectures and distributed RSVP-TE module. There is an explanation given about how the previous work in these areas is related to the proposed architecture.

In this chapter we describe the distributed routers modules and explain their basic characteristics. Industrial next generation high-speed distributed routers [39] like Juniper [29], Nexabit [42],

Cisco [38] and Avici [18], operating with packet forwarding rates on the order of gigabits or even terabits per second are also discussed.

The continuously growing need for new services and protocols also affects the control plane of routers. Adding new services to a router often is a complex task from the control plane point of view. The number of protocols that has to be supported by a router is already significant [33]. These protocols, together with other functions needed in a router, constitute large amounts of complex software. This may consequently result in high demands on the control processor. Thus, it is hard to add new functionality in the control plane and meanwhile maintain a high degree of reliability and efficient execution of the software. Our intention is to take this decentralization further and study how a router can be distributed for allowing physical separation of control and forwarding functions into multiple modules.

For example, in 2005, HyperChip Router [24] was introduced with a new core router model which may support a very large number of LCs and control card (e.g., 64,000) and provide a very high throughput up to 1280 Gbps. The software architecture for next generation routers should therefore be distributed and highly scalable.

3.1 Distributed Software Architecture

When it comes to the overall system design of a distributed router, the architecture suggested within the IETF ForCES working group [34] will serve as a starting point. The ForCES architecture is defined in terms of exchange of information between control elements (CEs) and forwarding elements (FEs). A group of CEs and FEs together forms a network element (NE).

The ForCES protocol is used to associate the CEs and FEs. It updates the FEs with configuration information from the CEs, queries for information by the CEs or sends asynchronous event notifications to CEs. Using the ForCES protocol, the CEs may also configure the processing functions on the FEs [35] [36].

Control functions of a router have been redefined based on the ForCES architecture, in order to share the processing tasks between the CC and the LCs. In [27] authors present a Distributed Control Plane architecture where message processing, particularly HELLO Message processing, is handled by the LC. With this new approach failures can be detected faster and Shortest Path First (SPF) calculations can be run as frequently as required without affecting the load on the control plane processor.

The current ForCES architecture does not consider the general purpose CPU and memory on LCs which are available on the next generation routers. Moving some control functions to the LCs will lead to reprogramming the network processor (NP). This is costly and if there is an error in running the code of the control function may shutdown the forwarding function.

Second, distributing the MPLS/RSVP-TE routing protocol functions on to the LCs would increase the scalability of the system. Since the network processor on LCs is not able to host many control functions due to its specific architecture, using the general purpose CPU and memory available on the LCs can help to improve the processing performance of the next generation routers.

ForCES architecture has not currently been adopted by the industry due to it's architectural limitations. In contrast, industrial routers are a step ahead mainly with regards to the fact that they overcome the limitations posed by the ForCES architecture.

Based on a general distributed software framework, MPLS/LDP [26] is distributed on the line cards in order to fully exploit the hardware platforms of the next generation routers [25]. The proposed distributed MPLS/LDP architecture moves the signaling and table management to the line cards, with additional mechanisms for resiliency and task sharing at the line card level in order to increase the scalability of distributed router architecture. The main contribution for this research is presented in [45].

The next section describes the advancement in industrial routers with high-speed interfaces (e.g., 10 or 40 Gb/s) and large switching capacity (e.g., multipetabit) [2].

3.2 Industrial Next Generation Routers and Prototype

This section provides a survey of the terabit and gigabit capacity routers available in the market [44]. Comparative analysis of all the major products classifies them into various categories based on architecture design as well as performance. Current Next generation routers by leading vendors like Avici [18], Juniper [12], Nexabit [42] and Cisco [38] claim to support up to several OC-192 interfaces as well as multi-terabit switching capacity. Other prototypes like Hyperchip [24] and Tiny Tera [43] have also been introduced.

3.2.1 Juniper Next Generation Router

Juniper's next-generation routing architecture provides the solid, reliable, high-performance foundation upon which today's real-time, critical networking applications can be delivered. In August 1998, Juniper networks began shipping the M-series Internet router. This was the first Internet- scale router built to offer uncompromising packet forwarding performance. All

M-series Internet router family are designed to meet demands of the new public network by delivering high reliability and scalability [28].

In April 2002, Juniper Networks began shipping the T640 Internet routers. The T640 delivers breakthrough OC-48/STM-16 and OC-192/STM-64 interface density in a multi-chassis capable system [28].

Architectural framework of M-series and T-series router:

The M-series router architecture successfully provides the performance and control that carries need to support their transition from OC-12/STM-4 to OC-48/STM-16 [28]. Figure 3.1 illustrates the architecture of an M-series router; each input interface is logically connected to a switch which distributes the cells of a packet into a centralized shared memory system. The cells are written into memory once, read from memory once, and sent through a switch to the proper output interface for transmission to the network.

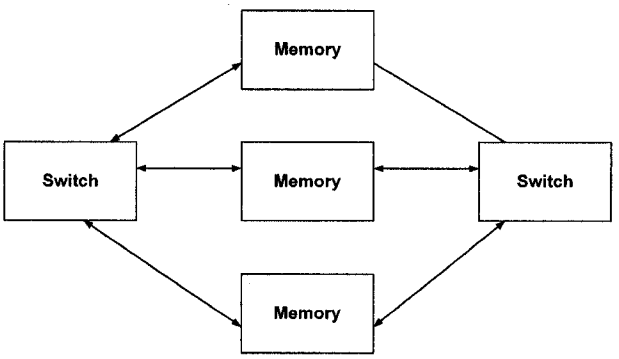

Figure 3.1 M-Series Router Architecture

Figure 3.2 illustrates the architecture of a T-series router. A T-series router uses a distributed architecture with packet buffering at the ingress Packet Forwarding Engine (PFE) before the switch fabric as well as packet buffering at the egress PFE before the output port.

A single chassis or multi-chassis T-series system consist of four components PICs, PFEs, the switch fabric and one or more routing engine.

- **Physical Interface Card**: The PICs connect a T-series router to the network and perform both physical and link layer packet processing.
- **Packet Forwarding Engine:** Each PFE performs routing table lookups, and packet filtering, forwarding packets to the correct output interface and manages output queues.
- **Switch Fabric**: It provides connectivity between the PFEs.
- **Routing Engine**: The routing engine executes the JUNOS software and creates the routing tables which are downloaded to each PFE.

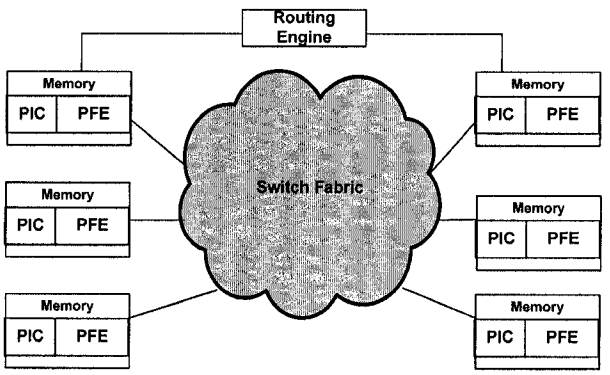

Figure 3.2 T-Series Router Entities

T-series router deliver industry-leading scalability, they provide the required scalability while maintaining feature and software continuity across all router platforms. T640 Internet Routing Node supports 32 OC-192c/STM-64 ports in a half-rack chassis, and offers eight 40 gigabits-per-second (Gbps) slots thus satisfying ever-increasing traffic in the networks.

3.2.2 Nexabit NX64000 Router

The NX64000 innovative switch fabric delivers 6.4 Tbps switch capacity per chassis. The NX64000 supports up to 192 OC-3, 96 OC-12, 64 OC-48 and 16 OC-192 lines [42]. The NX64000 allows Service Providers to even scale to higher speed like OC-768 and OC-3072 and port densities can also be increased further by interconnecting multiple chassis [41].

The NX64000 implements a distributed programmable hardware forwarding engine on each line-card. This approach facilitates wire-speed route lookup for full, 32-bit network prefixes-even at OC-192 rates. The distributed model enables support for 128 independent forwarding tables on each line-card. Each forwarding engine is capable of storing over one million entries.

The industry-leading routers deliver a powerful solution for carriers to achieve substantial technology breakthroughs in packet forwarding performance, bandwidth density, IP service delivery and system reliability. Not only do the next generation routers deliver industry-leading scalability, they provide the required scalability while still maintaining feature and software across all router platform.

3.2.3 Cisco Next Generation Router

The Cisco® XR 12000 Series routers accelerate the service provider evolution toward IP Next-Generation Networks [38]. Cisco XR 12000 Series provides intelligent routing solutions that scale from 2.5 to 10-Gbps capacity per slot, enabling next-generation IP/Multiprotocol Label Switching (MPLS) networks [37].

Cisco XR 12000 Architecture Components

The Cisco XR 12000 architecture consists of four basic building blocks:

- **Cisco IOS XR Software** - Cisco IOS XR Software takes full advantage of the distributed intelligence and hardware capacities on LCs and Cisco XR 12000 and 12000 Series Performance Route Processor-2 (PRP-2) technology on the Cisco XR 12000 Router.
- **Cisco XR 12000 multi- gigabit switch fabric** - The switch fabric is the high-speed interconnect between all router components, including the route processor and the Cisco intelligent programmable interface processors (Cisco ISE LCs).
- **Cisco XR 12000 intelligent programmable interface processors** - The Cisco intelligent programmable interface processors (Cisco LCs) terminate the physical interface and provide the forwarding decisions, payload processing, payload accounting, policy management, and security.
- **Cisco XR 12000 Series Control Plane Route Processor** - The Cisco XR 12000 Series Control Plane Processor provides chassis management, route protocol processing, and external management.

3.2.4 Hyperchip Prototype

The Hyperchip PBR-1280 Core IP System solves the problems of legacy routers to create a simple, low-cost IP core that provides the bandwidth quality needed not only for advanced IP services, but for traditional Voice and Frame Relay services as well. The PBR-1280 core IP system is fully compatible with legacy routers, allowing carriers to evolve their networks to reduce costs while preserving existing network architectures [19] [24].

PBR-1280 consists of a CC and a set of LCs connected via a switch fabric as shown in Figure 3.3. LCs contain very high speed interfaces (i.e., 10Gbps).

- One controller card that hosts all routing protocols. There can be an additional CC used for backup and redundancy.
- A given number of LCs which perform:
  - IP forwarding and/or MPLS label switching at hardware level, and
  - IP forwarding at software level for exceptional packets (e.g., control packets).

In this architecture there is a forwarding information table (FIT) on each LC. The RTM located on the CC receives the best routes learnt by routing protocols. Overall best routes are selected and then recorded to the FIT of the IP stack. The FIT on each LC is downloaded from the FIT on the CC via the switch fabric.

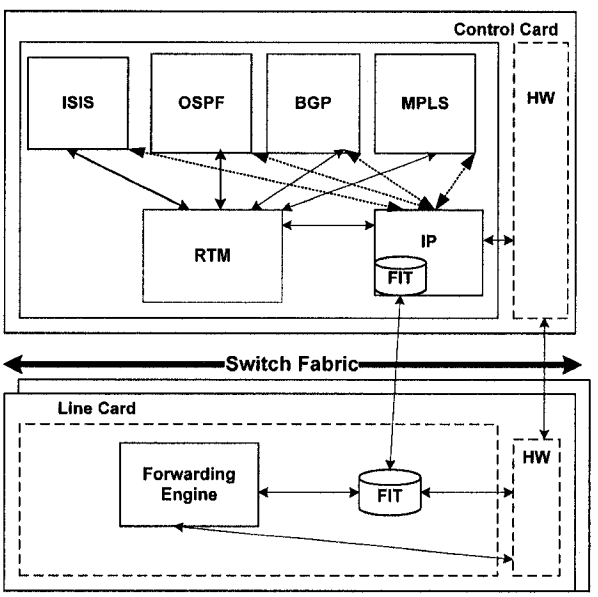

Figure 3.3 Hyperchip Architecture

3.2.5 Tiny Tera Prototype

Tiny Tera is a Stanford University research project, the goal of which is to design a small, 1 Tbps packet switch using normal CMOS technology [43]. The system is suited for an ATM switch or Internet core router. The current version has 32 ports each operating at a 10 Gbps (Sonet OC-192 rate) speed.

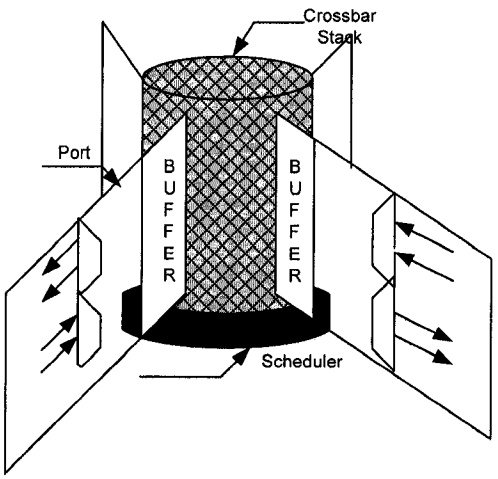

Figure 3.4 Architecture of Tiny Tera

In the next chapter we describe the distributed architecture for the Routing Table Manager (RTM) which is essentially one of the most important component of a router.

Chapter 4

4 Centralized and Distributed Architectures for RTM with RSVP-TE

This chapter gives a brief overview of the Routing Table Manager. The first requirement for a RTM distributed architecture came from RSVP-TE. RSVP-TE needs also to be distributed on LCs in order to increase the scalability and resiliency and to reduce the load of the CC. If RSVP-TE needs to compute the paths over network based on user-specific requirements, such as QoS. This is done with the help of routing protocols, like OSPF or BGP, through the RTM. Therefore, a distributed architecture for RTM is required to serve the RSVP-TE distributed architecture. In addition, the distributed architectures of RTMs may save the CPU resource and memory on the CC which are used to compute the best routes. Section 4.1 describes the centralized architecture for RTM. Section 4.2 describes a brief overview of partially distributed RSVP-TE with centralized RTM and limitation of this architecture. Section4.3 describes the distributed architecture for RTM with distributed RSVP-TE.

4.1 Centralized RTM Architecture

The current RTM architecture [25] used for next generation routers, consists of a Global RTM (G-RTM) module that manages the routing table for the whole system and several smaller RTMs, each devoted to a given protocol and therefore denoted by IGP-RTM or EGP-RTM, i.e.,

for each protocol. Each IGP/EGP-RTM is responsible for managing routes computed by a specific routing protocol (i.e. OSPF, BGP, MPLS), as shown in Figure 4.1. The G-RTM is a stand-alone process located on any CC in the system. It interfaces only with the IGP/EGP-RTMs and IP. It receives all the routes learned by the different IGP/EGP-RTMs and performs the selection of the overall best routes. Then it updates IP and the IGP/EGP-RTMs and finally performs route redistribution between protocols. The G-RTM also manages the configuration of static routes configured by users (through an external routing policy module) or traffic engineering based routes. The interface configuration and up/down status are handled by the G-RTM and broadcast to the routing protocol modules through the IGP/EGP-RTMs.

Each IGP/EGP-RTM is physically linked with a given protocol and gives the impressions to the protocol that it is a complete RTM. It contains all best routes of the protocol and the overall best routes of the system computed by the G-RTM used to update the protocol. It offers the same API, and keeps all pertinent information to the protocol for fast access and sustained performance.

Such architecture allows the routing protocols to have a flexible access to the routing tables managed by the G-RTM without being queued. The architecture does not require much modification on routing protocol modules and G-RTM regarding the traditional architecture. Each IGP/EGP RTM manages only a subset of the routing table that is related to a specific routing protocol. When a routing protocol receives a route update notification message through the corresponding signaling component on a LC, the control component located on the CC re-computes the best routes and updates its local IGP/EGP-RTM. The G-RTM is also notified through the link with the IGP/EGP-RTM. The overall best routes of the system are selected

among those provided by different protocols. The route update is advertised by the G-RTM to other routing protocols in order to notify the neighbors. Finally, the overall best routes are updated to the FIT on the LCs through the connection with the G-RTM.

