

2002

# Performance analysis of voice traffic in MPLS communication networks

Choo Chin Tan  
*San Jose State University*

Follow this and additional works at: [https://scholarworks.sjsu.edu/etd\\_theses](https://scholarworks.sjsu.edu/etd_theses)

---

## Recommended Citation

Tan, Choo Chin, "Performance analysis of voice traffic in MPLS communication networks" (2002). *Master's Theses*. 2378.  
DOI: <https://doi.org/10.31979/etd.3vph-jt7z>  
[https://scholarworks.sjsu.edu/etd\\_theses/2378](https://scholarworks.sjsu.edu/etd_theses/2378)

This Thesis is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Theses by an authorized administrator of SJSU ScholarWorks. For more information, please contact [scholarworks@sjsu.edu](mailto:scholarworks@sjsu.edu).

## **INFORMATION TO USERS**

**This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.**

**The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.**

**In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.**

**Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.**

**ProQuest Information and Learning  
300 North Zeeb Road, Ann Arbor, Mi 48106-1346 USA  
800-521-0600**

**UMI<sup>®</sup>**



**PERFORMANCE ANALYSIS OF VOICE TRAFFIC IN  
MPLS COMMUNICATION NETWORKS**

**A Thesis**

**Presented to**

**The Faculty of the Department of Electrical Engineering**

**San Jose State University**

**In partial fulfillment**

**of the Requirements for the Degree**

**Master of Science**

**by**

**Choo Chin Tan**

**December 2002**

**UMI Number: 1411631**

**UMI<sup>®</sup>**

---

**UMI Microform 1411631**

**Copyright 2003 by ProQuest Information and Learning Company.  
All rights reserved. This microform edition is protected against  
unauthorized copying under Title 17, United States Code.**

---

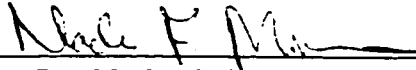
**ProQuest Information and Learning Company  
300 North Zeeb Road  
P.O. Box 1346  
Ann Arbor, MI 48106-1346**

© 2002

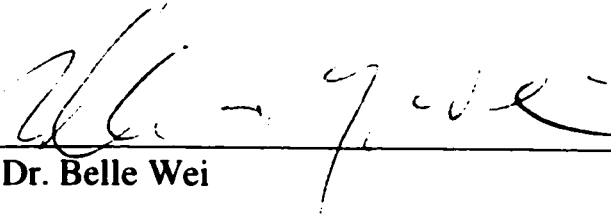
**Choo Chin Tan**

**ALL RIGHTS RESERVED**

**APPROVED FOR THE DEPARTMENT OF ELECTRICAL ENGINEERING**



Dr. Nader Mir

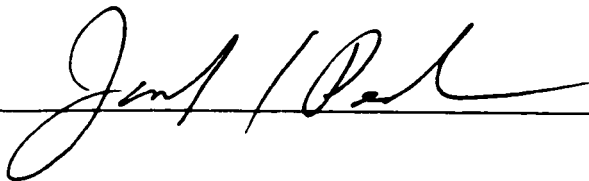


Dr. Belle Wei



Dr. Essam Marouf

**APPROVED FOR THE UNIVERSITY**



## **ABSTRACT**

### **PERFORMANCE ANALYSIS OF VOICE TRAFFIC IN MPLS COMMUNICATION NETWORKS**

by Choo Chin Tan

The performance of *real-time* transmission such as voice over data networks is not measured by throughput, but by the delay time that the network imposes on the voice packets. Multiprotocol Label Switching (MPLS) is the latest technology introduced to handle the convergence of voice and data networks. This thesis presents a detailed model analysis and simulation of MPLS for handling voice packets. From the models presented in this thesis, it can be observed how an MPLS switch segregates voice packets from data packets to achieve prioritized queuing. This technique would potentially minimize the delay time for voice packets. A mathematical representation of the MPLS queuing model is derived based on the stochastic Markov Chain theorem using models with non-preemptive priority queuing discipline. The queuing model is further implemented using OPNET simulation software. Performance metrics for voice packet transmission in an MPLS network are measured and analyzed under different traffic parameters.



# TABLE OF CONTENTS

<i>LIST OF FIGURES</i> .....	<i>viii</i>
<i>LIST OF TABLES</i> .....	<i>x</i>
<i>Chapter 1 Introduction</i> .....	<i>1</i>
1.1 Differentiated Service (DiffServ) .....	2
1.2 Convergence of Voice and Data .....	3
1.3 Multiprotocol Label Switching (MPLS).....	4
1.4 Organization of Thesis.....	5
<i>Chapter 2 Voice over Internet Protocol (VoIP)</i> .....	<i>6</i>
2.1 Voice Digitization.....	6
2.2 Voice Coder-Decoder (Codec) .....	7
2.2.1 Waveform Codecs.....	8
2.2.2 Source Codecs.....	8
2.3 Echo Cancellation .....	11
2.4 Voice Transportation Layers.....	12
2.5 Jitter Buffer .....	14
2.6 Audio Silence Detection .....	15
2.7 Loss Packet Compensation .....	15
2.8 Per-Call Bandwidth.....	16
2.9 VoIP Delay/Latency.....	18
2.9.1 Gateway-Incurred Delay.....	18
2.9.2 Network-Incurred Delay.....	20
<i>Chapter 3 Multiprotocol Label Switching (MPLS)</i> .....	<i>23</i>
3.1 Architecture of MPLS.....	24
3.1.1 Forwarding Module .....	25
3.1.2 Label Information Base (LIB) .....	27
3.1.3 Control Module.....	28
3.2 Benefits of MPLS .....	30
3.2.1 Simple Packet Forwarding Paradigm.....	30
3.2.2 Interoperability.....	30
3.2.3 Routing Hierarchy and Label Stacking.....	31
3.2.4 Traffic Classification .....	32
3.2.5 Quick Fault Recovery .....	32
3.2.6 Traffic Engineering Capabilities.....	33
3.3 MPLS versus ATM.....	34

3.4	Differentiated Services (DiffServ) .....	36
3.4.1	Per-Hop Behavior (PHB) .....	37
3.4.2	End-to-End Quality of Service (QoS) .....	41
3.5	MPLS and Differentiated Services (DiffServ) .....	44
3.6	MPLS-based Virtual Private Network (VPN) .....	46
3.6.1	Introduction.....	46
3.6.2	Overlay VPN Model .....	48
3.6.3	Peer-to-peer VPN Model .....	49
3.6.4	MPLS-based VPN.....	50
3.6.5	Benefits of MPLS-based VPN.....	55
 <i>Chapter 4 Queuing and Scheduling Analysis</i> .....		 57
4.1	Guaranteed Service Model.....	58
4.2	Scheduling Algorithms .....	59
4.2.1	First In First Out (FIFO) .....	60
4.2.2	Priority Queuing (PQ) .....	61
4.2.3	Weighted Fair Queuing (WFQ) .....	62
4.2.4	Earliest Deadline First (EDF) .....	65
4.2.5	Deficit Round Robin.....	65
4.2.6	Non-work Conserving Service Discipline .....	67
4.3	Buffer Management Schemes.....	68
4.3.1	Random Early Detection (RED) .....	69
4.3.2	Multi-Level Random Early Discard (MRED) .....	70
4.4	The MPLS Queuing Model.....	73
4.4.1	Packet Fragmentation.....	74
4.4.2	Differentiated Service (DiffServ) Module.....	75
4.4.3	Traffic Management.....	75
4.4.4	Priority Queuing with Traffic Policer .....	76
4.4.5	Class-Based Weighted Fair Queuing with MRED .....	77
4.5	Queuing Model Analysis .....	78
4.5.1	Notations and Assumptions .....	79
4.5.2	Performance Analysis of the Queuing Model.....	81
4.6	Theoretical Results.....	92
 <i>Chapter 5 OPNET Simulation Results</i> .....		 93
5.1	Modeling of Voice Sources .....	93
5.1.1	OPNET Voice Model.....	95
5.1.2	Notations and Abbreviations.....	96
5.1.3	ON-OFF Voice Model .....	197
5.2	Node Model Simulation.....	100
5.2.1	Analysis of EF Traffic with Different BE Traffic Parameters.....	100
5.2.2	Analysis of EF Traffic with Different Cross Traffic Parameters.....	122

5.3	Network Model Simulation.....	132
5.3.1	Analysis of Voice Traffic in an MPLS Network.....	134
5.3.2	MPLS Traffic Engineering Capabilities .....	141
<i>Chapter 6</i>	<i>Conclusion .....</i>	<i>153</i>
<i>Chapter 7</i>	<i>Future Work.....</i>	<i>155</i>
<i>Appendix A</i>	<i>The Implementation of Intranet and Extranet: A Case Study.....</i>	<i>156</i>
<i>REFERENCES</i>	<i>.....</i>	<i>159</i>

## LIST OF FIGURES

Figure 2.1: The Transmitter and Receiver in an IP Telephony System .....	7
Figure 2.2: A General Model of CELP Encoder.....	9
Figure 2.3: Normal VoIP Packet and Compressed RTP Header .....	13
Figure 3.1: The Control and Forwarding Component of an MPLS Router .....	25
Figure 3.2: MPLS Header .....	26
Figure 3.3: Elements in an MPLS Network.....	27
Figure 3.4: Interoperability of MPLS over Multiple Layer 2 Switching Technologies ...	31
Figure 3.5: Routing Hierarchy in an MPLS Network.....	32
Figure 3.6: Traffic Engineering Capability of an MPLS Network .....	34
Figure 3.7: IPv4 Type of Service versus Differentiated Service Byte.....	38
Figure 3.8: The DiffServ Boundary and Core Routers .....	41
Figure 3.9: RFC 2547bis BGP/MPLS-based VPN .....	50
Figure 3.3.10: Route Distinguisher of a VPN Address.....	52
Figure 3.11: 2-level Label Stack .....	54
Figure 3.12: MPLS-based Layer2 VPN .....	54
Figure 4.1: The Drop Probability of a RED.....	70
Figure 4.2: Block Diagram of a Traffic Conditioner .....	71
Figure 4.3: The parameter settings for MRED .....	72
Figure 4.4: MPLS Queuing Model Supporting Differentiated Services.....	73
Figure 4.5: The Result of Large Packet Fragmentation .....	74
Figure 4.6 The Token Bucket Scheme.....	76
Figure 4.7: Two-level Priority Queuing Model for Voice Performance Analysis .....	79
Figure 4.8: The Markov Chain for an M/M/1 Queuing System .....	81
Figure 4.9: Block Diagram of the M/M/1/K Queuing System .....	82
Figure 4.10: The Markov Chain for an M/M/1/K Queuing System .....	82
Figure 4.11: Block Diagram of the M/M/c/K Queuing System.....	85
Figure 4.12: The Markov Chain for an M/Mc/K Queuing System.....	86
Figure 4.13: Non-preemptive Priority Queuing Model .....	90
Figure 5.1: ON-OFF Voice Model.....	94
Figure 5.2: The Look-ahead Delay of the G.723.1 Codec.....	94
Figure 5.3: Process Model of the ON-OFF Voice Model.....	96
Figure 5.4: Time-Averaged Throughput (packets/sec) for the ON-OFF Voice Model....	98
Figure 5.5: Time-Averaged Throughput (bits/sec) for the ON-OFF Voice Model .....	99
Figure 5.6: Node Model for Simulation.....	100
Figure 5.7: Two-level Priority Queuing Model .....	101
Figure 5.8: Queuing Delay for EF Packets with Different BE Packet Sizes .....	105
Figure 5.9: Queuing Delay for EF Packets with Different BE Loads .....	109
Figure 5.10: Queuing Delay for EF Packets with Different Link Speeds .....	113
Figure 5.11: Queuing Delay for EF Packets with Different EF Loads.....	116
Figure 5.12: Packet Formats for Schemes A, B and C .....	119
Figure 5.13: Queuing Delay for EF Packets with Different EF Packet Sizes.....	121

Figure 5.14: Linear Multi-hop Network Topology for Simulation.....	123
Figure 5.15: OPNET Model of Tagged and Cross Traffics.....	124
Figure 5.16: Queuing Delay for Tagged Traffic with Different Cross Traffics .....	125
Figure 5.17: Distortion Level (Jitter) of the Tagged Traffic Stream .....	126
Figure 5.18: Queuing Delay for Tagged Traffic with Different Number of Nodes .....	127
Figure 5.19: Distortion Level (Jitter) of the Tagged Traffic Stream .....	128
Figure 5.20: Queuing Delay for Tagged Traffic with Different Traffic Profiles.....	130
Figure 5.21: Distortion Level (Jitter) of the Tagged Traffic Stream .....	131
Figure 5.22: The MPLS Network Topology for OPNET Simulation.....	132
Figure 5.23: Queuing Delay for Voice Traffic with Different Number of Voice Calls .	135
Figure 5.24: Throughput for Overall Voice Traffic with Different Number of Calls ....	136
Figure 5.25: Queuing Delay for Voice Traffic with Different Frames per Packet .....	137
Figure 5.26: Throughput for Overall Voice Traffic with Different Frames per Packet .	138
Figure 5.27: Queuing Delay for Voice Traffic with Different Coding Schemes.....	140
Figure 5.28: Throughput for Overall Voice Traffic with Different Coding Schemes ....	140
Figure 5.29: The MPLS Network Topology for OPNET Simulation.....	142
Figure 5.30: Throughput for TCP and UDP Flow .....	143
Figure 5.31: Throughput for the Two Different Paths in the Network .....	143
Figure 5.32: The MPLS Network Topology for OPNET Simulation.....	144
Figure 5.33: Throughput for TCP and UDP Flow .....	145
Figure 5.34: Throughput for the Two Different Paths in the Network.....	146
Figure 5.35: The MPLS Network Topology for OPNET Simulation.....	147
Figure 5.36: Throughput for the LSRs along Different Paths in the Network .....	149
Figure 5.37: Throughput for the Two Separate LSPs in the Network .....	149
Figure 5.38: Throughput for the LSRs along Different Paths in the Network .....	151
Figure 5.39: Throughput for the Three Separate LSPs in the Network.....	151

## LIST OF TABLES

Table 2.1: Summary of ITU Standardized Codecs .....	11
Table 2.2: Bandwidth Savings Using Various VoIP Packet Optimization Techniques ...	17
Table 2.3: Overall End-to-End Delay Calculation.....	22
Table 3.1: Original DiffServ AF Codepoints.....	39
Table 3.2: Characteristics of DiffServ Per-Hop Behaviors (PHB).....	40
Table 3.3: QoS Mapping Between DiffServ and MPLS.....	45
Table 5.1: Simulation Parameters of G.723.1 Voice Model.....	95
Table 5.2: Test Parameters for the BE Packet Size Test.....	102
Table 5.3: Theoretical Queuing Delay Calculations for EF and BE Traffic Flows.....	104
Table 5.4: Test Parameters for the BE Load Test.....	106
Table 5.5: Theoretical Queuing Delay Calculations for EF and BE Traffic Flows.....	108
Table 5.6: Test Parameters for the Link Speed Test.....	110
Table 5.7: Theoretical Queuing Delay Calculations for EF and BE Traffic Flows.....	112
Table 5.8: Test Parameters for the Link Speed Test.....	114
Table 5.9: Theoretical Queuing Delay Calculations for EF and BE Traffic Flows.....	115
Table 5.10: Test Parameters for the Link Speed Test.....	117
Table 5.11: Scheme C Packet Size .....	119
Table 5.12: Theoretical Queuing Delay Calculations for EF and BE Traffic Flows.....	120
Table 5.13: Test Parameters for the Different Number of Cross Traffics .....	124
Table 5.14: Test Parameters for the Different Number of Intermediate Nodes.....	127
Table 5.15: Test Parameters for the Different Traffic Profiles.....	129
Table 5.16: Test Parameters for the Different Number of Voice Calls .....	134
Table 5.17: Test Parameters for the Different Frames per Voice Packet .....	136
Table 5.18: Test Parameters for the Different Voice Coding Scheme .....	139

## **Chapter 1 Introduction**

The Internet was first designed to perform fast and efficient datagram forwarding for military and research institutions. It has had such great success that over the years, the Internet has evolved rapidly and become one of the most prominent carriers of information in our society. Besides traditional electronic mail and file transfer, the Internet is being used by a multitude of multimedia applications with very different network requirements such as Internet Protocol (IP) telephony and video conferencing. There is a remarkable demand for the Internet to support Quality of Service (QoS) for different traffic classes, as opposed to the single best-effort level of service provided by today's Internet. In order to extend the network beyond its current capabilities, it has to scale in terms of bandwidth, routing and customer service provisioning.

The Internet is a combination of networks that are interconnected together by a backbone network. To accommodate the ever-increasing traffic load in the network, Internet Service Providers (ISP) such as Global Crossing and Qwest have deployed fiber optics transmission lines, also known as optical carriers (OC) in their backbone networks. The OC-48 and OC-192 links can provide up to 2.5 gigabits per second (Gbps) and 10 Gbps of bandwidth respectively. As the Internet grows and spans across the globe, the performance of today's routing algorithm deteriorate due to the increasing size of the routing table. Extremely efficient software data structures and search algorithms are needed to perform efficient IP routing for high-speed networks. The solution to this problem is to integrate the scalability of connectionless IP with the performance of

connection-oriented networks, such as Asynchronous Transfer Mode (ATM) [1]. However, the overlay model of IP-over-ATM has introduced its own problems. Thus, a better protocol is needed to solve this issue.

### ***1.1 Differentiated Service (DiffServ)***

In recent years, ISPs have spent millions of dollars to build the Internet backbone with advance networking gears to provide the required bandwidth for the Internet users. However, today's Internet is free of charge, and is available to everyone. Average home users are satisfied with the best-effort service and are not willing to pay in order to have access to the Internet. As a result, the ISPs have to come up with a business model that can generate revenues by providing premium services to the corporate users, offering services such as Virtual Private Networks (VPN) and video conferencing. However, the lack of QoS in today's Internet prevented them to provide efficient customer service provisioning. Although ATM has provided the ISPs some traffic engineering capabilities to provide different level of service guarantees, but there are issues in ATM that make it not-so-scalable for wide area networks.

Two major QoS architecture have been proposed for the Internet. The Integrated Services [2] with Resource Reservation Protocol (RSVP) [3] as the signaling protocol was designed to offer end-to-end per-flow delay guarantees. However, this requires per-flow reservation state information in every node, causing scalability issues in its deployment for large networks. This led to the development of the Differentiated Services (DiffServ) architecture [4], which classifies packets into different service classes



based on the Differentiated Services Code Point (DSCP) in the IP header. The aggregation of flows that belong to the same traffic class will have the same forwarding treatment at each DiffServ-enabled router.

## **1.2 Convergence of Voice and Data**

According to a report from the RHK research group in 1998, the data traffic in the networks is going to grow exponentially into the year of 2002, compared to the relatively flat growth of the voice traffic. Data traffic over the Internet has already surpassed the voice traffic carried over the legacy circuit-switching network in some of the U.S. service provider networks [5]. The convergence of voice and data networks is imminent, and this has led to the introduction of IP telephony, better known as Voice over IP (VoIP). In order to address the scalability issues in the Internet and to support the convergence of voice and data networks, the Internet Engineering Task Force (IETF) has come up with a new standard – Multiprotocol Label Switching (MPLS). The architecture of MPLS specifications can be found in *RFC 3031* [6].

The performance of voice transmission over a data-packet network is not measured by throughput, but by the latency that the network imposes on the voice packets for switching and routing. This is because real-time speech conversations are delay and jitter sensitive. Once the one-way delay exceeds 250 milliseconds, the conversation parties are unable to tell whether the other party has finished speaking, and the situation will lead to both parties talking at the same time. The International Telecommunication Union's G.114 recommendation, which comprises the guidelines for voice transmission time

limits, specifies a one-way delay of 150 milliseconds. The overall end-to-end delay of an IP telephony system can be divided into two categories: gateway-incurred delay which consists of fixed delays such as speech coding delay and decompression delay, and network-incurred delay which consists of variable delays such as network congestion delay and queue scheduling delay.

### **1.3 Multiprotocol Label Switching (MPLS)**

MPLS is a label switching technique where packets are assigned a label based on their traffic classes as they enter an MPLS network through the Ingress Label Switch Router (inbound edge router). All subsequent packets forwarding at the core Label Switch Router (LSR) are based on that fixed-length label. Since the introduction of MPLS, it has caught the attention from a lot of researchers and industry experts [7] [8] [9] [10] [11] [12]. One of the most important benefits of MPLS is replacing the destination-based hop-by-hop forwarding paradigm of today's Internet with a label-swapping forwarding paradigm. This will solve the routing scalability issue by removing the use of routing tables in the core LSR.

Some of the features incorporated in the MPLS protocol are aimed at providing the tools to offer effective service differentiation over the Internet. The introduction of Forward Equivalency Class (FEC) to provide QoS in the Internet and the strong traffic engineering capabilities of MPLS has not only made transmission of real-time information possible in the Internet, but to perform it effectively. There are many researches on the subject of traffic engineering capabilities of MPLS [13] [14] [15] [16]

[17] [18] [19]. This is an important concept since it will solve the customer service provisioning dilemma faced by today's ISPs. The requirements for traffic engineering over MPLS can be found in [20].

In addition to the ability of providing Layer 2 traffic-engineering capabilities similar to ATM networks, MPLS supports various Layer 3 constrain-based routing protocols such as "QoS extensions to Open Shortest Path First" (QOSPF) and "Constraint based Routing Label Distribution Protocol" (CR-LDP). The ability to compute routes based on multiple constraints such as bandwidth and delay requirements enables the Internet Service Provider's (ISP) to provide different levels of customer service provisioning.

#### **1.4 Organization of Thesis**

The thesis is organized as follows. *Section 2* offers an overview of how voice signals are digitized, compressed and converted into IP packets to be transmitted over a data network. *Section 3* shows the features of DiffServ and MPLS that makes them a very good combination for providing end-to-end QoS provisioning. *Section 4* presents the analysis and discussion of the MPLS queuing model derived from the stochastic Markov Chain theorem using models with non-preemptive priority queuing discipline. *Section 5* shows the implementation of the queuing model using the OPNET simulation software. Performance metrics for voice packets transmission under different parameters in an MPLS network are measured and analyzed. *Section 6* summarizes the contribution of this thesis. Finally, *Section 7* discusses about some of the possible future work.

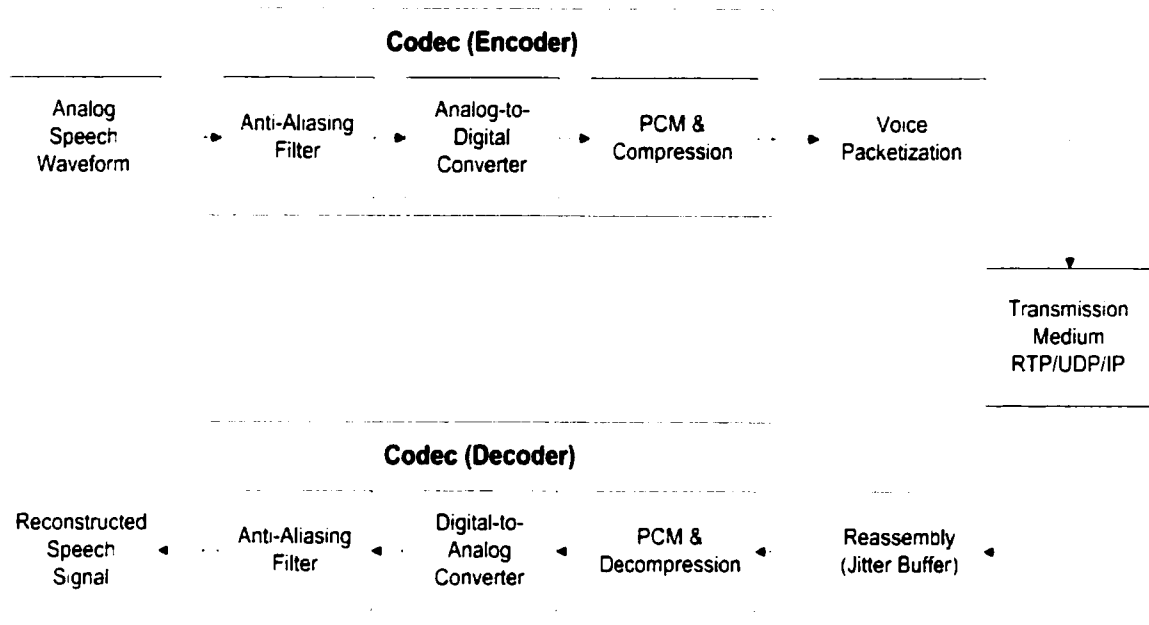
## **Chapter 2 Voice over Internet Protocol (VoIP)**

Internet Telephony, or better known as “Voice over IP” (VoIP), is a service to support normal telephone calls over a packet-switched network such as the Internet, rather than using the public switched telephone network (PSTN). One of the major concerns for the successful deployment of VoIP is the stringent end-to-end delay that is required by real-time voice conversation. Voice signals have to be digitized, processed and packetized before they can be transported through the packet-switched networks. These signal processing procedures will incur a fixed delay on the end-to-end transmission time, in addition to the variable delay imposed by the queuing and scheduling schemes from the data packet network. This section of the thesis presents a detailed discussion on the various digital signal processing (DSP) procedures required by a VoIP network.

### **2.1 Voice Digitization**

Analog speech signals have to be converted into digital bit streams before they can be transmitted through packet-switched networks. Normal narrow-band speech conversation is typically band-limited to 4000 Hertz (Hz). This conversion is done with a coder-decoder (codec), which consists of an anti-aliasing filter, an analog-to-digital converter (ADC), and a DSP unit to perform quantization and compression. Since Nyquist theorem states that analog signals need to be digitally sampled at twice the maximum signal frequency to avoid aliasing during signal reconstruction, the sampling

rate of the ADC is configured as 8000 Hz. *Figure 2.1* shows the signal processing performed on the voice packets along the transmission path.



*Figure 2.1: The Transmitter and Receiver in an IP Telephony System*

## 2.2 Voice Coder-Decoder (Codec)

A codec is a device that encodes or decodes a signal. Since speech samples are highly redundant, various codecs have come up with their unique compression schemes to produce good quality speech with efficient bandwidth utilization. Speech codecs can be broadly categorized into two classes: waveform codec and source codec. The former is used at high bit rates with very good quality speech, while the latter operates at very low bit rates with “synthetic” speech quality.

### **2.2.1 Waveform Codecs**

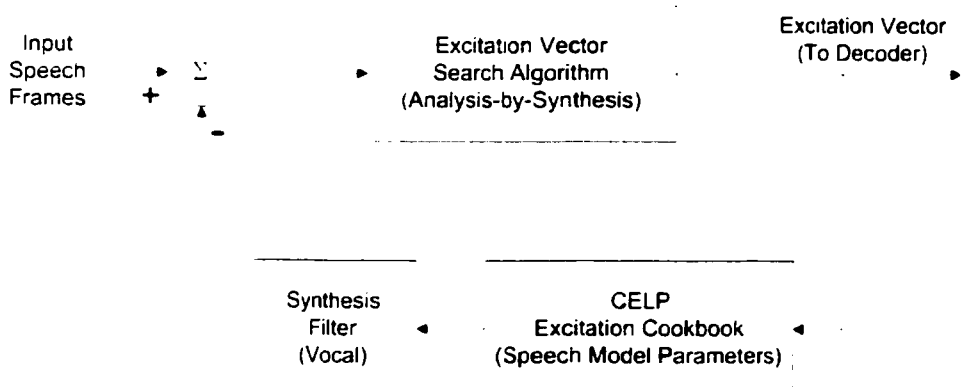
The simplest form of waveform coding is Pulse Code Modulation (PCM), which merely involves sampling and quantizing of the input signal. If linear quantization is used, 12-bit samples are needed to produce good quality speech, giving a bit rate of 96 kbps. However, this bit rate can be reduced by non-uniform quantization, using 8-bit samples and a bit rate of 64 kbps. Two widely used compression schemes in these non-uniform quantizers are the logarithmic  $\mu$ -law and A-law compression. PCM is specified by G.711 and has a bit rate of 64 kbps, which is similar to a digital phone line.

A commonly used technique in speech coding is to predict the value of the next sample based on the values of previous samples. If the predictions are effective, the error between the predicted sample and the actual speech sample will be small, and this error signal can be quantized using fewer bits than the actual speech sample. This is the basis for Adaptive Differential Pulse Code Modulation (ADPCM) codecs, which further reduce the bit rate by lowering the error through adaptive predictors and quantizers to match the changing characteristics of the speech that is being encoded. At the decoder, the quantized error signal is added to the predicted sample to reconstruct the speech sample. Standardized codecs that use ADPCM are specified by G.726 with multiple bit rate selections of 40, 32, 24 and 16 kbps.

### **2.2.2 Source Codecs**

Source coders for speech are also called voice coders (vocoders). They contain the models of how human speech is generated. The synthesis filter in a vocoder has the

function of the throat and mouth. The excitation vectors, which contain the parameters of the speech models, are applied to the filter to generate the desired speech signals. Speech samples are stored in frames of 10 – 30 milliseconds (ms), depending on the type of source codec used. The coder will analyze and extract the appropriate Code Excited Linear Prediction (CELP) model excitation vector for these frames. The approach for determining these excitation vectors is called the analysis-by-synthesis technique, where the encoder searches through the excitation cookbook for the vector that produces the best match for the synthesized signal with the original speech signal. The excitation vectors are then encoded and transmitted to the decoder, where they are applied to the filter to reconstruct the speech signal. A general model for the CELP encoder is shown in *Figure 2.2*.



*Figure 2.2: A General Model of CELP Encoder*

Another common type of source codec is the G.729 Conjugate Structure Algebraic CELP (CS-ACELP) [21] [22], which operate with speech frames of 10 ms. The advantage of this type of codec is that it is able to produce very high quality speech at a low bit rate of 8 kbps, while the major drawback is the computational complexity in

the excitation vector search algorithm. In addition to that, the determination of the excitation vectors requires an extra 5 ms look-ahead delay, giving a total processing delay of 15 ms for the first voice packet. Another variant to this codec is the G.728 Low Delay CELP (LD-CELP) [23] [24]. Instead of buffering 10 ms of speech frames as with CS-CELP, it determines the excitation vectors by incorporating the past reconstructed speech signals. Thus, this codec only requires a frame length of 5 samples, giving it a processing delay of less than 2 ms. However, the tradeoff for this low delay comes from a higher bit rate of 16 kbps to produce good quality speech.

A new dual rate source coder has been specified by G.723.1 [25]. It uses speech frames of 30 ms and encodes the speech samples using the linear predictive analysis-by-synthesis technique. The coder uses Multi-Pulse Maximum Likelihood Quantizer (MP-MLQ) excitation vectors for a higher rate of 6.3 kbps, and Algebraic CELP (ACELP) excitation vectors for a lower rate of 5.3 kbps. In addition to that, this codec requires an extra look-ahead delay of 7.5 ms, giving it a total processing delay of 37.5 ms for the first voice packet [26]. *Table 2.1* provides a brief description of the various International Telecommunication Union (ITU) standardized codecs.



*Table 2.1: Summary of ITU Standardized Codecs*

ITU-T Standards	Description	Bit Rate (kbps)	Conversion Delay (ms)	Look-Ahead Delay (ms)
G.711	PCM	64	0.125	N/A
G.726	ADPCM	16, 24, 32, 40	0.125	N/A
G.728	LD-CELP	16	2.0	N/A
G.729	CS-CELP	8	10	5
G.723.1	MP-MLQ	6.3	30	7.5
G.723.1	ACELP	5.3	30	7.5

A major drawback of processing data in frames is that the processor has to wait for the complete frame before any processing can be done. Since digitized speech signals are sampled at 8000 Hz, each sample has an interval 0.125 ms. Although a bigger frame size will result in better signal processing efficiency, but the frame size will directly affect the latency experienced by the codec.

### **2.3 Echo Cancellation**

Echo is caused by signal reflections of the speaker's voice back to the speaker's ear. Echo becomes a significant problem for speech conversations when the round-trip delay is more than 50ms. Since echo can quickly deteriorate speech quality, efficient echo cancellation has to be implemented in voice communication networks to provide clear calls.

There are two ways echo is generated: impedance mismatches in the circuit-switched PSTN network, or from acoustic coupling between the microphone and the earpiece of a telephone device. Impedance-induced echo normally happens at the

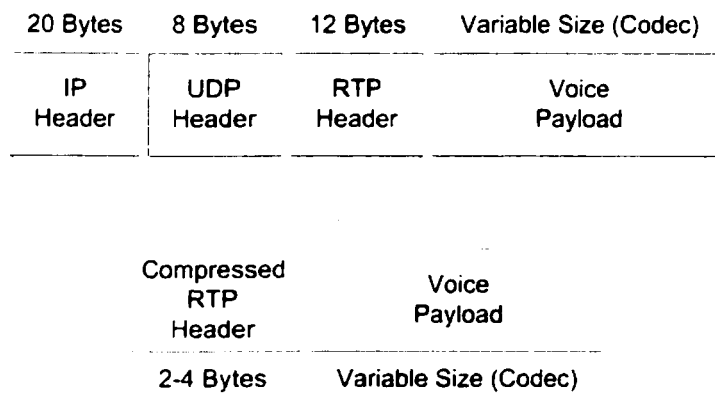
intersection of the PSTN network and the local loop. The conversion from a four-wire to two-wire telephone cable causes a less-than-perfect impedance match on the phone circuit. At the receiver end, a portion of the inbound audio is leak to the transmit path, and gets transmitted back to the speaker. The echo cancellers for legacy circuit-switched phone networks are normally implemented in hardware, whereas Voice over IP (VoIP) communication networks uses software DSP algorithms to implement its echo cancellation. Acoustic-induced echo is caused by audio that leaks into the microphone from the earpiece of the same telephony device or a nearby speaker. The DSP-based echo canceller can also remove this type of echo by removing the “extra” digitized audio at the receive path.

## ***2.4 Voice Transportation Layers***

Once digitized voice is encoded and packetized, IP is responsible for transporting the voice packets across a packet-based network. IP is a connectionless-oriented protocol, which does not guarantee the delivery of packets like a virtual circuit connection. Since voice transmission is a real-time application, it requires reliable transportation and some form of delay and packet arrival order guarantees. There are two types of transportation protocol: Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). The former is connection-oriented protocol, where a path is established before actual data transmission. In addition to that, TCP also offers error detection and packet reordering capabilities to provide reliable packet transportation. On the other hand, UDP is a connectionless protocol. It neither provides any packet ordering

nor reliable data transmission. At first glance, it seems that TCP is a much better protocol to be used, but UDP is chosen because TCP is too heavy for voice transmission. For example, real-time applications do not require the retransmission capability of TCP because any lost packet will not be processed.

Real-time applications require a mechanism to ensure that its packet stream to be reconstructed correctly at the receiver end. The individual packets making up the data stream experience a variable delay time when passing through a packet-based network. Real-Time Transport Protocol (RTP), specified by *RFC 1889*, is used to transport the voice packets across the network. *Figure 2.3* shows a typical VoIP packet using RTP.



*Figure 2.3: Normal VoIP Packet and Compressed RTP Header*

A RTP header is used as a prefix to these voice packets. It contains timing information that enables the receiver to place the inbound voice packets into the jitter buffer in the correct order. This is done to remove the timing variations of the voice packets and reconstruct the original voice stream. RTP does not provide any mechanism to ensure delay or QoS guarantees, but relies on the lower-layer services such as MPLS to perform those operations. The RTP/UDP/ IP header adds 40 bytes to the voice payload.

However, this header overhead imposed on a VoIP packet can be reduced by using compressed RTP or cRTP, which will compress the header to 2 - 4 bytes. This can be accomplished because much of the header information is redundant and does not change from packet to packet.

## **2.5 Jitter Buffer**

Packet-based IP networks can neither guarantee the delivery time nor the order of the transmitted voice packets. The variation in the packet inter-arrival time at the receiver's end is called jitter. A buffer is required at the receiver to compensate for the unpredictable nature of packet-based networks by storing the voice packets according to their order in time before they are decoded to reconstruct the speech signal. Any lost packet will not be processed, even if it is being retransmitted by the source, since voice is a real-time audio stream. Thus, User Datagram Protocol (UDP) is used as the transportation protocol for voice packet transmission instead of the more secured Transmission Control Protocol (TCP). Although the permissible jitter buffer setting range is between 0ms and 255ms, it is typically set between 10ms and 50ms. This delay is also called as the decompression delay.

## **2.6 Audio Silence Detection**

During a telephone conversation session, a speaker talks an average of approximately 65 percent of the time. Silence compression is an important feature to take advantage of bandwidth savings by generating voice packets only during periods of active conversation. Silence detection consists of three major components: voice activity detector (VAD), discontinuous transmitter (DTX) and comfort noise generator (CNG). VAD is responsible for determining whether the user is talking or under silence. The simplest form of VAD is the threshold-based detection method, which uses the magnitude of the speech signal in decibel as its determination factor. DTX is used to stop codec from transmitting packets when VAD detects a silent period. CNG generates a white noise signal to simulate background noise during those silent periods when both parties are not talking so that the listeners would not have the wrong assumption that the telephone line is dead. This is because telephone users are already accustomed to the background noise in the PSTN.

## **2.7 Loss Packet Compensation**

The typical reason for a loss packet during transmission is due to network congestion. In a packet-based IP network, the packets are stored in a buffer before they are serviced. Packets will be dropped if the buffer exceeds its capacity during periods of network congestion. Although this design works for data streams due to the retransmission capability of TCP, it will not work for lost voice packets. The consequences of a lost voice packet are much more severe than the loss of a data packet.

That is the reason why the notion of Quality of Service (QoS) has caught a lot of attention from networking researchers to support VoIP applications. The idea behind QoS is to classify the network traffic into multiple priority levels for traffic differentiation. The prioritized voice packets will experience a lower loss probability and an improvement in end-to-end delay when traversing the network.