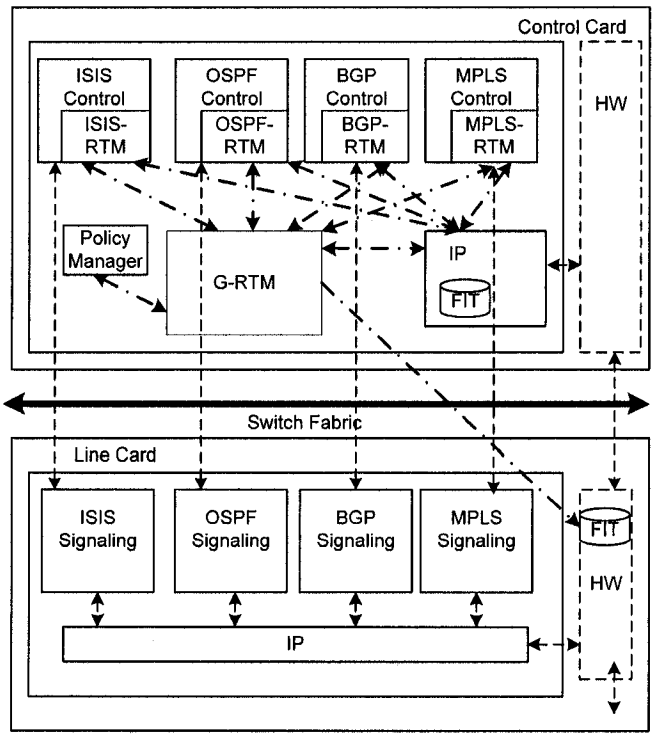

Figure 4.1 Current RTM Architecture: Distributed on Protocol Basis

The architecture does not reduce the number of messages sent to the CC, in other words, the congestion still remains. However, the resiliency and scalability are improved at the CC level because the routing protocol can still use the IGP/EGP-RTMs when the G-RTM temporary fails. The lookup operation is faster because each IGP/EGP-RTM contains only a portion of the global routing table. The main advantage of this scheme is the simplicity since not much modification is needed. Therefore, it can be suitable for the short-term migration from the current to the next

generation routers, where only the CC needs to be upgraded. However, there are some critical issues:

- Although the IGP/EGP-RTMs are distributed on a per protocol basis, they are basically independent processes running on the same CC. This leads to quite heavy resource consumption and to some overloading of the CC as the number of routes increases.
- Additional computing and memory resource on the LCs are not efficiently exploited to run the best route computations or to perform the route management tasks.
- The routing protocols will be distributed on the LCs in order to improve the scalability and fully exploit the available memory and CPU resource of the LCs [25], the IGP/EGP-RTM modules need also to be migrated to the LCs.
- It is not very efficient to perform the FIT update operations at the CC level by G-RTM as the FITs are hosted by the LCs.

In order to deal with these issues, in the following sections, new architectures for RTM will be proposed. We aim at a full distribution of RTMs on LCs where each LC has a RTM instance being responsible for the local routing protocols.

4.2 Partially Distributed RSVP-TE Architecture with Centralized RTM

Routing protocols, such as OSPF [6], IS-IS [7] or BGP [5] require path computations in order to produce best routes. The RSVP-TE protocol presented in the previous chapter is a signaling protocol for MPLS networks that performs simply the message exchanges between LSRs to establish the LSPs. RSVP-TE can be used alternatively as another signaling protocol for MPLS network that provides additional traffic engineering features. In traffic engineering we are

concerned with establishing LSPs that do not necessarily follow the IP best routes from the ingress to the egress computed by normal routing protocols like OSPF or IS-IS. This allows data to be sent by alternative routes to reduce bottle-necks and congestion, to increase the utilization of network-resources, and to avoid planned faults. The Resource Reservation Protocol (RSVP) was originally designed as a signaling for the Integrated Services (IntServ) model, wherein a host requests a specific QoS from the network for a particular flow. RSVP-TE is an extension of RSVP that has been adapted to support the traffic engineering within the MPLS network.

Signaling protocols run on LCs and can be used alternatively by the MPLS signaling module to establish the LSPs with the peer routers. Their concurrent access to the LIB table is controlled to avoid the data inconsistency. The labels provided by the L-LAT are given to RSVP-TE and LDP [26] according to their requirements. In Figure 4.2, there is only one RTM running on the CC that contains all the routes of the system and manages the interfaces of the routing protocols. Therefore, all path computation requests from RSVP-TE must go to the CC. The distributed MPLS architecture is therefore not deployed efficiently. In addition, the CC can be overloaded by the large number of requests when the number of LCs increases.

This issue can be dealt with a distributed RTM, where each LC has a RTM instance. The distributed RTM on each LC may contain the available routes of the routers, or all the routes out of the best routes that are directly related to the local LC. The RSVP-TE path computation request may therefore address the local RTM running on the same LC instead of the CC. Based on its routing table, the local RTM is able to call appropriate routing protocol to compute the connections to next-hop routes which satisfy user QoS requirements.

With a distributed RTM architecture, a RSVP-TE process will be able to consult the local routing database in order to obtain routes. Routing protocols determine where packets get forwarded; RSVP-TE is only concerned with the QoS of those forwarded packets. RSVP-TE sessions are launched according to user requests from application level. A host uses the RSVP-TE protocol to request specific qualities of services from a network for particular application data streams or flows. Routers use RSVP-TE to deliver QoS requests to all nodes along the paths of the flows and to establish and maintain a state to provide the requested services. RSVP-TE requests generally result in resources being reserved at each node in the data path.

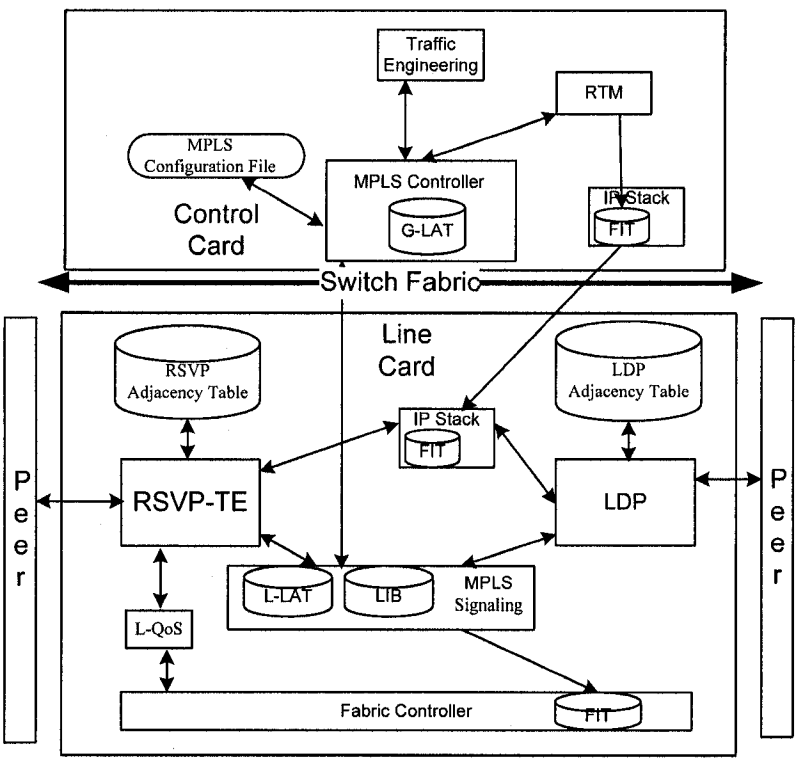

Figure 4.2 Distributed MPLS Architecture with a Centralized RTM

4.3 Fully Distributed RSVP-TE Architecture with Distributed RTM

The requirement of distributing RTM on line cards arise from the requirements of distributing signaling protocols such as LDP [26] and RSVP-TE. This section investigates the ability to distribute RTM on to the line cards [45].

A highly scalable router connects all ports on each LC to only one routing domain. In this context, proposed distributed architecture for RTM maintains a "master" for each cluster of LCs and each cluster is called a domain as shown in Figure 4.3. For example, domain D4 consists of line cards LC1, LC2, LC4 and LC5 and the master line card for this domain is LC5. The master LC instead performs the overall selection and management of the best routes from all routing protocols in its domain. This allows the master to be able to perform constrained shortest path first (CSPF) calculation on user-specific parameters (e.g., administrative weights) according to special requests (e.g., from RSVP). CSPF computation request (i.e., sent by a RSVP-TE module) from any LC in a domain will be forwarded to the master LC of that domain. The master LC-RTM performs the CSPF computation based on its database and on the IGP topology of the domain.

In this scheme, routes are managed at the LC level by LC-RTM process running on each LC. The LC-RTM handles the overall best routes of the domain supplied by the master. If a LC is assigned as master for a domain, its LC-RTM contains all available routes of the domain. The FIT on each LC is updated by the local LC-RTM. The LC-RTM of each LC has interfaces with local routing protocols modules so that route notifications can be sent from a given protocol to another one in order to advertise neighbors.

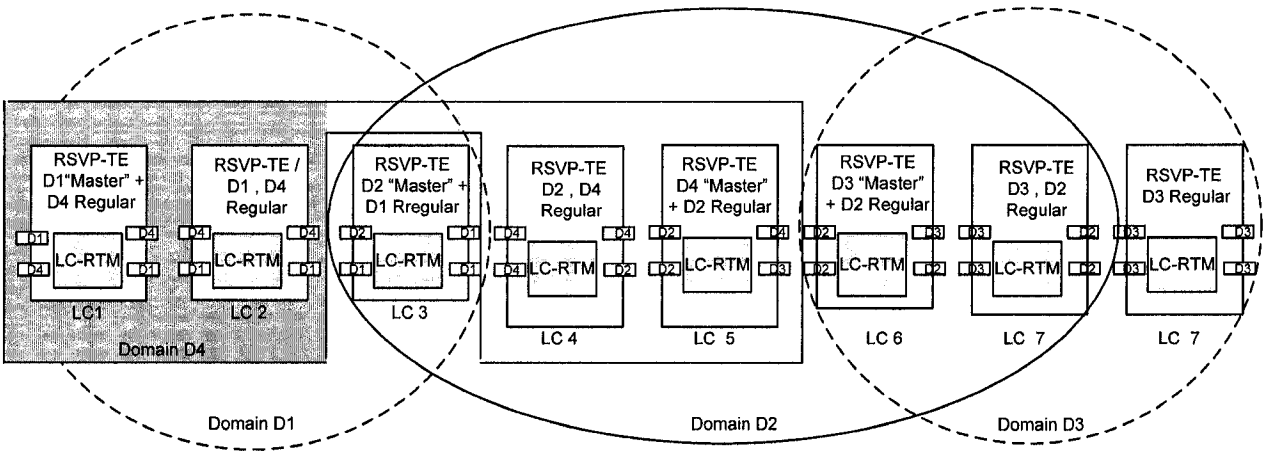

Figure 4.3 Distribution of LC-RTMs

Advantages

The main advantage of distributing RTM on the LC is that route information is handled on a master LC for each domain. Route information is available for RSVP-TE protocols without having to go up to G-RTM. In this case, CSPF computation is conducted at the level of the master LC.

Disadvantages

Large memory resource is required at the master LC in order to contain all available routes in the domain. In addition, the CSPF computation for every LSP in the domain requires message exchanges with the master: this result in an increase of the traffic among the LCs associated with the same domain.

4.4 Summary

In this chapter, we have presented the distributed architecture for the Routing Table Manager (RTM) which is essentially the most important component of a router. The RTM plays a decisive role for routing performance and connectivity of the network. The distribution of RTM was firstly required to serve the constrained shortest path first (CSPF) computation requests of a distributed RSVP-TE on LCs. Such distributed RTM architecture may also reduce considerably the load of the CC since parts of the path computation can be achieved on the LCs. This chapter also described the implementation architecture of the LC-RTM on the LCs and the use of such distributed architectures to compute CSPF path as RSVP-TE module requires. The communication among LC-RTMs and between LC-RTM and G-RTM is also discussed. In the next chapter we provide the detailed description of the various elements needed to be distributed for a MPLS/RSVP-TE distributed architecture.

Chapter 5

5 Elements for a Distributed and a Scalable Architecture for MPLS/RSVP-TE

In order to take advantage of the new generation router architecture, which provides ultra-high internal switching speed and additional processing and memory resource on LCs, we investigate the ability to distribute tasks from CC to the LC. This chapter targets next generation routers, with petabit **Error! Reference source not found.** [27] switching capacity, full memory and processing capabilities on LCs.

5.1 A General Framework for Distributed Software Architecture

The distributed routers have the following advanced features:

- The foundations of the proposed router architecture are built upon those of the next generation routers, where CC and LCs are separated in a distributed platform. Lines cards have full capacity of memory and processing power, and are able to perform data forwarding and some control tasks.

- Communication between CC and LCs is achieved through the switch fabric that is able to provide the required bandwidth and other QoS demands. The switching capacity is in order of a few petabits.
- There is a communication channel between LCs, enabling them to exchange control information with an insignificant impact on data flows. This channel is designed as an abstract layer called Distributed Service (DS). It provides a synchronization mechanism to manage module activations, monitoring and state transitioning facilities (active, backup, in-service upgrade, etc.).

The software implemented on the router platform is required to perform the following two primary procedures:

- **Data Forwarding:** This task is completely performed by the LCs in the current generation routers. The CC is no longer involved in this procedure. Once routes have been established and recorded into the routing table, data can be forwarded easily by network processors. Therefore the distributed software architecture we propose focuses on the routing and does not deal with data forwarding.
- **Routing and Signaling:** Information is exchanged (e.g., link state information) between different internetworking nodes in order to determine the paths through a network. In the centralized architecture, this task is performed by the CC. In the distributed software architecture we propose, this task is migrated to the LCs. The CC is required, to provide a global view of the network topology, according to the needs of the path computation.

Figure 5.1 presents the proposed architecture of RSVP-TE process located entirely on LC and its interactions with MPLS Controller and other independent modules on the CC and LC.

- Modification of the MPLS forwarder on the LC so that the MPLS swapping operation can be shared between ingress LC and egress LC.
- Replacing the connection between the G-RTM and MPLS controller by the connection between RSVP-TE and FIT (Forwarding Information Table) locally on the LC. This will reduce the number of message exchange between G-RTM and CC and LCs.
- Segmentation of the LAT table and its distribution into LCs, so that label allocation decisions can be made locally on LCs.
- Reallocating the LIB (Label Information Base) table into LC that enables MPLS data forwarding decisions to be done locally on LC without having to go through CC.
- Reallocating the RSVP-TE modules into LC. We merge the LIB and the RSVP-TE tables on the LCs. This will reduce the memory consumption to a certain extent.
- Balancing the RSVP-TE message processing task between the ingress LC and the egress LC. This task is performed only by CC in the centralized architectures of MPLS and RSVP-TE architecture. In the new proposed architecture, the Label Request processing task, including next hop and FEC lookups will be processed mainly by the ingress LC (upstream LC), while the Label Mapping processing task, including label allocation and Label Request matching will be processed mainly by the egress LC (downstream LC).