A few schemes have been introduced to compensate for the lost voice packets. If only one voice packet is lost during transmission, normal listener would not be able to differentiate the degradation in speech quality. A common and simple concealment strategy can be used to compensate for this kind of loss by just replaying the previous voice packet to avoid the gap of silence produced by the lost packet. However, if a string of voice packets are lost, this speech interpolation method will not work. Thus, QoS is an essential tool to the success of the deployment of real-time applications over the Internet.

## ***2.8 Per-Call Bandwidth***

One of the most important factors to be considered in a network planning is capacity provisioning – how much bandwidth is allocated for voice transmission. It is important for the network planners to compute the required bandwidth for each VoIP call, based on the type of codec, payload size and voice packet optimization methods deployed. The packet optimization techniques considered here are: compressed RTP header, VAD and packet fragmentation and interleaving. These optimization methods have to be supported on both ends of the VoIP gateway before they can be deployed. The assumptions made for our bandwidth calculation are listed below:

1. The size of the compressed RTP header is 2 bytes
2. VAD is assumed to reduce bandwidth utilization to 65% of full rate transmission
3. Multilink Point-to-Point Protocol (MLPPP) is used for packet fragmentation and the size of the header is 6 bytes
4. Voice payload size of 240 bytes using G.711 coding is calculated from the packetization delay of 30 ms.

The formulas for the calculation of per-call bandwidth are listed below [27], and the results are shown in *Table 2.2*.

$$\text{Voice Packet Size} = \text{MLPPP Header} + \text{compressed RTP Header} + \text{Voice Payload}$$

$$\text{Per - Call Bandwidth} = \text{Voice Packet Size} \times \frac{\text{Codec Bit Rate}}{\text{Voice Payload Size}}$$

*Table 2.2: Bandwidth Savings Using Various VoIP Packet Optimization Techniques*

Codec (Bit Rate)	Voice Payload (Bytes)	Bandwidth (kbps)			
		Without Optimization	cRTP	cRTP & VAD	cRTP, VAD & Fragmentation
G.711* (64kbps)	240	75	65	42	43
G.711* (64kbps)	160	80	65	43	44
G.726* (32kbps)	120	43	33	22	23
G.728 (16kbps)	20	48	18	12	15
G.729 (8kbps)	10	40	9.6	6.3	9.4
G.723.1 (6.3kbps)	24	16.8	6.9	4.5	5.5

\* These are waveform codecs that do not depend on speech frames. The voice payload is chosen to carry multiple samples to reduce packet overhead.

G.723.1 (5.3kbps)	20	15.9	5.9	3.8	4.9
----------------------	----	------	-----	-----	-----

## **2.9 VoIP Delay/Latency**

Real-time voice conversations are delay and jitter sensitive. Once the one-way delay exceeds 250 milliseconds, the conversation parties are unable to tell whether the other party has finished speaking, and this situation will lead to both parties talking at the same time. The International Telecommunication Union's G.114 recommendation, which comprise the guidelines for voice transmission time limits, specifies a one-way delay of 150 milliseconds. The "contributors" to the overall end-to-end delay of an IP telephony system can be divided into two categories: gateway-incurred delay which consists of fixed delays such as speech coding delay and decompression delay, and network-incurred delay which consists of variable delays such as network congestion delay and queue scheduling delay [28][29][30].

### **2.9.1 Gateway-Incurred Delay**

The VoIP gateway is the most important equipment in an IP telephony system. It is responsible for the digitization and encoding of voice signals, packetization of voice packets and transmitting the packets over the IP network. Most of the delays imposed by the VoIP gateway are fixed delays.

#### **1. Packetization Delay**

The packetization process consists of voice coding and the grouping of encoded samples into packets for transmission. Thus, the packetization delay can be



divided into two categories: algorithm delay and packet processing delay. The former is also called the codec delay. It is caused by the need to collect a frame of voice samples to be processed by the voice coder. The frame size is related to the voice coding technique utilized by the codec. A number of the standardized codecs and their frame times can be found in the *conversion delay* column of *Table 2.1*. As we can see from the frame times related to the codecs, there is a vast difference between waveform and source codec.

The packet processing delay is caused by the grouping of the encoded samples into a voice packet for transmission over the IP network. Since the algorithm delay time for the source codec is quite large, voice payload for these codecs will normally contain only a few encoded CELP codewords. For G.723.1, it is not advisable to have multiple codewords in a packet. For waveform codecs, they have the same sample time of 0.125  $\mu$ s. Thus, voice packets utilizing waveform coding will contain multiple samples to reduce packet header overhead. The number of samples packed into a voice packet will directly affect the amount of latency. For G.711, grouping 160 bytes of PCM samples into a packet equivalent to 20 ms of “speech” delay, while grouping 240 bytes of PCM samples, will produce 30 ms of delay.

The advantage of using this technique is the reduction of bandwidth utilization, at the expense of processing delay. However, this increases the severity of lost voice packets since more data are packed into a packet. Since grouping multiple codewords into a voice packet will lead to intolerable delay for

source coding due to the addition of later frame periods, an alternative technique is introduced to achieve better bandwidth efficiency. Voice streams from different speech channels originating from the same gateway going to the same destination can be multiplexed into a single voice packet [31]. Although this multiplexing scheme is still not standardized, it can be implemented as a proprietary solution to improve the overall performance of the IP telephony network.

## **2. *Jitter Delay***

Jitter is variation in the packet inter-arrival time at the receiver's end. A buffer is required to store the voice packets according to their order in time before they are decoded to reconstruct the speech signal. Without this buffer, there will be a very high chance that gaps will be heard in the reconstructed speech. The size of the jitter buffer determines the proficiency of the system to tolerate jitter caused the network. The tradeoff for this protection is the increase in overall delay.

### **2.9.2 Network-Incurred Delay**

After the VoIP gateway has encoded and packetized the original speech signal, the voice packets are transmitted over the IP network. Since it is a packet-based network, it neither guarantees the delivery time nor the order of the transmitted voice packets. Most of the delays imposed by the network are variable delays.

#### **1. *Transmission Delay***

This delay is caused by the time it takes to place bits and bytes onto the physical link. It is inversely proportional to the link speed and is relatively minimal

compared to other type of delays we discussed. However, if the connection to the IP network uses low-speed serial communications such as 28.8kbps dial-up modems, the transfer time of the data can add significant amount of delay compared to other high-speed links such as a 1.5Mbps T1 connection. The formula for transmission delay is as follows:

$$\text{Transmission Delay} = \frac{\text{Packet Size}}{\text{Link Speed}}$$

As an example, the transmission delay for a 10-byte packet using 28.8kbps low-speed connection is 2.78ms, while the delay for the same 10-byte packet using a T1 connection is reduced to 0.053ms. Consequently, although the transmission delay is unavoidable, using high bandwidth links can reduce the overall delay.

## **2. Propagation Delay**

This delay is caused by the time it takes an electrical signal to traverse the distance of a transmission line. It only becomes an issue when the signal travels a great distance, such as inter-continental communications. The formula for propagation delay is as follows:

$$\text{Propagation Delay} = \frac{\text{Distance}(m)}{\text{Speed of Light}(m/s)}$$

## **3. Network Delay**

This latency includes the queuing and scheduling delays incurred by the packet-switched network. The type of scheduling mechanism implemented in the network will have a great impact on the QoS guarantees that the network can provide, e.g. packets with higher priority will get preferential treatment. The

scheduler will decide which packet to be transmitted based on its scheduling algorithm. Please refer to *Section 4* for a detailed discussion on different queuing and scheduling schemes. In addition to that, the packet might experience extensive queuing delay if the node is congested.

*Table 2.3* below shows a sample calculation for the overall end-to-end delay for voice transmission using the G.729 CS-CELP codec, with link speed of 64kbps and 10-byte voice frames. Although the network delay is the only variable delay listed in the table, choosing an appropriate codec for a particular network can further reduce end-to-end delay to an acceptable level for voice transmission. In addition to that, efficient IP-based voice stream multiplexing schemes can also reduce the packetization delay and improve the overall performance of the VoIP network.

*Table 2.3: Overall End-to-End Delay Calculation*

	Fixed Delay (ms)	Variable Delay (ms)
Coder Delay - Look Ahead	5	
- Packetization	20	
Serialization Delay (64kbps)	2	
Propagation Delay	30	
Network Delay (64kbps)		43 <sup>†</sup>
Jitter Buffer	50	
<b>Total Delay</b>		<b>150 ms</b>

<sup>†</sup> This is just an assumption made in order to come up with a complete sample calculation that has one-way delay of 150ms.

## Chapter 3 Multiprotocol Label Switching (MPLS)

Over the years, the Internet has seen explosive growth from a modest data network for the research community to become a worldwide public network that supports a multitude of multimedia traffics such as voice and video transmission. Corporate users have come to understand the enormous potential of the Internet, leading to the mass deployment of virtual private networks (VPN) and electronic commerce sites. The Internet has to evolve from the current best-effort service towards a differentiated service framework that provides quality of service (QoS) assurances to support these new applications.

Traditional IP routers analyze the IP address of each incoming packet and search their routing table for the longest prefix match for choosing the next hop router. The maximum speed at which a router can make routing decisions and forward packets determines the efficiency of the link utilization. For example, taking the worst-case scenario of a packet stream with minimum packet size of 32 bytes, to achieve 100% utilization on 100 megabits per second (Mbps) Ethernet links, the minimum per-packet processing time would have been:

$$S_{\min} = \frac{32 \times 8}{100 \times 10^6} = 2.56 \times 10^{-6} \text{ seconds}$$

In other words, the router needs to perform routing decisions for  $3.9 \times 10^6$  packets per second. The processing speed required to perform routing in high-speed networks such as OC-48 links with transmission rate of 2.5 gigabits per second (Gbps) will increase to considerable proportion. To make matters worse, the growth of the Internet

increases the routing table size, which further increases the processing time of searching the longest-prefix match for an IP address. Consequently, a better routing scheme than the destination-based routing algorithm used by today's Internet has to be devised to solve this scalability problem.

### **3.1 Architecture of MPLS**

MPLS can be treated as a Layer 2.5 protocol – it integrates the Layer 3 routing with Layer 2 switching. It uses a label-switching packet forwarding technique that is very similar to the forwarding scheme that is used by ATM networks. The mapping of Layer 3 routing information to Layer 2 forwarding information enables fast packet forwarding and efficient traffic engineering.

An MPLS Label Switched Router (LSR) is composed of two distinct functional components – a control component and a forwarding component, which is shown in *Figure 3.1*. The control component uses routing protocols such as Open Shortest Path First (OSPF) and Border Gateway Protocol (BGP) to exchange information with other LSRs to build and maintain the forwarding table. When a packet arrives, the forwarding component uses the label of the packet as an index to search the forwarding table for a match, and directs the packet from the input interface to the output interface through the switching fabric. There are three main system components in an MPLS router [32]: the forwarding module, control module, and Label Information Base (LIB).

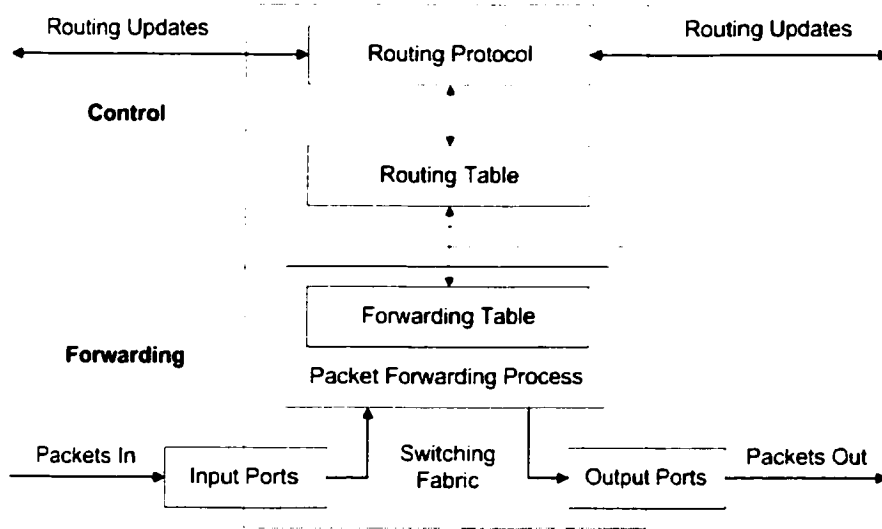


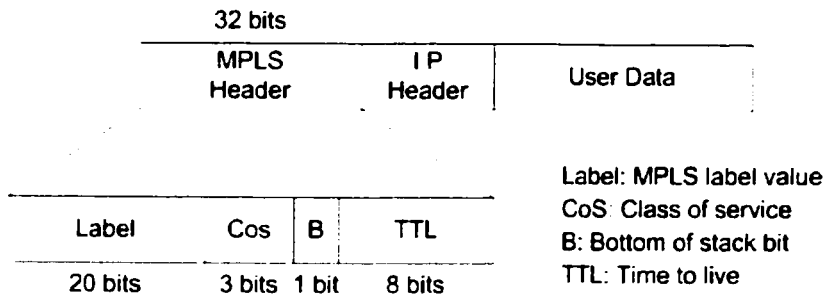
Figure 3.1: The Control and Forwarding Component of an MPLS Router

### 3.1.1 Forwarding Module

When an Ingress LSR (inbound edge router) received a packet from a traffic flow, a label will be allocated to the packet based on its traffic class. All subsequent packet forwarding at the core LSR is based on a label-swapping mechanism. In other words, MPLS has replaced the destination-based hop-by-hop forwarding paradigm of today's Internet with a label-swapping forwarding paradigm. It simplifies the routing process that we have today by replacing the longest-prefix match of IP routing with simple short-label exact match forwarding for faster routing, and replacing big routing tables with small index tables for labels for more efficient routing table searching algorithm.

When an IP packet enters an MPLS domain, the Ingress LSR will encapsulate the IP packet with an MPLS header. This 4-byte header contains a 20-bit label to act as the index for the forwarding table, a 3-bit Class of Service (CoS) field for traffic management and QoS indication, a 1-bit bottom-of-stack indicator, and an 8-bit Time-

To-Live (TTL) field to prevent packets from looping forever in the network. *Figure 3.2* shows the MPLS header encapsulation for an IP packet.



*Figure 3.2: MPLS Header*

There are three label manipulation instructions in an MPLS domain. The Ingress LSR creates a new label and pushes it to the label stack a packet, the core LSR swaps the incoming label with a corresponding next-hop label found from the forwarding table, and the Egress LSR (outbound edge router) pops a label from the label stack. Only the label at the top of the stack determines the forwarding decision. *Figure 3.3* shows the label-switching paradigm in an MPLS network. Label stacking enables multi-level hierarchical routing. For example, BGP labels would be used for higher-level hierarchical packet forwarding from one BGP speaker to the other, while Interior Gateway Protocol (IGP) labels would be used for packet forwarding within an autonomous system (AS).



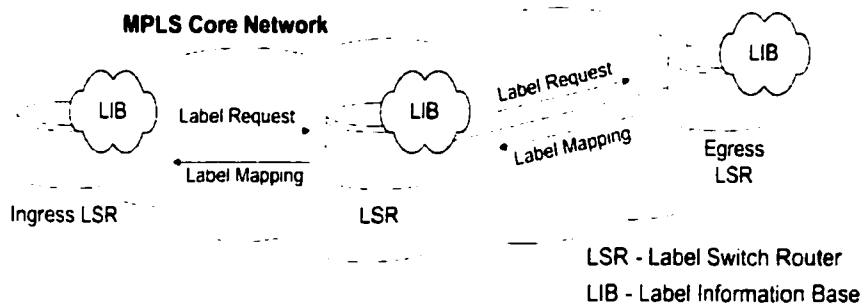
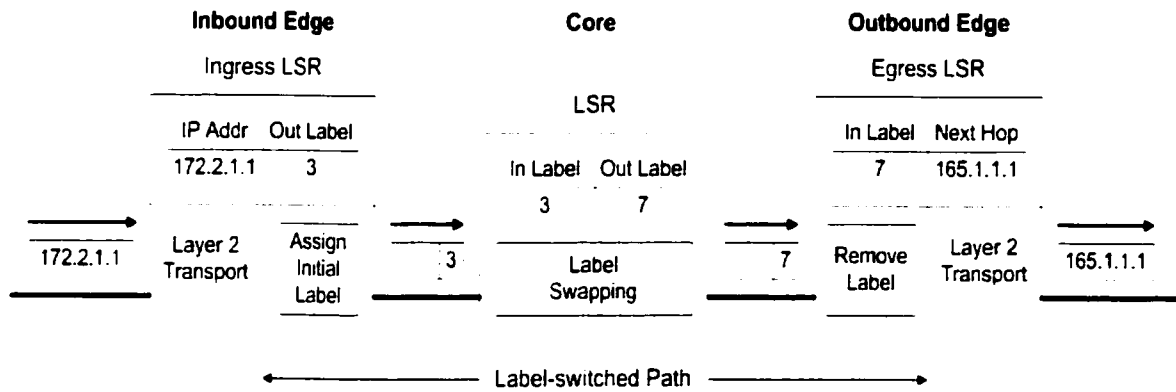


Figure 3.3: Elements in an MPLS Network

### 3.1.2 Label Information Base (LIB)

The Label Information Base (LIB) consists of all the information regarding the labels that have been allocated. The functionality of this module in an MPLS LSR is the same as the routing table in an IP router – it enables the router to make forwarding decision for the packets. MPLS has introduced the concept of Forward Equivalency Class (FEC) to provide QoS in the network. FEC can be defined as a group of packets that have the same forwarding treatment in an MPLS domain. The label assignment for the incoming packets is based on the FEC for the traffic flow. The packets will then be encapsulated with the MPLS header and forwarded to the next-hop router.

When the core LSR receives the labeled packet, it will use the label as an index to search the Incoming Label Map (ILM) table for the corresponding next-hop label, which is stored in the form *<in label, out label>* known as a Next-Hop Label Forwarding Entry (NHLFE). The label in the MPLS header will be replaced by the *out label* and be forwarded to the next-hop. When the packet arrives at the Egress LSR, the MPLS header will be decapsulated and the packet will be routed to its destination. The control module will distribute the label information during the Label-switched Path (LSP) setup phase, which will be discussed in *Section*.

The FEC-to-NHLFE Map (FTN) table is updated at the Ingress LSR so that the packet will have the correct forwarding treatment with the other traffic flows that are of the same FEC. The FECs specified are a global entity for the entire MPLS network, and they can be set up with different levels of granularity such as IP Prefix for standard IP routing, and different levels of QoS. The routing protocol uses the FTN to forward unlabeled packets, while the label binding and distribution uses ILM to forward packets.

### **3.1.3 Control Module**

The route determination or control module is used to construct, remove or update the LIB entries. The “multiprotocol” nature of the control component enables the support of various routing and signaling protocols such as OSPF, BGP, ReSerVation Protocol (RSVP), and Label Distribution Protocol (LDP). One of the major benefits from the separation of the control and forwarding component is to ensure interoperability, since the Internet is consists of a combination of networks with different Layer 2 technologies.

Furthermore, each component can be independently modified or upgraded, providing a more attractive upgrade path for the ISPs.

In order to provide more efficient QoS and traffic engineering for the Internet, the routing protocols will have to perform constraint-based rather than shortest-path routing in the network. Constraint-based routing is the ability to make routing decision based on different performance constraints such as bandwidth utilization, delay guarantees and QoS requirements. Various routing protocols have begun making extensions to support the distribution of relevant state information to perform constraint-based routing. Some of the popular ones are QoS for OSPF (QOSPF) [33], MPLS RSVP (RSVP-TE) [34], and Constraint-based Routing LDP (CR-LDP) [35]. These signaling protocols are responsible for reserving the appropriate resources in a network during the Label-switched Path (LSP) setup phase before actual data transmission. They should be responsible for the establishment, teardown and maintenance of the LSP.

Within an MPLS domain, a LSP is established for a packet based on its FEC. It is analogous to a virtual circuit (VC) in an ATM network. A LSP is setup as follows. First, the Ingress LSR will bind a label to the arriving IP packet and begins to set up a “logical” path to forward the packet to its destination. The Ingress LSR will advertise its routing information and label bindings to its “upstream” neighbors towards the Egress LSR through a *Label Request* message. The process is repeated by core LSR until an Egress LSR that can service the IP packet is found. In response, the Egress LSR will send a *Label Mapping* message back to the Ingress LSR, and during the propagation of this response message, the ILM of the core LSRs along the LSP will be updated. Thus, a

Label Switching Path (LSP) is established in the MPLS network to forward the IP packet to its destination. *Figure 3.3* shows the establishment of a LSP in the MPLS network.

## **3.2 Benefits of MPLS**

MPLS has been hailed as the latest step in the evolution of multilayer switching technology for the core of the Internet. It enables the integration of the scalability of IP with the simplicity of Layer 2 packet switching paradigm. In this section, we will discuss about the various benefits of an MPLS network.

### **3.2.1 Simple Packet Forwarding Paradigm**

MPLS uses a label-swapping forwarding algorithm to simplify its routing process. This simple short-label exact match forwarding mechanism enables faster routing and smaller forwarding tables. This ensures the scalability of the network in the future.

### **3.2.2 Interoperability**

Interoperability is the ability of two or more systems or components to exchange information and to use the information that has been exchanged [36]. The separation of the control and forwarding components in the MPLS design provides the foundation for multilayer, multiprotocol interoperability between various advanced Layer 3 routing protocols and Layer 2 switching technologies. *Figure 3.4* shows an MPLS network that supports multiple Layer 2 switching technologies, using a standard IP routing protocol.

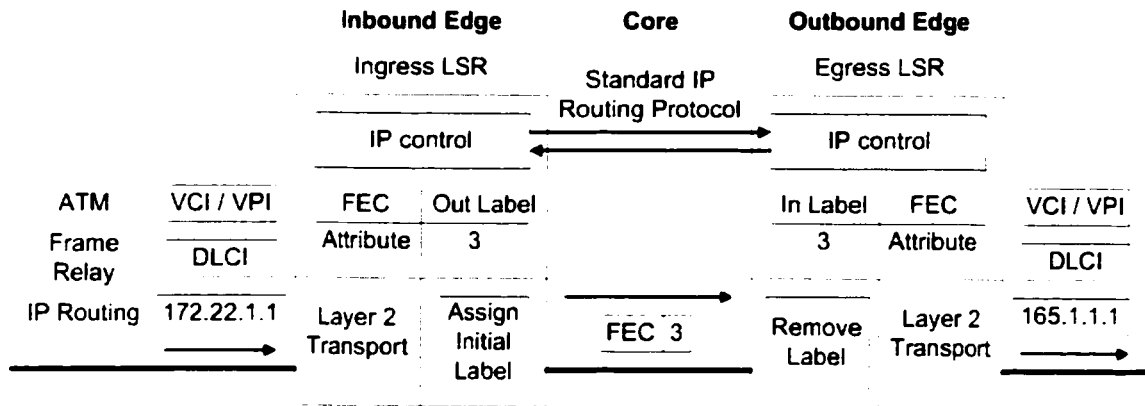


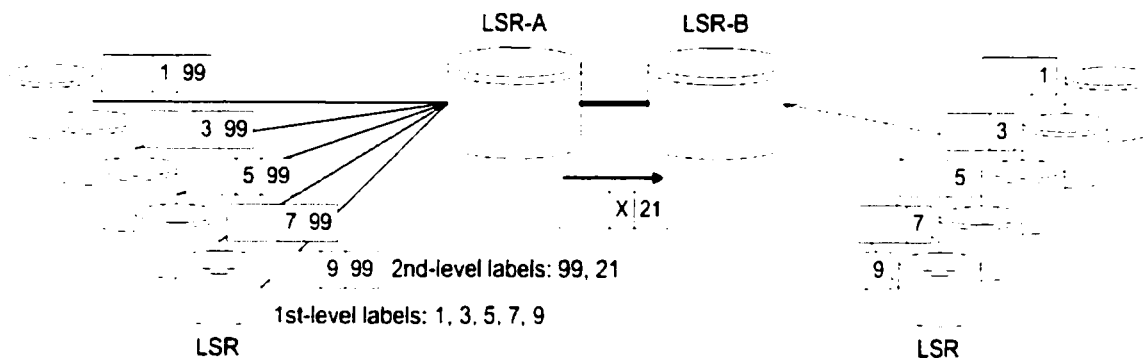
Figure 3.4: Interoperability of MPLS over Multiple Layer 2 Switching Technologies

### 3.2.3 Routing Hierarchy and Label Stacking

Routing hierarchy is used to perform LSP encapsulation within another LSP. This is done through the use of label stacking. As an example, label stacking provides the capability comparable to IP-in-IP tunneling in VPN implementation. This has led to the class-based forwarding, where different streams with the same FEC can be aggregated and transported using the same LSP. Stacking maintains the identity of each aggregated streams at the higher-level routing hierarchy. This is similar to the concept of Virtual Path/Circuit Identifiers (VPI/VCI) in ATM networks, although ATM only supports a 2-level routing hierarchy. With routing hierarchy, it simplifies the aggregation and de-aggregation of LSP flows.

The example in *Figure 3.5* shows a 2-level hierarchical routing topology. The first and second level labels are obtained during each of their respective LSP establishment. Traffic Aggregation occurs at LSR-A for the streams with the same FEC, and uses the same label #21 for a particular FEC. Traffic de-aggregation occurs at LSR-B, where the

second-level label is removed and the packets are forwarded to their respective destination based on the first-level label.



*Figure 3.5: Routing Hierarchy in an MPLS Network*

### **3.2.4 Traffic Classification**

At the Ingress LSR, each incoming IP packet is analyzed and classified into different FEC. This traffic classification scheme provides the capability to partition the traffic for service differentiation. With the introduction of constraint-based routing, FEC is able to segregate the traffic into different level of QoS, each with different service constraints to support different type of services such as latency-based voice traffic and security-based VPN.

### **3.2.5 Quick Fault Recovery**

A signaling protocol is required to be robust and scalable. In addition to LSP establishment/teardown/maintenance operations, the signaling protocol is also required to support path recovery operations. They should provide the capability to specify different path attributes during a LSP setup phase for fault recovery purposes. For example, LSP

priority and preemption level is used by the high-priority paths to preempt the lower-priority paths during a path recovery scenario. The flexibility on path setup options and rerouting mechanism makes the network more reliable. A detailed discussion on reliable service for an MPLS network is presented in [37].

### **3.2.6 Traffic Engineering Capabilities**

Traffic engineering can be categorized into two different areas based on their objective: traffic-oriented or resource-oriented. The former relates to the optimization of key traffic performance parameters such as the minimization of packet loss and delay, maximization of throughput, quick fault recovery when node or link fails, and the enforcement of Service Level Agreement (SLA), while the later relates to optimizing the network utilization to avoid congestion due to inefficient traffic mapping. The requirements for traffic engineering for an MPLS network are defined in RFC 2702.

Layer 2 switching schemes such as ATM and Frame Relay (FR) possess traffic engineering capabilities to provide a certain QoS guarantees for the network. ATM supports a very rich set of QoS infrastructure that supports different traffic contracts such as Constant Bit Rate (CBR), Variable Bit Rate (VBR), real-time VBR (rt-VBR) etc. Although these QoS mechanisms already exist in the Layer 2 switching technologies, true end-to-end QoS is only achievable with Layer 3 routing protocol. The overlay model of IP-over-ATM has its shortcomings such as the use of ATM Adaptation Level (AAL). MPLS solve this issue by integrating the scalability of Layer 3 routing algorithms with the speed and traffic engineering capability of Layer 2 switches through its control and forwarding components.

Figure 3.6 provides an example the capability of traffic engineering. Assume that Router R1 has a packet to send to R2. For OSPF, the routing algorithm will route the packet through the shortest path, regardless of whether R3 is experiencing congestion. In an MPLS network, a LSP can be set up explicitly to avoid the congested node, or if a constraint-based routing algorithm such as CR-LDP is used, a LSP that avoids the congested node will be set up dynamically even though the routing path is longer. This path management capability is great for traffic engineering.

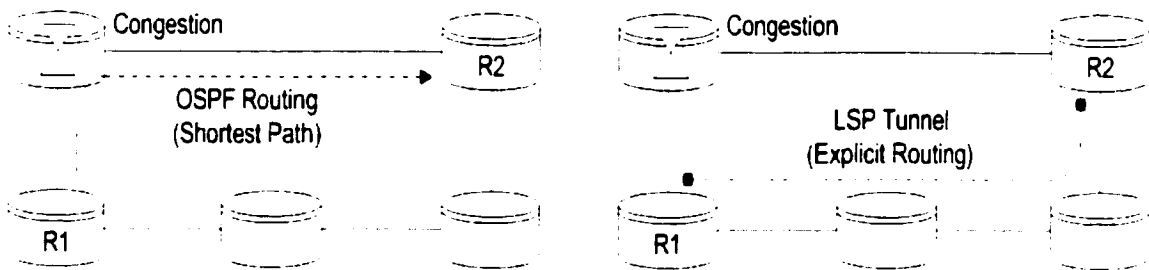


Figure 3.6: Traffic Engineering Capability of an MPLS Network

### 3.3 MPLS versus ATM

In recent years, the industry has been searching for an approach to combine the best features of IP routing and the throughput of ATM switching. A number of overlay models have been proposed: the classical IP-over-ATM model, LAN Emulation (LANE) and multiprotocol over ATM (MPOA). However, these approaches were inefficient in that they require the use of additional servers for address mapping and routing, and did not take full advantage of the QoS capabilities of ATM. Although the label-swapping forwarding paradigm and Layer 2 traffic engineering capabilities of MPLS is very similar



to the ones in ATM, there are still subtle differences between the two that make MPLS outperforms the latter technology.

One of the major advantages of MPLS is that it has better integration of the routing and switching layers. ATM, as a pure Layer 2 switching technology, is separated from the Layer 3 routing algorithms. In other words, it could not “see” the entire network topology to provide efficient path management. Moreover, the separation of the MPLS control and forwarding components has led to multilayer, multiprotocol interoperability between Layer 2 and Layer 3 protocols. ATM has to depend on AAL to interact with the IP layer, but has problems inter-operate with other protocols such as frame relay.

MPLS supports variable-length inbound traffic packets, unlike ATM that only supports fixed-length 53-byte cells. This will cause inefficient bandwidth utilization since the 5-byte header overhead for ATM cells is very large. For an ATM overlay model, the reliance on the segmentation and reassembly process in the ATM Adaptation Layer (AAL) increases the packet processing time. Although both technologies use connection-oriented packet forwarding, ATM does not provide loop protection for its cells since there is no TTL field in the cell header.

The ATM network in an overlay model has to be meshed. However, the flooding of link-state updates has very bad scalability in a meshed network, making IP-over-ATM impossible to scale to large network. Furthermore, MPLS provides multi-hierarchical routing through label stacking. This enables simple traffic aggregation and de-aggregation, which fits well with the trend of class-based routing paradigm. However, ATM has cell-interleaving problems when merging cells from multiple virtual circuits.

Although this problem can be overcome by using a virtual path as a second-level routing hierarchy to merge those virtual circuits, but it is not efficient and increases complexity in the forwarding table.

### **3.4 Differentiated Services (DiffServ)**

The Internet has had such great success that over the years, it has evolved rapidly and become one of the most prominent carriers of information in our society. Besides traditional electronic mail and file transfer, the Internet is being used by an increasingly large number of multimedia applications with very different network requirements such as Voice over IP (VoIP) and video conferencing. There is a remarkable demand for the Internet to support Quality of Service (QoS) for different classes of traffic, as opposed to the single best-effort level of service provided by today's Internet.

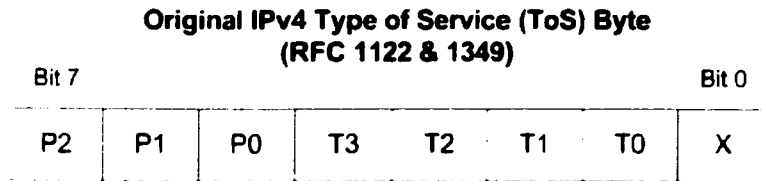
Two major QoS architectures have been proposed. The Integrated Services, with Resource Reservation Protocol (RSVP) as the signaling protocol, was designed to offer end-to-end per-flow delay guarantees. However, this requires end-to-end per-flow reservation state processing in every node, causing scalability issues in its deployment for large networks. In addition to that, the difficulty of upgrading the infrastructure of the size of today's Internet as a whole to support strong per-flow service guarantees weakens the cause for the wide deployment of RSVP [38]. A *simple* QoS guarantee is the new requirement. This has led to the development of the Differentiated Services (DiffServ) architecture, which defines a model for implementing scalable differentiation of QoS in the Internet. DiffServ uses a relatively simple and coarse method to classify packets into

different service classes based on the Differentiated Services Code Point in the IP header. The scalability of DiffServ mechanism is achieved by implementing complex classification and traffic conditioning functions only at the boundary nodes and simple class-based forwarding in the core nodes. The aggregation of flows that belong to the same traffic class will have the same forwarding treatment, formally called Per-Hop Behavior (PHB), at each DiffServ-enabled router. As a result, the amount of state information at each node is reduced to the number of aggregated classes rather than the number of flows [39].

Although pre-flow guarantees require complex signaling support, the strong QoS guarantees provided by this model is warranted in an access switch or router. But such fine control might not be feasible if it is used in a backbone switch with OC-48 links, since the amount of state information to be processed for each flow is just paramount [40]. Thus, the potential for traffic aggregation offered by the DiffServ model is very beneficial in the backbone of the Internet, whereas the Integrated Services and RSVP model is more appropriate for smaller networks that require stringent service guarantees.

### **3.4.1 Per-Hop Behavior (PHB)**

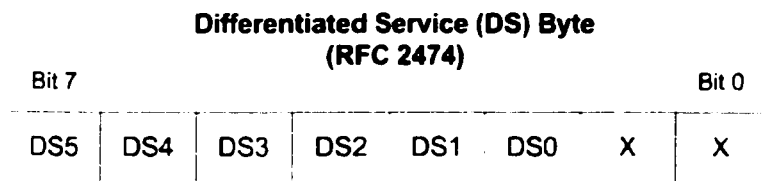
Differentiated Services (DiffServ) is a classification mechanism that provides traffic differentiation based on aggregating traffic flows into classes or service priorities. This classification is based on the Differentiated Service Code Point (DSCP), and can be found in the Type of Service (ToS) field of the IPv4 packets. *Figure 3.7* shows the original IPv4 Type of Service versus Differentiated Service byte.



P[2..0] = IP Precedence

T[3..0] = IP Type of Service (ToS)

X = Unused Bit



DS[5..0] = Differentiated Service Code Point (DSCP)

X = Unused Bit

*Figure 3.7: IPv4 Type of Service versus Differentiated Service Byte*

A DSCP determines the PHB associated with a packet. A PHB specifies both the queuing and scheduling treatment that it will receive from the server and the drop precedence from the congestion avoidance mechanism. Three categories of PHB that have been defined for DiffServ: Expedited Forwarding (EF), Assured Forwarding (AF) and Best-Effort (BE) Forwarding.

### 3.4.1.1 Expedited Forwarding (EF)

EF PHB [41] provides both bandwidth and delay service guarantees. It is used for applications that require low-latency, low-loss, and low-jitter performance guarantees. Other traffic flows could not share the bandwidth that has been reserved for EF flows. EF traffic class can be implemented using priority queuing (PQ) with the combination of a rate limiting policer.

### 3.4.1.2 Assured Forwarding (AF)

AF PHB [42] is divided into several priority classes for service differentiation. Each service class is assured to have a certain minimum service guarantee. AF flows can be divided into 4 different AF forwarding classes, each with 3 levels of drop precedence. It is also an IP precedence-based classification scheme because the first 3 digits of the DSCP correspond to the original IP precedence defines for today's Internet [43]. The level of assurance for AF PHB depends on resource allocated to the service class, the current load of the class and the drop precedence of the packet. *Table 3.1* specifies the original DSCP for the AF PHB.

*Table 3.1: Original DiffServ AF Codepoints*

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

### 3.4.1.3 Best-Effort (BE) Forwarding

If a packet arrived at a DS-compliant node without any DSCP specification, it will get mapped to the BE PHB. It is the as the default forwarding scheme, and the packets will only get transmitted when excess bandwidth is available, or when the network is not congested. This traffic type does not have any QoS guarantees.