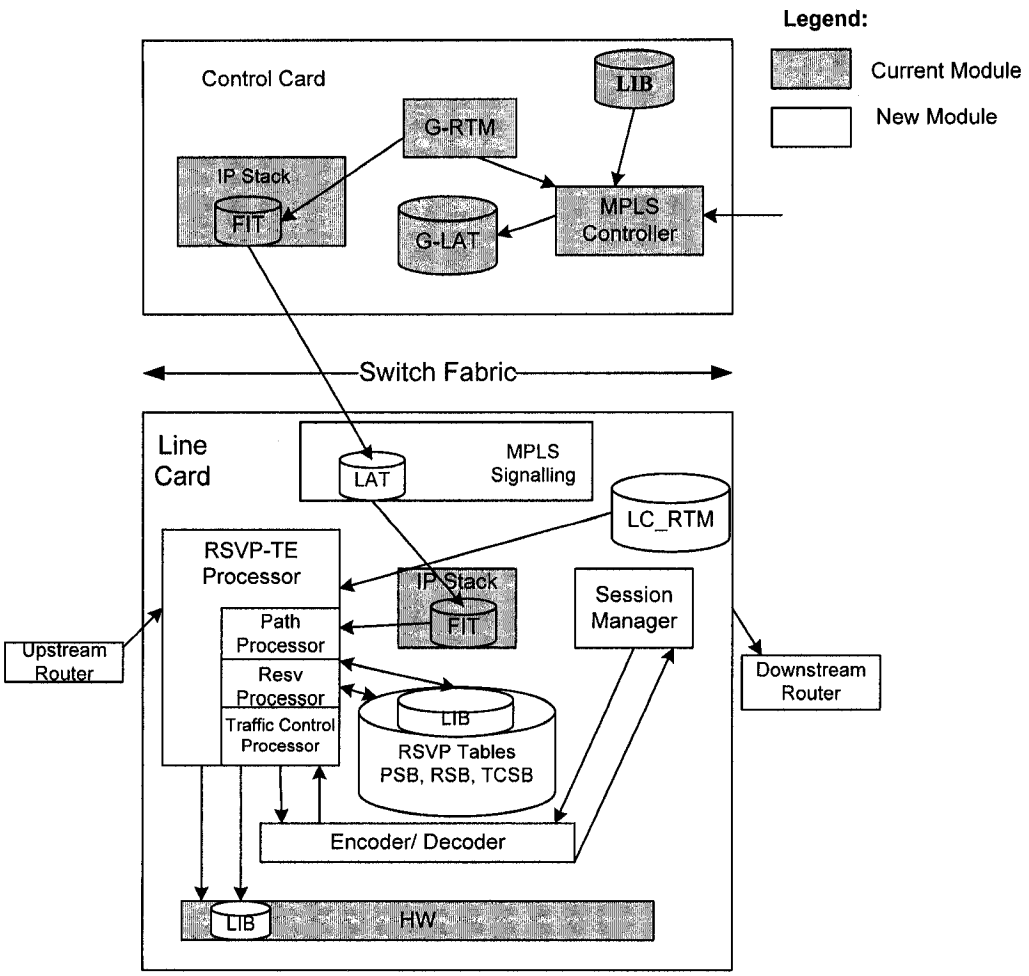

Figure 5.1 RSVP-TE Components on the LC

5.2 Table Management

This section describes the various tables that are used within the LC and CC and interactions with the RSVP-TE module.

Forwarding Table (FIT) used by the IP stack and the hardware components responsible of forwarding IP packets. We believe the FEC (IP route) lookup can be achieved by local FIT on LC in order to accelerate the processing speed and reduce the number of message exchanges between LC and CC. If we duplicate FIT on each LC, then any packets arriving on any of the LC

interface would not have to be directed to the central processor for route calculation and forwarding to the output links. RSVP-TE module will be able to find the egress interface of the path. This will reduce the major bottleneck on the CC.

Label Information Base (LIB):

The LIB contains information for labeling a data packet, changing the current label, or removing a label when the packet reaches the destination. There are two main tables: FTN (FEC-TO-NHLFE) and ILM (Incoming Label Mapping).

The FTN table is used for making MPLS forwarding decisions for unlabeled packets. When the ingress LC receives an unlabeled packet, it classifies the packet using a Flow Qualifier. Criterion used to qualify can be QoS class, VPN ID, and so on. If the packet is qualified for an LSP, the MPLS forwarder looks up the FTN table to find an entry that has the NHLFE corresponding to the LSP.

- If an FTN entry is found, the FTN table returns one or more NHLFEs associated with the entry used as instructions for packet forwarding.
- If the entry is not found, the LC performs IP based forwarding.

Figure 5.2 shows the structure of a FTN's record.

The ILM table is used for making MPLS forwarding decisions for labeled packets. When the ingress LC receives an MPLS labeled packet, it lookups the ILM table for the next hop and outgoing label using the incoming label as searching key.

- If the lookup succeeds, the ILM table returns the NHLFE associated with this label used as an instruction for packet forwarding.
- If the lookup fails, the packet is simply dropped.

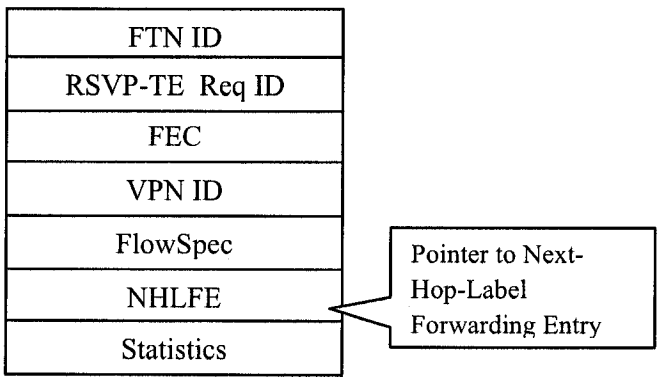

Figure 5.2 FEC-TO-NHLFE (FTN) Table Structure

Figure 5.3 shows the structure of a ILM's record.

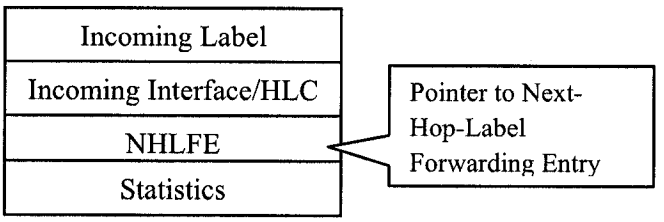

Figure 5.3 ILM Table Structure

Label Allocation Table (LAT): is used by MPLS Controller to manage label spaces and to keep track of all allocated labels. The LAT table is segmented in order to distribute the global label space into the LC. The MPLS LC-Controller uses the LAT object to manage label spaces and to track all allocated and ready-to-use labels. The LAT object is configured during initialization of

the MPLS LC-Controller interfaces and is consulted when a Label Switched Path (LSP) is established or removed. By reallocating the MPLS signalizing module onto the LC, label allocation decisions can be made locally on the LC.

G-RTM:

- The Global RTM (G-RTM) is stand alone, and can be located on any CC in the system. It interfaces only with the LC-RTM and IP. It receives all the routes learned by the different LC-RTMs running independently on each protocol and performs the selection of the overall best routes. Then it updates IP and the LC-RTM. It finally performs route redistribution between protocols.

- LC-RTM is located on the CC, the number of messages exchanged between the LC and the CC will be very enormous. In the distributed approach we duplicate the G-RTM to the LC, assuming there are no memory constraints. It deploys the link state database in order to compute the LSPs locally. In addition, the database lookup operation becomes more efficient because the database contains only paths going through the related LC. In future we would consider moving a subset of the G-RTM on each LC, due to space requirements on each LC.

5.3 Message Processing

The RSVP-TE processor is responsible for processing messages coming from both the MPLS Controller and the underlying IP stack. The RSVP-TE Processor includes:

- The RSVP-TE Path Processor, which is responsible for the Path direction processing.
- The RSVP-TE Resv Processor, which is responsible for the Resv direction processing.

- The RSVP-TE Traffic Controller, which is responsible for Traffic Control related operations, including:
  - Creating/modifying/deleting Traffic Control State blocks.
  - Summarizing reservations committed for different senders.
  - Interacting (via the MPLS Controller) with the Traffic Control mechanisms
  - Implemented in the Layer 2 Interface module.

The RSVP-TE processor maintains three databases Path State Block (PSB), Reservation State Block (RSB) and Traffic Control State Block (TCSB) for processing a RSVP-TE request [10].

- **Path State Block (PSB):** Each entry in the PSB table holds a path state for a particular session, sender pair, defined by SESSION and SENDER_TEMPLATE objects, respectively, received in a Path Message. If no entry corresponds to this session (characterized by a Session, Sender_Template pair), a new one is created. Otherwise, the old entry is updated if necessary. A Session object is required in every RSVP-TE message; Sender_Template describes the sender's data flow. Then, the Path Message is forwarded in order to propagate the message to the session's receiver. If this session corresponds to a reservation listed in the reservation state, then reservation refreshes are sent to previous hops (as the control architecture only supports unicast, only one reservation refresh may be sent).
- **Reservation State Block (RSB):** Each entry in the table RSB holds a reservation request that arrives in a particular Resv Message, corresponding to the (session, next hop, Filter_spec_list). The processing of Resv Messages is more complex. They are the messages responsible for reservations, so the RSVP-TE process not only has to record an incoming

Resv Message in the reservation state, but it may also merge the new reservation with a reservation already in place.

First of all, the RSVP-TE process has to find the corresponding path state entry. If no corresponding session has been found in the path state or if the entry found does not match the reservation made because of a port or style incompatibility, a Resv Error message will be sent back to the originator of the message. The information about the reservation in a Resv Message is contained in a series of flow descriptors which are located at the end of the message. The flow descriptor format depends on the reservation style.

- **Traffic Control State Block (TCSB):** Each entry in the TCSB table holds the reservation specification that has been handed to traffic control for a specific egress interface. In general, TCSB information is derived from RSB's for the same egress interface. Each TCSB defines a single reservation for a particular (session, Filter_spec_list).

5.4 RSVP-TE Data Base Distribution

The main problem in the RSVP-TE distributed architecture is the fact that any changes in the data base in an ingress/egress interface must be reported to the corresponding(s) ingress/egress interface(s).

In each entry in the tables LIB, PSB, RSB and TCSB, there is an indicating field corresponding to the ingress interface that forwards (inside the router via DS) the message, or the egress interface to which the message was sent (inside the router).

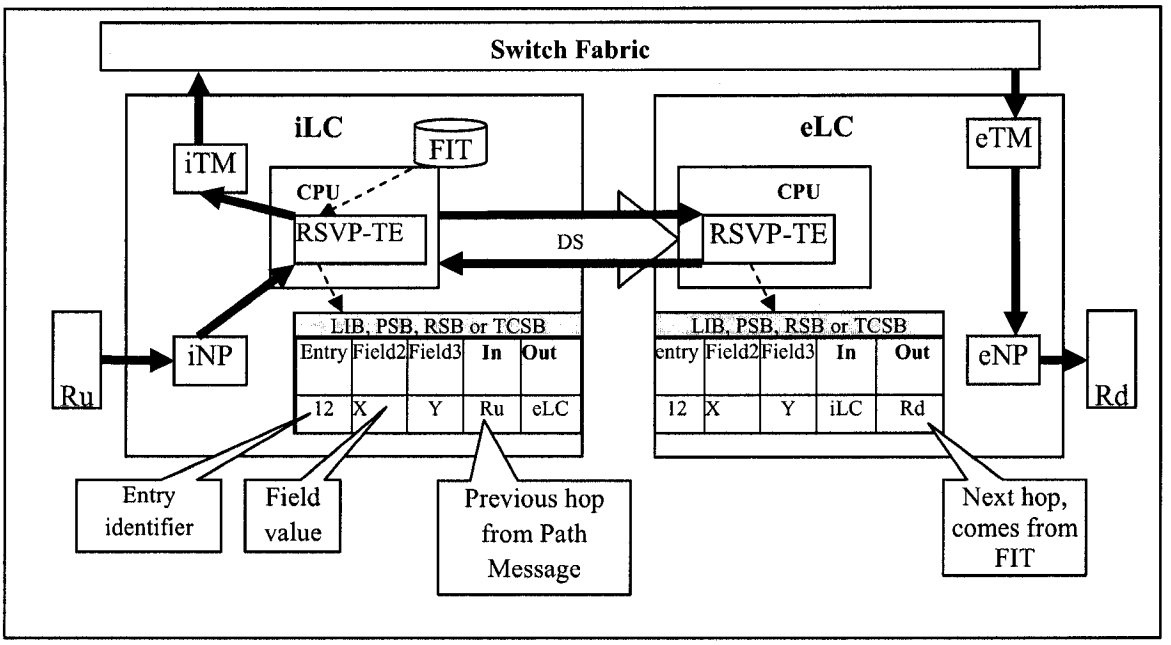

Figure 5.4 Distributing RSVP-TE Data Base in LCs

Let us consider the necessity of consistency among the databases of the different LCs between different instances of RSVP-TE data base. Figure 5.4 shows two LCs, with a same key entry in their table. The condition that must hold all the time before performing any operation on RSVP-TE messages is that if two entry IDs have the same value in the tables of an egress and ingress LCs then the values in the remaining fields must be identical before performing any operation on messages exchanged within a RSVP-TE process.

5.5 Resource Reservation with QoS Module

QoS is the ability to differentiate diverse classes of traffic based on predefined or user-defined criteria and assign priorities based on traffic variables that affect the treatment of traffic on each router in the network [1]. QoS, when implemented, becomes a requirement for end-to-end service delivery. In some cases, QoS might not be chosen for implementation in networks with excess bandwidth on links.

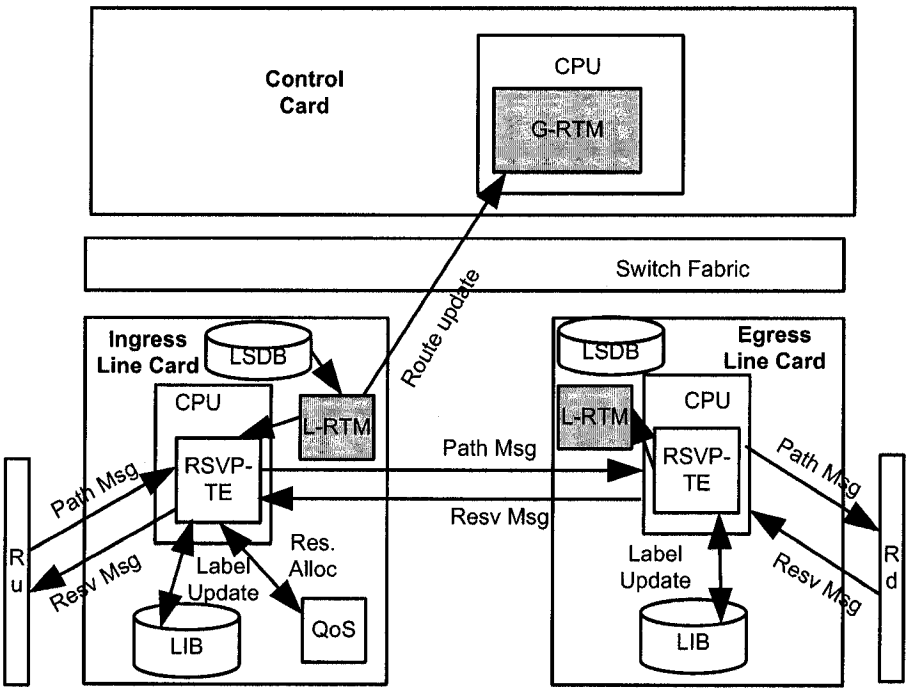

Figure 5.5 Resource Reservation for a Distributed RSVP-TE

In order to make resource reservation, RSVP-TE module is traditionally located on the CC. This is due to the fact that CC is aware about the resource consumption and takes responsibility of contacting the QoS module to reserve bandwidth on LC. Potential congestions can be experienced on CC due to the fact that the centralized architecture is not able to satisfy the huge demands of RSVP-TE sessions (e.g.., up to thousands of transit tunnels.)