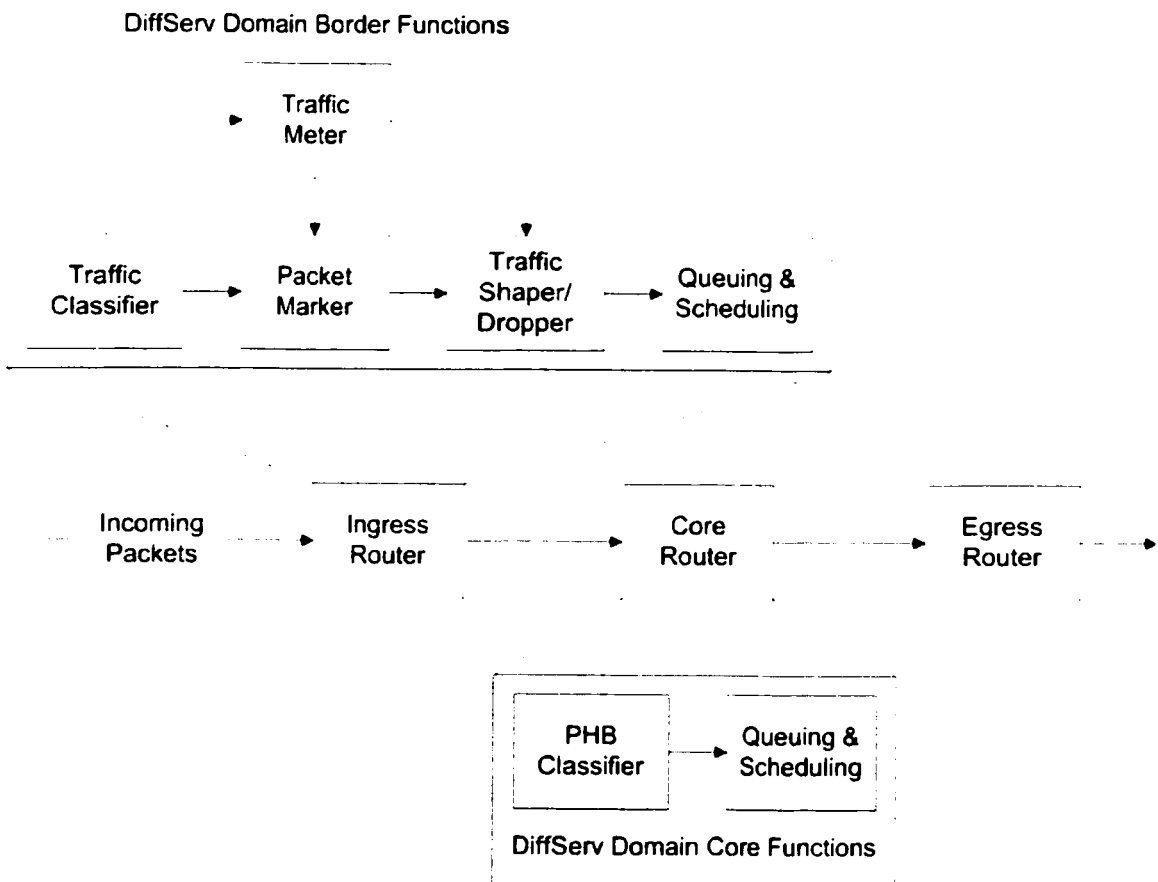
In general, higher priority traffic flows will get better transmission treatment than lower priority traffic flows. *Table 3.2* summarizes the characteristics for each of the three DiffServ PHBs [44].

*Table 3.2: Characteristics of DiffServ Per-Hop Behaviors (PHB)*

<b>PHB</b>	<b>Priority Level</b>	<b>Description</b>
EF	High	<ul style="list-style-type: none"> <li>• EF flow can preempt AF and BE flows</li> <li>• EF traffic flows are guaranteed for performance</li> <li>• Good for real-time applications such as voice transmission</li> </ul>
AF	Medium	<ul style="list-style-type: none"> <li>• AF flow can preempt BE flows</li> <li>• Assures a minimum bandwidth for each flow</li> <li>• Each AF flow can be further divided into 3 priority classes</li> <li>• A high priority AF flow can preempt a low priority AF flow</li> <li>• Good for premium data traffic</li> </ul>
BE	Low	<ul style="list-style-type: none"> <li>• BE flow can be preempted by EF and AF flows</li> <li>• It is the default forwarding scheme</li> <li>• Good for regular data traffic</li> </ul>

### 3.4.2 End-to-End Quality of Service (QoS)

An incoming packet flow will be classified when it enters a DiffServ domain. In order to provide end-to-end QoS, the DiffServ domain has to contain the proper traffic management mechanism to enforce the service guarantees specified by the different types of PHB. These traffic management functions can be broadly categorized into two classes: the domain border and core functions. *Figure 3.8* shows a block diagram of the boundary and core routers in a DiffServ domain.



*Figure 3.8: The DiffServ Boundary and Core Routers*

### **3.4.2.1 DiffServ Domain Border Functions**

The main function of the traffic management modules in a border router is to apply the conditioning functions such as traffic metering, shaping, and dropping to the different classes of traffic. The basic building blocks for a DiffServ Boundary router are listed below:

#### ***1. Packet Classification***

In DiffServ, the classification of packets is done through the aggregation of the same traffic class into multiple priority levels or classes of service (CoS). The collection of packets that have the same DSCP value is called a Behavior Aggregate (BA). These packets in the same BA will experience the same forwarding treatment in the DiffServ domain, such as packet scheduling, queuing, policing, and shaping.

#### ***2. Packet Metering***

This module performs in-profile/out-of-profile checking on each incoming packet to determine whether the packet is compliance to the traffic profile. It will pass the result to traffic marker and shaper/dropper to trigger the proper traffic management procedure. This policing function is done through a token bucket scheme. Depending on the traffic policy specified, the packet in question can be transmitted, dropped or remarked with a different DSCP.

#### ***3. Packet Marking and Remarking***

The packets are marked into EF, AF, or BE forwarding classes by writing the appropriate DSCP into the packet header. For the packets that are out-of-profile,



they can be remarked into BE traffic or to a lower precedence, since they have violated the SLA.

#### **4. *Traffic Dropper***

The functionality of a dropper is to drop packets in a traffic flow to ensure that it is compliance with the traffic profile. If an EF traffic flow is out-of-profile, the packets can be dropped directly by the dropper since any delay in real-time traffic will not be tolerated. The traffic dropper also works with the congestion control mechanisms such as Random Early Detection (RED) for implementing their packet dropping policies.

#### **5. *Traffic Shaper***

The function of a shaper is to delay the packets in a traffic flow to achieve the target flow rate. This delay is done through the “store and forward” process using shaping buffers and token bucket. The shaper might be able to eliminate jitter if used accordingly.

#### **6. *Congestion Avoidance***

Two of the most popular queue managements for congestion avoidance are the Random Early Detection (RED) and Weighted Random Early Detection (WRED). These algorithms overcome the disadvantages of the tail-drop policy by selectively dropping the packets when the queue size exceeds a certain minimum threshold. The idea is to rely on the adaptive nature of TCP traffic flows to treat packet drop as a mean to show network congestion and reduce their transmission

rate. Another benefit of RED is from the statistical point of view: random packet drops is better than lots of packet being dropped when the queuing buffer is full.

#### **3.4.2.2 DiffServ Domain Core Functions**

The queuing discipline is the most important element in a DiffServ core router since service differentiation is achieved through the scheduling algorithm. Strict priority queuing (PQ), weighted round robin (WRR), weighted fair queuing (WFQ) and class-based queuing are different scheduling schemes that can be implemented to provide QoS in the network. A detail discussion on this topic will be presented in *Chapter 4*.

### **3.5 MPLS and Differentiated Services (DiffServ)**

The combination of MPLS and DiffServ provides a very attractive strategy for providing end-to-end QoS. DiffServ offers scalable end-to-end QoS while MPLS performs traffic engineering and efficient packet forwarding. After DiffServ has performed traffic classified on a packet, the Ingress LSR will map the incoming DSCP to the corresponding Class of Service (CoS) within the MPLS network. For EF and BE PHB, we can have a direct mapping of the DiffServ DSCP into MPLS CoS. However, since the CoS field in the MPLS header is only 3-bit wide, there can be only 8 BA specified within an MPLS network. This means that MPLS has less service granularity than the one DiffServ proposed. As a result, some work has been done to group the traffic flows that have similar service level in AF into a single MPLS service class [39]. For AF PHB, MPLS has derived from *RFC 2597* to come up with an Olympic service offering

that consists of three service classes – Gold, Silver and Bronze, each with two level of drop precedence. Table 3.3 summarizes the QoS mapping between DiffServ and MPLS.

*Table 3.3: QoS Mapping Between DiffServ and MPLS*

<b>DiffServ</b>		<b>MPLS</b>	
<b>PHB</b>	<b>DSCP</b>	<b>Class of Service</b>	<b>Service Class</b>
EF	101110	111	Premium
AF11	001010	110	Gold
AF12	001100	101	
AF13	001110		
AF21	010010	100	Silver
AF22	010100	011	
AF23	010110		
AF31	011010	010	Bronze
AF32	011100	001	
AF33	011110		
AF41	100010		
AF42	100100		
AF43	100110		
BE	000000	000	Best Effort

### **3.6 MPLS-based Virtual Private Network (VPN)**

With the growth of the Internet, virtual private network (VPN) has rapidly evolved into a mainstream network option compared to the traditional wide area network (WAN) for corporations that require remote access and site-to-site connectivity. Since VPN is such an important service to be offered by ISP, we will devote this section to discuss the benefits and methods of implementation of an MPLS-based VPN.

#### **3.6.1 Introduction**

VPN is a network in which multiple remote sites of a corporation are interconnected together on a public infrastructure, with the same access or security policies as a private network. AVPN can be deployed over the Internet or built on a service provider's existing IP, Frame Relay or ATM infrastructure. However, besides having to ensure that the inter-site connectivity is always private, ISP has to satisfy a broad range of other customer requirements such as different security levels, number of sites, number of users, bandwidth allocations, traffic patterns, routing complexity and mission-critical applications. As a result, QoS is a very important issue for the successful deployment of VPN. There are three categories of VPN:

##### **3.6.1.1 Remote-Access VPN**

Remote-access VPN connects telecommuters and mobile users from remote sites to the enterprise WAN. With *Client-initiated Access VPNs*, telecommuters establish an encrypted IP tunnel from their remote sites to the corporate network across a service

provider's shared network. Client-initiated VPNs ensure end-to-end security from the client to the host. As for a *Network Access Server-initiated Access VPN*, the telecommuters connect to the ISP through telephone line dial-up session or ISDN, and the ISP initiates a secure, encrypted tunnel to the corporate network using the Network Access Server. Although this VPN implementation relieves companies from the details involved in managing a secured network, but the drawback for this implementation is the lack of end-to-end encryption.

### **3.6.1.2 Intranet**

An intranet is an intra-company VPN. It interconnects the branch offices within an enterprise with their headquarters to form a "private" network, enabling the sharing of corporate information for different remote sites.

### **3.6.1.3 Extranet**

An extranet is an inter-company VPN. It provides business partners limited access to the corporate WAN, providing a secured means to allow transactional business-to-business activities.

The last two categories of VPN can be deployed over several architecture choices. The first implementation method is through the establishment of IP tunnels based on IPSec (IP Security). For this implementation, Intranet and Extranet VPNs create tunnels across an IP network to provide private and secured data forwarding between different sites in the VPN. Another popular VPN implementation method is through the use of

virtual circuits (VC), based on Layer 2 switching technologies such as ATM or Frame Relay.

Traditionally, VPN is deployed using leased lines interconnecting the various remote sites. However, leased lines are expensive, bandwidth inefficiency and are difficult to scale. As a result, virtual circuits have replaced leased lines as the preferred method in VPN implementation, mainly based on Layer 2 switching technologies. The two widely used VPN models are the overlay and peer-to-peer model.

### **3.6.2 Overlay VPN Model**

In this VPN model, the ISP provides the customer a set of point-to-point or point-to-multipoint links between the customer's remote sites. Layer 2 overlay model make use of the current Layer 2 switching technologies such as ATM and Frame Relay to provide "L2 tunneling" for the VPN service, whereas Layer 3 overlay model provides "L3 tunneling" at the network layer through packet encapsulation such as IP-over-IP tunneling. In ATM, the emulated leased lines can either be permanent virtual circuit (PVC) or switched virtual circuit (SVC).

There are two major advantages for using this VPN model. It provides some form of QoS guarantees in terms of per-flow bandwidth guarantees in terms of Committed Information Rate for a virtual circuit. The Constant Bit Rate (CBR) and Variable Bit Rate (VBR) are two examples of QoS offered by ATM networks. In addition to that, it also can provide traffic-engineering capabilities through manual virtual circuit setup.

However, as the number of sites grows, the topology of the Layer 2 switching network becomes very complex, and this is very bad for the management and

provisioning of large number of VCs. Furthermore, the provisioning of the bandwidth requirement for each VC between different remote sites involves the detailed knowledge of the site-to-site traffic profiles. Finally, this model limits the VPN infrastructure to a single medium, such as ATM. This model is mostly deployed within an ATM or Frame Relay network.

### **3.6.3 Peer-to-peer VPN Model**

In this VPN model, the ISP and its customers exchange routing information between the edge routers, i.e. between the Customer Premises Equipment (CPE) and the Provider Edge Router (PE-router). Using the routing information from the CPE for each VPN, the ISP backbone network will forward the data without the customers' network involvement.

The advantages of this model are based on the utilization of simple and optimal routing algorithm such as OSPF for the exchange of routing information. The routing algorithm easily handles topology modifications such as the addition of new VPNs.

Although this model solved the scalability issue of the overlay model, but it still contains a few disadvantages that requires further modifications. Since customers might use private IP addresses in their local networks, some of those private addresses might no longer be unique when they arrived at the VPN network. Moreover, the lack of isolation between the customer's VPN is also a detriment to the success of this model. However, this model is still deployed in the IP networks, with the support of IP-over-IP tunneling or Network Address Translation to prevent the overlapping of private IP addresses and some form of secured transmission.

Since the overlay model and the peer-to-peer model have their own fair share of drawbacks that cannot simply be disregarded, modifications of the VPN architecture have to be made. This has led to the development of MPLS-based VPN.

### 3.6.4 MPLS-based VPN

There are two types of VPN architecture based on MPLS. They are the BGP/MPLS-based VPN model defined in *RFC 2547bis* and the MPLS-based Layer 2 VPN model.

#### 3.6.4.1 BGP/MPLS-based VPN

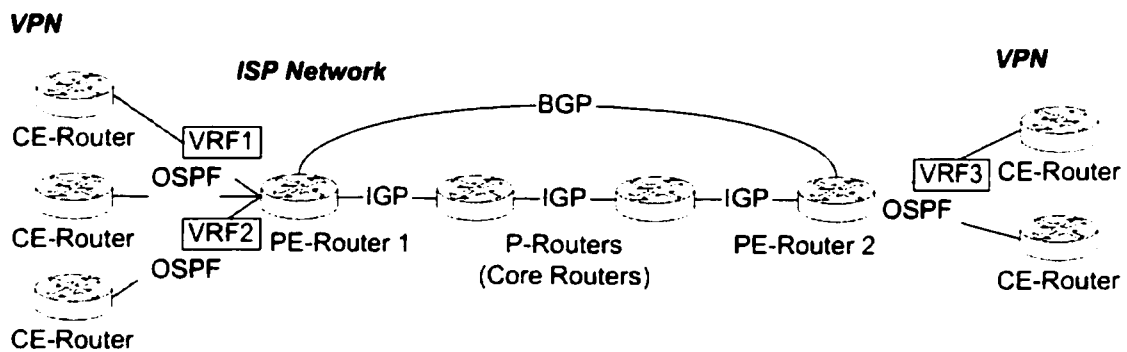


Figure 3.9: RFC 2547bis BGP/MPLS-based VPN

*RFC 2547bis* defines a mechanism for the ISP to use their network to provide VPN services [45]. It is also known as the BGP/MPLS VPN because BGP is used to distribute the routing information between PE-routers, whereas MPLS is used to forward the inter-site VPN traffics. *Figure 3.9* shows a sample topology for a BGP/MPLS-based VPN.

Each VPN remote site has its own “virtual” routing and forwarding table (VRF) in the PE-router, solving the problem of overlapping private IP addresses amongst the



customer sites. Furthermore, MPLS enables the use of simple routing algorithms at the network layer, solving the scalability issue in overlay model while maintaining the capabilities of virtual circuits at the transportation layer. The basic building blocks for the BGP/MPLS VPN are listed below:

**1. VRF (Virtual Routing and Forwarding Table)**

The VRF contains a list of routes that is available to a particular VPN connected to the PE-router. If two or more VPN remote sites have the same routing table, they can share the same VRF.

**2. CE-Router (Customer Edge Router)**

A CE-router provides the data link connection from the customer site to the ISP network. The CE-router will advertise its *local* VPN routes to the PE-router that it attaches to, and receives *remote* VPN routes from the PE-router.

**3. PE-router (Provider Edge Router)**

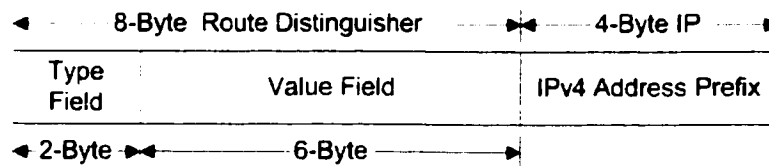
The PE-router is able to exchange routing information with the CE-routers attached to it through various Internal Gateway Protocols (IGP) such as OSPF. The PE-router only maintains the VPN routes for those VPNs that are attached to it. Exchange of VPN routing information between PE-routers is done through Border Gateway Protocol (BGP).

**4. P-routers (Provider Router)**

The P-routers are only required to maintain the routes between the PE-routers since the forwarding of VPN traffic is done through a two-layer label stack.

Each PE-router is only required to maintain the VRFs for the VPNs directly attached to it. The capability of supporting multiple VRFs enables the different VPN sites that connect to the same PE-router to use the same IP address space, i.e. private addresses. In addition to that, multiple VRF prevents communication between sites that are not in the same VPN, preserving the isolation of network traffic. This enables the establishment of both Intranet and Extranet for a particular VPN site.

When a *local* route is advertised from a CE-router to its corresponding PE-router, the address is appended with a 64-bit prefix, which is also known as a VPN route distinguisher (RD). This results in the exchange of a unique 96-bit VPN address between the PE-routers. The encapsulation of RD and IPv4 is shown in *Figure 3.10*.



*Figure 3.3.10: Route Distinguisher of a VPN Address*

PE-routers exchange routing information through Border Gateway Protocol (BGP), since the information is only needed by the edge routers (PE-routers), but not required by the core routers (P-routers). The PE-routers will perform their routing decision based on the VPN address. Keeping the VPN routing information out of the core routers will result in better scalability and stability in the core routers. In addition to that, the RD is a globally unique VPN-IPv4 address to support carrier-of-carriers implementation so that each ISP will not have conflicting VPN addresses with other ISP. The procedures for the establishment of a BGP/MPLS-based VPN is listed below [46]:

#### ***3.6.4.1.1 Step I: Routing Information Exchange***

The CE-router advertises its *local* VPN route information to the PE-router it is attached to. The VRF in the PE-Router is updated, and a unique label is allocated for each VPN route. After that, the Ingress PE-router will advertise the VPN routes in its VRF to other PE-routers by propagating the MPLS labels allocated for each route and the IP address of the Ingress PE-router as the BGP next hop throughout the MPLS network through BGP. This is because the P-Routers in the backbone do not need to look into the VPN address for forwarding decision. Finally, the Egress PE-Router will determine which entries in its VRF are required to be updated with the routing information from the Ingress PE-router by performing route filtering (route target).

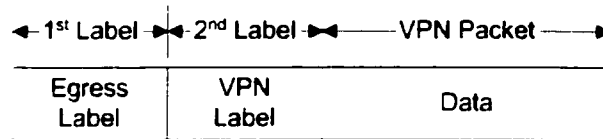
#### ***3.6.4.1.2 Step II: LSP establishment***

A Label Switching Path (LSP) must be established between the Ingress PE-Router and the Egress PE-Router by IGP, such as Label distribution Protocol (LDP) or Resource Reservation Protocol (RSVP). The label associated with the Ingress PE-Router will be propagated throughout the MPLS network across the P-Routers to the Egress PE-router. Then, the LSP between two PE-Routers is established once the Ingress receives a confirmation from the Egress. The Ingress PE-Router will now have the label to forward packets to the Egress PE-Router.

#### ***3.6.4.1.3 Step III: VPN Packet Forwarding***

When Ingress PE-Router receives a VPN packet, the VRF associated with that VPN is examined. The label associated with the destination VPN, given by the Egress

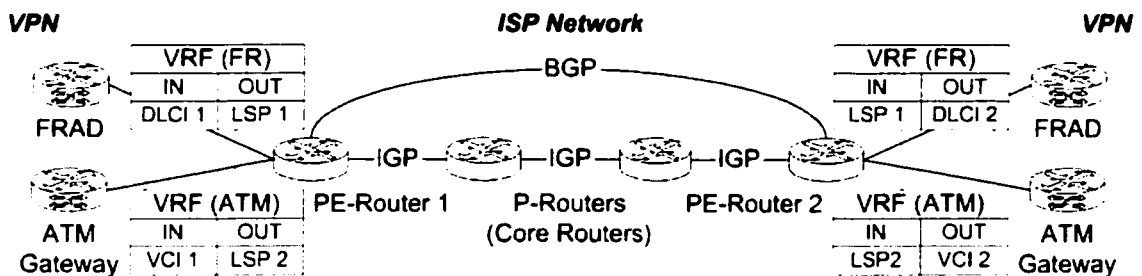
PE-Router in step 1, is push onto the label stack. Next, the label pointing to the Egress PE-Router, obtained from step 2, is push onto the label stack. This will create a 2-level label stack MPLS packet encapsulation shown in *Figure 3.11*. The core routers (P-routers) will forward the packet according to the Egress PE-Router label. When Egress PE-Router receives the packet, it will pop the first label, and examine the second label. The packet will then be sent to the correct VPN.



*Figure 3.11: 2-level Label Stack*

### 3.6.4.2 MPLS-based Layer 2 VPN

In this VPN architecture, the ISP is using the MPLS network to provide layer-2 services to the customers [47]. The ISP is only responsible for the layer-2 connectivity, whereas the customers themselves are responsible for the layer-3 routing. As a result, the customer edge (CE) devices can be configured as Frame Relay or ATM. *Figure 3.12* shows a sample topology for an MPLS-based Layer 2 VPN.



*Figure 3.12: MPLS-based Layer2 VPN*

The PE-routers in the MPLS network will maintain the routes to the destination CE devices for each VPN. This layer-2 encapsulation method enables the ISP to transparently transport any traffic type over type MPLS network. This enables a “tunnel” to be created between two PE-routers to provide a private and secured connection between two remote sites.

The implementation MPLS-based layer 2 VPN is very similar to the overlay architecture that uses ATM or Frame Relay. However, one of the major advantages of MPLS-based Layer 2 VPN is that the ISP only requires to maintain and manage a single MPLS network but still able to support various VPN implementations. The converged network is not only interoperable with the current VPN designs, it even supports the older Customer Premises Equipment (CPE) such as the Frame Relay Access Device (FRAD) and ATM gateways that the customers have already invested. In addition to that, the ISP can also offer the BGP/MPLS-based VPN solution described in the previous section.

### **3.6.5 Benefits of MPLS-based VPN**

The most significant benefit of deploying an MPLS-based network is that the ISP only needs to maintain and manage a single network to support multiple VPN implementations and other network services using a differentiated service framework. The strong integration between the control and the forwarding components in MPLS enables the ISP to provide Layer 3 services such as QoS, video conferencing and IP telephony for a VPN.

Currently, privacy is provided through tunneling in connection-oriented networks such as ATM. In a connection-oriented topology, it is hard to modify or configure the VC

mapping table. MPLS-based VPN can be considered as a “connectionless” service because it relies on Layer 3 routing protocols such as OSPF to take care of the complex path setup procedures. This enables easy creation of new VPNs or modifications to the topology of an existing VPN. Furthermore, class-based routing in MPLS is better than per-flow routing in ATM to provide QoS guarantees, since ATM has cell-interleaving problems when aggregating the same-class traffic.

MPLS-based VPN has great scalability due to its network design structure. PE-routers are only required to maintain the VPN routes for the VPNs that are directly attached to them, and P-routers are not required to maintain any VPN routes because of the 2-level label stack implementation. Furthermore, CE-routers from customer sites do not need to exchange routing information with each other directly. Finally, the ability to support overlapping private address through label encapsulation eliminates IP-in-IP tunneling implementation or the use of Network Address Translator (NAT).

## **Chapter 4 Queuing and Scheduling Analysis**

Successful deployment of a differentiated service network requires new data path and control path mechanisms. The data path is responsible for classifying and mapping incoming packets to their respective service classes and controlling the amount of network resources that a service class can consume. The control path is required to setup the service level agreement (SLA) between the user and the Internet Service Provider (ISP). The ISP is required to provide the performance guarantees specified in the SLA as long as the user honors its part of the contract by not sending excessive traffic, but is allowed to reject the client's request for bandwidth due to insufficient network resources or administrative constraints [48].

The International Telecommunication Union's H.323 recommendation is a set of protocols that specifies the procedures for real-time multimedia communication over packet-based networks that do not provide guaranteed quality of service (QoS) [49]. It includes the specifications for audio codecs, video codecs, data conferencing, call control and signaling mechanisms, and transport protocols such as Real-Time Protocol (RTP). From a Voice over IP (VoIP) network perspective, the areas of interest are the audio codec standards and RTP discussed in *Chapter 2* and the call admission control (CAC) algorithm. The later is used for the provision call signaling between the various components in a H.323 environment. The CAC includes functions such as call setup, call authorization and admission, call tear down and other supplementary services such as voice mail and call transfer that are found today's Public Switched Telephone Networks

(PSTN). Moreover, the H.323-standardized CAC ensures the signaling interoperability between VoIP networks and legacy PSTN. For detailed information on H.323, please refer to [50] and [51].

#### **4.1 Guaranteed Service Model**

With the convergence of voice and data networks, provision of service guarantees is becoming increasingly important for the Internet. Packet queuing and scheduling algorithms in switches and routers play an important role in providing Quality of Service (QoS) required by various time-critical applications. This guaranteed service model is a two-step procedure: the connection admission control algorithms *reserve* resources during connection establishment time, while the packet service disciplines *allocate* resources according to the reservation during data transfer [52].

There are several aspects to be considered when choosing a packet service discipline to achieve the performance guarantees required. A flow is a stream of packets that traverses the same route from the source to the destination that required the same Per-Hop Behavior (PHB) at each node along the path. The *fairness* of a scheduling algorithm ensures every active flow gets its fair share of bandwidth. An active flow is considered as a flow that has packets to be transmitted in the queuing buffer. The excess bandwidth from the inactive flows will be distributed fairly among the active flows. In addition to that, the scheduling algorithm has to provide *isolation* among the different flows in service. This is to protect the conforming flows from other ill-behaving flows. The packet processing complexity of a scheduling algorithm measures the amount of time



complexities to enqueue or dequeue a packet. The algorithm must be “simple” so that it can be implemented in high-speed networks. Last but not least, the *delay bound* of a scheduling algorithm will have a great impact on the end-to-end delay experienced by a traffic flow.

There are two elements in the data path mechanism to ensure QoS for a network – the scheduling algorithm and buffer management schemes. However, packet scheduling is a greater concern than buffer management. This is because as link speed increases, the processing time available for each packet decreases in proportion. This is especially essential for high-speed networks, where most of the switching mechanism is implemented in hardware. For a packet flow with minimum packet size of 32 bytes to achieve 100% utilization on an OC-48 links with transmission rate of 2.5 gigabits per second (Gbps), the processor will have to service almost 10 million packets per second.

## **4.2 Scheduling Algorithms**

The type of scheduling mechanism employed will have a great impact on the QoS guarantees that the network can provide. The basic function of a scheduler is to decide which incoming packet is selected for transmission on the output link. It can be broadly classified into two categories: sorted priority schedulers and frame-based schedulers. The sorted priority schedulers require a global state variable called the virtual time to determine the timestamp for each incoming packet. The packets are then scheduled in the increasing order of their timestamps. Examples of sorted priority schedulers are Weighted Fair Queuing (WFQ) [53], Self-clocked Fair Queuing (SCFQ) [54], and Worst-case Fair

WFQ (WF<sup>2</sup>Q) [55]. They provide good fairness and low delay guarantee, but are not very efficient due to the complexity involved in virtual time computation and packet sorting. The best known algorithm for inserting into a sorted queue required  $O(\log n)$  work complexity, where  $n$  is the number of active flows [56]. For frame-based schedulers such as Deficit Round Robin (DRR) [57], the scheduler visits all the active flows in a round-robin fashion. The service received by an active flow during its round of service is proportional to its fair share of the bandwidth specified by the weights for each flow. The work complexity of the frame-based scheduling discipline is  $O(1)$ .

#### **4.2.1 First In First Out (FIFO)**

The simplest form of scheduler is FIFO. It is also known as the First Come First Served scheduler. Incoming packets are served in the order in which they arrive. FIFO is very simple to implement. Insertion and deletion from the queue has a work complexity of  $O(1)$ . Although FIFO by itself cannot provide service differentiation, it is still the most commonly implemented scheduling policy. With the help of a buffer management scheme, it is possible to control the sharing of bandwidth among different classes and traffic.

The traffic flows under a pure FIFO scheduling scheme will neither have fair treatment nor isolation. A higher-speed link will tend to take up more spaces in the output buffer compared to lower-speed links, and ill-behaving flows will consume more than its fair share of bandwidth. Although some form of congestion control can be implemented using the combination of Random Early Detection (RED) and Transmission Control

Protocol (TCP), ill-behaving User Datagram Protocol (UDP) flows can still cause other well-behaved TCP flows to lose their share of bandwidth.

Delay bound of a FIFO scheduler is limited to the size of the queuing buffer. Control algorithms such as Resource Reservation Protocol (RSVP) can be used with FIFO to reserve the appropriate link capacity for a particular traffic flow. Although tight delay bounds are required by real-time applications, when high-speed links are concerned, even the worst-case delays are still considerable small and tolerable by the application. As an example, the worst-case delay caused by a 1 Megabyte queuing buffer serving an OC-48 link is less than 3.2ms. Thus, the delay bound for a FIFO scheduler is:

$$\theta_{FIFO} \leq \frac{B}{\gamma}$$

where B is the buffer size and  $\gamma$  is the link speed.

#### **4.2.2 Priority Queuing (PQ)**

This scheduling policy incorporates the simplicity of FIFO schedulers with the ability to provide different service classes. Incoming packets are classified as one of the static service priorities. This scheduling scheme retains the  $O(1)$  work complexity in that a packet is selected for transmission based only on the number of priority levels rather than the number of flows that are multiplexed together. One unique characteristic for PQ is that the packets in the lower priority queues will only be served after all the packets from higher priority queues are transmitted. As a result, only the queue with the highest priority will have similar delay bound as with the simple FIFO scheduler. The lower priority queues will have a delay bound that includes the delays incurred by the higher-

priority queues. In other words, the queues with lower priority will be prone to “bandwidth starvation” if the traffic rates for the higher priority queues are not controlled. There are two major types of PQ: preemptive and non-preemptive PQ. Preemption is defined as the service interruption of a lower-priority packet by an incoming higher-priority packet. Non-preemptive PQ scheme is more commonly used because of its relative implementation simplicity.

### **4.2.3 Weighted Fair Queuing (WFQ)**

The fair queuing (FQ) service discipline has been very popular because of its ability to provide per-flow delay guarantees. Although it has been seen from previous discussion that class-based flow-aggregated service guarantee has been chosen to provide QoS provisioning, per-flow guarantee is still required for applications that requires tight end-to-end delay bound. Most variant of FQ service discipline are comparable to the Generalized Processor Sharing (GPS) scheduler, constructed based on the idea of processor sharing. However, this scheduler is defined for a fluid traffic model. Since no fluid flows exist for a real network, Packetized Generalized Processor Sharing (PGPS) [58], better known as Weighted Fair Queuing (WFQ) is established for packet-based networks.

The basic idea of WFQ is as follows: a weight  $w_i$  is allocated for each flow  $i$ , where  $i = 1$  to  $N$ , and  $N$  is the total number of flows. The link capacity  $\gamma$  is shared among the flows with respect to their allocated weights. Thus, each flow  $i$  is guaranteed to have a minimum service rate of:

$$r_{\min}(i) = \frac{w_i}{\sum_{j=1}^N w_j} \gamma$$

A flow is considered as an active flow if there are packets in the queuing buffer waiting to be transmitted. When inactive flows exist, the unutilized bandwidth will be redistributed among the active flows with respect to their allocated weights, giving a service rate of:

$$r_{\text{active}}(i) = \begin{cases} \frac{w_i}{\sum_{j \in B(t)} w_j} \gamma & B(t) \in \text{Set of Active Flow} \\ 0 & \text{otherwise} \end{cases}$$

where  $B(t)$  is the set of active flows at time  $t$ .

The WFQ scheme is based on the notion of a global state variable called virtual time,  $V(t)$ . It is used to calculate the departure time, or the “virtual finish time” for each incoming packet. The WFQ scheduler will choose the packet with the smallest departure time to be transmitted on the output link. The complexity of the enqueueing and dequeuing operations are  $O(\log n)$ . If a packet has arrived as the  $k^{\text{th}}$  packet for flow  $i$ , the virtual start time  $s_i^k$  and virtual finish time  $f_i^k$  for this packet are:

$$s_i^k = \max\{V(a_i^k), f_i^{k-1}\}$$

$$f_i^k = s_i^k + \frac{l_i^k}{w_i}$$

where  $a_i^k$  denotes the arrival time and  $l_i^k$  denotes the packet length. The delay guarantee for flow  $i$  [59] is shown below:

$$\theta_{WFQ} \leq \frac{\rho_i}{r_{\min}(i)} + \sum_{h=1}^H \left( \frac{M_i}{r_{\min}(i)} + \frac{L_{\max}^h}{\gamma^h} \right)$$

where  $H$  is the number of nodes traversed by flow  $i$ ,  $M_i$  denotes the maximum packet length for flow  $i$ ,  $L_{\max}^h$  denotes the maximum packet length in link  $h$ , and  $\gamma^h$  denotes the link speed with  $h = 1$  to  $H$ . The term  $\frac{\rho_i}{r_{\min}(i)}$  is the burst delay, and the final term is the maximum waiting time for the transmission of a prior packet already in the output link  $h$ . This scenario is also known as the Head of Line (HoL) delay. Since the delay bound is independent of the number of connections  $n$ , WFQ is considered as one of the best queuing schemes for providing tight delay bounds.

Worst-case Fair WFQ (WF<sup>2</sup>Q) [55] is a variant of WFQ. It uses both the virtual start time and end time to perform in its scheduling mechanism. Unlike WFQ, when WF<sup>2</sup>Q chooses the next packet to be serviced, the server only selects from the set of packets that should have already started service by considering both the virtual start time and end time, rather than selecting the next packet with the smallest virtual end time from all the queues. Thus, although WF<sup>2</sup>Q has the same delay bound as WFQ, the scheduling mechanism is quite different, resulting in different service order. Therefore, WF<sup>2</sup>Q is a more accurate packet discipline that approximates the fluid FQ discipline.

Keshav [56] shows that only one packet per active flow will be contenting to be inserted into the sorted queue, resulting in  $O(\log n)$  work complexity. Fair queuing is “expensive” to implement at high-speed networks, since the sorting operation depends on number of flows  $n$ . Another disadvantage of WFQ is that the weight allocation is

proportion to the bandwidth required for each flow. As a result, a traffic flow that requires tight delay bound will also has high bandwidth allocated to it. However, low-delay flows for voice transmissions has low traffic throughput. Although some work has been done to decouple the relationship of the two components [60], more studies are required for further verification.

#### **4.2.4 Earliest Deadline First (EDF)**

The Earliest Deadline First (EDF) scheduler is a dynamic priority scheduler. It computes the departure deadline for incoming packets, and maintains a sorted list based on the packet deadlines to ensure transmission rate and delay guarantees. The key lies in the assignment of the deadline so that the server will provide a delay bound for the packets according to the SLA specified. Thus, the deadline for a packet can be defined as the summation of the packet's expected arrival time and the server's delay guarantee associated with the flow that the packet belongs to.

#### **4.2.5 Deficit Round Robin**

Both WFQ and EDF scheduling schemes require the processing of flow-specific state information such as the last transmission time of a packet. Furthermore, the reliance on a sorting operation that grows with the number of flows is a concern for scalability as speed increases. In general, scheduling mechanism that can provide tight delay guarantees are used in lower speed links. For high-speed networks, it is more desirable to have a simpler implementation for better scalability even if at the cost of some decrease in performance guarantees [61].

Deficit Round Robin (DRR) is a scheduling mechanism that is similar to the Weighted Round Robin (WRR) used by ATM networks. In WRR, each flow is served in a round-robin fashion in proportion to a weight assigned for each flow without considering the length of packets since ATM cells in all the flows have fixed length of 52 bytes. DRR is introduced to accommodate the variable-length packets in a packet-switching network. The work complexity of this round-robin serving is only  $O(1)$ . In addition to that, the service received by an active flow during its round of service is proportional to its fair share of the bandwidth specified by the weight for each flow.

In DRR, each flow  $i$  is allocated  $Q_i$  bits in each round of service, and  $Q_{\min}$  is defined as the minimum bit allocation value amongst all the flows. Thus, we will have:

$$Q_{\min} = \min(Q_i)$$

$$Q_i = w_i Q_{\min}$$

where  $w_i$  is the weight allocated for each flow. Active flows are placed in an *Active List*, serviced in a round-robin order. If a packet cannot to be completely serviced in a round of service without exceeding  $Q_i$ , it is deferred to the next service round, and the unused portion of  $Q_i$  in this round will be added to the next round, hence the name “Deficit” Round Robin. If a flow becomes inactive, the deficit will be dropped and not accumulated to the next round of service.

Although DRR is fair in terms of throughput, it lacks any reasonable delay bound. Another major disadvantage of DRR is that the delay bound for a flow with a small share of bandwidth can be very large. Assume a DRR scheduler with  $n$  flows, with all the flows active. If a low-weighting flow is located at the end of the *Active List*, it will have to wait



for all the other flows to be serviced before its turn, even if it is transmitting a minimum-sized packet. The delay bound of a flow  $i$  served by the DRR scheduler is [62]:

$$\theta_i^{DRR} \leq \frac{(W - w_i)M + (n-1)(m-1)}{\gamma} + (m-1) \left( \frac{1}{\sigma_i} + \frac{1}{\gamma} \right)$$

where  $W$  is the total weights of the active flows,  $M$  is the maximum size packet potentially serviced,  $m$  is the maximum size packet already serviced, and  $\sigma_i$  is the reserved rate for flow  $i$ . For WFQ, this will not have happened since the packets are scheduled using the virtual finish time.

In order to improve on the flow-service delay bound, a modified scheduling discipline called Nested Deficit Round Robin is proposed in [62]. For DRR, each flow receives its entire service in its round of service opportunity. However, Nested DRR splits this service into several rounds of service, reducing the waiting time for the flows with low weighting, while preserving a fair bandwidth allocation. All in all, it contains all the characteristics of DRR, but with a significantly lower flow-service delay bound.

#### 4.2.6 Non-work Conserving Service Discipline

While the service disciplines described above are all work conserving, non-work conserving service disciplines have begun to gain attention due to the performance guarantees required by real-time application. With work-conserving discipline, a server is never idle when there is any packet to be serviced from the input links. For non-work conserving discipline, the packets are only sent when they met with their *eligibility* time.

Non-work conserving service discipline can be expressed by a general class of discipline called rate-controlled service discipline [63]. It consists of two components, a

rate controller and a scheduler. The rate controller acts as the regulator or traffic shaper for the incoming packets by allocating an eligibility time for each of the incoming packet, much like the concept of virtual time of WFQ. The scheduler will then select from the list of packets that are eligible to be transmitted through some form of scheduling algorithm. The server will be idle unless some packets have become eligible for transmission.

This service discipline has seldom been studied in the past because performance parameters such as the average throughput and average delay for a network are catered for best-effort Internet traffic. It makes no sense to hold a packet in the queue even when there is idle resource to serve the packet. However, with the introduction of real-time applications, the performance parameters for this type of traffic have become the end-to-end delay bound and jitter in a networking environment. Thus, a packet may be held in the buffer even if the server is idle to reduce the end-to-end jitter.

### **4.3 Buffer Management Schemes**

While scheduling algorithms provide QoS guarantees to traffic flows by controlling the transmission opportunities that each individual flow gets, sufficient buffer space has to be allocated to hold the incoming packets, especially in high-speed networks. Some form of protection mechanism has to be implemented to provide flow *isolation* for preventing ill-behaving flows from occupying the entire queuing buffer. In addition to that, the mechanism is also required to make packet discard decisions based

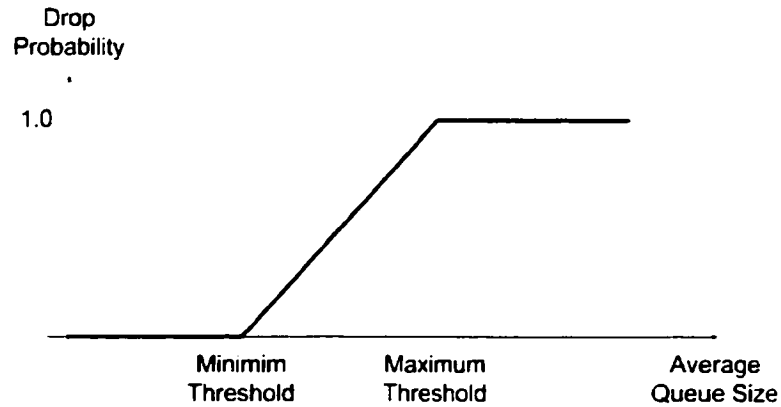
on the congestion level of the network. Thus, buffer management scheme is required to provide rate guarantees for a network.

### **4.3.1 Random Early Detection (RED)**

When FIFO queues experience congestion, they will discard the incoming packets that could not be hold in the buffer. This tail-drop policy will lead to two serious problems: global synchronization of TCP sessions and prolonged congestion in the network.

The Random Dearly Detection (RED) is a congestion avoidance mechanism. It overcomes the disadvantages of the tail-drop policy in FIFO queues by *randomly* dropping the packets when the average queue size exceeds a given minimum threshold. From statistical point of view, random packet drops is better than lots of packet being dropped at once when the queue buffer is full. RED works as a feedback mechanism to inform the TCP sessions form the source to anticipate congestion and reduced its transmission rate.

The drop probability for a packet is calculated based on the weight allocation on its flow, i.e. heavy flows will experience a larger number of dropped packets. The average queue size is computed using an exponentially weighted moving average so that the RED should not react to spontaneous transitions caused by bursty Internet traffic. When the average queue size exceeds the maximum threshold, all further incoming packets will be discarded. *Figure 4.1* shows the drop probability of a RED.



*Figure 4.1: The Drop Probability of a RED*

### 4.3.2 Multi-Level Random Early Discard (MRED)

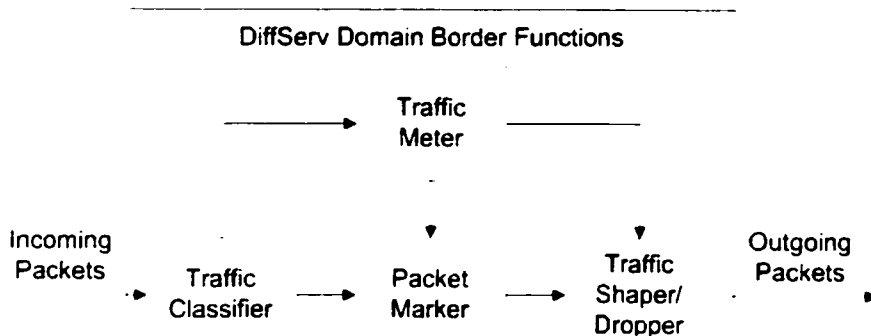
The Differentiated Service model (DiffServ) discussed in *Chapter 3* is the preferred method to provide QoS for the Internet. There are various methods proposed for extending the RED to support Assured Forwarding Per-Hop-Behavior (AF PHB) with different drop precedence levels. The Weighed RED (WRED) and RED with In/Out and Coupled Virtual Queues (RIO-C) are two popular MRED schemes.

Based on the information from the traffic classifier, the marker in a border DiffServ node is used to set the appropriate DiffServ Code Point (DSCP) for the AF class and its drop precedence. *Figure 4.2* shows the block diagram of a traffic conditioner for the DiffServ domain. The marker can be defined as a Two-Rate Three-Color Marker (trTCM), proposed in [64]. The packets with different drop precedence levels are marked as Green, Yellow or Red. The drop precedence of a packet is set based on two token buckets<sup>‡</sup> in the traffic meter: the Peak Information Rate (PIR) with its corresponding

---

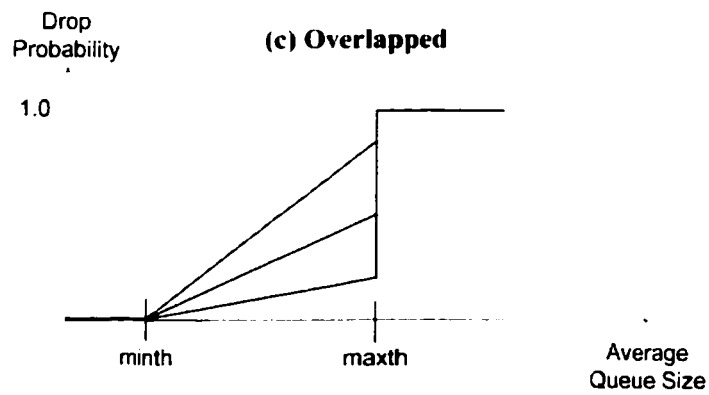
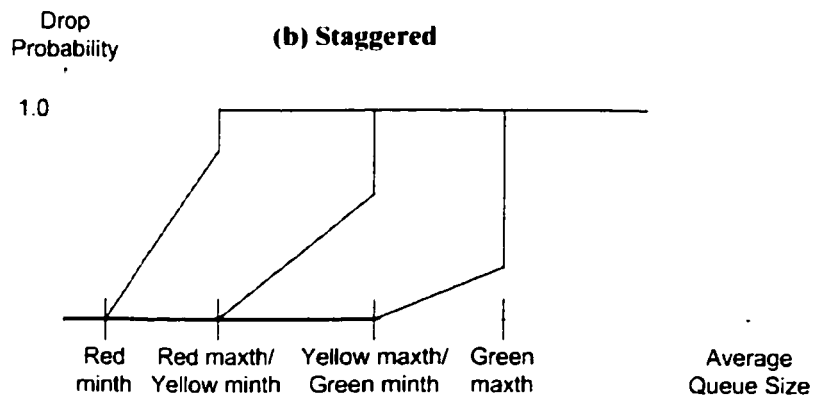
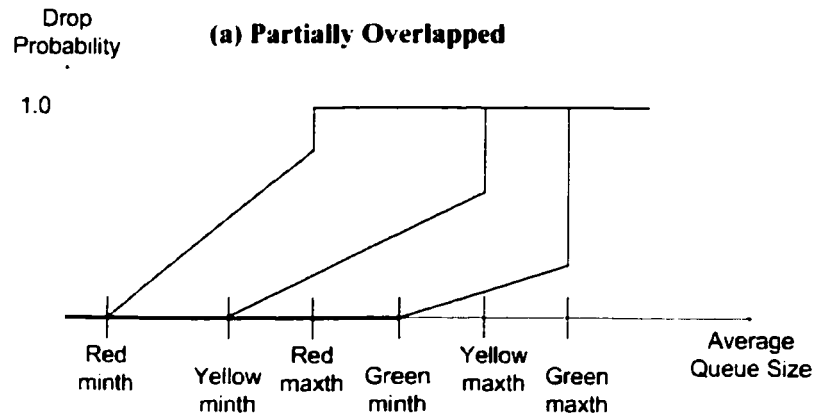
<sup>‡</sup> Please refer to *Section 4.4.3* for the detailed discussion of token buckets.

Peak Burst Size (PBS), and the Committed Information Rate (CIR) with its Committed Burst Size (CBS). A packet will be marked Red if it exceeds the PIR, otherwise it is marked Yellow or Green depending on whether it exceeds the CIR [65].



*Figure 4.2: Block Diagram of a Traffic Conditioner*

Traditional RIO contains two sets of configuration parameters to determine whether the packets are In-profile or Out-of-profile. For RED with In/Out and Coupled Virtual Queues (RIO-C), the average queue for packets of different colors are computed by adding its average queue to the average queue of the colors with lower drop precedence [66]. As an example, the average queue for *Red* will be computed using *Red*, *Yellow* and *Green* packets. Multiple RED parameters are maintained, one for each color. The parameters can be configured in the following methods: partially overlapped, overlapped and staggered. They are shown in *Figure 4.3*.



minth = Minimum Threshold      maxth = Maximum Threshold

*Figure 4.3: The parameter settings for MRED*

## 4.4 The MPLS Queuing Model

The combination of DiffServ and MPLS provides a very attractive strategy for providing end-to-end QoS in which DiffServ offers scalable service differentiation, while MPLS performs traffic engineering and efficient packet forwarding. Since end-to-end delay requirement is very stringent for voice traffic, voice packets have to be segregated from data packets to achieve prioritized queuing. *Figure 4.4* shows the queuing model that enables traffic segregation.

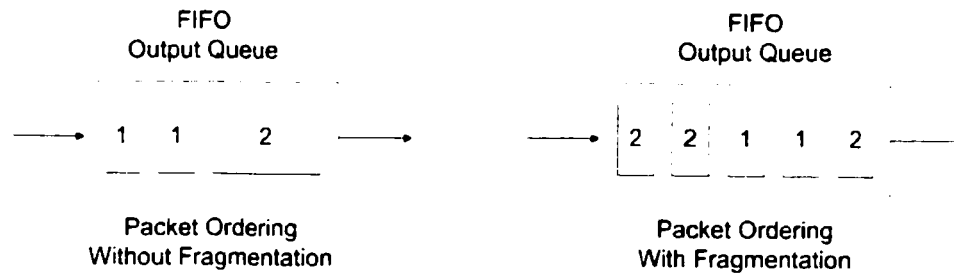


*Figure 4.4: MPLS Queuing Model Supporting Differentiated Services*

The model consists of a Priority Queue (PQ) to expedite the service for the delay-sensitive voice packets, and a Class-Based Weighted Fair Queuing (CB-WFQ) to provide service differentiation for the data packets. The Best Effort (BE) traffic is transmitted only when excess bandwidth is available. The model consists of the following modules:

### 4.4.1 Packet Fragmentation

Although Priority Queuing (PQ) is specifically reserved for voice packets, it is non-preemptive – the high-priority voice packets in the queue will still have to wait for the packet currently in transmission before they are serviced. As a result, there is still a blocking delay experienced by the voice packets, which is the “residual” transmission time of the big data packet. If the data packet is fragmented into smaller frames, the smaller voice packets will be able to interleave between the fragments of the large packets, reducing the blocking delay experienced in the initial scenario. *Figure 4.5* shows the packet ordering in the output queue with and without fragmentation.



*Figure 4.5: The Result of Large Packet Fragmentation*

There are two major reasons behind the packet fragmentation scheme:

1. The large packets are fragmented to ensure that the blocking delay will be minimal to meet the end-to-end delay requirements for voice packet transmission. For example, if the maximum blocking delay for a node with 56 kilobits per second (kbps) link speed is set to be 20 milliseconds (ms), the maximum fragmented data packet size to satisfy the delay requirements is 140 bytes.

$$Latency = \frac{Packet\ Size}{Link\ Speed}$$



$$\text{Fragmented Packet Size} = 20 \text{ ms} \times 56 \text{ kbps} = 140 \text{ bytes}$$

2. In addition to meeting the delay requirement, the fragmentation process will ensure that the voice packets are transmitted in a more regular fashion, without the long blocking delay caused by the background traffic. This will help in reducing jitter in the transmission of voice packets.

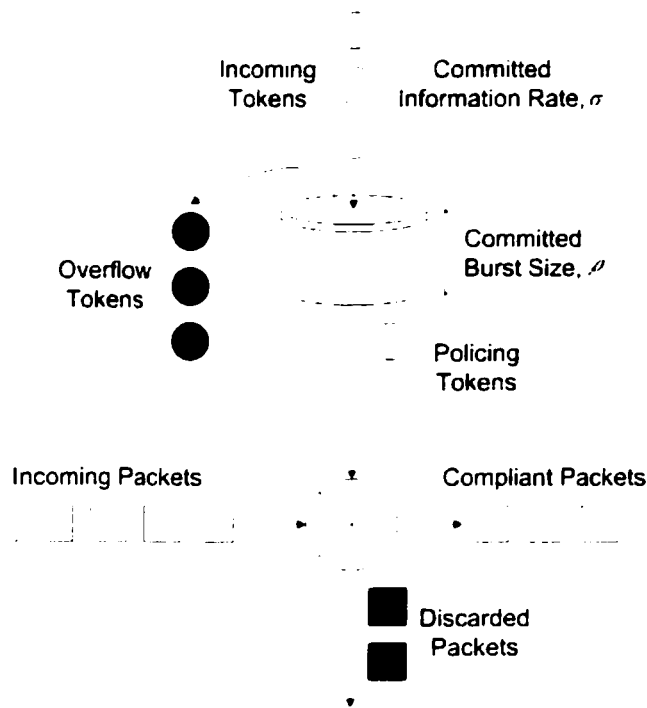
The idea of packet fragmentation to support voice transmission in a data packet network can be found in FRF.12 [67], a new standard set by the Frame Relay Forum to support Voice over Frame Relay (VoFR).

#### **4.4.2 Differentiated Service (DiffServ) Module**

The DiffServ traffic conditioner module shown in *Figure 4.2* can be found in an Ingress router of the network. Traffic flows are assigned with different levels of service priority based on the Per-Hop-Behavior (PHB) to provide Quality of Service (QoS) in the network. DiffServ Code Points (DSCP) is used to support this traffic classification. The three PHB defined are the expedited forwarding (EF), assured forwarding (AF) and best-effort forwarding (BE). Please refer to *Section 3.4.2: End-to-End QoS* for a detailed discussion of the functions of a DiffServ module at the edge and the core of the network.

#### **4.4.3 Traffic Management**

The token bucket scheme is used to provide traffic management functions such as metering, policing, and shaping. There are two key parameters in the token bucket scheme: Committed Info Rate (CIR),  $\sigma$  and Committed Burst Size (CBS),  $\rho$ . *Figure 4.6* shows a traffic policer based on the token bucket scheme.



*Figure 4.6 The Token Bucket Scheme*

The number of tokens corresponds to the amount of data that can be transmitted per interval. The size of the token bucket is equal to the Committed Burst Size,  $\rho$ . For each interval, the bucket is refilled with  $\sigma$  tokens, which represents the Committed Information Rate. Any unused tokens exceeding the maximum burst size  $\rho$  will be thrown away as overflow tokens. Thus, a flow is allowed to have a maximum burst size of  $\rho$ , but then must reduce its transmission rate to  $\sigma$ . Thus, the traffic profile for a flow  $i$  is given as  $(\sigma_i, \rho_i)$ .

#### **4.4.4 Priority Queuing with Traffic Policer**

In our queuing model, voice packets are aggregated and classified as the Expedited Forwarding (EF) PHB. This PHB has the highest service priority among all the

PHBs. It is good for transmitting traffic with low-latency and low-jitter requirements. Bandwidth reserved for the EF flows cannot be preempted by any other flow to guarantee transmission performance. As a result, the PQ scheduling scheme is selected to service the packets with EF PHB.

There are two different varieties in PQ: strict priority and alternating priority. The drawback for a strict priority PQ is that under heavy traffic loads, the high-priority flow might consume the entire bandwidth of the queuing system, causing prolonged congestion for the lower-priority flows. The alternating priority PQ allows the packets from the lower-priority flows to interleave between the higher-priority packets. When combined with packet fragmentation, it is able limit the jitter experienced by the EF traffic flow.

The function of the traffic policer is to monitor the profile of the incoming traffic on a per-flow basis, and ensures that the traffic flow conforms to the profile that is specified for that particular flow. It is a rate-limiting device used for traffic metering purposes. Any “extra” traffic will be discarded by the policer.

#### **4.4.5 Class-Based Weighted Fair Queuing with MRED**

This queuing algorithm utilizes weights as a measurement of bandwidth allocation to different queues serviced by the scheduler. It guarantees a minimum service rate to all the queues based on the weight assigned for each queue. Assured forwarding (AF) and best-effort (BE) PHB traffic classification will be directed to this scheduler.

AF PHB in an MPLS network differentiates the traffic into three classes, each with a two-level drop precedence. Multi-level Random Early Detection (MRED) is used

as the congestion avoidance module for this class-based WFQ. Please refer to *Section 4.3.2* for a detailed discussion on how DiffServ and MRED perform traffic classifications and marking for the AF PHB.

The BE PHB is the default forwarding mechanism. The weight allocated to the BE PHB as one of the WFQ can be zero if there are EF and AF PHB traffic to be serviced. In other words, the performance of the BE PHB depends on the excess bandwidth availability in a network. On the other hand, a small weight can always be allocated to it to assure a guaranteed minimum service rate.

#### **4.5 Queuing Model Analysis**

The thesis focuses on the performance analysis of voice transmission in an MPLS network. From the queuing model shown in *Figure 4.4*, we can simplify the model to a two-level Priority Queuing (PQ) system shown in *Figure 4.7*. The voice packets will be treated under EF PHB using the higher-priority queue, whereas the packets with AF and BE PHB will be lumped together as the “background traffic” using the lower-priority queue. From this simplified model, we will be able to analyze the performance parameters for the voice packets having EF PHB.

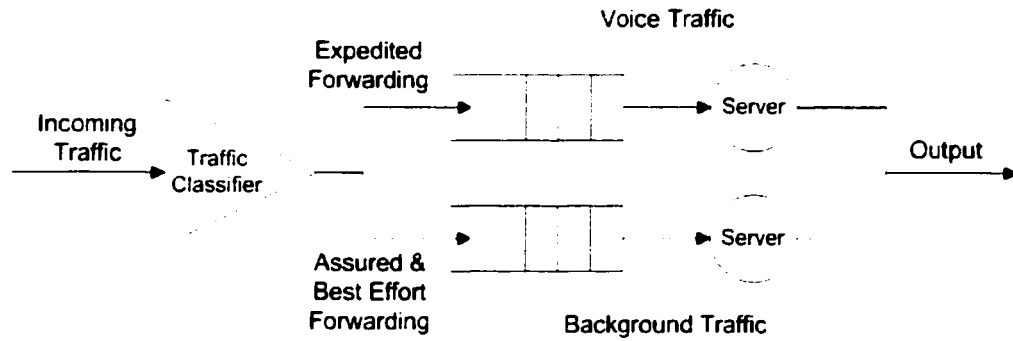


Figure 4.7: Two-level Priority Queuing Model for Voice Performance Analysis

### 4.5.1 Notations and Assumptions

Before we proceed with the analysis of the queuing model, this section provides a list of notations used in the later part of the analysis.

- $T$  : Average total waiting time in system
- $N$  : Average number of packets in system
- $W_Q$  : Average waiting time in queue
- $N_Q$  : Average number of packets in queue
- $P_B$  : Probability of blocking for a queue
- $P_n$  : Probability of  $n$  packets in the system
- $\lambda$  : Poisson arrival rate
- $\mu$  : Poisson service rate or departure rate
- $\rho$  : Utilization factor for the system

#### 4.5.1.1 Queuing System Classification

Queuing systems are classified in the  $a/b/m/K$  notation, where  $a$  denotes arrival process,  $b$  the service time distribution,  $m$  the number of servers and  $K$  the maximum

occupancy in the queuing system. For example, M/M/1/K corresponds to a queuing system with exponentially distributed inter-arrival time ( $M$ )<sup>§</sup>, exponentially distributed service time ( $M$ ), single server ( $1$ ), and at most  $K$  occupancies in the system.

#### 4.5.1.2 Queuing Analysis Assumptions

There are a few assumptions made for the queuing analysis:

1. The packets arrive according to a Poisson process of  $\lambda$  with exponentially distributed inter-arrival times.
2. The service times are independent and exponentially distributed with Poisson process of  $\mu$ .
3. Inter-arrival times and service times are independent with each other
4. The incoming packets for a system are homogeneous
5. The system is *memoryless* – the additional time required to complete a service in progress is independent of the time the service has gone through

#### 4.5.1.3 Markov Chain Formulation

Let  $N_k$  represent the number of customers in system at time  $k\delta$ , where  $\delta$  is a small interval length. Thus, the number of arrivals and departures in any interval length  $\delta$  is  $\lambda\delta$  and  $\mu\delta$  respectively. Using the notation from [69], the transition probability within a small interval length  $\delta$  in a Markov chain is defined as:

$$P_{i,j} = P\{N_{k+1} = j \mid N_k = i\}$$

---

<sup>§</sup> The notation  $M$  is used for exponential distribution because it leads to a Markov process model [4.21].

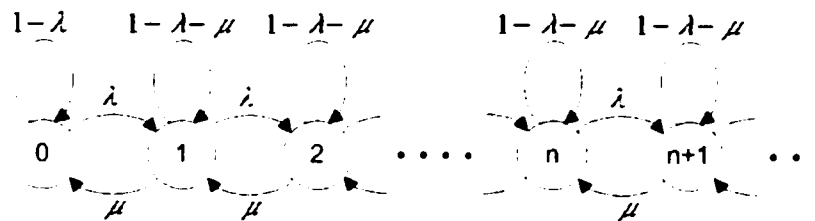
Transition of  $N_k$  is triggered by arrivals and departure. It can only happen between neighboring states. The transition probabilities for different cases are listed below:

$$P_{i,i+1} = P\{1 \text{ arrival}, 0 \text{ departure}\} = \lambda\delta$$

$$P_{i,i-1} = P\{0 \text{ arrival}, 1 \text{ departure}\} = \mu\delta$$

$$P_{i,i} = P\{0 \text{ arrival}, 0 \text{ departure}\} = (1 - \lambda\delta) \times (1 - \mu\delta) = 1 - \lambda\delta - \mu\delta$$

The term  $\lambda\mu\delta^2$  is taken out from the self-transition probability  $P_{i,i}$  because the term  $\delta^2$  is very small compares to the other terms in the equation. A Markov Chain transition diagram is shown in *Figure 4.8*. The notation  $\delta$  is taken out from the figure for clarity.



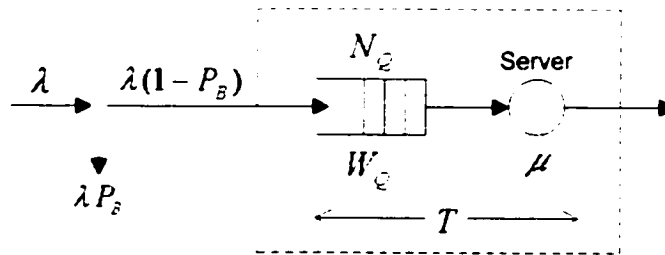
*Figure 4.8: The Markov Chain for an M/M/1 Queueing System*

### 4.5.2 Performance Analysis of the Queueing Model

In order for us to derive the performance parameters for the queueing model, we will start with the analysis of an M/M/1/K queueing model. This model represents the single-server FIFO scheduler used by the “background traffic”. Next, we will perform the analysis for the M/M/c/K queueing model, which represents a c-server FIFO scheduler to minimize the latency for voice packets transmission. The analysis is done with the help from the references [68], [69], [70] and [71]. The simplified model shown in *Figure 4.7* will be analyzed as a two-level non-preemptive priority queueing system.

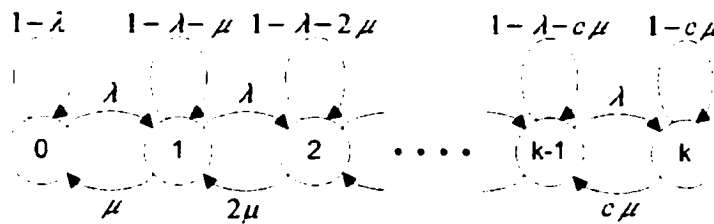
### 4.5.2.1 M/M/1/K Queuing System

The block diagram for an M/M/1/K queuing system is shown in *Figure 4.9*. The maximum number of occupancies for the queuing system is  $K$ . Any packet that arrives after the  $K^{\text{th}}$  packet will be dropped or blocked. Thus, we can say that the blocking probability  $P_B$  is equal to the probability of the queuing system in the state with maximum occupancy,  $P_K$ .



*Figure 4.9: Block Diagram of the M/M/1/K Queuing System*

The steady-state probability states that for any given time interval, the total number of transitions from state  $n$  to  $n+1$  must differ from the total number of transitions from  $n+1$  to  $n$  by at most 1 [69]. As an example,  $\lambda P_0 = \mu P_1$ , where  $P_0$  and  $P_1$  represent the probability of zero and one packet in the system respectively. The Markov chain for the M/M/1/K queuing model is shown in *Figure 4.10*.



*Figure 4.10: The Markov Chain for an M/M/1/K Queuing System*



The utilization factor for this queuing model is defined as  $\rho = \frac{\lambda}{\mu}$ , where  $\rho = 1 - P_0$ . The

global balance equations for the steady-state probabilities are:

$$\lambda P_0 = \mu P_1$$

$$\lambda P_1 = \mu P_2$$

$$\lambda P_n = \mu P_{n-1}$$

This leads to the following equations:

$$P_2 = \frac{\lambda}{\mu} P_1 = \frac{\lambda}{\mu} \left( \frac{\lambda}{\mu} P_0 \right) = \left( \frac{\lambda}{\mu} \right)^2 P_0 = \rho^2 P_0$$

$$P_n = \frac{\lambda}{\mu} P_{n-1} = \frac{\lambda}{\mu} \left( \left( \frac{\lambda}{\mu} \right)^{n-1} P_0 \right) = \left( \frac{\lambda}{\mu} \right)^n P_0 = \rho^n P_0$$

From the above equation, we are able to determine the probability of  $n$  number of packets in the system if we know  $P_0$ , which is found from the boundary condition:

$$\sum_{n=0}^K P_n = 1.$$

$$\sum_{n=0}^K \rho^n P_0 = 1$$

$$P_0 = \frac{1}{\sum_{n=0}^K \rho^n} = \frac{1 - \rho}{1 - \rho^{K+1}}$$

Assuming that the system is stable when  $\rho = \frac{\lambda}{\mu} < 1$ , we will obtain  $P_n = \frac{\rho^n (1 - \rho)}{1 - \rho^{K+1}}$ .

Furthermore, using the  $P_0$  result, we are able to obtain the following formula:

Probability of blocking or loss:  $P_B = P\{Loss\} = P_K = \rho^K P_0$

Probability of queuing or server is busy:  $P\{Queuing\} = P\{n \geq 1\} = 1 - P_0$

The average number of customer in system can be found as:

$$\begin{aligned}
 N &= \sum_{n=0}^K nP_n = \sum_{n=0}^K n\rho^n (1 - \rho) = \rho(1 - \rho) \sum_{n=0}^K n\rho^{n-1} \\
 &= \rho(1 - \rho) \frac{\partial}{\partial \rho} \left( \sum_{n=0}^K \rho^n \right) = \rho(1 - \rho) \frac{\partial}{\partial \rho} \left( \frac{1 - \rho^{K+1}}{1 - \rho} \right) \\
 N &= \begin{cases} \frac{\rho}{1 - \rho} - \frac{(K+1)\rho^{K+1}}{1 - \rho^{K+1}} & \rho \neq 1 \\ \frac{K}{2} & \rho = 1 \end{cases}
 \end{aligned}$$

Little's Formula [71] relates the average waiting time in the system,  $T$  to the arrival rate,  $\lambda$  and the average number of packets in system,  $N$  with the equation  $N = \lambda T$ . Since we have found  $N$ , we can derive the following equations:

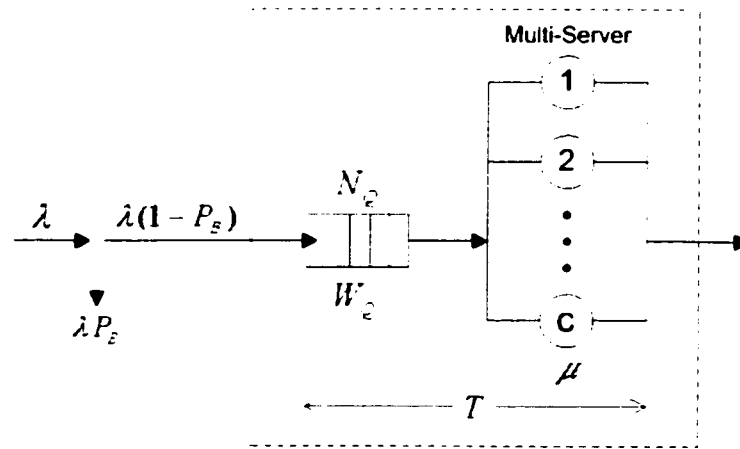
Average delay time in system:  $T = \frac{N}{\lambda'}$ , where  $\lambda' = \lambda(1 - P_B)$

Average waiting time in queue:  $W_Q = T - \frac{1}{\mu}$

Average number of customers in queue:  $N_Q = \lambda' W_Q$

### 4.5.2.2 M/M/c/K Queuing System

The  $M/M/c/K$  queuing model is very similar to the  $M/M/1/K$  model, with the difference in number of servers in the system. The block diagram for an  $M/M/c/K$  queuing system is shown in *Figure 4.11*.



*Figure 4.11: Block Diagram of the M/M/c/K Queuing System*

The  $M/M/c/K$  queuing model has the following characteristics:

$$\lambda_n = \begin{cases} \lambda & 1 \leq n < K \\ 0 & n \geq K \end{cases} \quad (\text{packet blocked or dropped}) \quad (\text{Equation 4.1})$$

$$\mu_n = \begin{cases} n\mu & 0 \leq n < c \\ c\mu & c \leq n \leq K \end{cases} \quad (\text{Equation 4.2})$$

where  $\lambda_n$  and  $\mu_n$  are the Poisson arrival and departure rates for  $n$  number of packets in the system. The Markov chain for the  $M/M/c/K$  queuing model is shown in *Figure 4.12*.

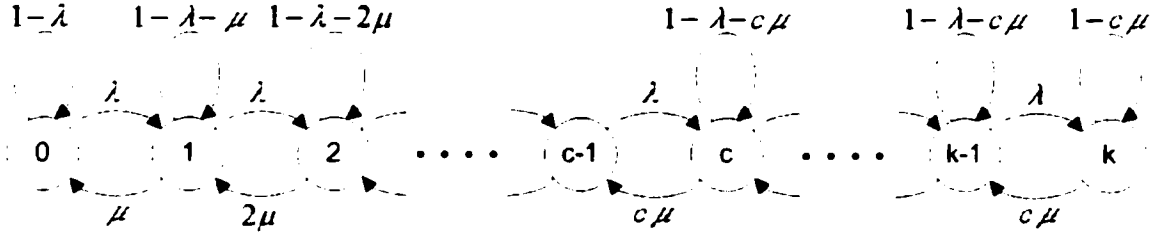


Figure 4.12: The Markov Chain for an M/Mc/K Queuing System

The global balance equations for the steady-state probabilities are:

$$\lambda P_{n-1} = \begin{cases} n\mu P_n & 1 \leq n < c \\ c\mu P_n & c \leq n \leq K \end{cases}$$

We will look at both of the cases in order to derive the probability of  $n$  number of packets in the system,  $P_n$ . For the case of  $1 \leq n < c$ :

$$\lambda P_0 = 1\mu P_1$$

$$\lambda P_1 = 2\mu P_2$$

$$\lambda P_2 = 3\mu P_3$$

$$\lambda P_{n-1} = n\mu P_n$$

This leads to the following equations:

$$P_2 = \frac{1}{2} \frac{\lambda}{\mu} P_1 = \frac{1}{2} \frac{\lambda}{\mu} \left( \frac{\lambda}{\mu} P_0 \right) = \frac{1}{2!} \left( \frac{\lambda}{\mu} \right)^2 P_0$$

$$P_3 = \frac{1}{3} \frac{\lambda}{\mu} P_2 = \frac{1}{3} \frac{\lambda}{\mu} \left( \frac{1}{2!} \left( \frac{\lambda}{\mu} \right)^2 P_0 \right) = \frac{1}{3!} \left( \frac{\lambda}{\mu} \right)^3 P_0$$

$$P_n = \frac{1}{n} \frac{\lambda}{\mu} P_{n-1} = \frac{1}{n} \frac{\lambda}{\mu} \left( \frac{1}{(n-1)!} \left( \frac{\lambda}{\mu} \right)^{n-1} P_0 \right) = \frac{1}{n!} \left( \frac{\lambda}{\mu} \right)^n P_0 \quad \text{for } 1 \leq n < c$$

From the equations  $\lambda_n$  and  $\mu_n$  specified in (1) and (2) respectively, we will obtain  $P_n$  for the case of  $c \leq n \leq K$  :

$$P_n = \frac{1}{c!} \left( \frac{\lambda}{\mu} \right)^c \frac{1}{c^{n-c}} \left( \frac{\lambda}{\mu} \right)^{n-c} P_0 = \frac{1}{c! c^{n-c}} \left( \frac{\lambda}{\mu} \right)^n P_0 \quad \text{for } c \leq n \leq K$$

Given that the utilization factor for an M/M/c/K queuing model is  $\rho = \frac{\lambda}{c\mu}$ , we are able to

reduce the formula for  $P_n$  as:

$$P_n = \begin{cases} P_0 \frac{(c\rho)^n}{n!} & 0 \leq n < c \\ P_0 \frac{c^c \rho^n}{c!} & c \leq n < K \end{cases}$$

One of the parameters that is of interest for an M/M/c/K queuing model is the probability that an arriving packet finds all  $c$  servers busy and has to wait in the queue. This probability is also known as the *Erlang C* formula and is derived below:

$$\begin{aligned} P\{\text{Queuing}\} &= P\{n \geq c\} = \sum_{n=c}^K P_n = \sum_{n=c}^K \frac{1}{c! c^{n-c}} \left( \frac{\lambda}{\mu} \right)^n P_0 \\ &= \sum_{n=c}^K \frac{c^c \rho^n}{c!} P_0 = \frac{c^c \rho^c}{c!} P_0 \sum_{n=c}^K \rho^{n-c} \end{aligned}$$

Let  $m = n - c$ ,

$$\begin{aligned} &= \frac{(c\rho)^c}{c!} P_0 \sum_{m=0}^{K-c} \rho^m \\ &= P_0 \frac{(c\rho)^c}{c!} \frac{1 - \rho^{K-c+1}}{1 - \rho} \end{aligned}$$

In order to determine the average number of packets in queue,  $N_Q$ , we are only dealing with  $P_n$ 's that are in the region  $c \leq n \leq K$ :

$$N_Q = \sum_{m=0}^{K-c} m P_{m+c} = \sum_{n=c}^K (n-c) P_n = \sum_{n=c}^K (n-c) \frac{1}{c! c^{n-c}} \left( \frac{\lambda}{\mu} \right)^n P_0$$

$$= \sum_{n=c}^K (n-c) \frac{c^c \rho^n}{c!} P_0$$

Let  $m = n - c$ ,

$$= P_0 \frac{c^c}{c!} \sum_{m=0}^{K-c} m \rho^{m+c}$$

$$= P_0 \frac{c^c \rho^c}{c!} \sum_{m=0}^{K-c} m \rho^m$$

$$= P_0 \frac{(c\rho)^c}{c!} \rho \sum_{m=1}^{K-c} m \rho^{m-1}$$

$$= P_0 \frac{(c\rho)^c}{c!} \rho \left[ \sum_{m=0}^{K-c} m \rho^{m-1} - 1 \right]$$

$$= P_0 \frac{(c\rho)^c}{c!} \rho \frac{\partial}{\partial \rho} \left[ \frac{1 - \rho^{K-c+1}}{1 - \rho} - 1 \right]$$

$$= P_0 \rho \frac{(c\rho)^c}{c!} \frac{1}{(1-\rho)^2} \left[ (1 - \rho^{K-c+1}) - (1 - \rho)(K - c + 1) \rho^{K-c} \right]$$

As with the case for  $M/M/1/K$  queuing model, the boundary condition is used to find  $P_0$ :

$$\sum_{n=0}^K P_n = 1$$

$$\sum_{n=0}^{c-1} \frac{1}{n!} \left( \frac{\lambda}{\mu} \right)^n P_0 + \sum_{n=c}^K \frac{1}{c^{n-c} c!} \left( \frac{\lambda}{\mu} \right)^n P_0 = 1$$

$$P_0 = \left[ \sum_{n=0}^{c-1} \frac{1}{n!} \left( \frac{\lambda}{\mu} \right)^n + \sum_{n=c}^K \frac{1}{c^{n-c} c!} \left( \frac{\lambda}{\mu} \right)^n \right]^{-1}$$

Assuming that the system is stable for  $\rho = \frac{\lambda}{c\mu} < 1$ , we will obtain the following formula:

$$P_0 = \left[ \sum_{n=0}^{c-1} \frac{(c\rho)^n}{n!} + \frac{(c\rho)^c}{c!} \frac{1 - \rho^{K-c+1}}{1 - \rho} \right]^{-1}$$

Since we have found  $N_Q$ , we can use Little's Formula to derive the following equations:

Average waiting time in queue:  $W_Q = \frac{N_Q}{\lambda'}$ , where  $\lambda' = \lambda(1 - P_B)$

Average delay time in system:  $T = W_Q + \frac{1}{c\mu}$

Average number of customers in system:  $N = \lambda' T = \lambda' \left( W_Q + \frac{1}{c\mu} \right) = N_Q + \rho(1 - P_B)$

#### 4.5.2.3 Non-preemptive Priority Queuing System

For priority queuing (PQ) systems, arriving packets are divided into  $x$  different priority classes. Each priority class is serviced by a separate FIFO queue, with the priority level set as Priority Class 1 > Priority Class 2 > ... > Priority Class  $x$ . "Non-preemptive" means the packet currently in service is not interrupted even if a customer of higher priority arrives at the system. From the two-level PQ queuing model presented in *Figure 4.7*, the  $M/M/c/K$  model is applied for Priority Class 1 voice traffic while the  $M/M/1/K$  model is for the Priority Class 2 background traffic. This queuing model is shown in *Figure 4.13*.