In the distributed architecture, RSVP-TE module is moved to the LC as shown in Figure 5.5. Relocating of signals and reservations on the LC, enables messages to be exchanged and QoS provisioned locally. This enables RSVP-TE to handle control on specific QoS modules located on the LCs. Decision for resource reservation can therefore be made at the traffic manager chipsets on the LCs instead of going to the CC. Reallocating control task on the CC in order to

compute TE paths are performed by using the global information available on the CC. This information is provided by routing protocols.

5.6 Summary

This chapter discussed the mechanisms to reallocate MPLS/RSVP-TE functions on router's components namely CC and LCs. Such architecture requires solutions for additional challenges, such as synchronization, consistency between data and control planes, distribution of MPLS labels, and the restoration of tables in case of failures. Designs for MPLS/RSVP-TE tables have also been discussed which are used for MPLS forwarding. Detailed description of RSVP-TE protocol message processing has been addressed in the next chapter.

Chapter 6

6 A New Distributed & Scalable MPLS/RSVP-TE Architecture

This chapter presents the mechanisms to achieve MPLS/RSVP-TE functions on router's components, namely CC and LCs. Such architecture requires solutions for additional challenges, such as synchronization, consistency between data and control planes, and distribution of MPLS label.

This chapter is organized as follows. Section 6.1 gives a brief overview of ingress LCs and egress LCs. Section 6.2 provides a detailed description of RSVP-TE messages processing.

6.1 Definition of Ingress and Egress Line Card

In order to identify the ingress and egress LCs, a function called Is_Egress_IF() has been implemented. Is_Egress_IF() determines if the current LC is egress or ingress. This function can be based on the source where data is coming to the LC's CPU: if data comes from DS then the LC is egress, otherwise if data is from I/O interface then it is ingress as shown in Figure 6.1.

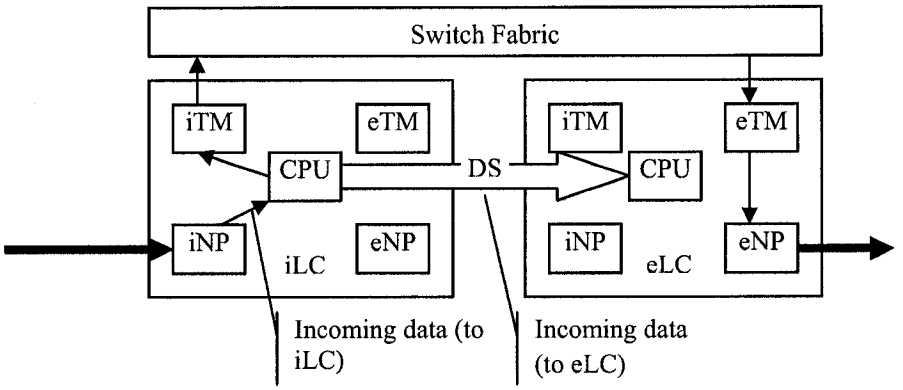

Figure 6.1 Ingress LC and Egress LC

In general, RSVP-TE message is received by the iNP. A filter is implemented on iNP to detect RSVP-TE messages and forward them to the CPU. The CPU of the ingress LC extracts information from the RSVP-TE message then forwards it to the CPU of the egress LC using the DS layer.

Egress router can also be identified by a function called Is_Egress_Router(), which is based on examining the data packet. If no label is found in the header of the data packet then it reaches the egress router.

6.2 Detailed description of a RSVP-TE Protocol Message Processing

The two most important messages of the RSVP-TE protocol are the Path Message and the Resv Message. RSVP-TE Path Message is responsible for initiating a request made by the ingress node to bind labels to a specific LSP tunnel. For this purpose, the RSVP-TE Path Message is augmented with a LABEL_REQUEST object. Labels are allocated downstream and distributed (propagated upstream) by means of the RSVP-TE Resv Message. For this purpose, the RSVP-TE Resv Message is extended with a special LABEL object.

To establish an LSP tunnel, the sender creates a Path Message with a LABEL_REQUEST object. The LABEL_REQUEST object indicates that a label binding for this path is requested and provides an indication of the network layer protocol that is to be carried over this path. This permits non-IP network layer protocols to be sent down an LSP. The LABEL_REQUEST object is stored in the Path State Block. When the Path Message reaches the receiver; the presence of the LABEL_REQUEST object triggers the receiver to allocate a label and to place the label in the LABEL object for the corresponding Resv Message.

If a label range was specified, the label must be allocated from that range. A receiver that accepts a LABEL_REQUEST object must include a LABEL object in Resv Messages pertaining to that Path Message. If a LABEL_REQUEST object was not present in the Path Message, a node must not include a LABEL object in a Resv Message for that Path Message's session and PHOP. A node that sends a LABEL_REQUEST object must be ready to accept and correctly process a LABEL object in the corresponding Resv Messages.

6.2.1 Downstream Processing of Path Message Requests

Figure 6.2 shows the processing of the Path Message from ingress LER (iLER) to egress LER (eLER). Details of the processing of these messages are provided in the following sub-sections. RSVP-TE messages can be processed differently depending on the location of the LC and router on the LSP. Each LC can perform the functionalities of iLER, LSR or eLER LC.

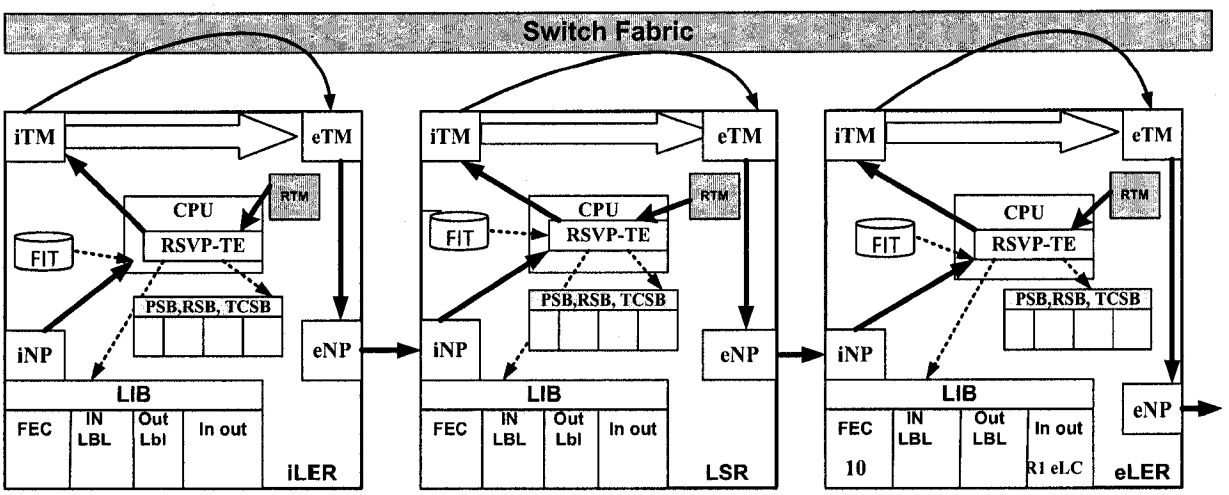

Figure 6.2 Downstream Processing of Path Message

6.2.1.1 Processing a Path Message at LER

Interactions between the CLI and the MPLS controller takes place on the CC for MPLS configuration or Targeted Discovery command. G-RTM send an IP route (FEC) update notification or FIT lookup. FIT is updated through IP stack. G-RTM receives the static route update through QoS module. Then Label allocation takes place through the LAT on the LC. MPLS controller on LC sends a label request message to the RSVP-TE processor with the FEC value included in the Label_ Request object as shown in Figure 6.3. The message sequence is described below.

- Firstly interactions between the CLI and the MPLS controller takes place for MPLS configuration or Targeted Discovery command (1).
- G-RTM then sends an IP route (FEC) update notification or FIT lookup (2). FEC is updated through IP stack.
- Then G-RTM receives the static route update through QoS module (3).

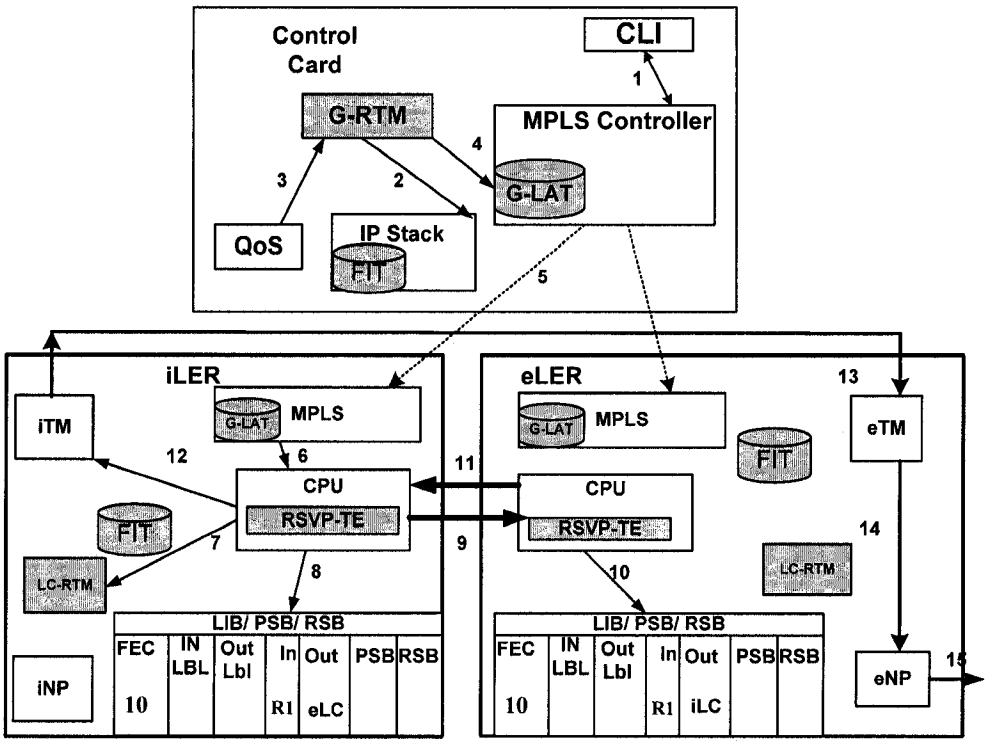

Figure 6.3 Processing RSVP-TE Path Message on LER

- Next, Label allocation takes place through the LAT on the LC. MPLS controller on the LC and on the CC have been synchronised together for consistency (4).
- RSVP-TE processor receives the LABEL REQUEST message is sent by MPLS controller containing FEC 10 (5).
- The LABEL REQUEST message is then forwarded to the RSVP-TE module (6).
- The RSVP-TE processor then determines the address of the outgoing interface by looking up the LC-RTM with FEC as search key (7)
- It then creates an entry in the LIB (8) where the fields contain the following values:
  - **FEC**: the FEC which needs a label,

- **Incoming label**: NULL, this label will be generated when the LABEL MAPPING message arrives in the return path,
- **Outgoing label**: NULL, this is used to indicate that this entry is incomplete and it is waiting for LABEL MAPPING reply,
- **Incoming interface**: address of the sender from which the LABEL REQUEST message is sent, that is the address of the upstream router for the ingress LC or the address of the ingress LC for the egress LC,
- **Outgoing interface**: the location of the egress LC.
- It then creates a Path State Block (PSB) entry for the new LSP to be established. The entry of the new PSB is disabled until an acknowledgement from the egress interface is received. It then generates the RSVP-TE Path Message for this PSB.

- Use DS to forward the original LABEL REQUEST message to the corresponding egress LC, followed by the address of next hop found from FIT (9).
- When the egress LC receives the LABEL REQUEST message from the ingress LC sent over DS, it performs the following operations:
- Create an entry in the LIB (10) with identical fields as in the ingress LIB:
- Send an ACK back to the ingress LC. The ACK message contains also the nexthop (11).
- After receiving an ACK message from the egress LC (11), forward LABEL REQUEST message to iTM (12).
- The message travels the switch fabric (12) in order to reach the downstream router (13), (14), (15).

- Wait for Resv Message reply from the downstream router. Resv Message includes the LABEL MAPPING object .The behavior of LCs when receiving the Resv Message is described in Figure 6.6.

6.2.1.2 Processing a Path Message at LSR (Transit Router)

Figure 6.4 shows the processing of a Path Message which includes the LABEL REQUEST object sent from the upstream router. The LABEL REQUEST message is sent by a remote peer (1). It is filtered by the local iNP and sent to the RSVP-TE process (2). The process looks up the local LIB to determine whether an entry with the same FEC and PSB entry is already there (3). The searching key is the FEC contained in the LABEL REQUEST message.

- If an entry is found with all the label fields completed, the ingress LC sends a Resv Message upstream by the binding it finds from the LIB.
- If an entry is found with an entry in the FEC field, PSB field and Label fields are still waiting. If PSB is found with a matching sender host but the SrcPorts differ and one of the SrcPorts is zero, it then builds and sends an "Ambiguous Path" PERR message.
- If an entry is found with an entry in the FEC field, PSB field and Label fields are still waiting for Resv Message, which includes the Label _Mapping object, it processes the message as a Path Refresh Message. It updates the PSB fields in the LIB and executes steps from (4 -13) as mentioned below. The RSVP-TE module in the ingress interface starts a periodical timer, with a configurable time interval. Path Refresh message will be sent thrice if the label fields are still empty and waiting for a Label Mapping request. Then the timer times out and then the PSB will be destroyed.

- If no entry is found, the ingress LC performs the following tasks:
  - Determines the address of the outgoing interface (on the egress LC) by looking up the FIT table with FEC as searching key (4).
  - Creates an entry in the LIB (5) where the fields contain the following values:
    - **FEC:** the FEC which needs a label,
    - **Incoming label:** NULL, this label will be generated when the LABEL MAPPING message arrives in the return path,
    - **Outgoing label:** NULL, this is used to indicate that this entry is incomplete and it is waiting for LABEL MAPPING reply,
    - **Incoming interface:** address of the sender from which the LABEL REQUEST message is sent, that is the address of the upstream router for the ingress LC or the address of the ingress LC for the egress LC,
    - **Outgoing interface:** the location of the egress LC.
    - It then creates a Path State Block (PSB) entry for the new LSP to be established. The entry of the new PSB is disabled until an acknowledgement from the egress interface is received. It then generates the RSVP-TE Path Message for this PSB.
- Uses DS to forward the original LABEL REQUEST message to the corresponding egress LC, followed by the address of next hop found from FIT (6).
- When the egress LC receives the LABEL REQUEST message from the ingress LC sent over DS, it performs the following operations:
- Create an entry in the LIB (7) where the fields contain the following values:
  - **FEC:** the FEC which needs a label, this field is used as key to lookup the requester in the return path,

- **Incoming label:** NULL, this label will be generated when the LABEL MAPPING message arrives in the return path,
- **Outgoing label:** NULL, this is used to indicate that this entry is incomplete and it is waiting for LABEL MAPPING reply,
- **Incoming interface:** address of the ingress LC,
- **Outgoing interface:** the address of the downstream router that is provided by the ingress LC following the LABEL REQUEST message.
- It then creates a Path State Block (PSB) entry for the new LSP to be established.

- Send an ACK back to the ingress LC. The ACK message contains also the nexthop (8).
- After receiving an ACK message from the egress LC (8), forward LABEL REQUEST message to iTM (9).
- Wait for LABEL MAPPING reply from the egress LC.
- The message travels the switch fabric (10) in order to reach the downstream router (11), (12), (13).
- Wait for Resv Message reply from the downstream router. Resv Message includes the LABEL MAPPING object .The behavior of LCs when receiving the Resv Message is described in Figure 6.6.

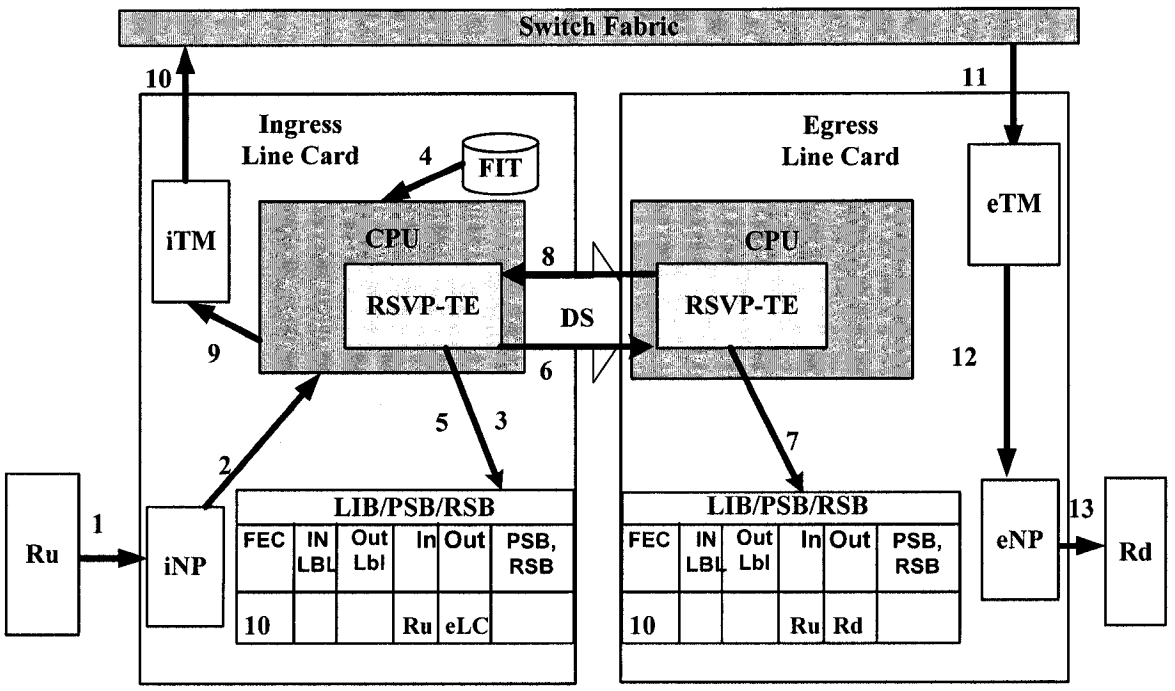

Figure 6.4 Path Message Processing on LSR

6.2.2 Upstream Processing of Resv Message Requests

In the following Figure 6.5 shows the flow of the Resv Message from iLER through LSR to eLER. These messages are processed differently depending on the position of the LC and router on the LSP. Three cases can be taken into account:

- Ingress LER LC
- egress LSR LC of an LSP intermediate router (transit router)
- egress LER LC of an egress router

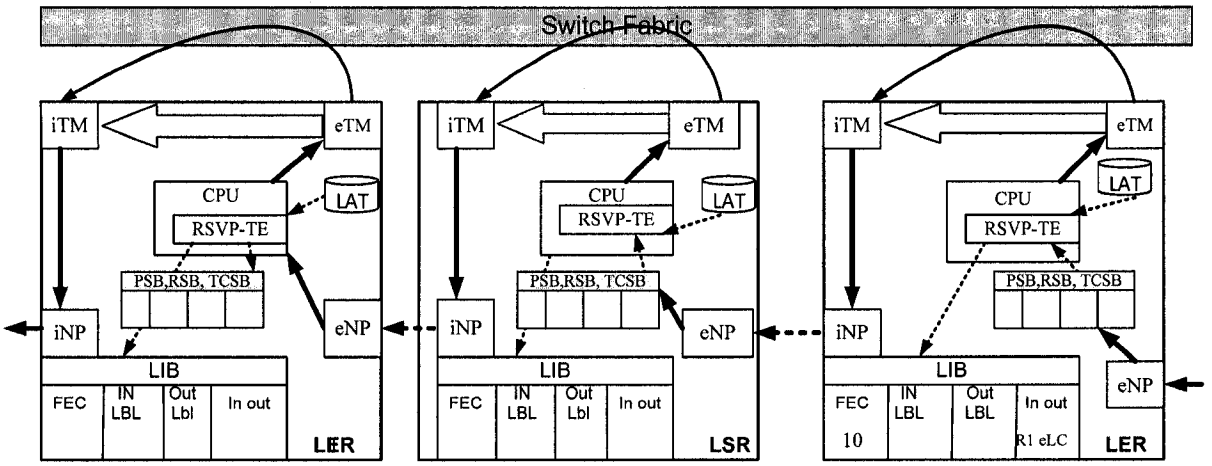

Figure 6.5 Processing Resv Message (Upstream On demand Mode)

6.2.2.1 Processing a Resv Message at LER

Processing of Resv message on iLER and eLER are similar to the steps performed at the transit router. These steps have been discussed in the next section.

6.2.2.2 Processing a Resv Message at LSR (Transit Router)

The return path of the Resv Message which includes the LABEL MAPPING object is shown in Figure 6.6. Upon receiving a LABEL MAPPING message (1) (2) corresponding to the previous LABEL REQUEST, the egress LC looks up the local LIB (3) to determine if there is an entry waiting for this mapping request. The searching key is the FEC contained in the LABEL MAPPING message.

Figure 6.6 illustrates the Resv Message processing procedure on iLC and eLC on a transit router.

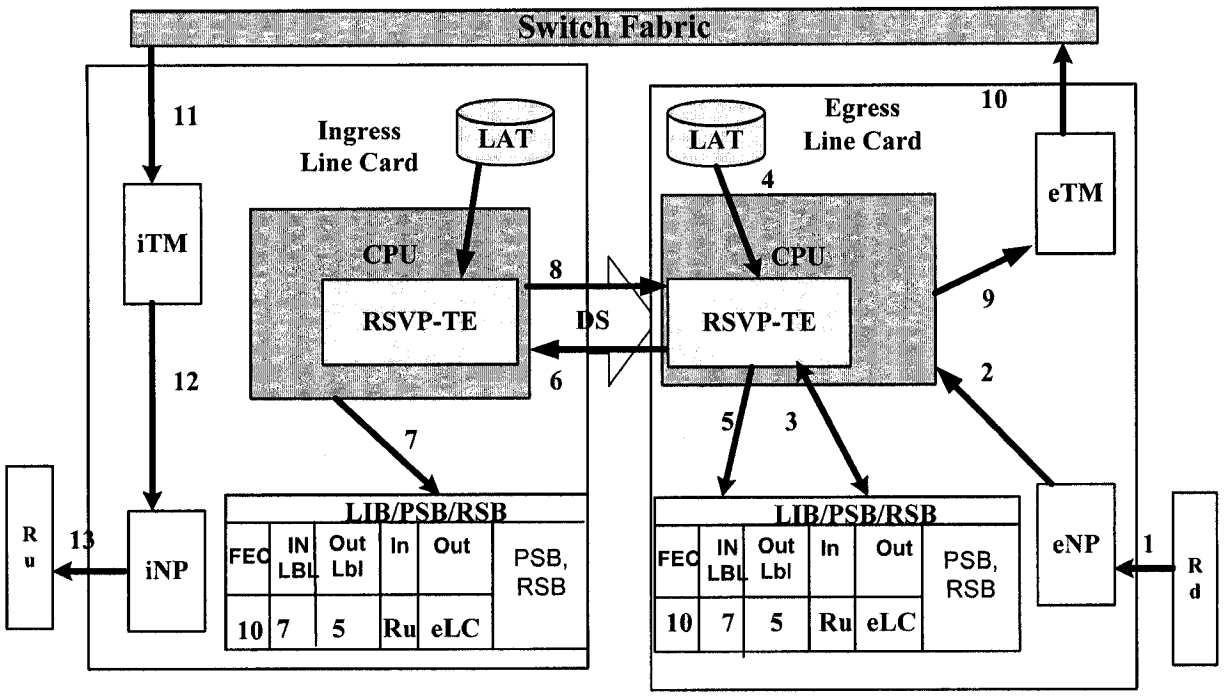

Figure 6.6 Processing RSVP-TE Resv Message on LSR

- If an entry is found in the label fields, it processes the message as a Resv Refresh Message. It updates the RSB fields in the LIB and executes steps from (6 -13) as mentioned below and send a Resv Confirmation Message downstream. The RSVP-TE module in the ingress interface starts a periodical timer, with a configurable time interval. The timer will time out if Resv Refresh messages have been sent thrice and no Resv Confirmation Message were received. Then the timer times out and, the PSB will be destroyed.
- If there is no existing PSB's entry found.
  - Send a RERR message specifying "No path information", drop the Resv Message.
  - If a PSB is found with a matching sender host but the SrcPorts differ and one of the SrcPorts is zero, then build and send an "Ambiguous Path" PERR message, drop the Resv Message.
- If an entry is found (with the label fields still empty), the egress LC performs the following:

- Generate a new label (taken from local L-LAT) (4),
- Update the entry; the outgoing label field is filled with the label contained in the LABEL MAPPING message, and the incoming label field is filled with the new generated label (5),
- Find or create a reservation state block (RSB) and call this the "active RSB".

- Using DS to forward the original LABEL MAPPING message to the corresponding ingress LC, including the new label the egress LC has generated. The address of the ingress LC is found in the entry of the LIB (6),
- After receiving an ACK message from the ingress LC (8), the egress LC replaces the label in the original LABEL MAPPING message by the new generated label, then forward LABEL MAPPING message through the iTM to the upstream router (9).
- When the egress LC receives a LABEL MAPPING message from the ingress LC sent over DS, followed by a label, it looks up the local LIB to determine if there is an entry waiting for this mapping previously. The searching key is FEC contained in the LABEL MAPPING message (6).

  If an entry is found (where the label fields are empty), the egress LC performs:

  - Update the entry (7); the outgoing label field is filled with the label contained in the LABEL MAPPING message, and the incoming label field is filled with the second label.
  - Send an ACK back to ingress LC (8),
- If no entry is found, the following is done:
  - Create a new entry in the LIB table,
  - Lookup the FIT for the upstream router. It is in fact a reverse lookup.

- Fill in the new entry with the second label (to the incoming label field), original label (to the outgoing label field) and FEC (to the FEC field) in the LABEL MAPPING message, address of the egress LC (to the incoming interface field) and address of the router from which the LABEL MAPPING message comes (to the outgoing interface field).
- Find or create a reservation state block (RSB) and call this the "active RSB".

- Send an ACK back to the ingress LC (8).
- The LABEL MAPPING will then be forwarded to the upstream in case of transit LSPs.

6.3 Summary

This chapter describes a distributed architecture for MPLS/RSVP-TE developed from the general distributed framework for signaling protocols described in Chapter 5. MPLS support is one of the emergent requirements for the next generation routers. Details processing of RSVP-TE message and synchronization issues have been discussed in detailed. Next Chapter provides the performance analysis of RSVP-TE signaling messages in terms of CPU and memory consumption, load distribution and bandwidth utilization.

Chapter 7

7 Performance Analysis of MPLS/RSVP-TE Messages (Distributed vs. Centralized Architectures)

Performance has been evaluated in order to compare the distribution architecture and the centralized architecture by estimating the number of exchanged messages, the number of consumed CPU cycles and the amount of required memory in the two architectures through some experiments with different router configurations. We propose a mathematical model in order to evaluate the number of RSVP-TE messages in centralized and distributed architectures as shown in Figure 7.1 and Figure 7.2.

Multiple plane switch fabric has been introduced in order to increase the forwarding performance and the bandwidth density. We propose a deterministic mathematical model to balance the load on the switch fabric. Performance evaluation in terms of bandwidth utilization is estimated between the centralized and distributed architectures.

7.1 CPU Cycles and Memory Consumption

In this section we evaluate the total number of CPU cycles required and the total amount of memory utilized in centralized and distributed architecture.

Let:

N_{LC}: Number of LCs in a router.

P: Number of ports in a router

N_C: Number of needed CPU cycles in order to run the route computation algorithm (either on the LC or on the CC to compute the RSVP-TE messages.

N_{Cmsg}: Total number of CPU cycles used to process a message on the LC or on the CC.

N_{LCi} : Number of LCs per domain

N_D : Number of domains

N_{LSP} : Number of LSPs per port

A_{LSP} : Number of established LSPs per port

M: Required memory to store one route on the LC or on the CC

Using the notations we deduce that the total number of ports in a router is $P\times N_{LC}$. For the centralized architecture, maximum number of RSVP-TE messages that can be processed on a given port on a LC $\sum_{i=1}^{N_{LSP}} A_{LSP}$.

Therefore the overall number of messages is:

$$N_{Message} = \sum_{i=1}^{N_{LSP}} A_{LSP}$$

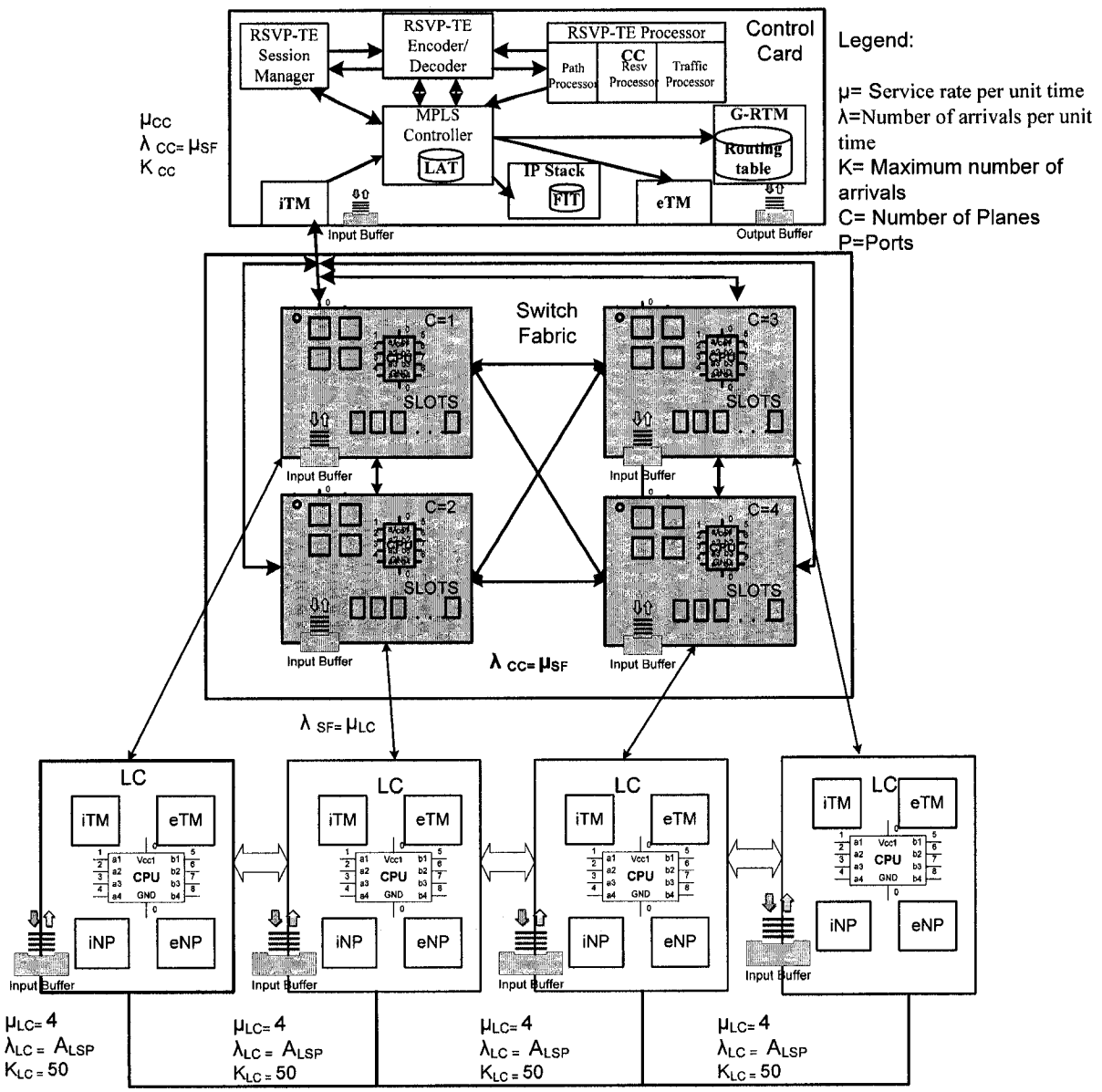

Figure 7.1 Centralized Architecture with Multiple Planes in a Switch Fabric

Number of CPU Cycles:

In a centralized architecture, for the control card to process and compute RSVP-TE messages, we need:

- N_{Cmsg}CPU cycles to determine the egress line card and the downstream router.
- N_C CPU cycles to compute the RSVP-TE messages.
- N_{CFIT} CPU cycles to determine the egress line card and the downstream router.
- N_{Label} CPU cycles to allocate a label from the LAT. N_{CLIB} CPU cycles to record the LSP to the LIB.