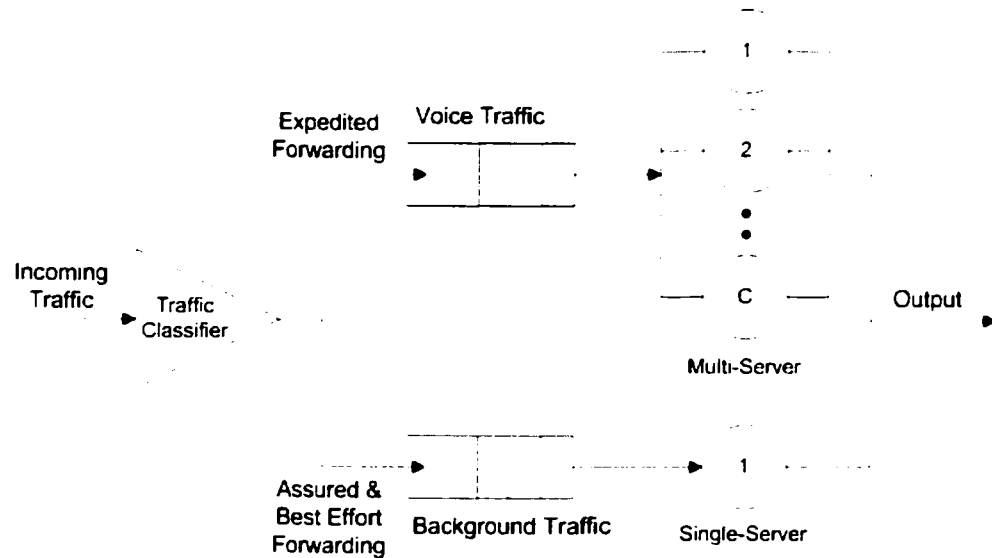


Figure 4.13: Non-preemptive Priority Queuing Model

Before we proceed with the analysis of this queuing model, a summary of the notations used is listed below:

$\lambda_x$  = Arrival rate for priority  $x$

$\mu_x$  = Departure rate for priority  $x$

$\rho_x$  = Utilization factor for priority  $x$

$N_Q^x$  = Average number in queue for priority  $x$

$W_Q^x$  = Average queuing time for priority  $x$

$R$  = Mean residual service time (mean waiting time for a higher priority packet for the current customer in service)

The following assumptions have been made for the analysis of this non-preemptive priority queuing:

1. The system is stable, i.e., the overall system utilization is  $\rho_1 + \rho_2 + \dots + \rho_x < 1$



2. All the servers in the system are identical, i.e. they have the same service rate.
3. The incoming packets are homogeneous, i.e. they are of the same size.

The average waiting time in queue for an incoming  $i^{\text{th}}$  Priority Class 1 packet is taken as the summation of the residual service time of the current packet in service and the total transmission time required for the packets in queue that arrived prior to this  $i^{\text{th}}$  packet. The residual service time  $R$  is bounded by the transmission time of a maximum size packet. The equation is derived below:

$$W_Q^1 = \text{Residual Service Time, } R +$$

$$(\text{Num of packets in Queue} \times \text{Service Rate of Queue})$$

$$W_Q^1 = R + N_Q^1 \frac{1}{c\mu_1}$$

From Little's Formula, let  $N_Q^1 = \lambda_1 W_Q^1$

$$W_Q^1 = R + \rho_1 W_Q^1, \text{ where } \rho_1 = \frac{\lambda_1}{c\mu_1}$$

$$W_Q^1 = \frac{R}{1 - \rho_1}$$

The average waiting time in queue for an incoming  $i^{\text{th}}$  Priority Class 2 packet is more complex. In addition to the residual service time for the current packet in transmission and the total transmission time required for the packets already in Class 2 queue, we need to take into account the additional delay due to packets with higher priority. This delay is composed of the total transmission time required for the packets in

Class 1 queue and the additional queuing delay due to Class 1 packets that arrive while the  $i^{\text{th}}$  Class 2 packet is waiting in queue. Thus, the equation is derived as:

$$W_Q^2 = R + N_Q^2 \frac{1}{\mu_2} + N_Q^1 \frac{1}{c\mu_1} + \text{Additional queuing delay due to packet of higher priority that arrive while Class 2 packet is waiting}$$

$$W_Q^2 = R + N_Q^2 \frac{1}{\mu_2} + N_Q^1 \frac{1}{c\mu_1} + \lambda_1 W_Q^2 \frac{1}{c\mu_1}$$

$$W_Q^2 = R + \rho_2 W_Q^2 + \rho_1 W_Q^1 + \rho_1 W_Q^2$$

$$W_Q^2 = \frac{R + \rho_1 W_Q^1}{1 - \rho_1 - \rho_2} = \frac{R(1 - \rho_1) + \rho_1 R}{(1 - \rho_1 - \rho_2)(1 - \rho_1)} = \frac{R}{(1 - \rho_1 - \rho_2)(1 - \rho_1)}$$

where  $\rho_1 = \frac{\lambda_1}{c\mu_1}$  for Class 1  $M/M/c/K$  queue and  $\rho_2 = \frac{\lambda_2}{\mu_2}$  for Class 2  $M/M/1/K$  queue.

## 4.6 Theoretical Results

Theoretical queuing delay calculations for voice and data traffic flows are done based on the equations derived in this chapter and the listed parameters for each of the different simulation test cases in the following chapter. As a result, these analytical results will be presented along with OPNET simulation results later in *Chapter 5*.

## **Chapter 5 OPNET Simulation Results**

OPNET (Optimum Network Performance) is a software tool that provides a comprehensive development environment supporting the modeling of communication networks and protocols. The MPLS queuing model is implemented using the OPNET simulation environment. Based on the model library provided by OPNET, an MPLS network is constructed to analyze the end-to-end queuing delay of the voice traffic traveling in the network. This simulation can be divided into three stages. First, a G.723.1 voice model with silent suppression is implemented to act as our voice source. Next, a simple node model is implemented to examine the relationship between the Expedited Forwarding (EF) voice traffic and the Best-Effort (BE) data traffic under different performance parameters. Lastly, we implement an MPLS network topology to examine the queuing delay of the voice traffic based on various testing parameters.

### **5.1 Modeling of Voice Sources**

Voice over IP refers to real-time transmission of digitized voice streams in packet networks. The modeling of a voice source is important to our simulation design. In a conversation, voice activities are alternating between two states: talk spurt and silence period. This model is known as the ON-OFF voice model, shown in *Figure 5.1*. Coded voice packets are generated during ON periods, while no packets are transmitted in the OFF periods. This can be done through the audio silent detector (VAD) discussed in the previous chapter.

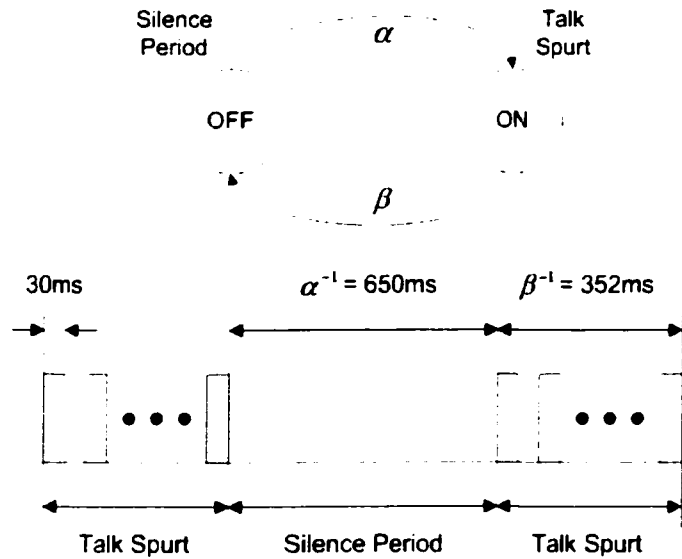


Figure 5.1: ON-OFF Voice Model

The talk spurts and the silence periods of a voice source are exponentially distributed [72]. In other words, the duration of a voice source in the *ON* state will have a Poisson distributed mean of  $\alpha^{-1}$ , whereas the *OFF* state has a Poisson distributed mean of  $\beta^{-1}$ . As a result, the transitional rate from the *OFF*-to-*ON* state and *ON*-to-*OFF* state will be  $\alpha$  and  $\beta$  respectively.

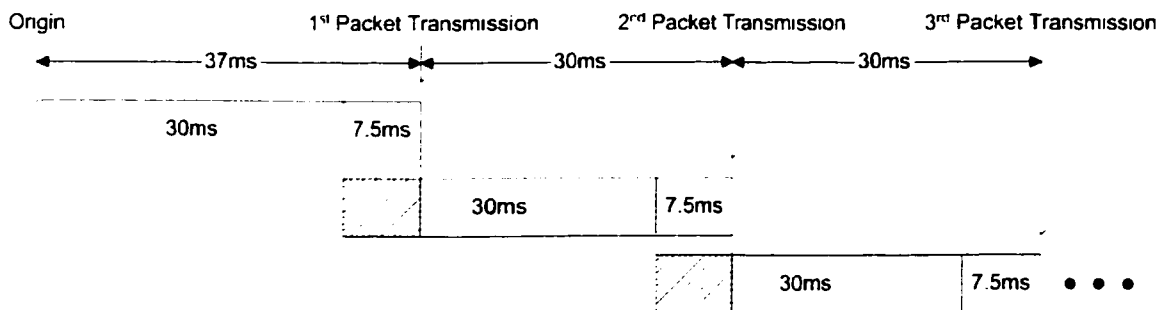


Figure 5.2: The Look-ahead Delay of the G.723.1 Codec

In our modeling, the voice source is encoded with 5.3 *kbps* G.723.1 source coder with silent suppression. This codec uses speech frames of 30 ms and encodes the speech samples into 20-byte coded voice blocks. This codec requires an extra look-ahead delay

of 7.5 ms, giving it a total processing delay of 37.5 ms. *Figure 5.2* shows that the 20-byte coded blocks are generated periodically with a 30 ms interval, except for the first code block. Thus, the packet generation rate during the *ON* state can be considered as 30 ms. The exponentially distributed mean values commonly used for talk spurt and silence period are  $\alpha^{-1} = 352$  ms and  $\beta^{-1} = 650$  ms respectively [73][74]. The summary of the parameters of the G.723.1 voice model can be found in *Table 5.1*.

*Table 5.1: Simulation Parameters of G.723.1 Voice Model*

<i>Parameters</i>	<i>Values</i>
Codec rate	5.3 kbps
Coded voice block	20 bytes
Packet inter-arrival time during talk spurt	30 ms
Codec look-ahead delay	7.5 ms
Average length of talk spurt ( $\alpha^{-1}$ )	352 ms
Average length of silence period ( $\beta^{-1}$ )	650 ms
RTP header size	40 bytes

### 5.1.1 OPNET Voice Model

OPNET provides a 3-layer modeling hierarchy. The highest layer is the network domain, where system topologies are defined. The second layer is the node domain, where node architectures such as the ON-OFF voice model and various queuing systems are defined. The third layer is the process model, where the control flow of a node model is defined in the form of Finite State Machine (FSM). *Figure 5.3* shows the states transitional diagram of the voice model. The *Init* state is used to initialize the parameters for the voice source and other user-specific statistics gathering. The *OFF* state represents

the silence periods in a voice conversation, with an exponentially distributed mean value of  $\beta^{-1} = 650$  ms. No packets are generated in this state. The *ON* state represents the talk spurts, with an exponentially distributed mean value of  $\alpha^{-1} = 352$  ms. Coded voice blocks are generated with a constant 30 ms interval while in the *ON* state. Finally, the *Stop* state will be activated at the end of the simulation.

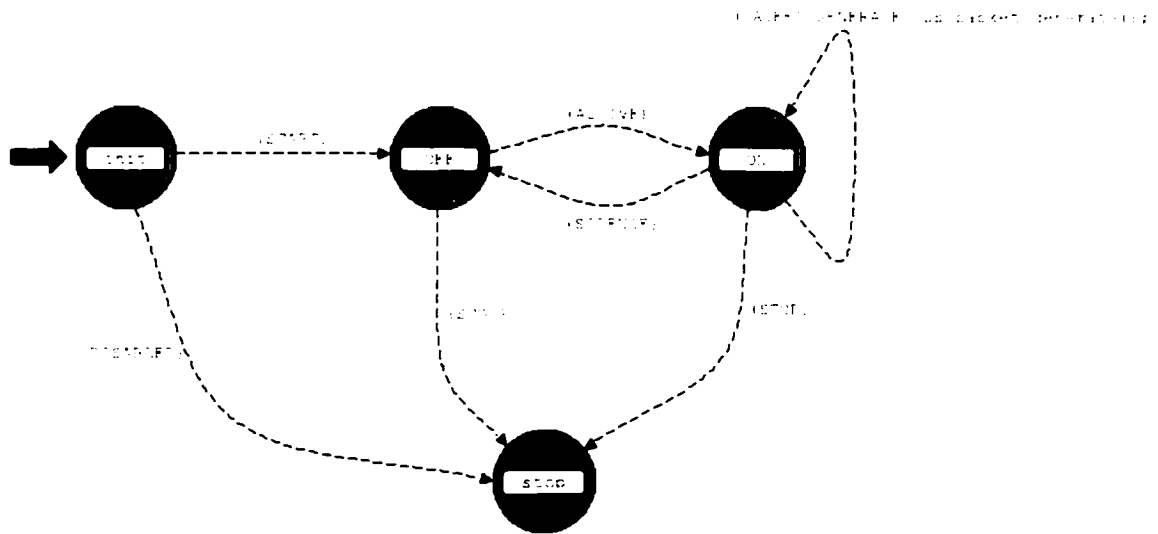


Figure 5.3: Process Model of the ON-OFF Voice Model

### 5.1.2 Notations and Abbreviations

This section provides a list of notations and abbreviations used in this chapter.

*PHB* : Per-Hop Behavior

*EF* : Expedited Forwarding PHB for voice traffic

*BE* : Best-Effort Forwarding PHB for data traffic

$\lambda_{ON}$  : Poisson arrival rate during talk spurt

$\lambda_{EF}$  : Average Poisson arrival rate for voice traffic

$\mu$  : Poisson service rate or departure rate

- $\rho$  : Utilization factor for the system
- $C$  : Server capacity or line rate
- $N$  : Number of EF Streams
- $\tau_x$  : Average service time
- $\tau_x^2$  : Second moment of service time
- $L_{EF}$  : EF packet size or length
- $L_{BE}$  : BE packet size or length
- $BW_{EF}$  : EF traffic bandwidth consumption
- $BW_{BE}$  : BE traffic bandwidth consumption
- $\lambda_x$  : Arrival rate for priority  $x$
- $\mu_x$  : Departure rate for priority  $x$
- $\rho_x$  : Utilization factor for priority  $x$
- $N_Q^x$  : Average number in queue for priority  $x$
- $W_Q^x$  : Average queuing time for priority  $x$
- $R$  : Mean residual service time

### 5.1.3 ON-OFF Voice Model

From the parameters listed in *Table 5.1*, we are able to determine the following:

$$EF \text{ Packet Size} = RTP \text{ Header} + G.723 \text{ Voice Packet} = 480bits$$

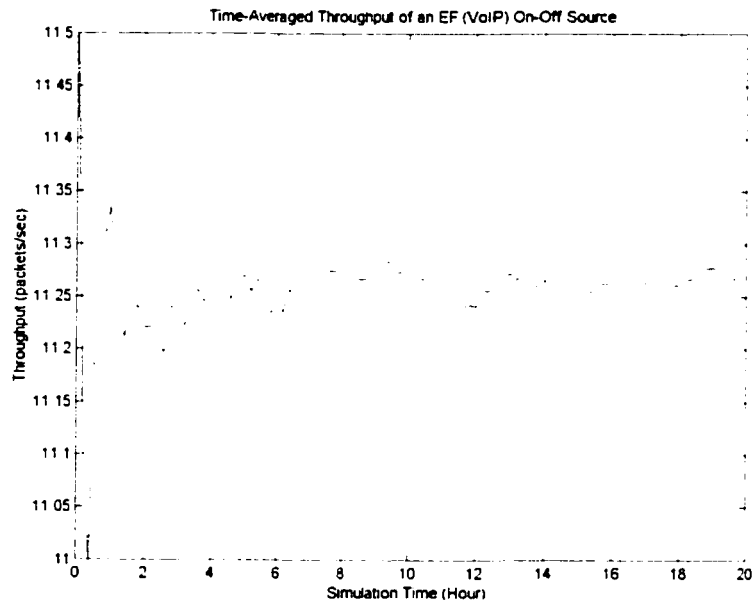
$$\lambda_{OV} = \frac{1}{30ms} \approx 27 \text{ packets / sec}$$

Thus, the average Poisson arrival rate during a voice conversation is:

$$\lambda_{EF} = \frac{0.352ms}{0.352ms + 0.65ms} \times 27 \text{ packets/sec} \approx 11.7 \text{ packets/sec}$$

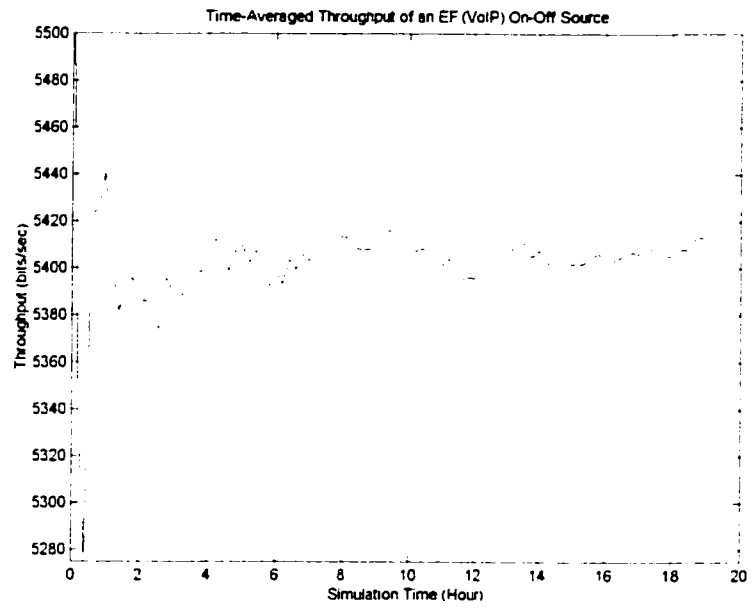
$$EF \text{ Traffic Bandwidth} = 11.7 \text{ packets/sec} \times 480 \text{ bits/packet} \approx 5.6 \text{ kbps}$$

The ON-OFF voice model is implemented in OPNET based on the process model shown in *Figure 5.3*. The model is simulated for duration of 20 hours. The time-average throughput obtained for the voice model is plotted in *Figure 5.4* and *Figure 5.5*, and they match with the analytical results above.



*Figure 5.4: Time-Averaged Throughput (packets/sec) for the ON-OFF Voice Model*



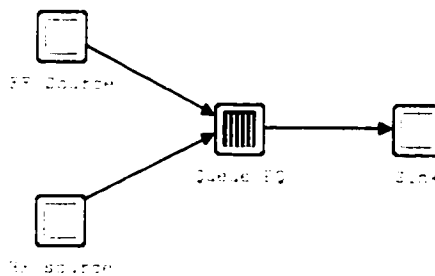


*Figure 5.5: Time-Averaged Throughput (bits/sec) for the ON-OFF Voice Model*

## 5.2 Node Model Simulation

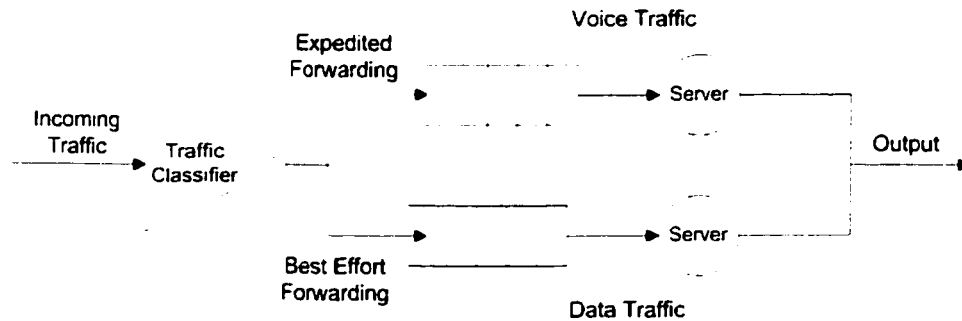
### 5.2.1 Analysis of EF Traffic with Different BE Traffic Parameters

A simple node model is implemented to examine the performance parameters for the Expedited Forwarding (EF) voice traffic under different Best-Effort (BE) data traffic conditions. This node model, shown in *Figure 5.6*, is consists of the following modules:



*Figure 5.6: Node Model for Simulation*

- i) EF\_Source – This is the node model for generating EF voice traffic. The parameters for this module are summarized in *Table 5.1*.
- ii) BE\_Traffic – This is the node model for generating BE data traffic. It is just a simple source that generates constant-sized packets with an exponentially distributed inter-arrival time.
- iii) Queue\_PQ – This is the node model for a two-level non-preemptive priority queuing system shown in *Figure 5.7*. The voice packets will be served under EF PHB using the higher-priority queue, whereas the packets with BE PHB will be using the lower-priority queue. From this model, we will be able to analyze the performance parameters for the voice packets.



*Figure 5.7: Two-level Priority Queuing Model*

- iv) Sink – This node model is used to gather statistics and “destroy” the packets generated from both of the EF and BE sources.

### 5.2.1.1 Simulation Assumptions

In order to simulate the environment to analyze the EF voice traffic under different network constraints, we need to modify the parameters for the EF and BE sources for each test case. The following are the assumptions for the simulations in this section:

1. The BE packet size is selected to be 600 bytes. Based on the Internet data collected by the Measurement & Operations Analysis Team from the National Library for Applied Network Research project during the month of February, 2001 [75], the size of the packets falls into two distinct regions: under 600 bytes and around 1500 bytes. We made the packet size selection based on the assumption that the transmission line has a lower maximum transfer unit.
2. The total line utilization is set at 80%.
3. Based on the previous assumption, the BE traffic load is set to be the total line utilization subtract the corresponding EF traffic load.

4. The queuing system is assumed to have infinite queuing buffer.
5. The BE traffic has constant packet size, which is set as the maximum packet size permitted by the packet segmentation module to simulate worst-case scenarios.

### 5.2.1.2 Test Case: Different BE Packet Sizes

The testing parameters to analyze the performance of the EF traffic under different BE packet sizes are listed in *Table 5.2*.

*Table 5.2: Test Parameters for the BE Packet Size Test*

Transmission Line		Expedited Forwarding			Best-Effort Forwarding
Line Rate (kbps)	Utilization Factor (%)	Num of Streams	Packet Size (bits)	EF Load (kbps)	Packet Size (bits)
64	80	1	480	5.62	Variable*

\* The BE packet sizes are selected to be 1500, 1000, 500, 100 bytes respectively

Given that the line rate,  $C$  is  $64kbps$ , we are able to derive the following:

$$\mu_1 = \frac{C}{L_{EF}} = \frac{64kbps}{480bits} = 133.3\bar{3} \text{ packets / sec}$$

$$\mu_2 = \frac{C}{L_{BE(\text{variable})}}$$

Given that the average arrival rate of EF traffic  $\lambda_1 = 11.7 \text{ packets / sec}$ ,

$$BW_{EF} = \lambda_1 \times L_{EF} = 11.7 \times 480 = 5.6kbps$$

Since we have assumed the total link utilization is 80% of line rate,

$$BW_{BE} = (80\% \times C) - BW_{EF}$$

Thus, the average arrival rate of BE traffic is:

$$\lambda_2 = \frac{BW_{BE}}{L_{BE(\text{variable})}}.$$

From the non-preemptive priority queuing analysis presented in *Section 4.5.2.3*, we obtained the following average queuing delay formula:

$$\text{EF PHB: } W_Q^1 = \frac{R}{1 - \rho_1}, \text{ where } \rho_1 = \frac{\lambda_1}{\mu_1}$$

$$\text{BE PHB: } W_Q^2 = \frac{R}{(1 - \rho_1 - \rho_2)(1 - \rho_1)}, \text{ where } \rho_2 = \frac{\lambda_2}{\mu_2}$$

Since we are using constant packets for both the EF and BE traffic with infinite queuing buffer, the queuing analysis will be performed under the M/D/1 queuing model [76][77].

Given that average service time  $\tau = \frac{1}{\mu}$ , the second moment of average service time is

found to be  $\tau^2 = \frac{1}{\mu^2}$ . The residual service time,  $R$  for an M/D/1 queuing system is:

$$R = \frac{\lambda \tau^2}{2}, \text{ where } \lambda = \lambda_1 + \lambda_2 \text{ and } \tau^2 = \frac{\lambda_1}{\lambda} \tau_1^2 + \frac{\lambda_2}{\lambda} \tau_2^2.$$

The overall end-to-end queuing delay for our node model is  $W_Q + \frac{1}{\mu}$ . The theoretical

queuing delay calculations for EF and BE flows are listed in *Table 5.3*.

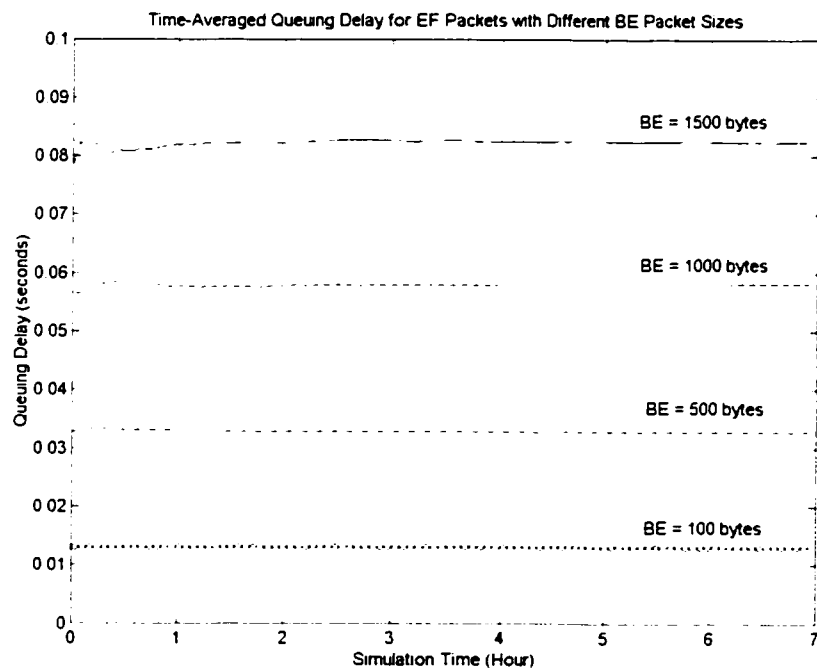
Table 5.3: Theoretical Queuing Delay Calculations for EF and BE Traffic Flows

Link Speed ( <i>bps</i> )	64000
Utilization Factor	80%

EF Bandwidth Utilization ( <i>bps</i> )	5620	<i>All unspecified units in seconds</i>	
EF Packet Size ( <i>bits</i> )	480	EF Transmission Delay	0.0075
		EF Number of Packets	11.710
		EF Inter-arrival Time	0.030

BE Bandwidth Utilization ( <i>bps</i> )	45580	<i>All unspecified units in seconds</i>			
Different BE Packet Size ( <i>bits</i> )	12000	BE Transmission Delay	0.188	EF Queuing Delay	0.081
		BE Number of Packets	3.798	BE Queuing Delay	0.555
		BE Inter-arrival Time	0.263		
	8000	BE Transmission Delay	0.125	EF Queuing Delay	0.057
		BE Number of Packets	5.670	BE Queuing Delay	0.371
		BE Inter-arrival Time	0.176		
	4800	BE Transmission Delay	0.075	EF Queuing Delay	0.037
		BE Number of Packets	9.500	BE Queuing Delay	0.223
		BE Inter-arrival Time	0.105		
	4000	BE Transmission Delay	0.0625	EF Queuing Delay	0.032
		BE Number of Packets	11.395	BE Queuing Delay	0.186
		BE Inter-arrival Time	0.088		
800	BE Transmission Delay	0.0125	EF Queuing Delay	0.013	
	BE Number of Packets	56.974	BE Queuing Delay	0.039	
	BE Inter-arrival Time	0.0175			

The BE packet size directly affects the residual service time experienced by the EF priority queue. Thus, we observe from *Figure 5.8* that the time-averaged end-to-end queuing delay increases with BE packet size. As a result, the packet fragmentation module in the queuing model is used to constrain the residual packet delay caused by the BE packets so that the EF packet stream will meet the delay requirements for voice packet transmission.



*Figure 5.8: Queuing Delay for EF Packets with Different BE Packet Sizes*

### 5.2.1.3 Test Case: Different BE Load %

The testing parameters to analyze the performance of the EF traffic under different BE loads are listed in *Table 5.4*.

*Table 5.4: Test Parameters for the BE Load Test*

Transmission Line		Expedited Forwarding			Best-Effort Forwarding	
Line Rate (kbps)	Utilization Factor (%)	Num of Streams	Packet Size (bits)	EF Load (kbps)	Packet Size (bits)	BE Load (% Link Rate)
64	80	1	480	5.62	4800	Variable*

\* The BE loads are selected to be 80%, 70%, 60%, 50%, 40% of line rate respectively.

Given that the line rate,  $C$  is  $64kbps$ , we are able to derive the following:

$$\mu_1 = \frac{C}{L_{EF}} = \frac{64kbps}{480bits} = 133.3\bar{3} \text{ packets / sec}$$

$$\mu_2 = \frac{C}{L_{BE}} = \frac{64kbps}{4800bits} = 13.3\bar{3} \text{ packets / sec}$$

Given that the average arrival rate of EF traffic  $\lambda_1 = 11.7 \text{ packets / sec}$ ,

$$BW_{EF} = \lambda_1 \times L_{EF} = 11.7 \times 480 = 5.6kbps$$

Since we have assumed the total link utilization is 80% of line rate,

$$BW_{BE(\text{variable})} = BE \text{ Load \%} \times C$$

Thus, the average arrival rate of BE traffic is:

$$\lambda_2 = \frac{BW_{BE(\text{variable})}}{L_{BE}}$$



Then, we are able to calculate  $\rho_1$ ,  $\rho_2$ , and  $R$ , which leads to the derivation of  $W_Q^1$  and  $W_Q^2$ .

The theoretical queuing delay calculations for EF and BE flows are listed in *Table 5.5*.

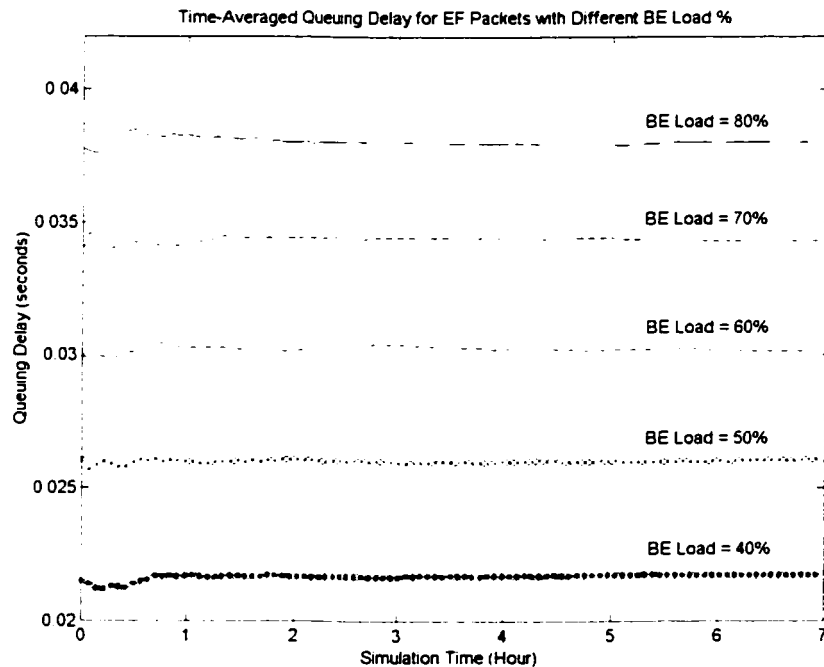
Table 5.5: Theoretical Queuing Delay Calculations for EF and BE Traffic Flows

Link Speed ( <i>bps</i> )	64000
BE Packet Size ( <i>bits</i> )	4800

EF Bandwidth Utilization ( <i>bps</i> )	5620	<i>All unspecified units in seconds</i>	
EF Packet Size ( <i>bits</i> )	480	EF Transmission Delay	0.0075
		EF Number of Packets	11.710
		EF Inter-arrival Time	0.030

BE Transmission Delay ( <i>sec</i> )	0.075	<i>All unspecified units in seconds</i>			
Link Utilization	80%	BE Bandwidth ( <i>bps</i> )	45579	EF Queuing Delay	0.037
		BE Number of Packets	9.496	BE Queuing Delay	0.223
		BE Inter-arrival Time	0.105		
	70%	BE Bandwidth ( <i>bps</i> )	39179	EF Queuing Delay	0.033
		BE Number of Packets	8.162	BE Queuing Delay	0.160
		BE Inter-arrival Time	0.123		
	60%	BE Bandwidth ( <i>bps</i> )	32779	EF Queuing Delay	0.029
		BE Number of Packets	6.829	BE Queuing Delay	0.129
		BE Inter-arrival Time	0.512		
	50%	BE Bandwidth ( <i>bps</i> )	26379	EF Queuing Delay	0.025
		BE Number of Packets	5.496	BE Queuing Delay	0.110
		BE Inter-arrival Time	0.181		
40%	BE Bandwidth ( <i>bps</i> )	19979	EF Queuing Delay	0.021	
	BE Number of Packets	4.162	BE Queuing Delay	0.097	
	BE Inter-arrival Time	0.240			

The BE load, in terms of percentages of line rate, affects the arriving frequency of the BE packet. A higher utilization factor for BE traffic increases the contention probability of the BE packets, which in turn affects the residual service time experienced by the EF priority queue. We observe from *Figure 5.9* that the time-averaged end-to-end queuing delay increases with BE load percentage. This is not a good controlling parameter to achieve the required delay requirements for voice packet transmission, since service providers want to maximize their bandwidth utilization.



*Figure 5.9: Queuing Delay for EF Packets with Different BE Loads*

### 5.2.1.4 Test Case: Different Link Speed

The testing parameters to analyze the performance of the EF traffic under different link speeds are listed in *Table 5.6*.

*Table 5.6: Test Parameters for the Link Speed Test*

Transmission Line		Expedited Forwarding			Best-Effort Forwarding
Line Rate (kbps)	Utilization Factor (%)	Num of Streams	Packet Size (bits)	EF Load (kbps)	Packet Size (bits)
Variable*	80	1	480	5.62	4800

\* The link speeds are selected to be *64kbps, 256kbps, 512kbps, 1.54Mbps* respectively.

Given that the line rate,  $C$  is a variable, we are able to derive the following parameters:

$$\mu_1 = \frac{C_{(variable)}}{L_{EF}}$$

$$\mu_2 = \frac{C_{(variable)}}{L_{BE}}$$

Given that the average arrival rate of EF traffic  $\lambda_1 = 11.7 \text{ packets / sec}$ ,

$$BW_{EF} = \lambda_1 \times L_{EF} = 11.7 \times 480 = 5.6 \text{ kbps}$$

Since we have assumed the total link utilization is 80% of line rate,

$$BW_{BE} = (80\% \times C_{(variable)}) - BW_{EF}$$

Thus, the average arrival rate of BE traffic is:

$$\lambda_2 = \frac{BW_{BE}}{L_{BE}}$$

Then, we are able to calculate  $\rho_1, \rho_2$ , and  $R$ , which leads to the derivation of  $W_Q^1$  and  $W_Q^2$ .

The theoretical queuing delay calculations for EF and BE flows are listed in *Table 5.7*.