Therefore the total number of CPU cycles required to compute RSVP-TE messages is:

$$N_C = N_{CFIT} + N_{Label} + N_{CLIB}. \quad (7.1)$$

Number of CPU Cycles Required for Message Processing:

The number of RSVP-TE messages that may arrive on a given port is:

$$N_{Message} = N_{Path\ Message} + N_{Resv\ Message} + N_{Tear\ Message} + N_{Error\ Message} + N_{Conf\ Message} + N_{Refresh\ Message}.$$

Maximum number of RSVP-TE messages that can be processed on all ports in a router is:

$$N_{Message} \times P \times N_{LC}. \quad (7.2)$$

The CC has to process all the RSVP-TE messages coming from all the LCs. Therefore, the number of CPU cycles used is:

$$N_{Message} \times P \times N_{LC} \times N_{Cmsg}. \quad (7.3)$$

Therefore the total message processing required in the centralized architecture is, processing of RSVP-TE messages received on the LCs and processing of all RSVP-TE messages forwarded by the LCs to the CC.

$$Total_{MessageProcessing} = 2 \times (N_{Message} \times P \times N_{LC} \times N_{Cmsg}). \quad (7.4)$$

Path Computation:

Maximum number of CPU cycles required to compute the RSVP-TE messages received on the CC is:

$$Total_{PathComputation} = N_{Message} \times P \times N_{LC} \times N_C. \quad (7.5)$$

The CC updates all the LCs after computing the RSVP-TE messages. The numbers of required updates is equal to the number of LCs in a router. The Maximum number of CPU cycles required to process all the update messages is:

$$N_{LC} \times N_{Cmsg}. \quad (7.6)$$

Using relations (7.4), (7.5) and (7.6), we deduce that the total number of CPU cycles required to process and compute RSVP-TE messages is:

$$\underbrace{2 \times N_{Message} \times P \times N_{LC} \times N_{Cmsg}}_{(7.4)} + \underbrace{P \times N_{Message} \times N_{LC} \times N_C}_{(7.5)} + \underbrace{N_{LC} \times N_{Cmsg}}_{(7.6)}.$$

Memory Requirement:

In centralized architecture, LSPs are managed by the MPLS Controller. Therefore, the amount of memory needed for the global LIB on the control card is equal to the memory needed to process all the RSVP-TE messages received on all the ports of the router is:

$$M \times P \times N_{LC} \times N_{Message}. \quad (7.7)$$

The memory needed to process all the update messages sent to all the LCs by CC

$$N_{LC} \times M. \quad (7.8)$$

Using relations (7.7) and (7.8) we deduce that the total memory required is:

$$M_{Total} = M \times N_{Message} \times P \times N_{LC}. \quad (7.9)$$

Distributed Architecture

Let:

Total number of ports on master LC = P

Number of ports on regular LCs = $(N_{LC_i} - 1) \times P$

The maximum number of RSVP-TE messages processed by each port on a router is:

$$N_{Message} = \sum_{i=1}^{N_{LSP}} A_{LSP}.$$

In the distributed architecture, the maximum number of RSVP-TE messages received by the non-master LCs in domain *i* is:

$$\left(N_{LC_i} - 1\right) \times P \times N_{Message}. \tag{7.10}$$

The number of CPU cycles required to process RSVP-TE messages received on non master LCs is:

$$\left\{N_{LC_i} - 1\right\} \times P \times N_{Message} \times N_{Cmsg}. \tag{7.11}$$

The total number of RSVP-TE messages received by master LCs in domain *i* are the messages received from regular LCs as described in Equation (7.10) and messages received on all the ports of master LC itself. Therefore the total numbers of RSVP-TE messages received is

$$N_{LC_i} \times N_{Message} \times P \tag{7.12}$$

i.e.,

$$\underbrace{\left\{N_{LC_i} - 1\right\} \times P \times N_{Message}}_{\text{Regular LCs}} + \underbrace{P \times N_{Message}}_{\text{Master LC}}$$

Using relation (7.12), we calculate the total number of CPU cycles required to process the RSVP-TE messages on mater LC, it is equal to:

$$P \times N_{LC_i} \times N_{Message} \times N_{Cmsg}. \tag{7.13}$$

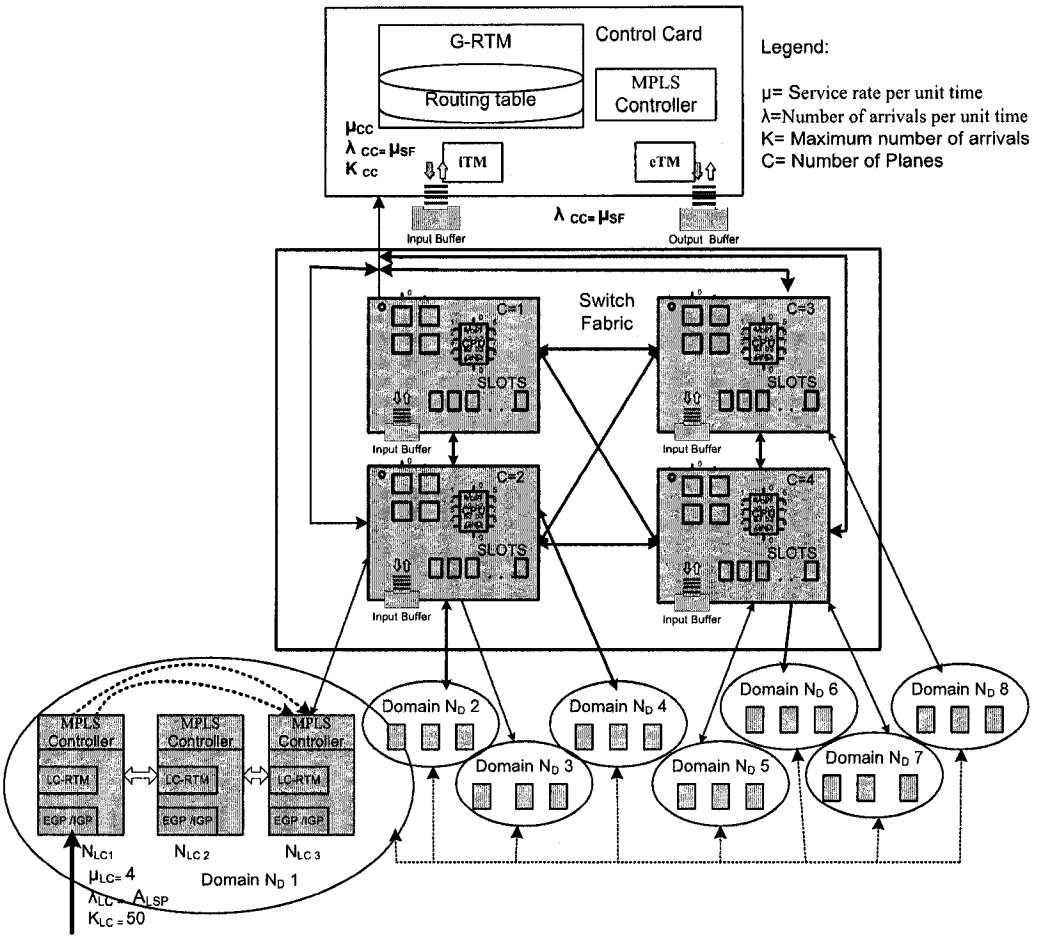

Figure 7.2 Distributed Architecture with Multiple Planes in a Switch Fabric

Using relation (7.13), we calculate the total computation time required by each master LC on a given domain. Therefore the total number of CPU cycles consumed on a given master LC for the path computation is:

$$P \times N_{LC_i} \times N_{Message} \times N_C. \qquad (7.14)$$

For all the Domains in a Router

In the proposed distributed architecture, the total number of RSVP-TE messages received on non- master LCs for all domains is equal to:

$$\sum_{i=1}^{N_D}(N_{LC_i} - 1) \times P \times N_{Message}. \quad (7.15)$$

Using relation (7.15), we deduce that the total number of required CPU cycles to process the RSVP-TE messages on non master LCs is:

$$\sum_{i=1}^{N_D}(N_{LC_i} - 1) \times P \times N_{Message} \times N_{Cmsg}. \quad (7.16)$$

The total number of RSVP-TE messages received on master LCs for all domains are the messages received from the non master LCs as described in Equation 7.16 and the one's received on master LC itself:

$$\underbrace{\sum_{i=1}^{N_D}(N_{LC_i} - 1) \times P \times N_{Message}}_{(7.15)} + N_D \times P \times N_{Message}. \quad (7.17)$$

Using relation (7.17) we deduce that the total number of required CPU cycles to process RSVP-TE messages received on master LCs is:

$$P \times N_{Message} \times N_{Cmsg} \times \left(\sum_{i=1}^{N_D}(N_{LC_i} - 1) \times + N_D\right). \quad (7.18)$$

Using relation (7.18) we deduce that the total number of required CPU cycles to compute RSVP-TE messages is:

$$P \times N_{Message} \times N_C \times \left(\sum_{i=1}^{N_D} N_{LC_i} - 1 \times + N_D\right). \quad (7.19)$$

Updates are sent by the master LC to the CC. Since there are N_D domains, the total number of updates sent is $N_D \times N_{Cmsg}$.

Memory Requirement:

Each port P_i on a given domain i receives $N_{Message}$ possible messages. Therefore, we deduce that the total memory needed to compute the all the RSVP-TE messages is:

$$M \times N_{Message} \times P \times N_{LC_i}. \quad (7.20)$$

The memory required to store RSVP-TE messages received by all domains in a router is:

$$M \times N_{Message} \times P \times \sum_{i=1}^{N_D} N_{LC_i}. \quad (7.21)$$

Route update is sent to G-RTM by master LC in a domain. Therefore, the memory needed for the CC is $N_D \times M$. Using relation (7.20), we deduce that the total memory required is:

$$M \times \left(N_{Message} \times P \times \sum_{i=1}^{N_D} N_{LC_i}. + N_D\right) \quad (7.22)$$

Router configuration for CPU cycles for centralized and distributed architecture is shown in Table 7-1 and memory consumption is shown in Table 7-2.

- The number of interfaces (ports) located on a LC. They are optical interfaces with high capacity (10-40Gbps). In practice, a LC has about 10 ports. In our evaluation, we use this configuration.
- The number of LSPs that can be set up per port is 10.

- Number of LCs the router supports (N_{LC}). The router has higher connectivity by increasing the number of LCs.
- CPU cycles required to process each RSVP-TE messages on the LCs and CC in the centralized architecture.
- CPU cycles required to process RSVP-TE messages on the non master LCs, master LCs and CC.
- Number of domains (N_D).

Parameter		Centralized		Distributed		
N_{LC}	**N_D**	**CC Centralized**	**LC Centralized**	**CC Distributed**	**Non - Master Line Card Distributed**	**Master Line Card Distributed**
16	4	166,400	32	8	600	40,800
32	4	332,800	64	8	1,400	81,600
64	8	665,600	128	16	1,400	81,600
128	32	1,331,200	256	64	600	40,800
256	32	2,662,400	512	64	1,400	81,600
512	64	5,324,800	1,024	128	1,400	81,600
1,024	64	10,649,600	2,048	128	3,000	163,200
2,048	64	21,299,200	4,096	128	6,200	326,400
4,096	128	42,598,400	8,192	256	6,200	326,400
8,192	128	85,196,800	16,384	256	12,600	652,800
16,384	256	170,393,600	32,768	512	12,600	652,800
32,768	256	340,787,200	65,536	512	25,400	1,305,600
65,536	256	681,574,400	131,072	512	51,000	2,611,200

Table 7-1 CPU Resource Consumption

N_{LC}	N_D	Total Memory CC (Centralized) Mbytes	Total Memory LC Distributed/Per Domain
16	4	320,000	80,000
32	4	640,000	160,000
64	8	1,280,000	160,000
128	32	2,560,000	80,000
256	32	5,120,000	160,000
512	64	10,240,000	160,000
1024	64	20,480,000	320,000
2048	64	40,960,000	640,000
4096	128	81,920,000	640,000
8192	128	163,840,000	1,280,000
16384	256	327,680,000	1,280,000
32768	256	655,360,000	2,560,000
65536	256	1,310,720,000	5,120,000

Table 7-2 Memory Consumption for Centralized and Distributed Architecture

The overall performance of the distributed and centralized architectures, expressed in terms of CPU cycles consumed to process the RSVP-TE messages has been shown in Figure 7.3 .For the centralized architecture, there is no RSVP-TE module running on the LCs, so therefore, the CPU cycles required for computation of RSVP-TE messages are consumed primarily on the CC. On the other hand, the CPU cycles required for computation of RSVP-TE messages are consumed primarily on master LCs in the proposed distributed architecture. In other words, the distribution allows the load on the CC to be transferred to the master LCs and the congestions on CC can be

eliminated. Each master LC serves only a small set of LCs, so its capacity can satisfy the current demand.

In the distributed architecture, each domain has master line card for path computation. Therefore, the total number of master LCs in a given router equals to N_D. If there is an increase in the number of domains, the total number of CPU cycles consumed on a given domain is significantly reduced as shown in the Figure 7.2.

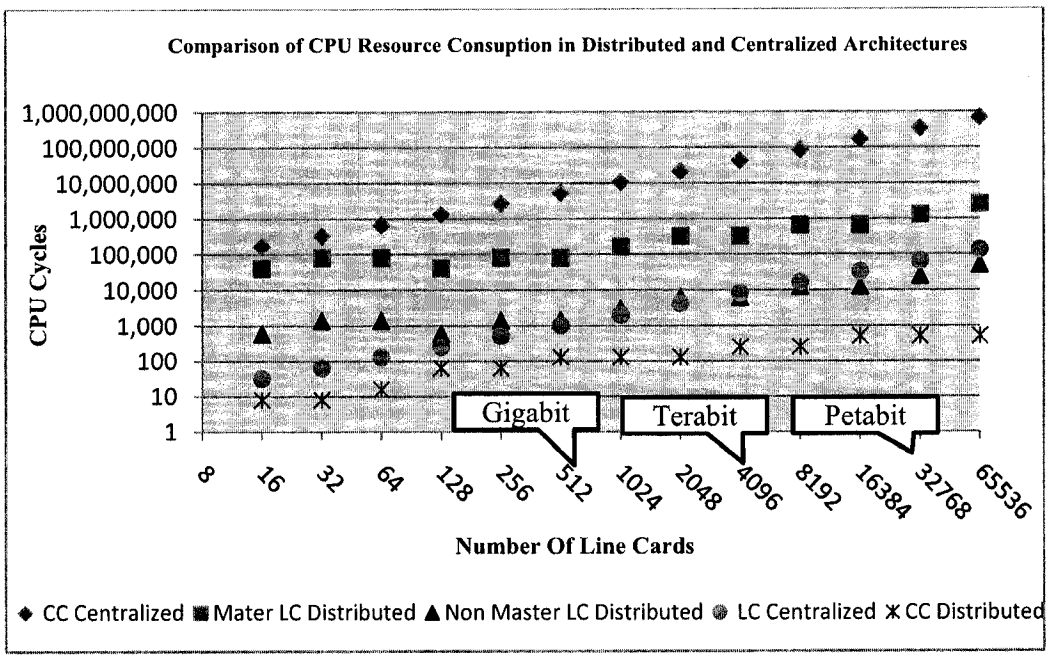

Figure 7.3 CPU Resource Consumption

In the centralized architecture, all the received RSVP-TE messages are processed on the CC, thus it occupies a lot of memory as shown in Figure 7.4. In the proposed distributed architecture, RSVP-TE messages of each domain are kept by a master LC by utilizing the memory available on the LC. We can see that the memory requirement for each line card is much less, especially when the number of line cards increases.

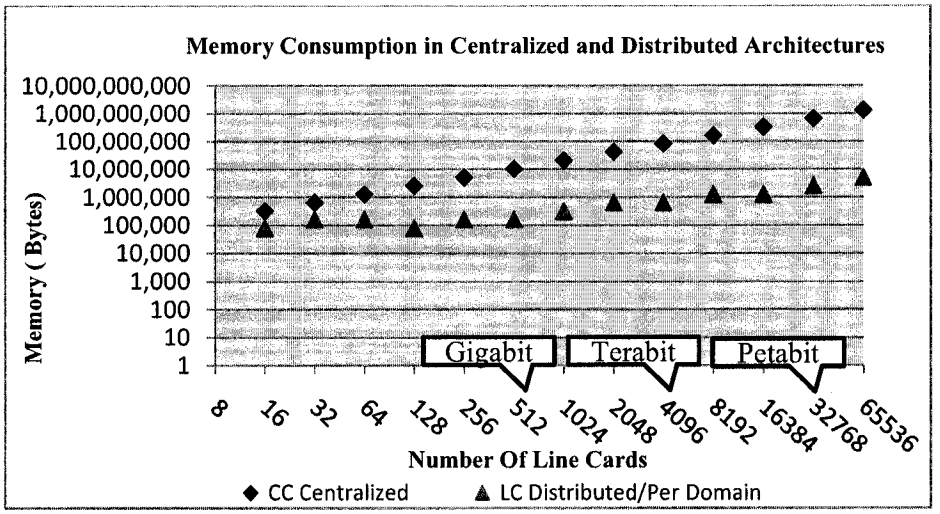

Figure 7.4 Memory Consumption

7.2 Load Balancing with Multi-Plane Switch Fabric

The required input and output bandwidth of the switch fabric exceeds the I/O capabilities of a single plane. To overcome this limitation, the fabric is implemented as a multiple switch planes [29], each plane connects to multiple LCs in a centralized architecture as shown in Figure 7.1 and multiple domains in a distributed architecture as shown in Figure 7.2. To guarantee that packets are evenly loaded in multiple planes in a switch fabric we consider queuing models and Kendall's notation [32] to achieve a distributed load balance on each plane.

A queue is described in shorthand notation by A/B/C/K/N/D.

A: Inter- arrival time distribution.

B: The service time distribution.

C: The number of service channels (or servers).

K: The capacity of the system, or the maximum number of RSVP-TE messages allowed in the system including those in service. When the number is at this maximum, further arrivals are turned away.

Note: This is sometimes denoted C+K where K is the buffer size.

N: The size of calling source. The population size from where the RSVP-TE messages arrive. A small population will significantly affect the effective arrival rate, because, as more jobs queue up, the capacity left for messages to arrive in the queue has reduced.

D: The Service Discipline or Priority orders in the queue that are served:

The codes used to describe these notations are:

- M (Markovian): Poisson process (or random arrival process).
- D (Degenerate distribution): A deterministic or fixed arrival time.
- G (General distribution): Rrefers to independent arrivals

In our model we assume deterministic queuing model D/D/1/K/ Round Robin [31][32]. Let us consider that, at t=0, there are no RSVP-TE messages and the queue is empty.

λ: Number of arrivals per unit time

$1 / \lambda$: Time taken for one message to arrive

μ : Service rate per unit time

$1/ \mu$: Time taken to service one message

Cases: $\lambda > \mu$ or $(1 / \lambda < 1/ \mu)$

If the arrival rate is greater than service rate, then the RSVP-TE messages have to be queued. In this case the length of the queue will increase with the increase in the number of RSVP-TE messages and grow beyond any bound. So we impose balking where the number of RSVP-TE messages in a system gets to a certain point.

Let us keep the limit of the system set to K. If the total number of arrived RSVP-TE messages is greater than the value K, entry is rejected in the queue. In our system we calculate for D/ D/1/K/Round Robin model. The number of RSVP-TE messages in the system at a given time t is n(t). We consider that, as soon as a RSVP-TE message is serviced, another one begins. The number of RSVP-TE messages in a system (including the one in service) at time t =

n (t)= {Number of arrivals in interval $(0, t)$} – {Number of services completed in $(0, t)$}

$$n(t) = \left[\frac{t}{1/\lambda}\right] - \left[\frac{t - 1/\lambda}{1/\mu}\right]$$

n (t) is valid only till the first balk occurs and let t_i be the time until the first balk occurs.

$$n(t_i) = K = [\lambda t_i] - \left[\mu t_i - \mu/\lambda\right]$$

$$t_i = \frac{\{\lambda \times K\} - \mu}{\lambda \times \{\lambda - \mu\}} \tag{7.23}$$

n(t) will never go below (K-2), since we assume $\mu < \lambda$ and an arrival comes before another service is completed.

$$n(t) = \begin{cases} 0 & \left(t < \frac{1}{\lambda}\right) \\ \left\{[\lambda t] - \left[\mu t - \mu/\lambda\right]\right\} & \left(1/\lambda \leq t < t_i\right) \\ K - 1 & (t \geq t_i) \end{cases}$$

We observe that wait times in a queue, until the service begins are W_q^n and W_q^{n+1} .

$$W_q^{n+1} = \begin{cases} W_q^n + S^n - T^n & (W_q^n + S^n - T^n) > 0 \\ 0 & (W_q^n + S^n - T^n) \leq 0 \end{cases}$$

S^n : Service Time

T^n : Inter-arrival time between the nth and (n+1) RSVP-TE messages.

For the first finite difference $\Delta W_q^n = W_q^{n+1} - W_q^n$ or $\Delta W_q^n = S^n - T^n$

$$\Delta W_q^n = \left[\frac{1}{\mu} - \frac{1}{\lambda}\right]$$

A simple differential equation has the solution = $\Delta W_q^n = \left[\left(\frac{1}{\mu} - \frac{1}{\lambda}\right) \times n\right] + C$

To find this constant, we employ boundary conditions that $W_q^1 = 0$ which gives,

$$W_q^n = \begin{cases} \left[\frac{1}{\mu} - \frac{1}{\lambda}\right] \times (n-1) & (n < \lambda t_i) \\ (K-2)\frac{1}{\mu} & (n \geq \lambda t_i) \end{cases} \quad (7.24)$$

Relation (7.24) will be used to calculate the wait time for RSVP-TE messages on LCs, CC and switch fabric. t_i will be calculated by substituting the values of λ, μ, and K in Equation 7.23.

We conduct a comparison of the performance achieved by distributing the load on multiple planes vs. one plane in a switch fabric. The total processing cost and wait time are negligible due to the system configuration in next generation routers. The processing speeds are in petabit and terabit per second. T-Series Juniper Routers allows service providers to increase capacity without adding additional systems to the network [29]. The TX Matrix allows incremental expansion to a

2.5 Tbps system, and the new T1600 provides 1.6 Tbps capacity in a single chassis. Future proof architecture scales comfortably to well beyond this capacity as provider needs progress.

Signaling messages are comparably small when compared to real time traffic. Therefore, the total processing and the wait time on switch fabric are negligible and not considered in this thesis. Though, mathematical model have been presented for wait time and processing time of RSVP-TE messages.

Centralized Architecture

We evaluate the wait time, total processing time on LCs, CC and switch fabric by balancing the load on multiple planes in a switch fabric as shown in Figure 7.1.

Let :

W_{LC}^{n} : Wait time for a RSVP-TE message on a given port in a LC

W_{SF}^{n} : Wait time on Switch fabric

W_{CC}^{n} : Wait time on CC

$t_{Process}$: Processing time for a RSVP-TE message

$t_{Computation}$: Computation time for a RSVP-TE message

The total processing time required for RSVP-TE messages is:

$$\left(W_{LC}^{n} + W_{SF}^{n} + W_{CC}^{n} + t_{Process} + t_{Computation}\right)$$

Wait time on LCs, CC and switch fabric are calculated using Equation (7.24) and substituting values of μ and λ to calculate wait time on LCs, CC and switch fabric

$$W_q^n = \begin{cases} \left[\frac{1}{\mu} - \frac{1}{\lambda}\right] \times (n-1) & (n < \lambda t_i) \\ (K-2)\frac{1}{\mu} & (n \geq \lambda t_i) \end{cases}$$

The value of λ would be different on LCs, CC and switch fabric. According to the values described for λ Table 7-3 in we can calculate the wait time on LCs, CC and switch fabric in centralized and distributed architecture.

Number of Planes (C)	Wait Time (W_q^n)	Load (λ)
C = 1	Wait time on LC	$\lambda_{LC} = A_{LSP}$
	Wait Time on Switch fabric	$\lambda_{SF} = \mu_{LC} \times P \times N_{LC}$
	Wait time on CC	$\lambda_{CC} = \mu_{LC}$
C = Multiple Planes	Wait time on LC	$\lambda_{LC} = A_{LSP}$
	Wait Time on Switch fabric	$\lambda_{SF} = \left[\mu_{LC} \times P \times \left(\frac{N_{LC}}{C}\right)\right]$
	Wait time on CC	$\lambda_{CC} = \mu_{LC}$

Table 7-3 Wait Time Calculations in Centralized Architecture

Distributed Architecture

In distributed architecture we evaluate the wait time and total processing time on LC, CC and switch fabric by balancing the load on multiple planes in a switch fabric as show in Figure 7.2.

W^n_{NMLC} : Wait time for RSVP-TE messages received on non master LC

W^n_{MLC}: Wait time for RSVP-TE messages received on master LC

W^n_{CC} : Wait time on CC

W^n_{SF} : Wait time on Switch fabric

$t_{process}$: Processing time for RSVP-TE messages

$t_{computation}$: Computation time for a RSVP-TE message

Total processing time required for RSVP-TE messages is:

$$\left(W^n_{NMLC} + W^n_{MLC} + W^n_{CC} + W^n_{SF} + t_{process} + t_{computation}\right)$$

Wait time for a RSVP-TE Message received on non master LCs, master line card and switch fabric is calculated by the Equation described in (7.24).

$$W^n_q = \begin{cases} \left[\frac{1}{\mu} - \frac{1}{\lambda}\right] \times (n-1) & (n < \lambda t_i) \\ (K-2)\frac{1}{\mu} & (n \geq \lambda t_i) \end{cases}$$

The value of λ would be different on LCs, CC and switch fabric. According to the values described for λ in Table 7-4 we can calculate the wait time on LCs, CC and switch fabric in centralized and distributed architecture.

Distributed Single Plane Switch Fabric		
Number of Planes (C)	Wait Time (W_q^n)	Load (λ)
C = 1	Wait time on non master LC	$\lambda_{NMLC} = A_{LSP}$
	Wait Time on Switch fabric for RSVP-TE messages received from non master LC	$\lambda_{SF/Plane} = [\mu_{NMLC} \times P \times \sum_{i=1}^{N_D}(N_{LC_i} - 1)]$
	Wait Time on master LC for RSVP-TE messages received from switch fabric	$\lambda_{MLC} = \left(\mu_{NMLC}/_P + A_{LSP}\right)$
	Wait Time on switch fabric for RSVP-TE message computed on master LC and then sent back to the non master LC via switch fabric.	$\lambda_{SF/Plane} = \mu_{NMLC} \times N_D$
	Wait Time on non master LCs for RSVP-TE message sent from master LC through the switch fabric.	$\lambda_{NMLC} = \mu_{NMLC}$
Distributed Multiple Plane Switch Fabric		
C = Multiple Planes	Wait time on non master LC	$\lambda_{NMLC} = A_{LSP}$
	Wait Time on Switch fabric for RSVP-TE messages received from non master LC	$\lambda_{SF/Plane} = \left[\mu_{NMLC} \times P \times \frac{N_D}{C} \times NLCi - 1\right.$
	Wait Time on master LC for RSVP-TE messages received from switch fabric	$\lambda_{MLC} = \left(\frac{\mu_{NMLC}}{P} + A_{LSP}\right)$
	Wait Time on switch fabric for RSVP-TE message computed on master LC and then sent back to the non master LC via switch fabric.	$\lambda_{SF/Plane} = \mu_{NMLC} \times N_D$
	Wait Time on non master LCs for RSVP-TE message sent from master LC through the switch fabric.	$\lambda_{NMLC} = \mu_{NMLC}$

Table 7-4 Wait Time Calculations in Distributed Architecture

The router configuration parameters for the performance evaluation of the centralized and distributed architecture are given below:

- Number of LCs the router supports. The number of interfaces (ports) located on a LC (P).
- Number of RSVP-TE messages arrived per unit time (λ_{LC}= 1500)
- Service time for RSVP-TE message is (μ_{LC} =10Gbps slots, μ_{SF}=1.6 Tbps)

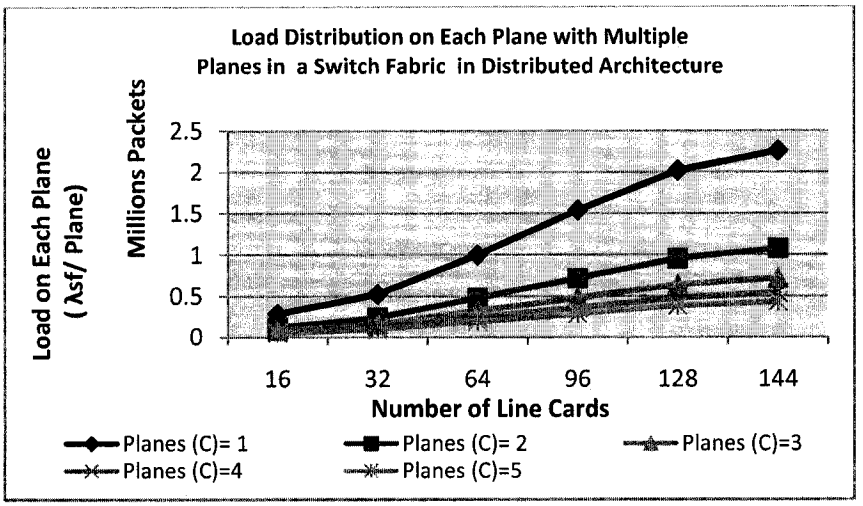

Figure 7.5 Load Distribution in Distributed Architecture

Figure 7.5 shows the effect of load being distributed on each plane in a multi-plane switch fabric. Increase in number of LCs increases the overall signaling messages sent to the switch fabric. For example, in a single plane switch fabric, 2 millions of packets are processed per second when the number of LCs is 128. By increasing the number of planes, load has been distributed and there is a drastic reduction in overall processing time. We need only a small number of planes (C=4) in order that the load on switch fabric on a single plane in centralized architecture is equal to the load on switch fabric with four planes.