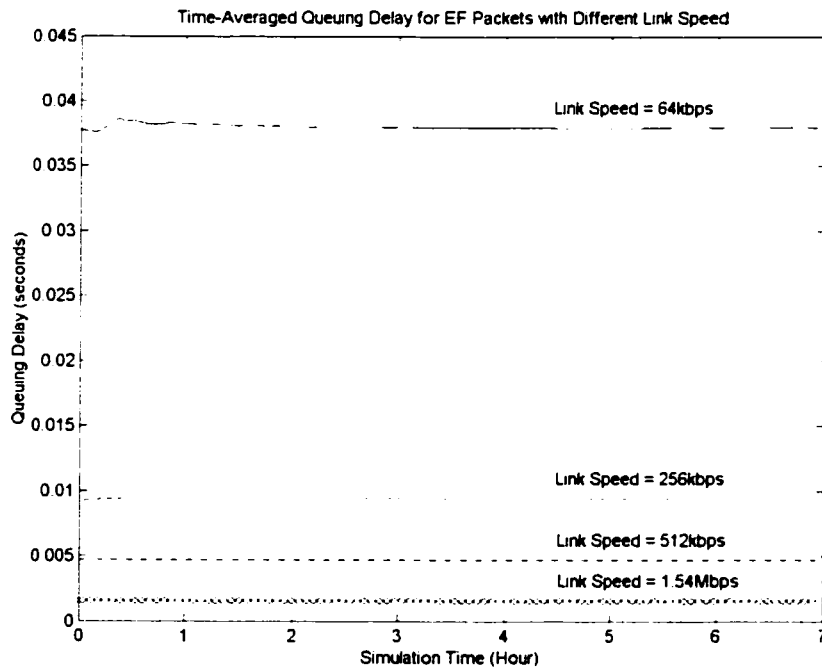
Table 5.7: Theoretical Queuing Delay Calculations for EF and BE Traffic Flows

Link Utilization	80%
BE Packet Size ( <i>bits</i> )	4800

EF Bandwidth Utilization ( <i>bps</i> )	5620	<i>All unspecified units in seconds</i>	
EF Packet Size ( <i>bits</i> )	480	EF Transmission Delay	0.0075
		EF Number of Packets	11.710
		EF Inter-arrival Time	0.030

Different BE Link Speeds		<i>All unspecified units in seconds</i>			
Modem ( <i>kbps</i> )	64	BE Bandwidth ( <i>bps</i> )	45579	EF Queuing Delay	0.0371
		BE Transmission Delay	0.075	BE Queuing Delay	0.2232
		BE Number of Packets	9.496		
		BE Inter-arrival Time	0.105		
Fractional T1 ( <i>kbps</i> )	256	BE Bandwidth ( <i>bps</i> )	199.2	EF Queuing Delay	0.0094
		BE Transmission Delay	0.019	BE Queuing Delay	0.0561
		BE Number of Packets	41.49		
		BE Inter-arrival Time	0.022		
Fractional T1 ( <i>kbps</i> )	512	BE Bandwidth ( <i>bps</i> )	403.9	EF Queuing Delay	0.0047
		BE Transmission Delay	0.009	BE Queuing Delay	0.0281
		BE Number of Packets	84.16		
		BE Inter-arrival Time	0.012		
Full T1 ( <i>Mbps</i> )	1.536	BE Bandwidth ( <i>bps</i> )	1223.2	EF Queuing Delay	0.0016
		BE Transmission Delay	0.003	BE Queuing Delay	0.0094
		BE Number of Packets	254.8		
		BE Inter-arrival Time	0.004		

The link speeds are selected based on the bandwidth commonly offered by the service providers – dial-up modem with 64kbps, fractional T1 line with 256kbps and 512kbps, and full T1 line with 1.54Mbps. Link speed directly affects the service rate and the transmission delay experienced by the packets. We observe from *Figure 5.10* that time-averaged end-to-end queuing delay decreases with higher link speed. Even though the queuing delay is reduced by half with the doubling of the link speed, it does not drop significantly with higher link speeds. This is because the reduction in delay has become a very small portion of the overall end-to-end queuing delay. As a result, we can remove the packet fragmentation module in high-speed networks, since it only plays a prominent role in networks with low link speed.



*Figure 5.10: Queuing Delay for EF Packets with Different Link Speeds*

### 5.2.1.5 Test Case: Different EF Load %

The testing parameters to analyze the performance of the EF traffic under different EF loads are listed in *Table 5.8*.

*Table 5.8: Test Parameters for the Link Speed Test*

Transmission Line		Expedited Forwarding			Best-Effort Forwarding
Line Rate (kbps)	Utilization Factor (%)	Num of Streams	Packet Size (bits)	EF Load (% Link Rate)	Packet Size (bits)
256	80	Variable*	480	Variable*	4800

\* The EF loads are selected to be 10%, 20%, 30%, 50% of link rate respectively, with the number of EF flows chosen to be 5, 9, 14, 18, 23 to control the EF load.

Given that the line rate,  $C$  is  $256\text{kbps}$ , we are able to derive the following parameters:

$$\mu_1 = \frac{C}{L_{EF}} = \frac{256\text{kbps}}{480\text{bits}} = 533.3\bar{3} \text{ packets / sec}$$

$$\mu_2 = \frac{C}{L_{BE}} = \frac{256\text{kbps}}{4800\text{bits}} = 53.3\bar{3} \text{ packets / sec}$$

Given the number of EF streams  $N$  and the average arrival rate of EF traffic  $\lambda_1 = 11.7 \text{ packets / sec}$ ,

$$BW_{EF(\text{variable})} = N_{(\text{variable})} \times \lambda_1 \times L_{EF} = 11.7 \times 480 = 5.6\text{kbps}$$

Since we have assumed the total link utilization is 80% of line rate,

$$BW_{BE} = (80\% \times C) - BW_{EF}$$

Thus, the average arrival rate of BE traffic is:

$$\lambda_2 = \frac{BW_{BE}}{L_{BE}}$$



Then, we are able to calculate  $\rho_1, \rho_2$ , and  $R$ , which leads to the derivation of  $W_Q^1$  and  $W_Q^2$ .

The theoretical queuing delay calculations for EF and BE flows are listed in *Table 5.9*.

*Table 5.9: Theoretical Queuing Delay Calculations for EF and BE Traffic Flows*

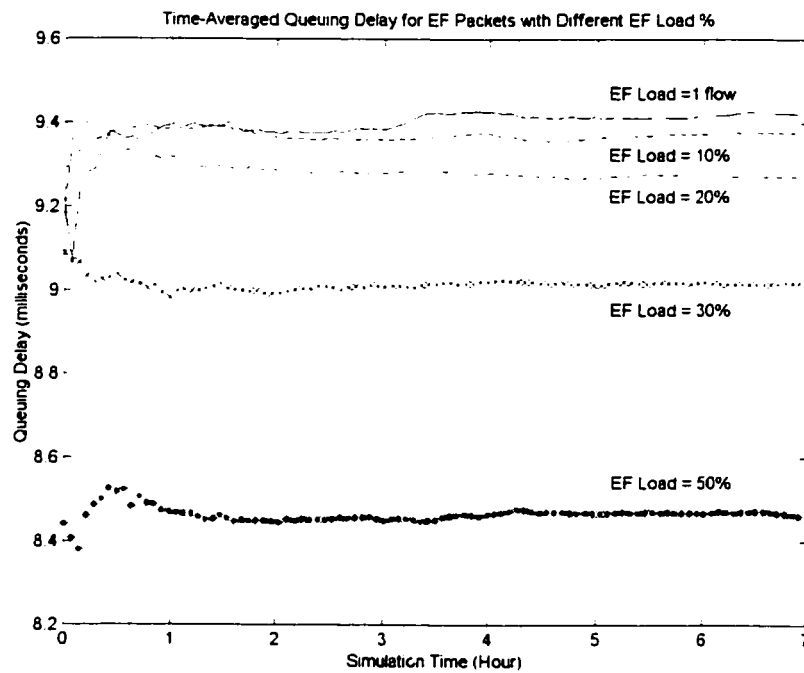
Link Speed ( <i>kbps</i> )	256
Link Utilization	80%

BE Traffic Parameters		EF Traffic Parameters	
BE Packet Size ( <i>bits</i> )	4800	EF Packet Size ( <i>bits</i> )	480
BE Transmission Delay ( <i>sec</i> )	0.019	EF Inter-arrival Time	0.03

EF Traffic Load		<i>All unspecified units in seconds</i>			
# Voice Flow = 5	10%	EF Number of Packets	58.55	EF Queuing Delay	0.0093
		EF Bandwidth ( <i>kbps</i> )	28.10	BE Queuing Delay	0.0556
		BE Number of Packets	36.81		
		BE Bandwidth ( <i>kbps</i> )	176.7		
# Voice Flow = 9	20%	EF Number of Packets	105.39	EF Queuing Delay	0.0091
		EF Bandwidth ( <i>kbps</i> )	50.59	BE Queuing Delay	0.0551
		BE Number of Packets	32.13		
		BE Bandwidth ( <i>kbps</i> )	154.21		
# Voice Flow = 14	30%	EF Number of Packets	163.94	EF Queuing Delay	0.0089
		EF Bandwidth ( <i>kbps</i> )	78.69	BE Queuing Delay	0.0541
		BE Number of Packets	26.27		
		BE Bandwidth ( <i>kbps</i> )	126.11		
# Voice Flow = 18	40%	EF Number of Packets	210.77	EF Queuing Delay	0.0087
		EF Bandwidth ( <i>kbps</i> )	101.17	BE Queuing Delay	0.0532
		BE Number of Packets	21.59		
		BE Bandwidth ( <i>kbps</i> )	103.63		

# Voice Flow = 23	50%	EF Number of Packets	269.33	EF Queuing Delay	0.0084
		EF Bandwidth (kbps)	129.28	BE Queuing Delay	0.0515
		BE Number of Packets	15.73		
		BE Bandwidth (kbps)	75.52		

Since the total line utilization is set at 80%, increasing the EF traffic load will cause the arriving frequency of the BE packets to decrease correspondingly. This will cause the residual service time of the EF traffic, imposed by the BE packets, to decrease. Thus, we observe from *Figure 5.11* that the time-averaged end-to-end queuing delay decreases with higher EF traffic load.



*Figure 5.11: Queuing Delay for EF Packets with Different EF Loads*

### 5.2.1.6 Test Case: Different EF Packet Sizes

The testing parameters to analyze the performance of the EF traffic under different EF packet sizes are listed in *Table 5.10*.

*Table 5.10: Test Parameters for the Link Speed Test*

Transmission Line		Expedited Forwarding			Best-Effort Forwarding
Line Rate (kbps)	Utilization Factor (%)	Num of Streams	Packet Size (bits)	EF Load (% Link Rate)	Packet Size (bits)
256	80	1	Variable*	Variable*	4800

\* The EF packet sizes are selected to be 480, 640, 1984, 3648, 6976 bits respectively.

EF load will be changing due to the different packet sizes.

Given that the line rate,  $C$  is  $256kbps$ , we are able to derive the following parameters:

$$\mu_1 = \frac{C}{L_{EF \text{ (variable)}}}$$

$$\mu_2 = \frac{C}{L_{BE}} = \frac{256kbps}{4800bits} = 53.33 \text{ packets / sec}$$

Given that the average arrival rate of EF traffic  $\lambda_1 = 11.7 \text{ packets / sec}$ ,

$$BW_{EF} = \lambda_1 \times L_{EF \text{ (variable)}}$$

Since we have assumed the total link utilization is 80% of line rate,

$$BW_{BE} = (80\% \times C) - BW_{EF}$$

Thus, the average arrival rate of BE traffic is:

$$\lambda_2 = \frac{BW_{BE}}{L_{BE}}$$

Then, we are able to calculate  $\rho_1, \rho_2$ , and  $R$ , which leads to the derivation of  $W_Q^1$  and  $W_Q^2$ .

The EF packet sizes are chosen based on the following packetization schemes [78]:

Scheme A: RTP header + 1 coded voice block

Scheme B: RTP header + 2 coded voice blocks

Scheme C: RTP header + multiplexing of  $M$  coded voice blocks

For Scheme A and B, a single voice packet stream is allocated for each voice flow (or voice source). Scheme A is inefficient due to the overhead of the uncompressed RTP header, where the 480-bit voice packet only carries one coded voice block. Scheme B reduces the overhead of the header through the encapsulation of 2 consecutive coded voice blocks in one voice packet, resulting in 640-bit packets. For Scheme C, multiple voice flows are multiplexed together and form the payload of a voice packet. However, this multiplexing method requires an additional Multilink Point-to-Point Protocol (MLPPP) packet header for each coded voice block to identify its voice flow. *Figure 5.12* shows the packet format for each of the scheme discussed above.

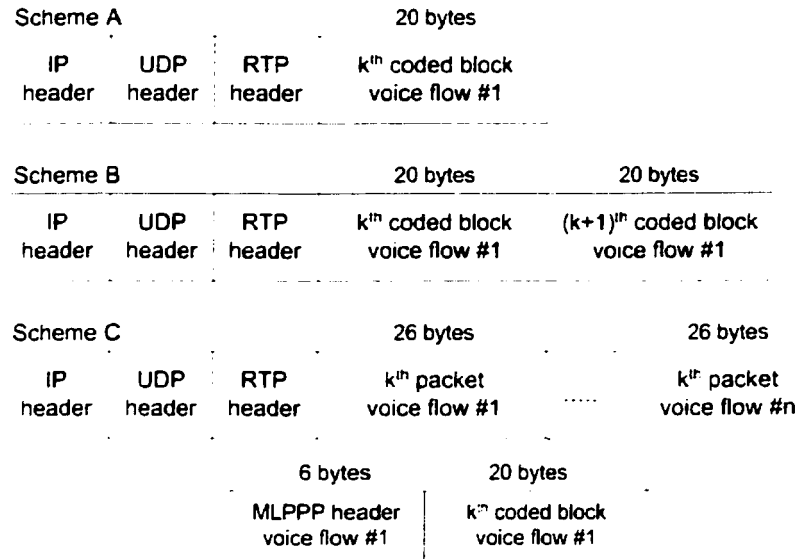


Figure 5.12: Packet Formats for Schemes A, B and C

Given that  $M$  is the number of voice flows being multiplexed together, the EF voice packet size for Scheme C can be calculated as:

$$\begin{aligned}
 EF \text{ Packet Size} &= RTP \text{ Header} + M \times (MLPPP + G.723.1 \text{ Voice Packet}) \\
 &= 40\text{bytes} + M \times 26\text{bytes}
 \end{aligned}$$

The Scheme C packet sizes selected for this simulation is listed in Table 5.11.

Table 5.11: Scheme C Packet Size

Number of Multiplexed Voice Flows, $M$	Scheme C Packet Size (bits)
8	1984
16	3648
32	6976

The theoretical queuing delay calculations for EF and BE flows are listed in Table 5.12.

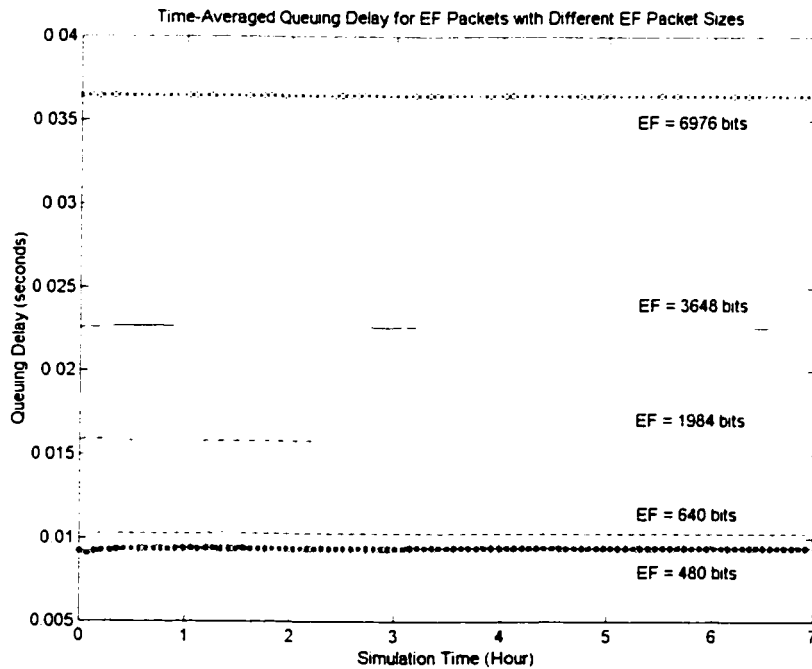
Table 5.12: Theoretical Queuing Delay Calculations for EF and BE Traffic Flows

Link Speed ( <i>kbps</i> )	256
Link Utilization	80%

BE Traffic Parameters		EF Traffic Parameters	
BE Packet Size ( <i>bits</i> )	4800	EF Inter-arrival Time	0.030
BE Transmission Delay ( <i>sec</i> )	0.019	EF Number of Packets	11.71

EF Packet Size ( <i>bits</i> )		<i>All unspecified units in seconds</i>			
<i>N</i> = 1 (Scheme A)	480	EF Transmission Delay	0.0019	EF Queuing Delay	0.009
		EF Bandwidth ( <i>kbps</i> )	5.62	BE Queuing Delay	0.056
		BE Number of Packets	41.50		
		BE Bandwidth ( <i>kbps</i> )	199.18		
<i>N</i> = 8 (Scheme C)	1984	EF Transmission Delay	0.0078	EF Queuing Delay	0.015
		EF Bandwidth ( <i>kbps</i> )	23.23	BE Queuing Delay	0.057
		BE Number of Packets	37.83		
		BE Bandwidth ( <i>kbps</i> )	181.57		
<i>N</i> = 16 (Scheme C)	3648	EF Transmission Delay	0.0145	EF Queuing Delay	0.023
		EF Bandwidth ( <i>kbps</i> )	42.717	BE Queuing Delay	0.061
		BE Number of Packets	33.77		
		BE Bandwidth ( <i>kbps</i> )	162.08		
<i>N</i> = 32 (Scheme C)	6976	EF Transmission Delay	0.0273	EF Queuing Delay	0.040
		EF Bandwidth ( <i>kbps</i> )	81.69	BE Queuing Delay	0.084
		BE Number of Packets	25.65		
		BE Bandwidth ( <i>kbps</i> )	123.11		
<i>N</i> = 2 (Scheme B)	640	EF Transmission Delay	0.0025	EF Queuing Delay	0.010
		EF Bandwidth ( <i>kbps</i> )	3.04	BE Queuing Delay	0.056
		BE Number of Packets	42.03		
		BE Bandwidth ( <i>kbps</i> )	201.76		

From *Figure 5.13*, the time-averaged end-to-end queuing delay decreases with smaller EF packet sizes. Although Scheme A and B have a lower queuing delay, the voice packets generated by these schemes have a big header overhead. Scheme C avoid this inefficiency through multiplexing different voice flows. With a larger payload, Scheme C voice packets increases the transmission efficiency of the network. In addition to that, the number of packets to be processed by the queuing system also decreases. This is a classical tradeoff issue between bandwidth efficiency and queuing delay requirements for voice transmission.



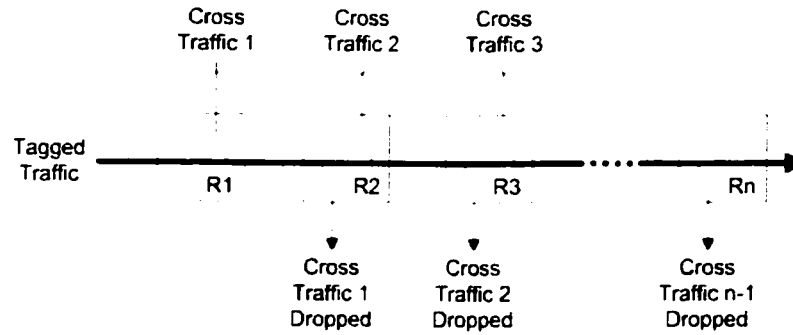
*Figure 5.13: Queuing Delay for EF Packets with Different EF Packet Sizes*

## 5.2.2 Analysis of EF Traffic with Different Cross Traffic Parameters

In a DiffServ model, traffic flows are classified and assigned with different levels of service priority based on Per-Hop-Behavior (PHB) to provide Quality of Service (QoS) in the network. The aggregation of flows that belong to the same traffic class will have the same forwarding treatment at each DiffServ-enabled core routers. This model is highly scalable but the rather coarse traffic aggregation levels provided may cause high level of distortion to the flows as they arrive at the destination edge router in a DiffServ domain. Furthermore, this jitter phenomenon will lead to conformance issues in terms of SLA for the flows as they cross a DiffServ domain, as discussed in the article [79].

All traffic flows are shaped to conform to the SLA as they arrive at the network Ingress node. Jitter is considered as the distortion to the distribution of the inter-packet arrival time at the network Egress due to traffic aggregation and other conditions in the core network. *Figure 5.14* shows the generic network topology that is used for the simulations in this section to look at the jitter effect on a particular voice stream with other voice streams that have the same characteristics. This linear multi-hop topology has been widely used by other studies to look at the effects of cross traffic on the tagged traffic flow.





*Figure 5.14: Linear Multi-hop Network Topology for Simulation*

In this topology, a tagged traffic stream enters the network at the Ingress node  $R_1$ , and travels through the network to reach the Egress node  $R_n$ . The tagged stream will be analyzed for the jitter effect. Cross traffics will be injected to interact with the tagged traffic stream at every node along the path. In our simulation, the cross traffics consist only of EF voice traffic as our focus is on the impact of aggregation of similar streams. Since we are using PQ as our scheduling mechanism, we can safely assume that the impact from BE traffic streams is minimum. The cross traffic entering at a given node interferes with the tagged stream for only one hop, and leaves the network at the next hop. The sink will monitor the arrival time of the packets and calculate the inter-arrival time of the packets. *Figure 5.15* shows the OPNET model implemented to examine the effects of cross traffics on the tagged traffic flow.

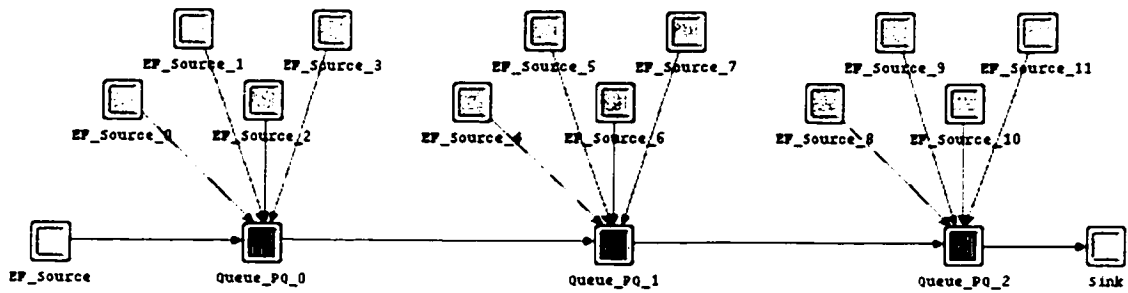


Figure 5.15: OPNET Model of Tagged and Cross Traffic

The parameters for the simulation in this section are listed below:

1. All the links in the network have transmission rate of 256kbps.
2. Tagged traffic packet size is 480 bits with no silent suppression.
3. Cross traffic packet size is 480 bits with silent suppression.

The reason the tagged traffic flow is chosen without silent suppression is to enable us to have a better view at the effect of inter-packet arrival time at the sink.

### 5.2.2.1 Test Case: Different Number of Cross Traffic

Table 5.13: Test Parameters for the Different Number of Cross Traffic

Transmission Line		Expedited Forwarding			
Line Rate (kbps)	Utilization Factor (%)	Number of Tagged Traffic	Number of Cross Traffic	Packet Size Dist. (bits)	Number of PQ Nodes
256	Variable*	1	Variable*	Const (480)	3

\* The number of cross traffics is selected to be 0, 4, 8, 13, 17, and 22 respectively.

The number of cross traffics selected corresponds to 0%, 10%, 20%, 30%, 40%, and 50% of link utilization respectively. We observe from Figure 5.16 that the time-averaged end-to-end queuing delay increases with a higher number of cross traffic.

Furthermore, the magnitude of change in queuing delay increases as the number of cross traffic increase. This shows that a higher utilization factor by the same traffic class increases the contention probability of the EF packets, which in turn affects the queuing delay of the EF priority queue.

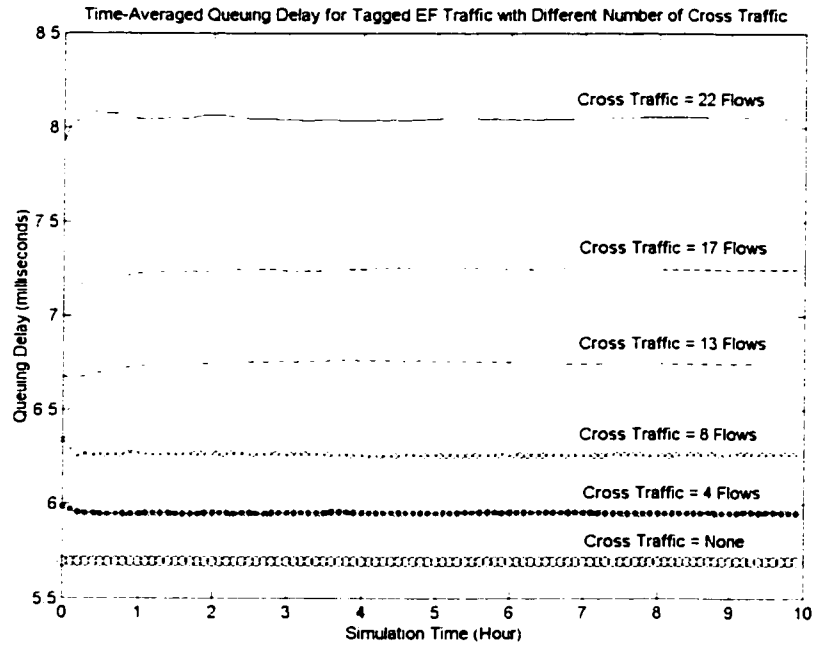
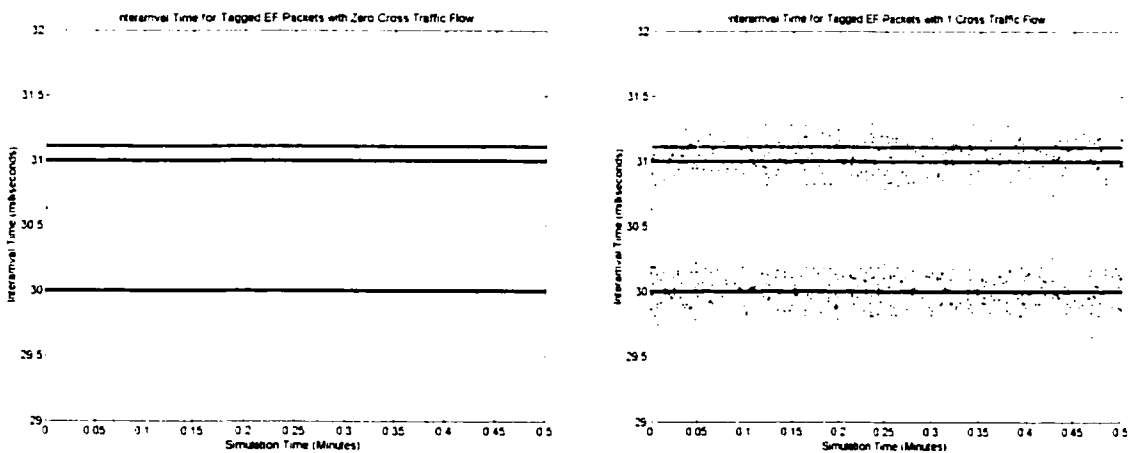
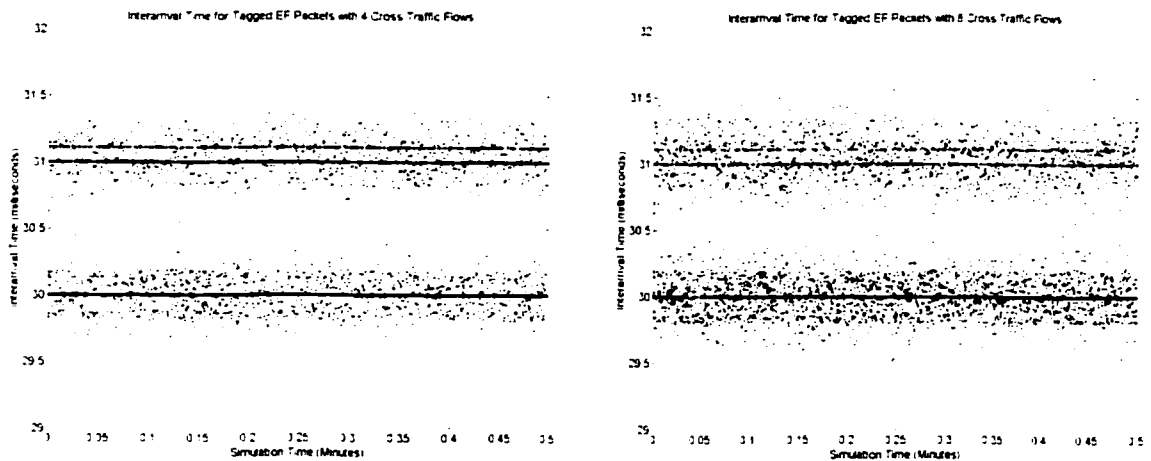


Figure 5.16: Queuing Delay for Tagged Traffic with Different Number of Cross Traffics





*Figure 5.17: Distortion Level (Jitter) of the Tagged Traffic Stream*

*Figure 5.17* shows the inter-arrival time for the tagged traffic based on different number of cross traffics. In our simulation, the tagged traffic stream is configured to have a constant inter-arrival time of 30 milliseconds. However, the simulation result shows three different inter-arrival time values (from the top left plot). This discrepancy is caused by the OPNET simulation engine, and can be ignored since we are looking at jitter effects. The inter-arrival time for the tagged traffic flow, as expected, gets more distorted as we increase the cross traffic flows. Thus, a shaper required at the network Egress for the flows to conform to the overall network-to-network traffic profile defined in the SLA.

### 5.2.2.2 Test Case: Different Number of Intermediate Nodes

Table 5.14: Test Parameters for the Different Number of Intermediate Nodes

Transmission Line		Expedited Forwarding			
Line Rate (kbps)	Utilization Factor (%)	Number of Tagged Traffic	Number of Cross Traffic	Packet Size Dist. (bits)	Number of PQ Nodes
256	10	1	4	Const (480)	Variable*

\* The number of nodes is selected to be 1, 5, 10 and 20 respectively

We observe from *Figure 5.18* that the number of intermediate nodes along a transmission path directly increases the queuing delay experienced by the tagged traffic. However, this result is not unexpected. Each queuing node will impose a certain amount of queuing delay on the tagged EF packets. As the number of intermediate nodes increases, the cross traffics will impose a bigger effect on the tagged traffic.

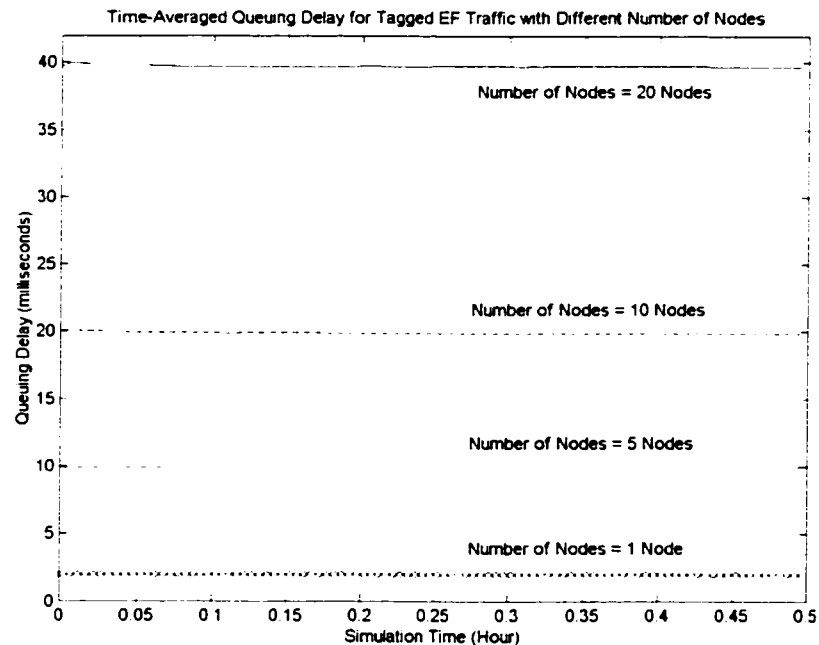
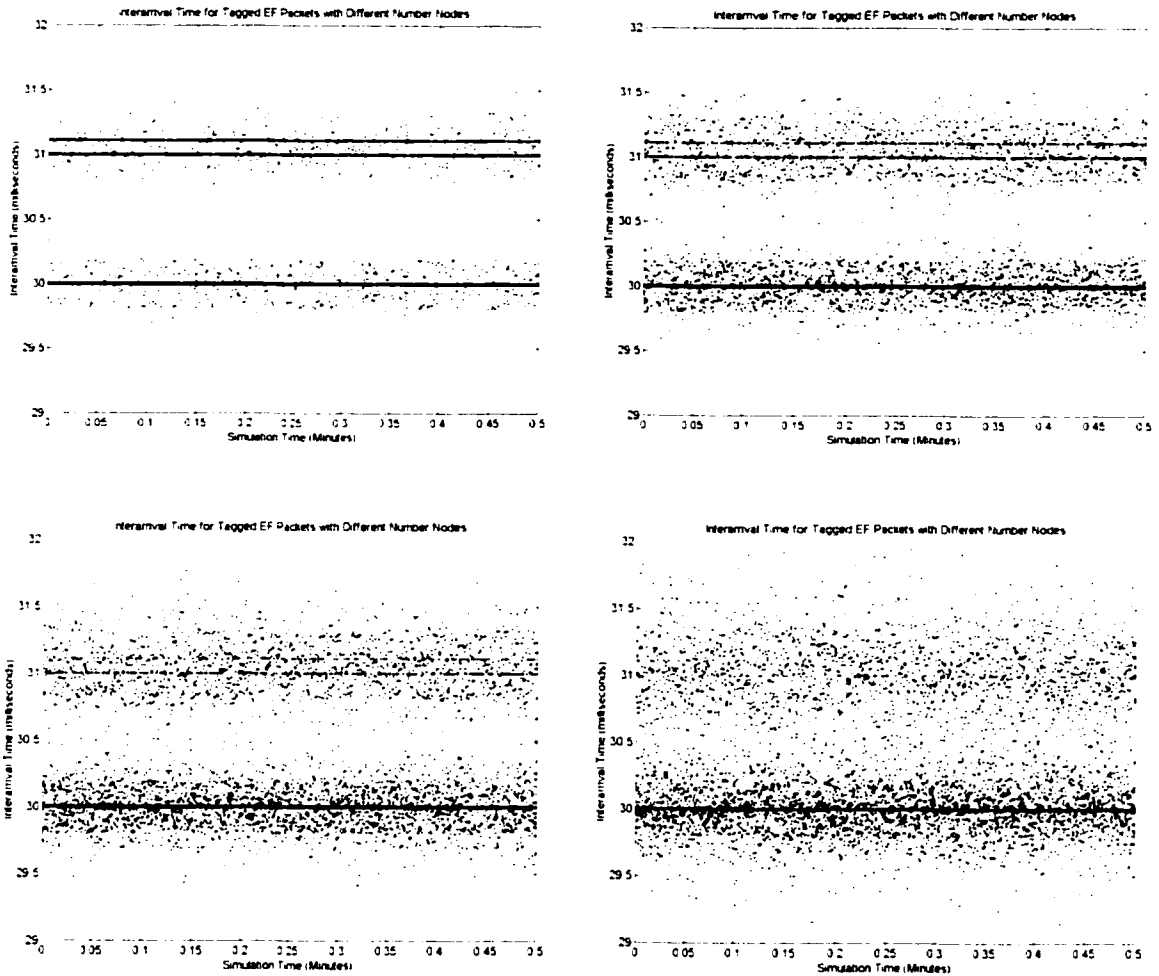


Figure 5.18: Queuing Delay for Tagged Traffic with Different Number of Nodes



*Figure 5.19: Distortion Level (Jitter) of the Tagged Traffic Stream*

We observe from *Figure 5.19* that the effect of cross traffics has a greater impact on the inter-arrival time of the tagged traffic with the increase in intermediate nodes. Thus, in a large network, traffic shapers have to be strategically placed to reduce the level of distortion in some part of the network to avoid the distorted flows from causing additional jitter to the rest of the network.

### 5.2.2.3 Test Case: Different Traffic Profiles

Table 5.15: Test Parameters for the Different Traffic Profiles

Transmission Line		Expedited Forwarding			
Line Rate (Mbps)	Utilization Factor (%)	Number of Tagged Flows	Number of Cross Flows	Packet Size Dist. (bits)	Number of PQ Nodes
1.540	10	1	4	Variable	3

There are four different combinations of traffic profile selected for this simulation

1. Cross traffic streams with constant-size packets and a Tagged traffic stream with fixed-size packets.
2. Cross traffic streams with variable-size packets and a Tagged traffic stream with fixed-size packets.
3. Cross traffic streams with constant-size packets and a Tagged traffic stream with variable-size packets.
4. Cross traffic streams with variable-size packets and a Tagged traffic stream with variable -size packets.

For all the above-mentioned cases, packet sizes for streams with fixed-sized packets are 480 bits, and in the case of streams with variable-sized packets, packet sizes are normally distributed with a minimum size of 480 bits and maximum size of 8000 bits.

We can clearly observe from *Figure 5.20* that there are two distinct sets of curve in the figure. The dominant effect on egress jitter is the internal packet variability from tagged traffic stream, compared to the impact of external packet variability from the cross traffic streams. Furthermore, we observe from *Figure 5.21* that the tagged traffic stream with variable-sized packets experience a higher level of distortion compared with the

results from the previous two sections. Thus, a shaper or policer is required at the egress node of the network to reduce the effect of jitter for the traffic streams that aggregate multiple flows, based on PHB, with different packet-size variability.

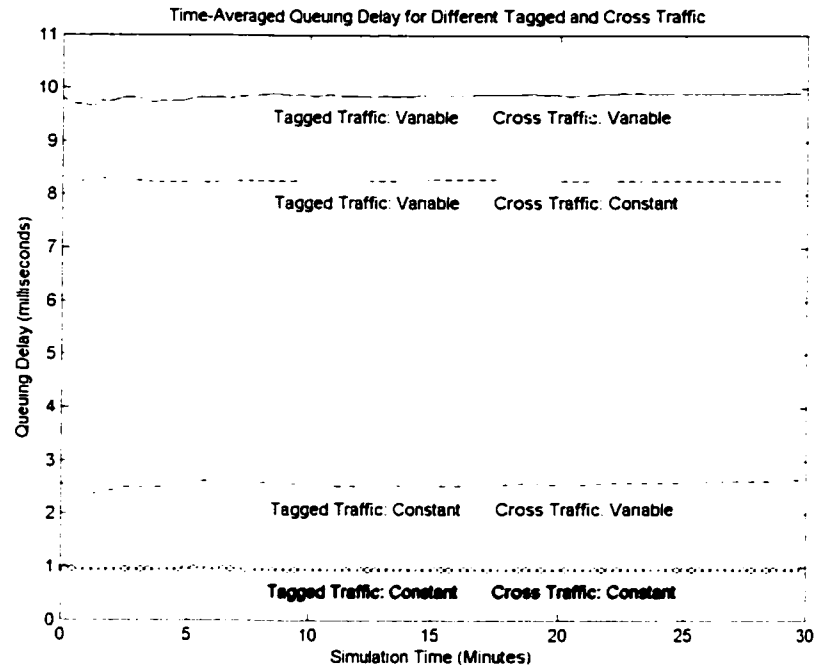


Figure 5.20: Queuing Delay for Tagged Traffic with Different Traffic Profiles



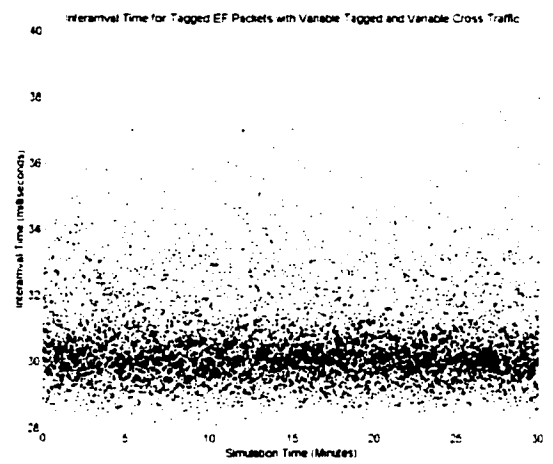
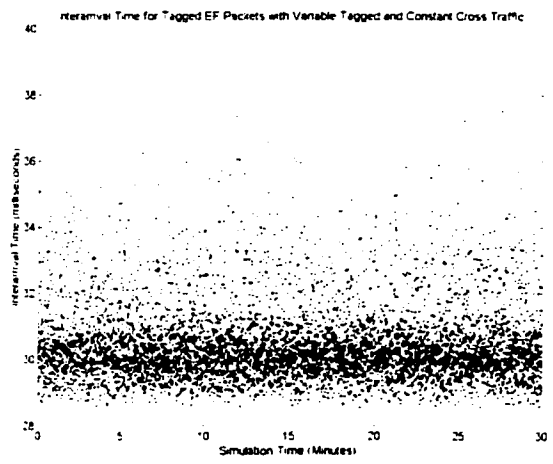
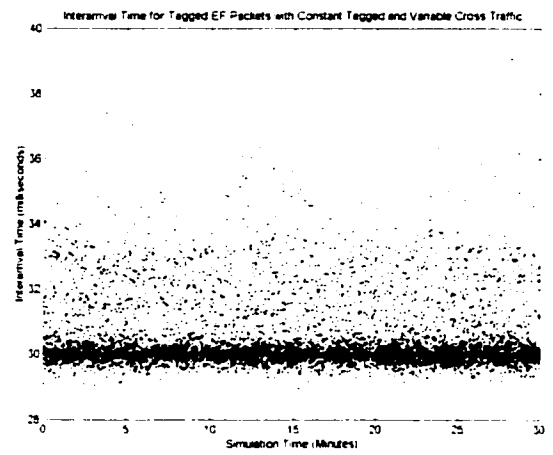
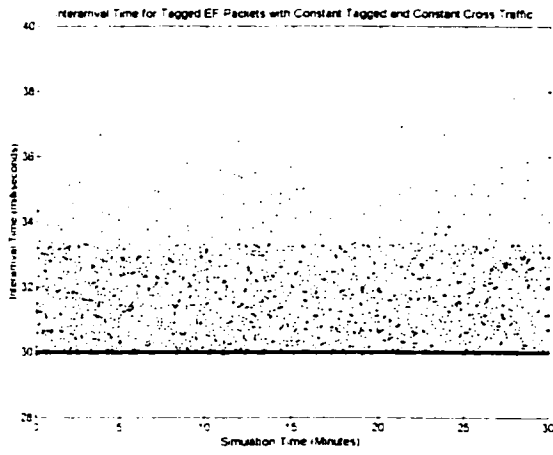
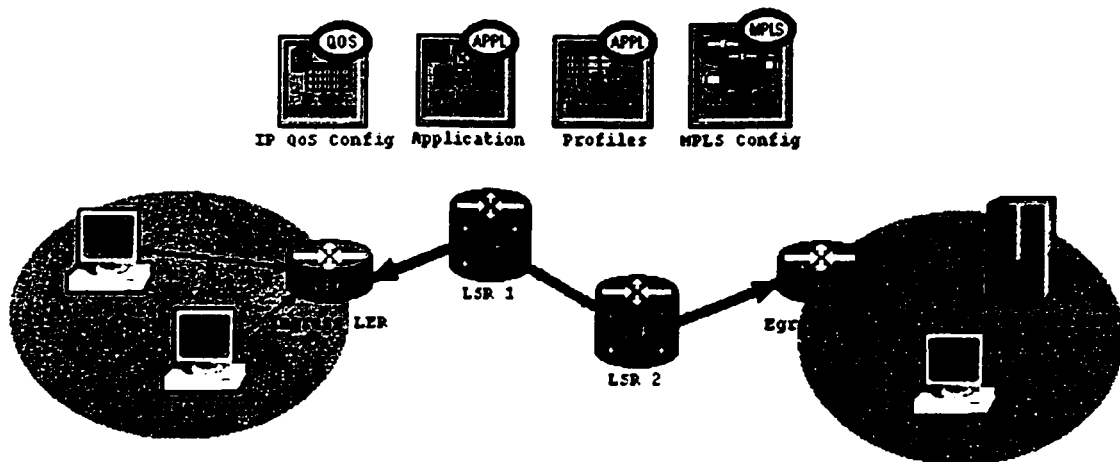


Figure 5.21: Distortion Level (Jitter) of the Tagged Traffic Stream

### 5.3 Network Model Simulation

We have seen how EF voice traffic is being affected by the BE traffic flows and EF cross traffics from the node model simulations. However, we would like to look at the performance of the voice traffic across an MPLS network, with multiple voice calls and different network topologies. Using the MPLS model suite provided by OPNET, we have created a network model of an MPLS network to perform the simulation. The basic network topology is shown in *Figure 5.22* below. The LSR will act as the intermediate nodes in the network, while the LER at the edge network connects the source and destination nodes to the core.



*Figure 5.22: The MPLS Network Topology for OPNET Simulation*

Once the network topology is constructed, we are ready to model the network traffic by setting up the Application Definition Module. OPNET provides a number of network applications with configurable traffic profile such as File Transfer Protocol (FTP), e-mail, and telnet to simplify the generation of network traffic. We have chosen

FTP to simulate the one-way data stream transfer from the “Data Source” to the “Data Server.” since we can regulate link utilization by specifying the file transfer size and inter-arrival time. The voice application model provided by OPNET has configurable parameters such as call generation rates and voice-encoding schemes, and includes a built-in caller-based performance statistics collection mechanism.

The Profile Definition Module describes the activity patterns of a source node in terms of the applications used over a period of time, with configurable parameters such as start time, duration and repeatability. For example, we can define a profile called “20× G.711 silence suppression” to represent twenty voice calls under G.711 encoding scheme with silence suppressing, executing simultaneously with different starting and ending time. The “Voice Source” will be the traffic generator that uses this profile to simulate the desired voice traffic in the network. A voice call is a two-way traffic flow between the source and its destination.

The MPLS Definition Module allows us to configure the LSP, FEC specifications, traffic trunks profiles and traffic engineering configurations. The LSP configuration consists of its directionality (unidirectional or bidirectional), type (static or dynamic) and path details. The FEC classifies and performs traffic aggregation to support QoS in the network by performing packet labeling. The traffic trunk profiles specify out-of-profile actions and act as CAR to control traffic characteristics such as peak rate, average rate and average burst size according to the SLA. The LER supports TE bindings, which specify the association of different FECs to their corresponding allocated

LSPs, which in turn related to a particular traffic trunk, based on the label of the incoming packet.

### 5.3.1 Analysis of Voice Traffic is an MPLS Network

In order to simulate the environment to analyze the voice traffic in an MPLS network, we need to modify the parameters for the voice source and various network constraints. The following are the assumptions for the simulations in this section

1. Voice stream has EF as its PHB.
2. Data stream has AF as its PHB.
3. Data stream consumes approximately 40% of link utilization.
4. The size of the data packets is 1500 bytes with an exponential distributed inter-arrival time of 0.02 seconds, consuming 600 *kbps* of bandwidth.
5. All links are T1 lines with 1.544 *Mbps*.
6. The simulation network domain consists of 4 intermediate nodes – 2 LERs at the edge and 2 LSRs at the core of the network.

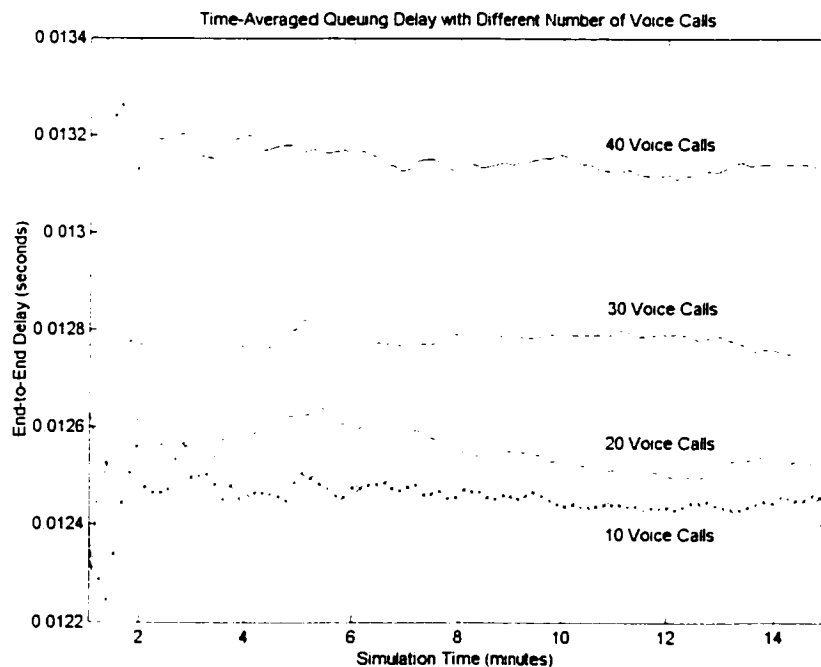
#### 5.3.1.1 Test Case: Different Number of Voice Calls

*Table 5.16: Test Parameters for the Different Number of Voice Calls*

Voice Flow Parameters			
Voice Encoding Scheme	Silence Suppression	Num Coded Blocks Per Voice Packet	Num of Voice Calls
G.723.1	Yes	1	Variable*

\*The number of calls is selected to be 10, 20, 30, and 40 calls respectively.

The time-averaged end-to-end queuing delay increases with a higher number of voice calls, since more voice packets are generated in the network with the increase in number of calls made. Furthermore, we observe from *Figure 5.23* that the magnitude of change in queuing delay increases as we increase the number of call, since this will increase the contention probability of the voice packets, which in turn affects the queuing delay of the voice EF priority queue. *Figure 5.24* shows the voice traffic throughput and packet rate for the network based on different number of voice calls. Although we use a voice-encoding scheme with silent suppression, the time-averaged throughput and packet rate increased with a linear scale with each 10-call increment.



*Figure 5.23: Queuing Delay for Voice Traffic with Different Number of Voice Calls*

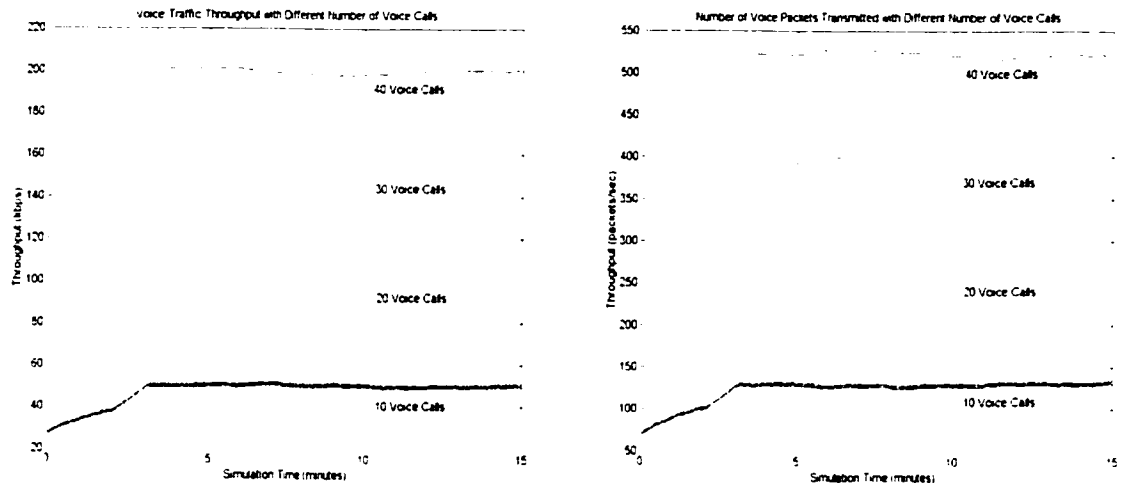


Figure 5.24: Throughput for Overall Voice Traffic with Different Number of Voice Calls

### 5.3.1.2 Test Case: Different Number of Frames Per Voice Packet

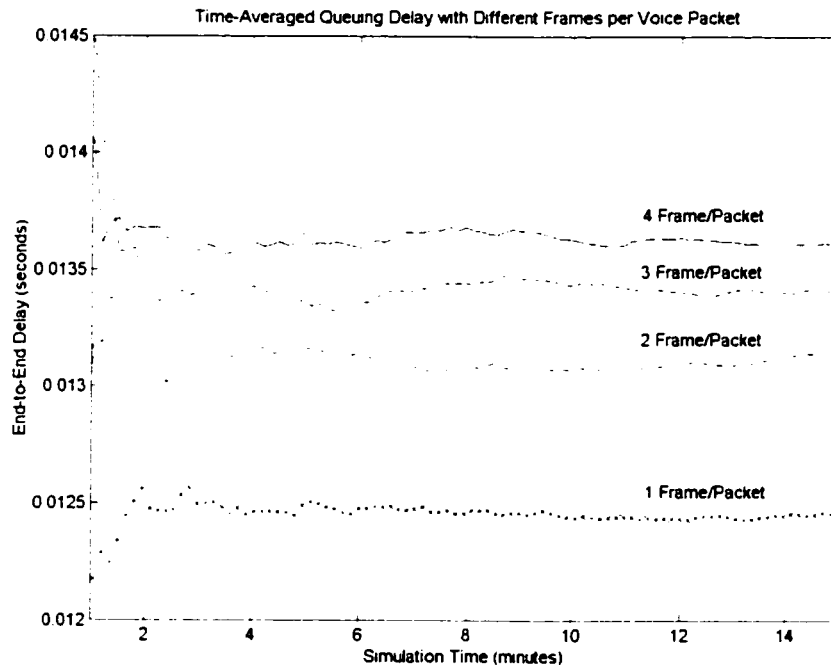
Table 5.17: Test Parameters for the Different Frames per Voice Packet

Voice Flow Parameters			
Voice Encoding Scheme	Silence Suppression	Num Coded Blocks Per Voice Packet	Num of Voice Calls
G.723.1	Yes	Variable*	20

\* The number of coded blocks per voice packet is selected to be 1, 2, 3, and 4 respectively.

Based on the discussion from *Section 5.2.6*, we have seen how a voice packet is generated with different number of coded blocks. Since an uncompressed RTP header size for a voice packet is 40 bytes and one voice-coded block for G.723.1 is only 10 bytes, the RTP header imposes a very high overhead. The overhead of the header can be reduced through the encapsulation of a number of consecutive coded blocks in one voice packet. In this simulation, we have chosen to encapsulate 2, 3, and 4 consecutive coded voice blocks in one voice packet.

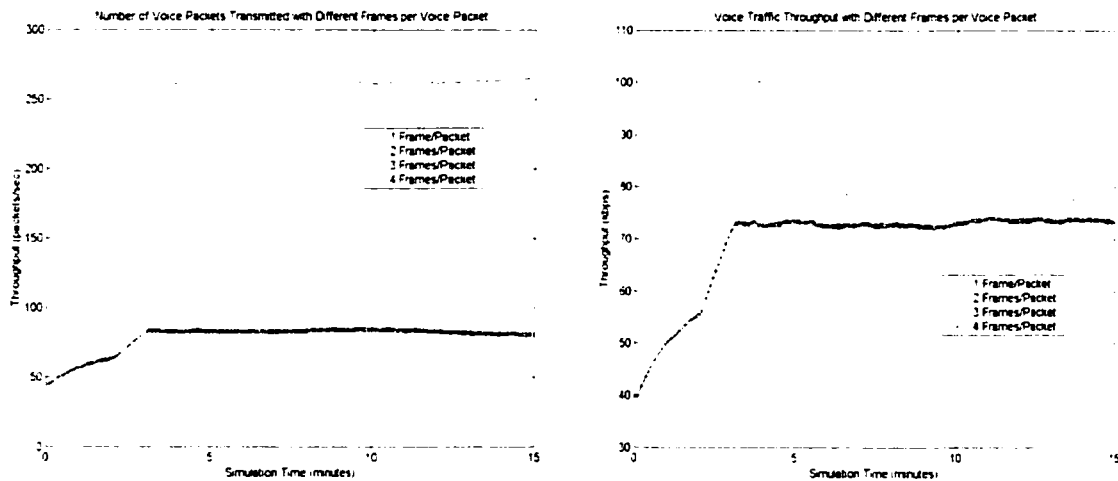
We observe from *Figure 5.25* that the time-averaged end-to-end queuing delay increases with the encapsulation of more coded voice blocks in a voice packet. This is due to the fact that the size of the voice packet increases with the encapsulation of more coded blocks, which corresponds with the simulation result presented in Section 5.2.6. As a reminder, this queuing delay does not include the coding delay for waiting the consecutive voice block to be included in the packet. Thus, it is not recommended to have too many coded blocks included in one voice packet.



*Figure 5.25: Queuing Delay for Voice Traffic with Different Frames per Voice Packet*

From *Figure 5.26*, we observe that the savings in throughput becomes less and less with the increase in the number of coded block included in one encapsulation. Furthermore, we also observe that the throughput (in terms of packets per second) for 3 and 4 coded blocks are almost identical, which indicate that no savings is obtained for the latter case.

Voice encoding scheme with silent suppression will only generate voice-coded blocks during active speech periods. According to the results, the G.723.1 scheme has only a slim chance to have 4 consecutive blocks in one encapsulation. Voice packet will still be generated even of there is less than 4 blocks if a silence period is encountered as the consecutive block to avoid long coding delay.



*Figure 5.26: Throughput for Overall Voice Traffic with Different Frames per Packet*

From these simulation results, we can conclude that at most 2 coded blocks are to be encapsulated in one voice packet, since the advantage of encapsulating more blocks in one packet quickly degenerated with more than 2 blocks. In addition to that, we also have to consider the effect of coding delay imposed on the overall end-to-end delay.



### 5.3.1.3 Test Case: Different Voice Coding Schemes

Table 5.18: Test Parameters for the Different Voice Coding Scheme

Voice Flow Parameters			
Voice Encoding Scheme	Silence Suppression	Num Coded Blocks Per Voice Packet	Num of Voice Calls
Variable*	Variable*	1	20

\* The type of voice-encoding scheme selected is G.729 and G.723.1.

Different type of voice-encoding schemes will have a different effect on the overall network performance. We have selected five different encoding schemes for comparative simulation analysis – G.729 with and without silent suppression, G.723.1 with and without silent suppression, and G.729 with 3 coded blocks per voice packet. Although the packet size is the same for a particular voice-encoding scheme, with or without silent suppression, we observe from *Figure 5.27* that voice-encoding schemes with silent suppression have a lower end-to-end queuing delay. This is due to the higher number of packets generated by the non-silent suppression coding scheme as observed in *Figure 5.28*, which causes packet contention in the EF priority queue, leading to higher queuing delay. Furthermore, *Figure 5.28* shows that the throughput or bandwidth consumption for coding schemes with silent suppression is lower, saving the network resources to be utilized for some other applications. Thus, silent suppression should be used with all the different standardized voice-encoding schemes.

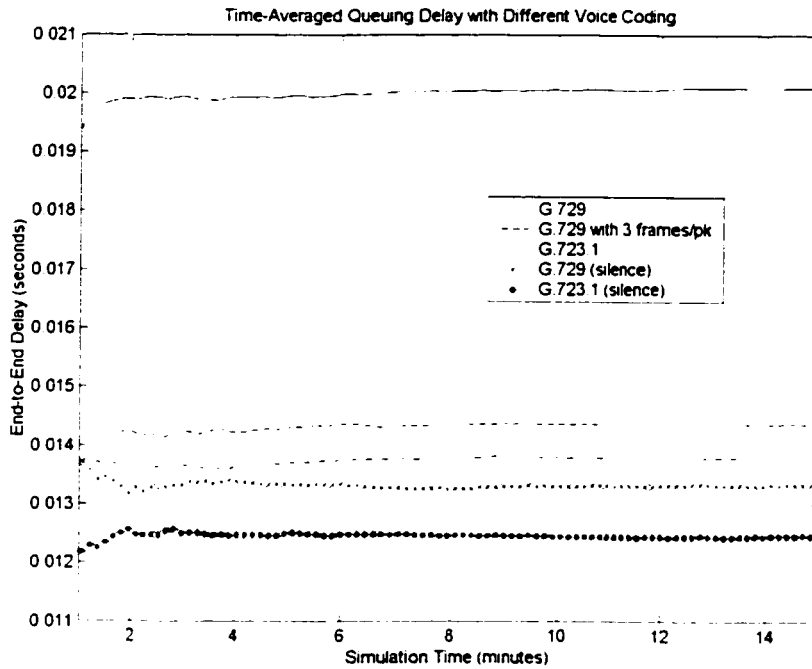


Figure 5.27: Queuing Delay for Voice Traffic with Different Coding Schemes

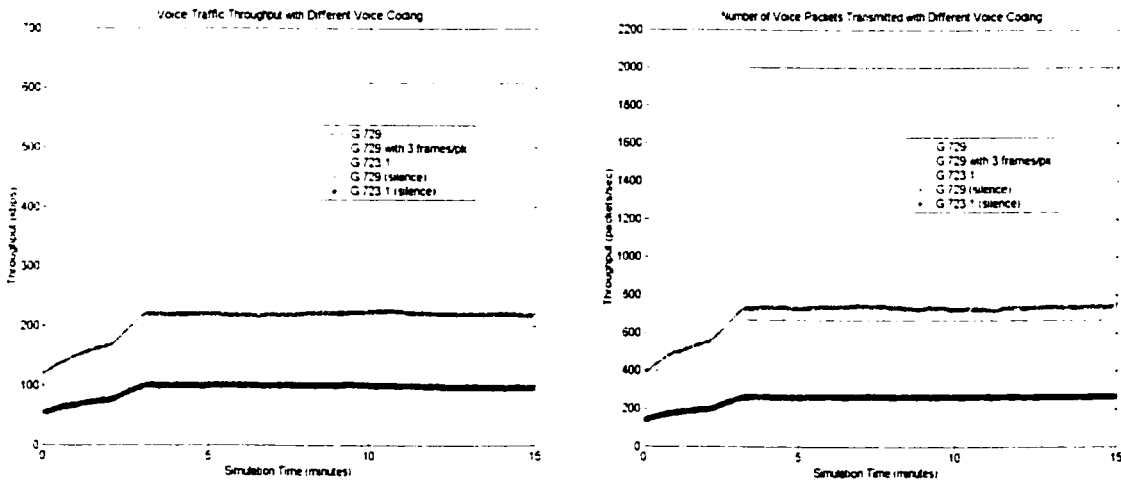


Figure 5.28: Throughput for Overall Voice Traffic with Different Coding Schemes

Comparing the normal G.729 scheme which carries 1 coded block per voice packet with the encapsulation of 3 consecutive coded blocks per voice packet, we observe from Figure 5.28 that the throughput or bandwidth savings is only by half (600 kbps

versus 300 kbps). This is due to the RTP header overhead. For example, the normal G.729 scheme will produce 50-byte voice packets – 40 bytes for RTP header and 10 bytes for coded voice block. With the encapsulation of 3 consecutive coded voice blocks, 70-byte voice packets will be generated. Assuming that the number of packets generated is 100 packets and 33 packets for each of the case, the throughput will be 40 *kbps* and 18.5 *kbps* respectively. This explains the result observe from the *Figure 5.28* plots.

### **5.3.2 MPLS Traffic Engineering Capabilities**

A well-known problem that exists in today's IP networks is the support of both congestion-insensitive UDP flows and the congestion-sensitive TCP flows along the same path. During network congestion periods, the performance of the TCP traffic will quickly deteriorate due to its congestion control mechanism. The traffic engineering capabilities of an MPLS network will be able to solve this bandwidth contention problem, and also provide QoS for the network and support network failures.

*Figure 5.29* shows the basic network topology for the simulations in this section. The "TCP Source" is the congestion-sensitive source and the "UDP Source" is the congestion-insensitive source. Both sources will generate a one-way traffic flow to their respective destinations, the "TCP Server" and "UDP Server." The LER and LSR will act as the intermediate nodes in the network. The core network consists of three LSRs, creating a one-node path (via LSR1), and a two-node path (via LSR2 and LSR3). All of the links in the network are T1 lines, except the two T3 lines connected to the servers to avoid congestion from happening in the network. The simulation results for a non-MPLS

network will be compared with those from an MPLS network to show the benefits of the traffic engineering and QoS capabilities of MPLS.

### 5.3.2.1 Test Case: Non-MPLS Network with OSPF

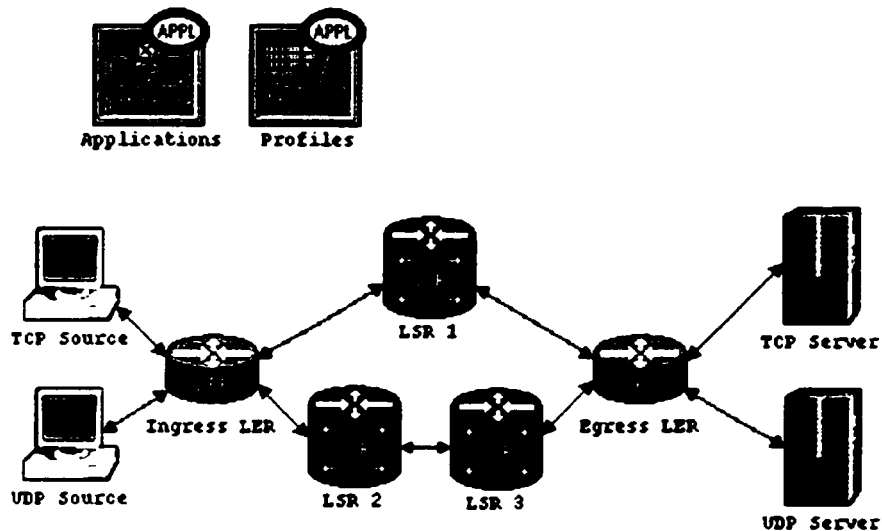


Figure 5.29: The MPLS Network Topology for OPNET Simulation

A non-MPLS network is constructed in OPNET as shown *Figure 5.29*. OSPF is chosen as the routing algorithm for this network. As a result, the TCP and UDP sources will follow the shortest path to their destinations, i.e., both traffic streams will flow from the Ingress LER to the Egress LER via LSR1.

The duration for this test case is 60 minutes. The “TCP Source” will start its transmission at the beginning of the experiment, having a constant transmission rate of 1.0 *Mbps*. The “UDP Source” will start its transmission at the 15<sup>th</sup> minute, with a transmission rate of 640kbps. At the 30<sup>th</sup> minute, the “UDP Source” will increase its transmission rate to 1.28 *Mbps*.

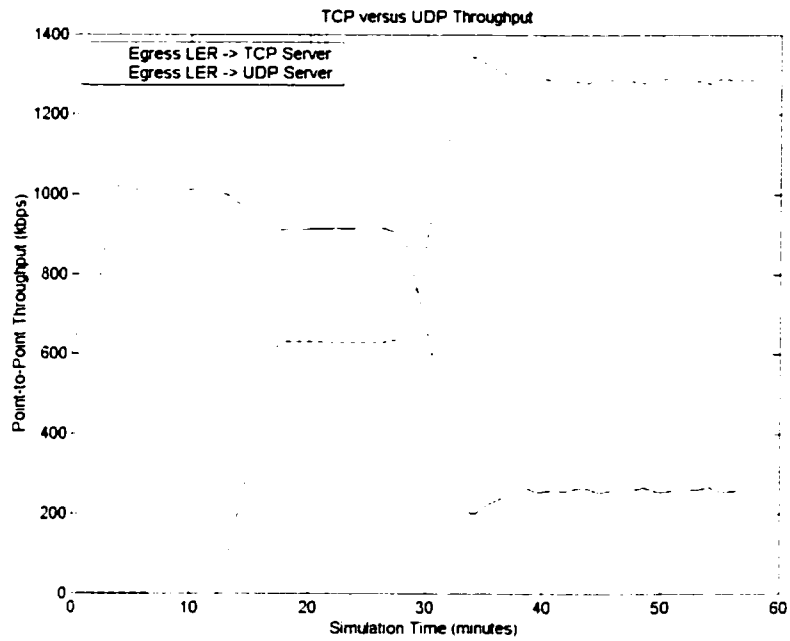


Figure 5.30: Throughput for TCP and UDP Flow

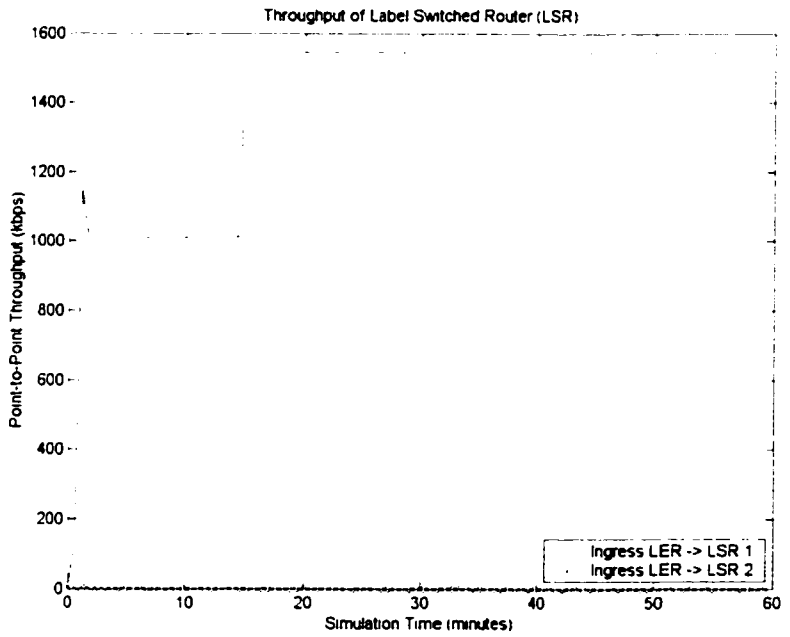


Figure 5.31: Throughput for the Two Different Paths in the Network

Figure 5.30 shows the throughput of the TCP and UDP flows, while Figure 5.31 shows the throughput of the LSRs that form the different paths in the core network. The simulation results show two problems with this network configuration. Firstly, from Figure 5.31, only TCP flow is penalized when there is congestion in the network. The UDP flow causes the TCP traffic to reduce its transmission so that the overall throughput is the TI limit, and the UDP flow consumes almost all of the bandwidth in the core network. Secondly, extra bandwidth is available in the network through the “longer” 2-node path, which is not used by any flow. This shows that the dynamic routing protocol such as OSPF does not maximize the network utilization.

### 5.3.2.2 Test Case: MPLS Network with Traffic Engineering

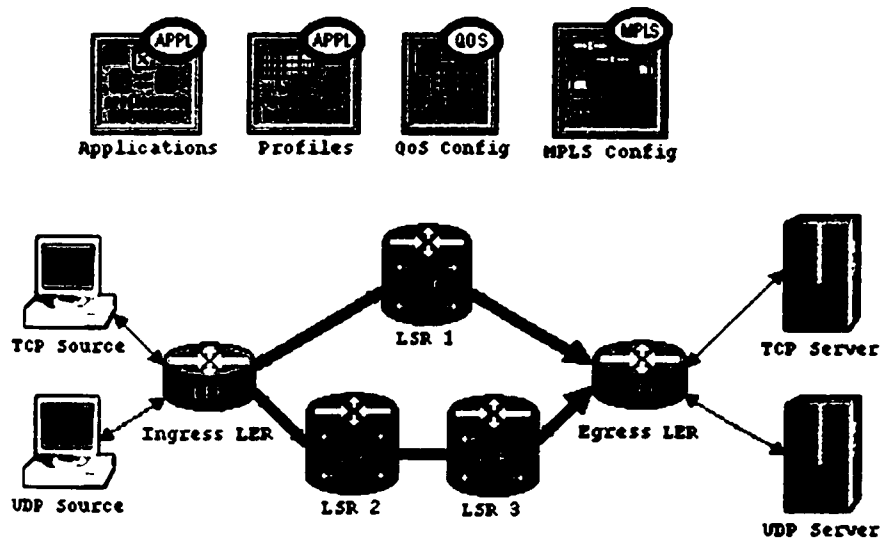


Figure 5.32: The MPLS Network Topology for OPNET Simulation

Using the MPLS Configuration Module, we can define two separate LSPs in the network. The first LSP is configured to carry TCP flow from “TCP Source” to its destination “TCP

Server” through the shorter path via LSR1, while the second LSP is configured to carry UDP flow from “UDP Source” to its destination “UDP Server” through the longer path via LSR2 and LSR3. Thus, we are able to separate the TCP and UDP traffic flows through two independent LSP based on different FEC, identified through their different applications in this case.

Figure 5.32 shows the network topology and the LSP defined for this simulation. The traffic parameters are the same as the previous case. The “TCP Source” will start its transmission at the beginning of the experiment, having a constant transmission rate of 1.0 Mbps. The “UDP Source” will start its transmission at the 15<sup>th</sup> minute, with a transmission rate of 640 kbps. At the 30<sup>th</sup> minute, the “UDP Source” will increase its transmission rate to 1.28 Mbps.