7.3 Bandwidth Utilization

Network links cover a large spectrum of speeds. Typically, backbone networks have link speeds ranging from OC-3 (155 Mb/s) to as high as OC-192 (10 Gb/s) [2], and are likely to operate at higher speed in the future, as the next-generation optical equipment is being deployed.

With the signaling or resource reservation protocols, the router along the route makes resource reservation and accepts the connection if there is enough bandwidth. The maximum amount of bandwidth allocated for signaling messages is comparably small in comparison to traffic flows. Due to reallocation of the control components on the LC, RSVP-TE messages are processed faster which in turn increases the signaling messages.

If the utilization of bandwidth is within the allocated percentage, then increase in signaling messages in the distributed architecture does not affect the overall performance of the system. The mathematical model proposed evaluates the bandwidth utilized by the signaling messages in the centralized and distributed architecture.

Let:

C = Capacity of the System

A_{BW} = Allocated Bandwidth

Throughput = No of messaged processed per unit time = μ

$(U(A_{BW}))_C = (U(A_{BW}))_D$ = Percentage of bandwidth utilized from allocated bandwidth is

$$\left[\frac{\mu}{A_{BW}} \times 100\right]$$

Difference in Percentage of Bandwidth utilized is

$$(U(A_{BW})) = (U(A_{BW}))_C - (U(A_{BW}))_D$$

The router configuration parameters for the performance evaluation of the centralized and distributed architecture are given below [29] [38]:

- 1.2 Tb/s of routing capacity.
- 10 Gb/s interfaces.

We assume 25% of bandwidth is allocated for signaling messages in distributed and centralized architecture. We receive 2 million signaling messages at given unit of time. Bandwidth allocated for RSVP-TE messages is 200 Gb/s.

As shown in Figure 7.6, bandwidth utilized for centralized architecture with 144 LCs is 24% and for the distributed architecture, the bandwidth utilized is 24.9%. Though, there is an increase in bandwidth utilization in distributed architecture, but it is within the limit of the bandwidth allocated for signaling messages. By increasing the number of planes in the switch fabric, the bandwidth utilized by the RSVP-TE messages have significantly reduced.

Therefore with increase in signaling messages, we are able to establish more LSP paths at a given time in the distributed architecture without additional bandwidth consumption.

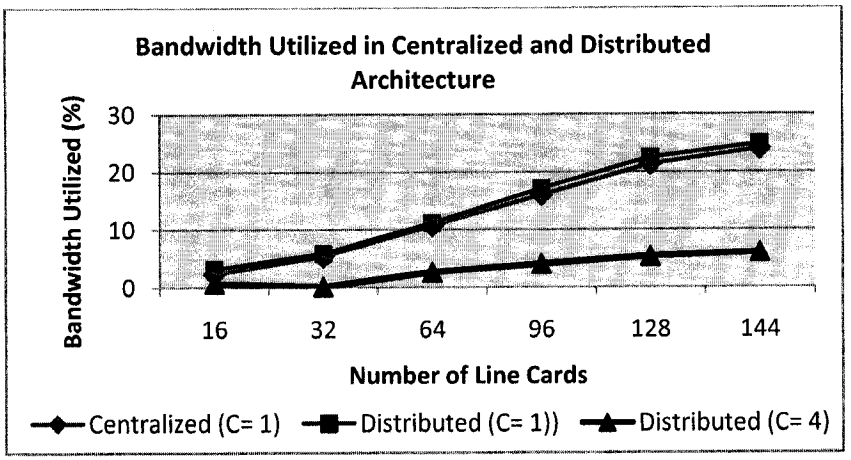

Figure 7.6 Bandwidth Utilization in Centralized and Distributed Architecture

7.4 Advantages of a Distributed/Scalable Architecture

The main benefit of distributed and scalable architecture is the ability of fully exploiting the next generation routers where LCs are provided with additional memory, computing resources and the switching capacity is enhanced to reach throughput of terabit or even petabit. Resiliency is significantly improved because message processing is achieved at the LC level. Failures acknowledged at the CC can be recovered transparently. The load on the CC is reduced by distributing the workload; hence there is increase in scalability.

Reallocating some of the processing tasks from CC to the LCs can reduce potential bottlenecks experienced on the CC when the number of requests is increased due to the increase in number of LCs and routes supported by the core router. In addition, the architecture we propose has the following advantages:

- **Performance:** Parallel processing is available in our proposed distributed model and wait time on queues are eliminated. Line cards can independently process the routes, without waiting for a reply from the control card.

- **Robustness:** LCs independently process the LSPs, without waiting for the reply from the CC in comparison with the centralized architecture. The LIB tables are optimized to contain only the LSPs and the neighbors directly related to the LC. Thus, the processing time for route lookup is significantly reduced.
- **Scalability:** Reallocation of control tasks, predominantly the RSVP-TE signaling messages facilitates a highly scalable next generation router. Processing speeds of petabit and terabit per second is achieved by increasing the number of line cards. Due to the increase in the number of line cards, there is an increase in number of signaling messages. If the bandwidth utilization of signaling messages is within the allocated bandwidth then the overall performance of the system is not affected and there is a significant improvement in throughput.
- **High Availability:** Route information and link state databases have a back up on LCs; which provides a high redundancy level for RSVP-TE module. Also, problems arising on the CC will not slow down the processes running on the LCs.
- **Resiliency:** Migration of signaling protocols to the LCs keeps the current session alive while there is a failure acknowledged on the CC. If the CC is required to perform and process all control tasks, system will totally shutdown when a failure acknowledged on CC. Maintaining a backup for CC is extremely an expensive solution. The fully distributed architecture proposed for RSVP-TE signaling messages is able to self restore without the help of the CC. Indeed, it is faster to recover from LC failures.

In addition, parallel processing and load balancing of signaling messages is achieved by introducing a multiple plane switch fabric.

The following table presents a qualitative comparison between the centralized and architecture for distributed RSVP-TE architecture:

Parameter	Centralized Architecture	Distributed Architecture
Path computation performance	Poor	Good
Message Processing	Some Delay	Fast
Memory consumption (Control Card)	High	Low
Memory consumption (Line Card)	Low	High
Route Lookup	Some Delay	Fast
Route advertisement speed	Some delay	Fast

Table 7-5: Qualitative Comparison between Centralized and Distributed RSVP-TE Architectures

With a distributed MPLS/RSVP-TE architecture, our Master thesis has covered the distribution of principal components: RSVP-TE signaling protocols message processing, route management, synchronization mechanism and MPLS. The methodology to design distributed software architecture has been presented, that can be applied for other protocol modules such as OSPF or BGP that may be considered in our future research and discussed in the next chapter.

Chapter 8

8 Conclusion and Recommendations

8.1 Conclusion

In this thesis, we have explored centralized and distributed architectures for MPLS/RSVP-TE targeting the next generation router platform. The main issue is to design routers with high scalability, resiliency and robustness. The main requirements that we have identified are related to scalability, flexibility, and availability of IP routers. To address these requirements, our approach is to exploit the new hardware capacity of the next generation routers in order to distribute control functions on all routers components.

The main benefit of this approach is the ability of fully exploiting the distributed hardware architecture of the next generation routers where LCs are provided with additional memory and computing resources and the switching capacity is enhanced to reach petabit per second. Resiliency is significantly improved because message processing is achieved at the LCs level so that failures of the CC can be recovered transparently. The load on the CC is reduced, so the scalability of the router is increased.

We investigated the specific functions of the router control plane, namely routing, signaling and routing table management, with the aim of redesigning them in a distributed way. Start from an overall distributed architecture for the control plane, we developed the distributed

implementations for MPLS/RSVP-TE with distributed RTM. Our approach is to review the current centralized architecture, point out the disadvantages, and then propose the new architecture regarding the new hardware features.

Performance evaluations are also performed in order to compare the proposed architectures to the old ones. For the next generation MPLS/RSVP-TE architectures, the most important goal is task sharing among LCs, and scalability. For RTM, we focus on the distributed computing of best routes, the storage, and the scalability of domains.

8.2 Future Work

Some of the areas for future work of the presented contributions are as follows.

- Propose distributed architecture for other protocols, such as OSPF or BGP. This thesis presented the distributed architectures for MPLS/RSVP-TE and RTM. OSPF and BGP are also two important protocols which core routers need to support. We may apply the general distributed architecture for routing protocol regarding the specific functions of OSPF or BGP.

- Propose and formalize the application programming interface (API), used between the control and interface instances running within a distributed and scalable architecture to avoid any dependency on the actual implementation, and potentially propose these APIs to organizations responsible for defining standards (IETF), Network Processor Forum (NPF), Service Availability (SA) Forum, etc.).

- Investigating the algorithms to group the LCs into domains and to assign the proxy or master LC in an optimal manner, so that the load on the LCs can be balanced.

- Ensure the defined architecture and actual implementation still allows in-service upgrades of the RTM module without interruption of service in real-time of the LC instances.

- Prove all concepts described in the preceding points by implementing all necessary modifications in the different modules. This aspect covers the actual description of the detailed design, definition of a test plan, test results summary, with tests performed against precise performance objectives under various stress conditions (routes flapping, high bandwidth traffic, etc.).

References

[1] Hidell, Markus, "Decentralized Modular Router Architecture," Doctoral Thesis, KTH Electrical Engineering, Sweden, 2004.

[2] Chao, HJ., "Next Generation Router," IEEE Communication Magazine, Vol. 90, No. 9, pp. 1518-1558, September 2002.

[3] "Cisco 12000 Series Internet Router Architecture," (Jun. 2007). http://www.cisco.com

[4] G. Huston, Next Steps for IP QoS Architecture, RFC2990, November 2000.

[5] Bu, T., Gao, L., Towsley, D., "On Characterizing BGP Routing Table Growth," Computer Networks, Vol. 45, Iss. 1, pages 45-54, 2004.

[6] Moy, J., "OSPF Version 2," RFC 2328, IETF - Network Working Group, April 1998.

[7] ISO, "Intermediate System to Intermediate System Routing Information Exchange Protocol for use in Conjunction with the Protocol for Providing the Connectionless-mode Network Service," ISO 8473, ISO/IEC 10589:2002, Second Edition.

[8] E. Rosen, A. Viswanathan, R. Callon, "Multiprotocol Label Switching Architecture", RFC3031, January 2001.

[9] R. Braden, Ed., L. Zhang, S. Berson, S. Herzog, S. Jamin, Resource Reservation Protocol (RSVP) – Version 1 Functional Specification, RFC2205, September 1997.

[10] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow, RSVP-TE: Extensions to RSVP for LSP Tunnels, RFC3209, December 2001.

[11] Adrien Farrel, Hyperchip, Protocol specification for RSVP-TE, February 2004.

[12] Juniper Networks Inc., "Juniper Networks Next Generation Network Core Solution for Service Providers," Solution Brochure, June 2007.

[13] Zini, A., "Cisco IP Routing," pp. 80-111, Addition-Wesley, 2002.

[14] Malkin, G., "RIP Version 2," RFC 2453, IETF - Network Working Group, November 1998.

[15] Rekhter, Y., Rosen, E., "Carrying Label Information in BGP-4," RFC3107, IETF – Network Working Group, May 2001.

[16] Medhi, D., Ramasamy, K., "Network Routing – Algorithms, Protocols, and Architectures," Morgan Kaufmann, Elsevier Edition, 2007.

[17] Kumar, VP, Lakshman, TV., Stiliadis, D., "Beyond best effort: router architectures for the differentiated services of tomorrow's Internet," IEEE Communications Mag., Vol. 36, pp. 152–164, 5/1998.

[18] Avici Systems Inc., "The Avici TSR®," 2006.

http://www.avici.com/products/tsr.shtml (Last access: 31/08/2007)

[19] Hyperchip Inc., "PBR-1280 Release 1.0 Performance Characterization System Engineering," Technical Report, 2004.

[20] Kaplan, H., "Non-Stop Routing Technology", White Paper, Avici Systems Inc., 2002.

[21] Aweya, J., "IP router architectures: an overview," International Journal of Communication Systems, vol. 14, part 5, pages 447-476, 2001.

[22] Marcela De Maria, Hyperchip, PBR1280 Chipset Architecture Release 0.3. December 19, 2001.

[23] Fan Lo-Jun , "Scalable Distributed Architecture of the Terabit router", Asian Green Electronics, AGEC, IEEE Conference, 2004.

[24] Duplaix, Jérôme, "Routing Software Architecture in the R1.0 PBR-1280 Router," Internal document, Hyperchip Inc., Oct. 2005.

[25] Nguyen, K.-K., Mahkoum, H., Jaumard, B., Assi, C., Lanoue, M., "Towards a Distributed Control Plane Architecture for Next Generation Routers," Universal Multiservice Networks, ECUMN, France, 2007.

[26] Andersson, L., Doolan, P., Feldman, N., Fredette, A., Thomas, B., "LDP Specification," RFC3036, IETF - Network Working Group, Jan. 2001.

[27] Deval, M., Khosravi, H., Muralidhar, R., Ahmed, S., Bakshi, S., Yavatkar, R., "Distributed Control Plane Architecture for Network Elements", Intel Technology Journal, vol. 7, iss. 4, Nov. 2003.

[28] Chuck Semeria, "T-series Routing Platform: System and Packet Forwarding Architecture", White Paper, Juniper Networks.

[29] "Juniper Networks M-series and J-series Routers", White Paper, Juniper Networks.

[30] "Virata "Performance Optimized MPLS™ Programmer's, Manual", Hyperchip.

[31] Dornald Gross, Carl M. Harris, "Fundamentals of Queuing Theory", January 1985.

[32] Leonard Kleinrock, "Queuing Systems", Volume I: Theory, March 1974.

[33] F. Baker, "Requirements for IP version 4 Routers", IETF Internet RFC 1812, June 1995.

[34] Doria, A., Haas, R., Salim, J.H., Khosravi, H., Wang, W. M., "ForCES Protocol Specification", IETF Draft, Work in Progress, IETF - Network Working Group, July 2007.

[35] Markus Hidell, Peter Sjodin, Olof Hagsand, "Reliable Multicast for Control in Distributed Routers ", High Performance Switching and Routing, HPSR, 2005.

[36] Markus Hidell, Peter Sjodin, Olof Hagsand, "Design and Implementation of a Distributed Router", Signal Processing and Information Technology, IEEE Symposium, 2005.

[37] "Migrating to Ethernet and MPLS: The Cisco Advantage", Cisco 12000 Series Routers, White Paper, Cisco Systems Inc, 1992-2007.

[38] "Cisco XR 12000 Series Router", White Paper, Cisco Systems Inc, 1992-2007.

[39] Henry C. B. Chan , Hussein M. Alnuweiri, Victor C. M. Leung, "Cost and Performance Optimization in IP Switched-Routers", Computers and Signal Processing, IEEE Pacific Rim Conference, 1999.

[40] Tang Kai, Xu Xin, Shao JunLi, "A New Software Architecture in Designing Multiprotocol Router", International Federation for Information Processing, IFIP, 2000.

[41] Nexabit, "The New Network Infrastructure : The Need for Terabit Switch/Routers, " 1999, 11 pages, http://www.nexabit.com/need.html

[42] Nexabit, "NX64000 Multi-Terabit Switch/Router Product Description," 1999, 18 pages, http://www.nexabit.com/proddescr.html.

[43] Nick McKeown, "A Fast Switched Backplane for a Gigabit Switched Router", Business Communications Review, Dec 1997, Vol. 27, No. 12, http://www.bcr.com/bcrmag/12/mckeown.htm

[44] Amit Singhal , Raj Jain, "Terabit Switches and Routers", Elsevier Science B.V., 2002.

[45] KimKhoa Nguyen, "Enabling Architectures for Quality of Service Provisioning", Phd thesis in progress, Concordia University.