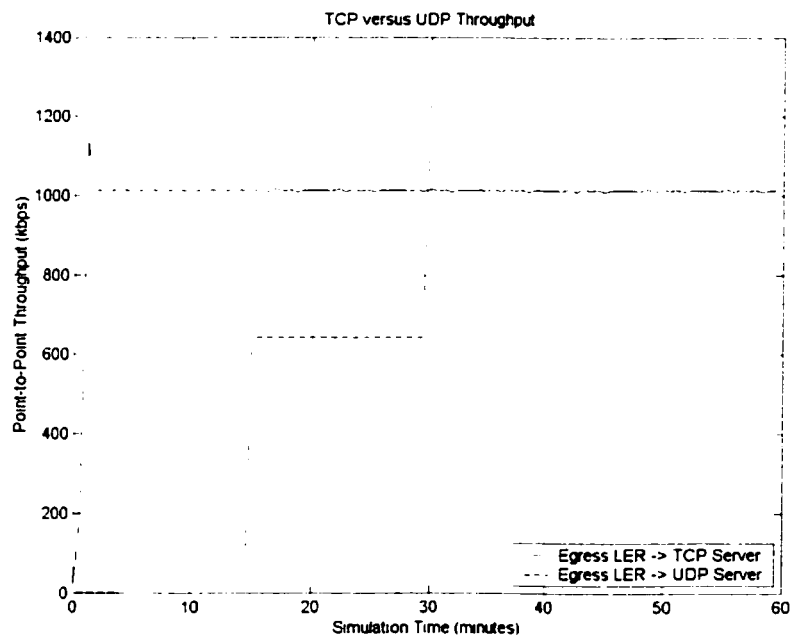
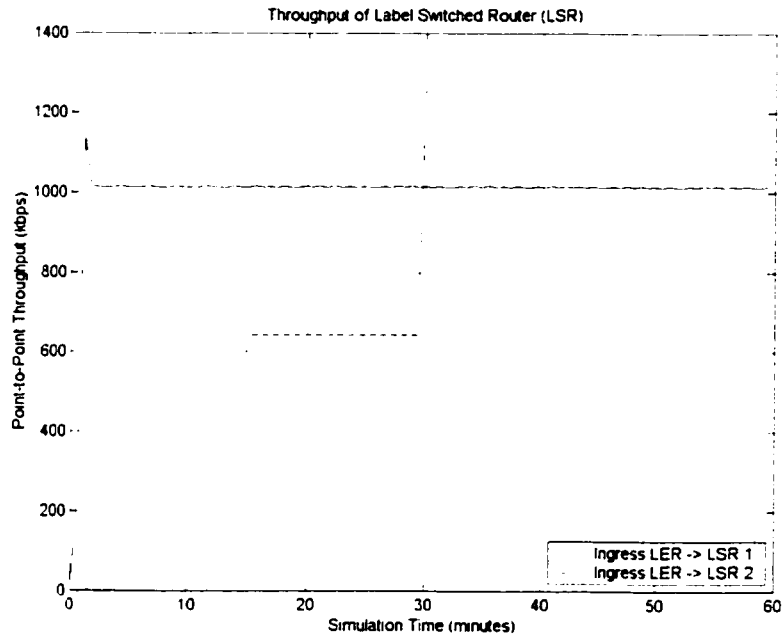


Figure 5.33: Throughput for TCP and UDP Flow

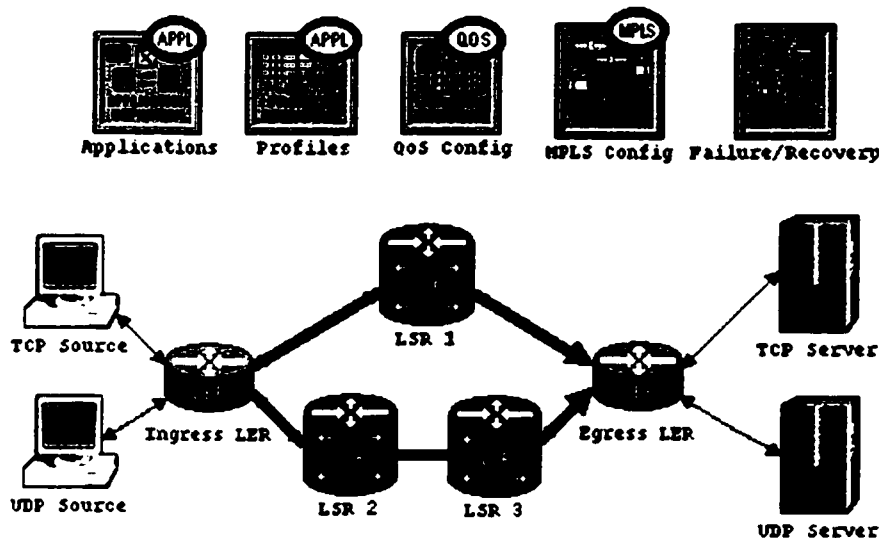


*Figure 5.34: Throughput for the Two Different Paths in the Network*

We observe from *Figure 5.33* and *Figure 5.34* that there is no congestion happening in the network, since the UDP flow has been redirected to the alternate path. Furthermore, the network utilization is better in this case. The ability to set up LSP to “force” the UDP traffic to travel through a “longer” path is an effective traffic-engineering trait to steer additional traffic away from the heavily utilized node in a network. This will prevent congestion from happening, improving the performance and utilization of the network.



### 5.3.2.3 Test Case: MPLS Network with Traffic Engineering and Failover



*Figure 5.35: The MPLS Network Topology for OPNET Simulation*

Any communication network should be robust in handling network link and node failures. MPLS provides the ability to specify backup paths in event of failure so that traffic flow can be switched to the backup path. The traffic flow will be switched back once the primary LSP has been reestablished. There will be two LSPs configured for the TCP flow. The “TCP Primary LSP1” is the main transmission path that goes through LSR1, while the “TCP Secondary LSP2” is the backup path that goes through LSR2 and LSR3. The UDP will only have one LSP configured in this case, and “UDP Primary LSP3” goes through LSR2 and LSR3.

The Failure-Recovery Module enables us to specify failure and recovery times for any node or link in the network. The OPNET simulation engine will automatically deactivate or reactivate the node or link at the specified times. The duration of this

simulation is 60 minutes. The “TCP Primary LSP1” link will fail at the 30<sup>th</sup> minute, recovering at the 40<sup>th</sup> minute.

Service Level Agreements (SLA) are defined for both UDP and TCP flows in order for us to observe the ability of MPLS to manage flow conformance. The TCP traffic flow will require a constant transmission rate of 1.0 *Mbps*, and the UDP flow will be treated as best-effort traffic flow. The “TCP Source” and “UDP Source” will start their transmissions at the beginning of the experiment, having transmission rates of 1.0 *Mbps* and 400 *kbps* respectively. At the 15<sup>th</sup> minute, the “UDP Source” will increase its transmission rate to 800 *kbps*.

When the “TCP Primary LSP1” fails at the 30<sup>th</sup> minute, the TCP flow will be redirected to the backup path “TCP Secondary LSP2”. However, this backup path shares a common path with the UDP flow. As seen in the previous case, the UDP flow will cause the TCP flow to reduce its transmission rate due to congestion. Two solutions can be used to solve this bandwidth contention issue in an MPLS network.

#### ***5.3.2.3.1 Committed Access Rate (CAR)***

CAR is used to manage flow conformance because RED not suitable for UDP flow. In order to uphold the SLA for the TCP flow, ample bandwidth will have to be preserved for the TCP flow in both its primary and secondary LSP to guarantee the transmission rate defined in the SLA. As a result, the CAR set for TCP flow is 1.0 *Mbps*, while UDP flow is 400*kbps*.

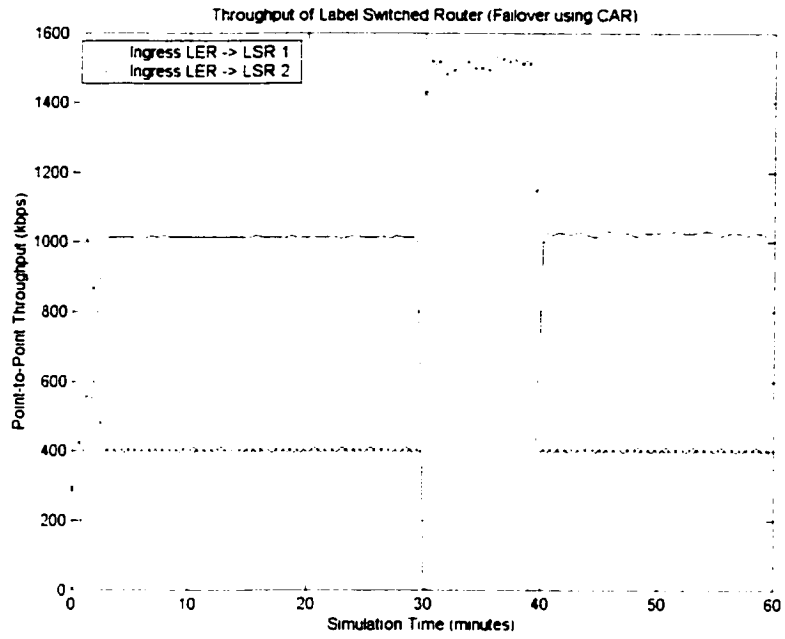


Figure 5.36: Throughput for the LSRs along Different Paths in the Network

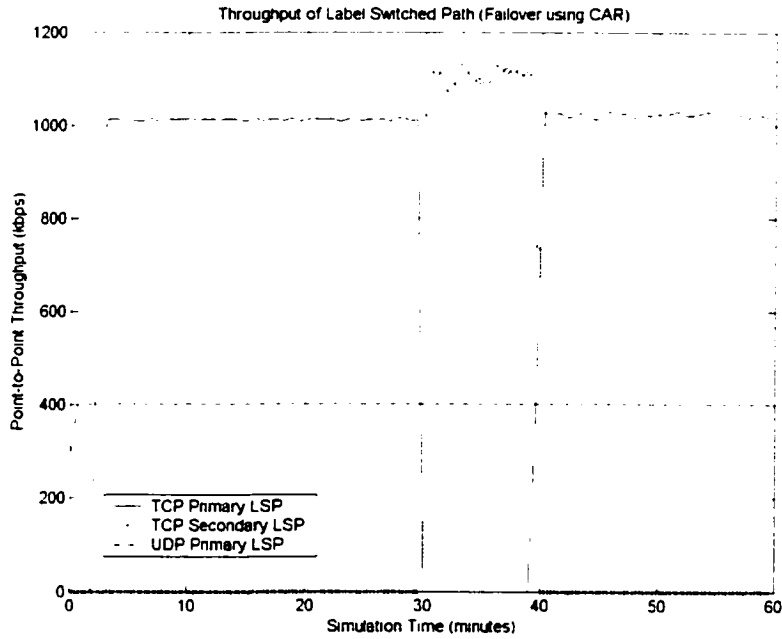


Figure 5.37: Throughput for the Two Separate LSPs in the Network

We observe from *Figure 5.36* that the “TCP Primary LSP1” fails and recovers according to the time specified in the Failure-Recovery Module. The TCP traffic flow will be switched over to the “TCP Secondary LSP2” as there is a jump in the total throughput of the path via LSP2 and LSP3. *Figure 5.37* shows the throughput for each of the three LSPs in the network. We observe that although the network is able to preserve the transmission rate of the TCP flow at *1Mbps*, the UDP traffic only consumes *400kbps* of bandwidth, even when the transmission rate is increased to *800kbps* at the 15<sup>th</sup> minute. This is due to the limitation of CAR. The disadvantage of using CAR is that the UDP flow will always be limited to *400kbps* even if excess bandwidth is available before the “TCP Primary LSP1” fails and after it recovers.

#### **5.3.2.3.2 Priority Queuing Scheme**

Since CAR has the disadvantage of not maximizing the utilization of the network, we can revert to the use of queuing mechanism to achieve the SLA defined for the different traffic flows. By applying QoS in the network through Priority Queuing (PQ), the Ingress LER is able to give priority to the TCP traffic flow. Excess UDP traffic will be discarded when both traffic flows are contenting to use the bandwidth from the same transmission path. However, this will only happen when the “TCP Primary LSP1” fails.

The “TCP Source” will start its transmission at the beginning of the experiment with a constant transmission rate of *1.0 Mbps*, and the “UDP Source” will have a transmission rate of *400 kbps*. At the 15<sup>th</sup> minute, the “UDP Source” will increase its transmission rate to *800 kbps*.

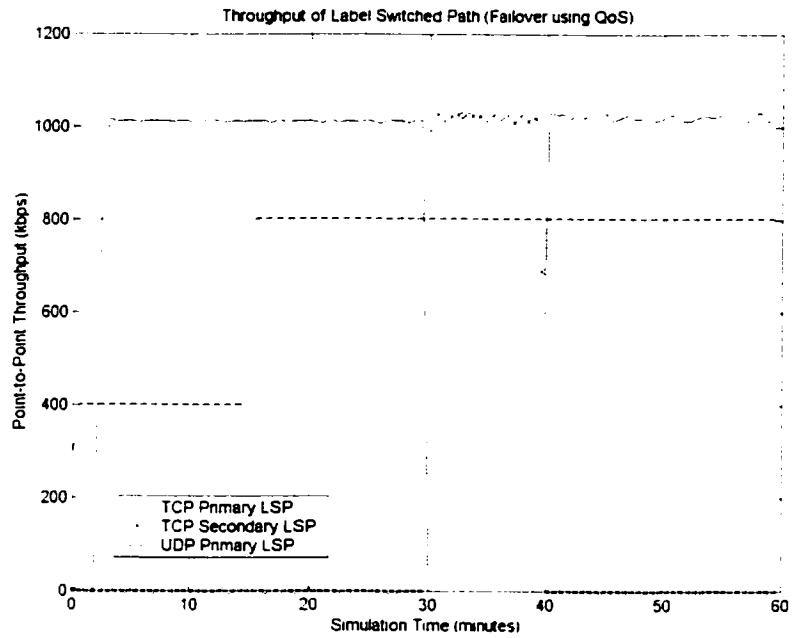


Figure 5.38: Throughput for the LSRs along Different Paths in the Network

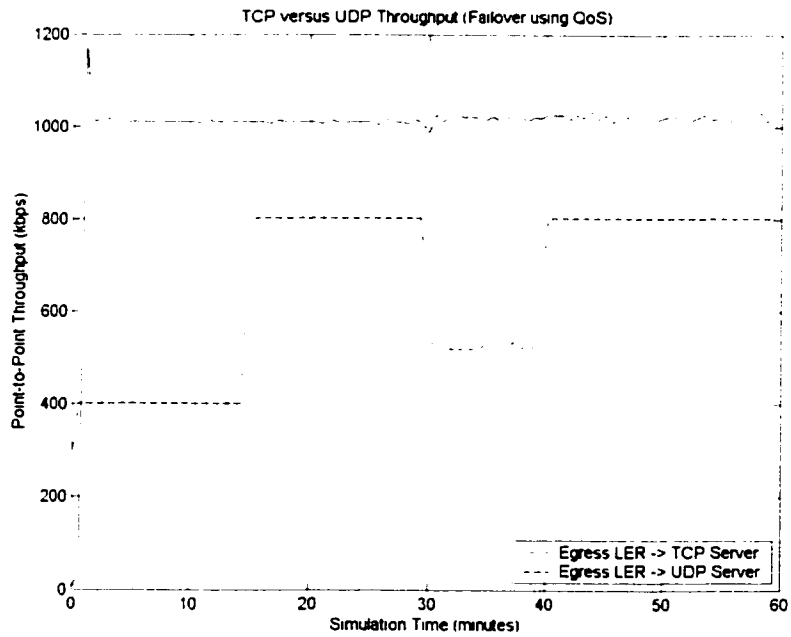


Figure 5.39: Throughput for the Three Separate LSPs in the Network

*Figure 5.38* shows the throughput for the LSPs in the network. We observe that the “TCP Primary LSP1” fails and recovers according to the time specified in the Failure-Recovery Module, and “TCP Secondary LSP2” kicks in when the primary path fails. *Figure 5.39* shows the throughput at the destination servers. It is interesting to find out that the UDP Server receives a transmission rate of 800 kbps except during the 10-minute “TCP Primary LSP1” failure period. The PQ scheme will give higher priority to the TCP packets to ensure the TCP Server maintains a transmission rate of 1 *Mbps*, and drop the UDP packets to limit its transmission rate so that the overall throughput is the T1 limit. This result shows that by incorporating QoS, the network will improve its performance significantly.

## Chapter 6 Conclusion

The convergence of voice and data networks has led to the introduction of Voice over IP (VoIP) applications. The goal of this thesis is to examine the various factors that affect the performance of voice traffic in an IP network. We describe a framework for providing QoS in the Internet to meet the stringent end-to-end delay requirement of the voice traffic. This framework consists of Priority Queuing to segregate voice traffic from data traffic to achieve prioritized queuing, DiffServ to provide service differentiation, and MPLS to provide traffic engineering and efficient packet forwarding in the Internet.

Firstly, we examine the performance of the EF voice traffic under non-preemptive Priority Queuing scheme. Simulation results in *Section 5.2.1* show that the end-to-end delay of the EF voice traffic is strongly related to traffic-related factors such as the traffic pattern of BE data traffic and the traffic profile of the EF traffic itself. Packet size and bandwidth utilization are two factors that affect the performance of voice traffic in the network. While Priority Queuing proved to be an effective mechanism to minimize queuing delay, careful network planning such as the number of intermediate nodes along a transmission path and the link speed of the core network is still required to achieve guaranteed end-to-end delay for EF voice traffic.

Next, we examine the effects of EF traffic aggregation in a DiffServ-based network. The study of the effect of traffic aggregation on end-to-end performance is to evaluate the capability of a DiffServ network in preserving the original voice traffic profile. Voice traffic is highly sensitive to delay and jitter imposed by the network, and

the simulation results in *Section 5.2.2* show that some form of traffic conditioning mechanisms such as policing or shaping are required to limit the traffic distortion level within a given traffic class. Number of cross traffics and intermediate nodes found along the transmission path contributes to the level of distortion of the aggregated EF voice traffic stream.

MPLS is the latest step in the evolution of multilayer switching technology for the core of the Internet. One of the most important benefits of MPLS is that it permits ISPs to deliver new services that cannot be readily supported by conventional IP networks. Some cost-reduction and revenue-generating services that can be deployed with MPLS include efficient traffic engineering, fast reroute during link failure, better SLA management, CoS-based forwarding and VPN. In *Section 5.3.1*, we showed how MPLS is combined with DiffServ and PQ to form a simple and efficient Internet model capable of providing applications with differential QoS and SLA.

Lastly, we examine the performance of the voice traffic across an MPLS network. The simulation results in *Section 5.3.2* show that number of voice calls directly affects the performance of the voice traffic. Different voice encoding schemes have different effects on the performance of the voice traffic. In addition to that, schemes for multiplexing different voice streams into a single voice packet stream were found to reduce the number of generated packets. Thus, the ISPs have to perform extensive studies to determine these voice application parameters to guarantee the end-to-end performance of voice traffic in their networks.



## **Chapter 7 Future Work**

Although this thesis has provided some important insights into the performance of voice traffic in data networks under different scenarios, there are still a number of aspects that can be addressed in future research. For example, it has been shown that traffic shapers are required in the network to remove jitter caused by traffic aggregation. Additional work can be done to analyze the size of the traffic shapers and the strategic placements of these traffic shapers to achieve optimal performance for the network. In addition to that, most of the OPNET simulations in this thesis are performed with a relatively small MPLS networks. Future work could develop large area MPLS networks in OPNET to simulate real-world ISP networks. As a final note, we would like to point out that this thesis focuses on the performance of the voice traffic, and little has been done to look in detail at the performance of different data flows under the DiffServ service differentiation discipline.

## Appendix A

### *The Implementation of Intranet and Extranet: A Case Study*

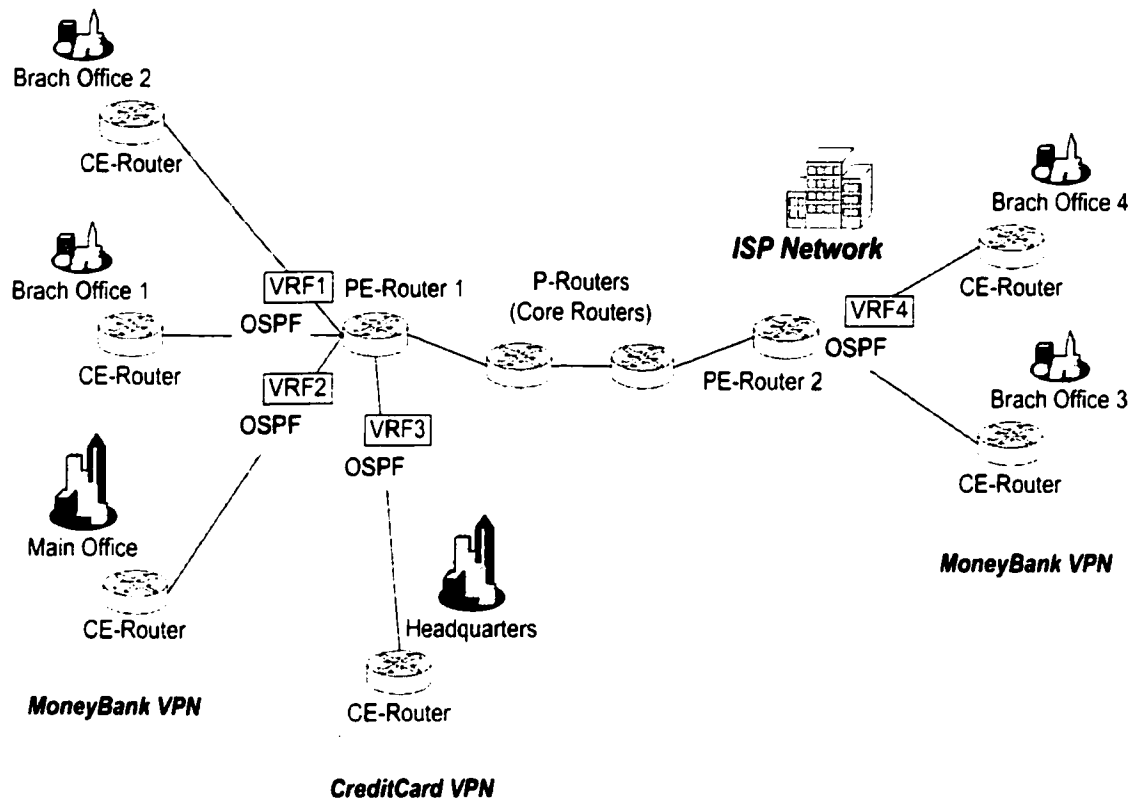


Figure A.1: VPN Topology for the Case Study

The ISP network is servicing VPNs for the corporations MoneyBank and CreditCard. The MoneyBank VPN consists of 5 different remote sites – Main Office, Branch Office 1, Branch Office 2, Branch Office 3, and Branch Office 4. The CreditCard VPN has only one site – Headquarters. All the sites in the MoneyBank VPN have to inter-communicate with each other, thus the formation of an *intranet* for MoneyBank. In addition to that, the Main Office of MoneyBank needs to form an *extranet* with the

Headquarters of CreditCard. However, the other branch offices of MoneyBank will not be in the extranet. The topology for this case study is shown in *Figure A.1*.

All the sites in a particular VPN that connects to a same PE-router will have the same routing information, e.g., Branch Office 3 and 4 have the same VRF at PE-router 2. However, at PE-router 1, Branch Office 1 and 2 has a different VRF than the Main Office's VRF. This is because Main Office can be found in both the intranet and extranet, there will be a "special" VRF setup for this particular site in PE-router 1 to include both the routes of that two VPN. The ability of the PE-router to identify the site that is located in more than one VPN and generate a separate VRF for it enables the ISP network to deploy intranet and extranet for customers.

When route information of a CE-router is advertised to its corresponding PE-router, the address is appended with a 64-bit prefix to make the address unique. This will result in a 96-bit address exchange between the PE-routers. PE-router 1 collects routing information advertised from the CE-routers from the two VPNs connected to it, running one copy of OSPF for each VPN to isolate the network traffic. However, to support a site (Main Office of MoneyBank) with an overlapping VPN, one copy of OSPF is only limited to run for each VRF in PE-router 1. As a result, the MoneyBank VPN will have two copies of OSPF running since it has two different VRF in PE-router 1. The routing information is distributed by PE-router 1 to other PE-routers in the ISP network through BGP, and they will insert the information into their VRF based on the route target attribute, e.g., the VRF in PE-router 2 will contain route target "MoneyBank VPN" so that the VRF is authorized to be updated with the routing information. Firewall is usually

deployed as the gateway of the customer network to prevent unauthorized traffic from going in and coming out of the network.

## REFERENCES

- [1] Dumortier P. (1998). Towards a New IP over ATM Routing Paradigm. *IEEE Communications Magazine*, January, 82-86.
- [2] Braden R., Clark D., Shenker S. (1994). Integrated Services in the Internet Architecture: an Overview. *RFC 1633*. Retrieved April 1, 2002, from <http://www.ietf.org/rfc/rfc1633.txt>
- [3] Braden R., Zhang L., Berson S., Herzog S., Jamin S. (1997). Resource ReSerVation Protocol (RSVP). *RFC 2205*. Retrieved April 1, 2002, from <http://www.ietf.org/rfc/rfc2055.txt>
- [4] Blake S., et al. (1998). An Architecture for Differentiated Services. *RFC 2475*. Retrieved April 1, 2002, from <http://www.ietf.org/rfc/rfc2475.txt>
- [5] (2001). Architecture for Voice, Video and Integrated Data. *Cisco Systems White Paper*. Retrieved November 10, 2001, from [http://www.cisco.com/warp/public/cc/so/neso/vvda/iptl/avvid\\_wp.pdf](http://www.cisco.com/warp/public/cc/so/neso/vvda/iptl/avvid_wp.pdf)
- [6] Rosen E., Viswanathan A., Callon R. (2001). Multiprotocol Label Switching Architecture. *RFC 3031*. Retrieved November 10, 2001, from <http://www.ietf.org/rfc/rfc3031.txt>
- [7] Semeria C. (2000). Multiprotocol Label Switching: Enhancing Routing in the New Public Network. *Juniper Networks White Paper*. Retrieved November 10, 2001, from <http://www.juniper.net/techcenter/techpapers/200001.html>
- [8] Viswanathan A., Feldman N., Wang Z., Callon R. (1998) Evolution of Multiprotocol Label Switching. *IEEE Communications Magazine*, May, 165-172.
- [9] Li T. (1999) MPLS and the Evolving Internet Architecture. *IEEE Communications Magazine*, December, 38-41.
- [10] Chen T., Oh T. (1999). Reliable Services in MPLS. *IEEE Communications Magazine*, December, 58-62.
- [11] Yoshihiro O. (1999). Issues on Loop Prevention in MPLS Networks. *IEEE Communications Magazine*, December, 64-68.
- [12] Lawrence J. (2001). Designing Multiprotocol Label Switching Networks. *IEEE Communications Magazine*, July, 134-142.

- [13] Xiao X. (2000). Providing Quality of Service in the Internet. *Doctor of Philosophy Dissertation from Michigan State University*.
- [14] Doyle J. (2000). Resolving Routes for MPLS Traffic Engineering. *Juniper Networks White Paper*. Retrieved November 10, 2001, from <http://www.juniper.net/techcenter/techpapers/200008.html>
- [15] Semeria C. (2000). Traffic Engineering for the New Public Network. *Juniper Networks White Paper*. Retrieved November 10, 2001, from <http://www.juniper.net/techcenter/techpapers/200004.html>
- [16] Awduche D. (1999) MPLS and Traffic Engineering in IP Networks. *IEEE Communications Magazine, December, 42-47*.
- [17] Ghanwani A., Jamoussi B., Fedyk D., Smith P., Li L., Feldman N. (1999) Traffic Engineering Standards in IP Networks Using MPLS. *IEEE Communications Magazine, December, 49-53*.
- [18] Swallow G. (1999) MPLS Advantages for Traffic Engineering. *IEEE Communications Magazine, December, 54-57*.
- [19] Metz C. (1999). IP QoS: Traveling in the First Class on the Internet. *IEEE Internet Computing, April, 84-88*.
- [20] Awduche D., Malcolm J., Agogbua J., O'Dell M., McManus J. (1999) Requirements for Traffic Engineering Over MPLS. *IETF RFC 2702*. Retrieved November 10, 2001, from <http://www.ietf.org/rfc/rfc2702.txt>
- [21] Kumar H., Sundaresan K. (2000). Implementation of the Code Excited Linear Predictive (CELP) Codec for VoIP. Retrieved April 1, 2002, from <http://www.acsu.buffalo.edu/~ks48/dspproject.pdf>
- [22] Reddy K., Vemuganti M., Jangi S. Efficient Implementation of a G.729 Speech Coder Using TMS320C6x DSPs. *RadiSys White Paper*. Retrieved April 1, 2002, from <http://www.radisys.com/files/efficient.pdf>
- [23] Tian W., Ni W., Hui G., Wang D. (1995). Implementation of a LD-CELP Codec with Echo Canceller Functions. *The Journal of China Universities of Posts and Telecommunications, Vol. 2, No. 2, December*.
- [24] Kumar A., Gersho A. (1997). LD-CELP Speech Coding with Nonlinear Prediction. *IEEE Signal Processing Letters, Vol. 4, No. 4, April, 89-91*.

- [25] (1996). ITU-T Recommendation G.723.1 Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.2 kbit/s. Retrieved April 1, 2002, from <http://www2.mplayerhq.hu/MPlayer/doc-tech/g723/G.723.1.pdf>
- [26] N. M. Sheikh, K. I. Siddiqui, H. Raza, M. A. F. Alam, Bhauddin. (2001). Real-Time Implementation of ITU-T's G.723.1 Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.3 kbits/s. *IEEE 4<sup>th</sup> International Multi-topics Conference INMIC*.
- [27] (1999). Fragmentation Size and Bandwidth Calculations. *Cisco Systems White Paper*. Retrieved April 1, 2002, from <http://www.cisco.com/warp/public/779/servpro/ms/evolve/docs/FragBWcalc.PDF>
- [28] Davidson J., Peters J., Gracely B. (2001). *Voice over IP Fundamentals*. Cisco Press.
- [29] Held G. (2000). Emerging Technology: Reducing Voice over IP Latency. *Network Magazine*. Retrieved April 1, 2002, from <http://www.networkmagazine.com/article/NMG20000710S0012>
- [30] Percy A. (1999). Understanding Latency in IP Telephony. *Brooktrout Technology White Paper*. Retrieved April 1, 2002, from [http://www.telephonyworld.com/training/brooktrout/iptel\\_latency\\_wp.html](http://www.telephonyworld.com/training/brooktrout/iptel_latency_wp.html)
- [31] Yamada H., Endo T., Oda T. (2001). IP-Based Voice Stream Multiplexing Schemes and their Performance Evaluation. *IEICE Transaction Communications, Vol.E84-B, No.8, August, 2256-2265*.
- [32] Gray E. (2001). *MPLS: Implementing the Technology*. Addison Wesley Professional.
- [33] Apostolopoulos G., et al. (1999). QoS Routing Mechanisms and OSPF Extensions. *RFC 2676*. Retrieved April 1, 2002, from <http://www.ietf.org/rfc/rfc2676.txt>
- [34] Awduche D., Berger L., Gan D., Li T., Srinivasan V., Swallow G. (2001). RSVP-TE: Extensions to RSVP for LSP Tunnels. *RFC 3209*. Retrieved April 1, 2002, from <http://www.ietf.org/rfc/rfc3209.txt>
- [35] Jamoussi, B., et al. (2001). Constraint-based LSP Setup Using LDP. *RFC 3212*. Retrieved April 1, 2002, from <http://www.ietf.org/rfc/rfc3212.txt>
- [36] (1990). Institute of Electrical and Electronics Engineers. *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries*.

- [37] Chen T., Oh T. (1999). Reliable Services in MPLS. *IEEE Communications Magazine*, December, 58-62.
- [38] Pezaros D., Hutchison D. (2001). Quality of Service Assurances for the Next generation Internet. *PG Net 2<sup>nd</sup> Annual Network Symposium*.
- [39] Horlait E., Rouhana N. (2000). Differentiated Services and Integrated Services Use of MPLS. Retrieved April 1, 2002, from <http://www-rp.lip6.fr/~eh/Files/mpls.pdf>
- [40] Guerin R., Peris V. (1999). Quality-of-Service in Packet Networks: Basic Mechanisms and Directions. *Computer Networks and ISDN Systems*, 169-179.
- [41] Jacobson V., Nichols K., Poduri K. (1999). An Expedited Forwarding PHB. *RFC 2598*. Retrieved April 1, 2002, from <http://www.ietf.org/rfc/rfc2598.txt>
- [42] Heinanen J., Baker F., Weiss W., Wroclawski J. (1999). Assured Forwarding PHB Group. *RFC 2597*. Retrieved April 1, 2002, from <http://www.ietf.org/rfc/rfc2597.txt>
- [43] Nichols K., Blake S., Baker F., Black D. (1998). Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. *RFC 2474*. Retrieved April 1, 2002, from <http://www.ietf.org/rfc/rfc2474.txt>
- [44] Yang C., Fu C., Tu Y. (2001). Enterprise Traffic with a Differentiated service Mechanism. *International Journal of Network Management*, 113-128.
- [45] Semeria C. (2001). RFC 2547bis: BGP/MPLS VPN Fundamentals. *Juniper Networks White Paper*. Retrieved April 1, 2002, from <http://www.juniper.net/techcenter/techpapers/200012.html>
- [46] Guichard J., Pepelnjak I. (2000). MPLS and VPN Architecture: A Practical Guide to Understanding, Designing and Deploying MPLS and MPLS-Enabled VPNs. *Cisco Press*.
- [47] Kompella K., Kolon M., Bichon P., Donnell A. (2001). MPLS-based Layer 2 Virtual Private Networks. *Juniper Networks White Paper*. Retrieved April 1, 2002, from <http://www.juniper.net/techcenter/techpapers/200009.html>
- [48] Yang C., Fu C., Tu Y. (2001). Enterprise Traffic with a Differentiated service Mechanism. *International Journal of Network Management*, 113-128.



- [49] Jones P. (2001). Introduction to H.323. *Packetizer: A resource for packet-switched conversational protocols*. Retrieved April 1, 2002, from [http://www.h323forum.org/papers/h323\\_intro\\_von\\_2001.zip](http://www.h323forum.org/papers/h323_intro_von_2001.zip)
- [50] *Packetizer: H.323 Information Site*. Retrieved April 1, 2002, from <http://www.packetizer.com/iptel/h323/>
- [51] Karim A. (1999). H.323 and Associated Protocols. Retrieved April 1, 2002, from <ftp://ftp.netlab.ohio-state.edu/pub/jain/courses/cis788-99/h323.pdf>
- [52] Zhang H. (1995). Service Disciplines For Guaranteed Performance Service in Packet-Switching Networks. *Proceedings of the IEEE, Vol. 83, No. 10, October*, 1374-1396.
- [53] Demers A., Keshav S., Shenker S. (1990). Analysis and simulation of a Fair Queuing Algorithm. *ACM SIGCOMM'89*, 3-12.
- [54] Golestani S. J. (1994). A Self-Clocked Fair Queuing Scheme for Broadband Applications. *IEEE INFOCOM'94, April*, 636-646.
- [55] Bennett J., Zhang H. (1996). WF2Q: Worst-case Fair Weighted Fair Queuing. *IEEE INFOCOM'96, March*, 120-128.
- [56] Keshav S. (1991). On the Efficient Implementation of Fair Queueing. *Internetworking: Research and Experiences, Vol. 2*, 157-173.
- [57] Shreedhar M., Varghese G. (1996). Efficient Fair Queuing using DRR. *IEEE/ACM Transactions on Networking, Vol. 4, No. 3, June*, 375-385.
- [58] Parekh K., Gallager R.. (1993). A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-node Case. *IEEE/ACM Transactions on Networking, Vol. 1, No. 3*, 344-357.
- [59] Guerin R., Peris V. (1999). Quality-of-Service in Packet Networks: Basic Mechanisms and Directions. *Computer Networks and ISDN Systems*, 169-179.
- [60] Francini A., Chiussi F. (2001). A Weighted Fair Queuing Scheduler With Decoupled Bandwidth and Delay Guarantees for the Support of Voice Traffic. *IEEE Globecom '01, November*.
- [61] Guérin R., Kamat S., Peris V., Rajan R. (1998). Scalable QoS Provision Through Buffer Management. *ACM SIGCOMM'98, October*.

- [62] Kanhere S., Sethu H. (2001). Fair, Efficient and Low Latency Packet Scheduling Using Nested DRR.
- [63] Zhang H, Ferrari D. (1994). Rate-Controlled Service Disciplines. *Journal of High Speed Networks*, 389-412.
- [64] Seddigh N., Nandy B., Piedad P. (1999). Bandwidth Assurance issues for TCP flows in a Differentiated Service Network. *Proceedings of Global Internet Symposium, GLOBECOM, December*.
- [65] Andrikopoulos I., Wood L. Pavlou. (2000). A Fair Traffic Conditioner for the AF in a DiffServ Internet. *IEEE ICC 2000, June*.
- [66] Makkar R., et al. (2000). Empirical Study of Buffer Management Scheme for DiffServ AF PHB. *IEEE ICCCN 2000*.
- [67] (1997). Frame Relay Fragmentation Implementation Agreement. *Frame Relay Forum*. Retrieved April 1, 2002, from <http://www.frforum.com/5000/Approved/FRF.12/frf12.pdf>
- [68] Leon-Garcia A., Widjaja I. (2000). Communication Networks: Fundamental Concepts and Key Architectures. *McGraw-Hill Osborne*.
- [69] Bertsekas D., Gallager R. (1992). Data Networks. *Prentice Hall*.
- [70] Gross D., Harris C. (1985). Fundamentals of Queuing Theory. *Wiley-Interscience*.
- [71] Leon-Garcia A. (1994). Probability and Random Processes for Electrical Engineering. *Addison Wesley*.
- [72] Brady P. (1969). A Model for Generating ON-OFF Speech Patterns in Two-Way Conversations. *Bell System Technology Journal, Vol.48, September*.
- [73] Chang X., Tan T., Subramanian K. (1999). Source Traffic Modeling in OPNET. *Proceedings of OPNETWORK'99*.
- [74] Kim D., Choi S, Choi J. (2001). Performance Analysis of Differentiated Service for Voice over IP. *IEICE Transaction Communications, Vol.E84-B, No.11, November*.
- [75] (2002). Journal of Internet Test Methodologies. *Agilent Router Tester Solution*. Retrieved April 1, 2002, from <http://advanced.comms.agilent.com/RouterTester/member/journal/Journal.pdf>

- [76] Bertsekas D., Gallager R. (1992). *Data Networks. Prentice Hall.*
- [77] Leon-Garcia A. (1994). *Probability and Random Processes for Electrical Engineering. Addison Wesley.*
- [78] Yamada H., Endo T., Oda T. (2001). *IP-Based Voice Stream Multiplexing Schemes and their Performance Evaluation. IEICE Transaction Communications, Vol.E84-B, No.8, August, 2256-2265.*
- [79] Guerin R., Pla V. (2000). *Aggregation and Conformance in Differentiated Service Networks. Proceedings ITC Specialist Seminar on IP Traffic Modeling, Measurement, and Management.*